Evaluating Deep Learning Models for Minute-Level Bitcoin Forecasting and Subsequent Trading

By Max Austin

With supervision by Dough Santry

Submitted in partial fulfilment of the requirements of the degree of
MSc Data Science, University of Kent, 2025

Summary

Purpose

Through this study, we will see if deep learning models can forecast Bitcoin price at minute levels and more importantly if such forecasts can lead to profitable trading strategies once realistic frictions (like transaction cost) are taken into account. Two neural architectures were considered: a Long Short-Term Memory (LSTM) network, which is suited to modelling short-range temporal dependencies, and a Transformer, an attention-based model designed to capture longer-range dynamics. Could forecasts be made accurately and if so, does predictive accuracy add value for trading? Since accurate forecasts will not add value unless they lead to a persistent improvement in the performance of a portfolio.

Methods

The analysis is based on 2012-2025 one-minute Bitcoin OHLCV data. Through exchange fees and network fees estimates, the optimal strategy approximates live trading conditions. The forecast horizons went from one minute to one week, allowing us to evaluate very short and longer time horizons. The decisions to trade were deterministic and rule-based, with the forecasted returns triggering buy and/or sell transactions. To avoid look-ahead bias and bring the procedure closer to real-life deployment rolling walk-forward model parameters were tuned. The accuracy of the forecasting was evaluated using numerous error measures such as the mean squared error (MSE), root-mean-square-error (RMSE), normalised RMSE (NRMSE), and mean absolute percentage error (MAPE). Comparison of outcomes versus a naïve Buy-and-Hold benchmark and assessment of the outcome in terms of equity growth, drawdown, and robustness in different months.

Key Findings

The LSTM was more accurate than the Transformer through all horizons (1–60 minutes). This suggests that volatility is most pronounced at short horizons. The difference in comparative measurements were more than noteworthy: the MSE for the LSTM is about 14 times smaller than the transformer and RMSE is about 1.5 times smaller. The Transformer produced more consistent forecasts but was systematically biased and unstable under non-stationary conditions and therefore was less effective for trading. LSTM forecasts incorporated into trading strategies led to improved profitability and higher robustness. They were able to take most of the upside, offer some protection to the downside when the market fell (such as during the fall of February 2025) and deliver a reasonable amount of compounding consistently over six months. Unlike the LSTM forecasts, strategies based on Transformer forecasts maintained an inactive position at critical points, generated whipsaw losses, and failed to align with the prevailing regimes, indicating sustained underperformance relative to the LSTM and Buy-and-Hold baseline.

Conclusion

The study concludes that LSTM architectures are better suited for high-frequency cryptocurrency forecasting and trading, given their adaptability to short-horizon volatility and capacity to preserve capital during adverse conditions. Transformers retain theoretical advantages in modelling long-range dependencies but require architectural refinements (e.g., autoregressive decoding, continuous sequence training) before achieving comparable results in practice. And in regards to the feasibility of using neural architecture to forecast and subsequently trade with, the evidence certainly suggests its more than possible and effective, however a significantly larger amount of testing would be needed to make it reliable to a level where it can be trusted with money

Future Directions

There are many options that can be tried to extend the research. These options include new model classes (e.g. TCN, TFT, CNN–LSTM) and reinforcement learning-style trading policies that optimise decisions directly. Further tuning of models and introducing more variables as inputs. Ultimately, progressing to live with exchange APIs, live execution and operational risk controls. The above-mentioned directions will help transition from back tests under control to production systems that are resilient.

Table of Contents

Introduction

Context and Motivation

Since 2009, Bitcoin has transitioned from a unique digital currency to a robust financial instrument that is traded globally in deep and liquid markets. Bitcoin is traded 24/7 on a decentralized network of exchanges. It is valued differently than stock or government bonds. Its daily price is far more flexible to change. The distinctive characteristics of the digital currency market, which include high liquidity, 24/7 availability, and extreme price fluctuations within a single day, make Bitcoin an ideal candidate for high-frequency forecasting and algorithmic trading research (Cheah & Fry, 2015; Corbet et al., 2019). Most research uses hourly or daily data, mainly because it is easier to work with and cheaper to compute. However, this methodological choice has been criticized for throwing away the very features that make cryptocurrency markets distinctive like intraday reversals, liquidity shocks, and transient arbitrages (Corbet et al., 2019). When you take minute-level fluctuations and collate them into daily aggregates, you risk creating models that might read well in error metrics but have no economic value in art. Models like ARIMA and GARCH can seem stable when viewed from afar. But it's a mirage! When looked at more closely at daily frequency data, they collapse. The focus of this study on the minute scale is motivated by the fact that this gap represents not only a lost opportunity but also a constraint in the literature. It has important meaning from both practical and academic points of view to obtain reliably accurate forecasts over short horizons. Researchers can check whether advanced machine learning architectures, deep neural networks in particular, can capture non-linear dependencies which are not captured by conventional econometric models. For traders, these forecasts form the foundation of automated trading systems because even small levels of predictive accuracy can make a large difference in profitability when leveraged over large volumes. Due to the strong non-stationarity of the Bitcoin price series and the volatility clustering, conventional statistic methods like ARIMA and GARCH have not performed well in these settings (Tsay, 2010; Cheah & Fry, 2015). The above challenges necessitate the study of deep learning architectures. They have been shown to be successful in modelling complex sequential dependencies in different domains.

Research Objectives

This dissertation attempts to assess the potential of deep learning models to forecast Bitcoin prices at the minute level over multiple horizons and if an improvement in forecast accuracy translates into gains in simulated trading performance. The goal is to evaluate if sequence based architectures like LSTM network and the transformer are suitable for high frequency forecasting and algorithmic trading requirement. The specific goals of this study are fourfold. First, the dissertation analyses and compares LSTM and Transformer models under identical experimental settings, removing other elements from the equation so differences can be attributed only to structural design. The models are assessed across twelve separate forecast horizons, from one minute to seven days ahead, in order to investigate how predictive skill changes as the time horizon lengthens. Another point is that forecasts are embedded in a deterministic, algorithmic-based trading strategy, thus providing a direct conduit for translating statistical performance into monetary results (Bailey et al., 2014). In the end, trading outcomes are compared with a simple Buy-and-Hold baseline under realistic transaction cost assumptions in order to give a real-world perspective on whether complex forecasting beats a passive investment strategy.

Research Questions

The two key research questions are the guideline of the inquiry. The first one is related to forecasting performance: which architecture, LSTM or Transformer, produces more reliable minute-scale forecasts, and do absolute error measures (e.g. mean squared error, root mean squared error) perform better than proportional measures (e.g. normalised RMSE, mean absolute percentage error)? Which model is more accurate is one question but does choice of metric matter for assessing success of our forecasting? The second question relates to trading performance: to what extent do improvements in forecast accuracy line up with any measurable gains in portfolio value when forecasts are used in an algorithmic trading system? Also, do specific market circumstances favour one model over the other? For example, bull, bear, or sideways. Answering these questions will reveal how useful deep learning forecasts are in actual trading settings; if a forecast has low error statistics, it does not mean it can make money if we use it.

Significance of the Study

This dissertation is significant since it tries to link two lines of research that are often followed independently: forecast accuracy and trading profits. Many financial time series modelling studies assess predictive performance in isolation, ignoring downstream implications for trading strategies. On the other hand, algorithms used for trading strategies provoke an interest. Their formulation is often heuristic in nature. Furthermore, there is a lack of rigorous linking. This linking occurs back to the forecast quality in the literature. By making those two domains explicit, this study provides a more integrative evaluation framework. It is also timely to compare LSTM and Transformer from an academic perspective. Although LSTMs are used often in financial deep learning, we already know about their slow training, difficulties learning very, long sequences since they were not designed for this purpose, and overfitting in non-stationary settings (Nelson et al., 2017; Fischer & Krauss, 2018).". Transformer-based architectures coming out in recent times are believed to solve the recurrence of sequence with global self attention, but their performance in finance has not been very good. Studies about equities show slight improvements. It is hard to forecast cryptocurrencies. Other Transformer variants (such as Informer, Auto former) become unstable with noise and regime shifts in the case of cryptocurrency markets (Wu et al., 2021; Zhou et al., 2021; Zhang et al., 2023). The inconsistency in the literature poses an important question. Do Transformers add genuine value in "high-frequency finance" or are they merely taking advantage of cleaner sequences in text or speech? This project examines whether the much-discussed advantages of attention are destroyed by the realities of minute-level cryptocurrency data by directly contrasting LSTM and Transformer in the same conditions. From practical view, this research work is designed with deployment approach in mind. The evaluation framework takes into account realistic transaction costs, includes walk-forward validation to prevent overfitting, and performs drawdown analysis to assess downside risks. These attributes guarantee results that are both sound and applicable to practitioners who want to design strengthened trading systems. The dissertation helps practitioners strike the right balance between model and real world utility by specifying the conditions under which complex deep learning architectures yield benefits over simpler baselines.

Scope and Limitations

The dissertation has been designed to allow a thorough investigation of the issue. Limiting the study of Bitcoin is both a strength and weakness. As mentioned above, the Bitcoin market has the best liquidity among all markets in the world and trades continuously which makes it the best and the most representative testing ground for high-frequency forecasting (Corbet et al., 2019). That said, critics of single-asset studies note that results tend not to generalize to other assets that operate under different

volatility regimes (Ethereum's, altcoins etc). Just like this, any comparative study done between two architectures ignores various other baselines that might perform on par as the tested ones but at a lower cost. Nevertheless, this conscious limitation is justified here: as a liquid and globally-sensitive asset, Bitcoin faces the hardest modelling challenge in the cryptocurrency markets. If something doesn't work here, it is unlikely to work anywhere. As such, the scope is not intended to be exhaustive but imposing a strict feasibility test under the severest possible conditions. There are, however, important limitations. Findings that were obtained based on Bitcoin can probably not be generalised to other cryptocurrencies or to traditional financial assets that have different volatility structures, liquidity conditions or regulatory constraints. In addition, all assessments are performed in simulation, meaning that market frictions from live execution, including slippage, order latency and exchange execution risk, are abstracted away. While transaction costs are included, these are also estimates. To wrap it up, the hyperparameters are chosen wisely but not exhaustively tuned, so the results should be seen as indicative rather than global optima.

Structure of the Dissertation

The rest of the dissertation is divided into five chapters. Part 0 contains the methodology which contains data collection, feature selection, preprocessing, and sequence preparation. Parts one and one-five present the forecasting models. Part one describes the design, training, and evaluation of the LSTM. Part one-five gives an equivalent treatment to the Transformer. Part 2 provides an overview of forecasts' integration into a deterministic algorithmic trading framework, with trading results and Buy-and-Hold comparison. The Discussion pulls together evidence from forecasting and trading, interpreting and contextualising it in relation to earlier works. The Conclusion sums up key contributions of this study and provides directions for future research that include extension to other model families, reinforcement learning–based strategies and live market tests.

Body

Part 0 – Methodology

Data Source

The analysis of this research is based on minute-level data of the Bitcoin trader from 2012 to 2025 that has been sourced from Kaggle (McZielinski, 2025). The dataset aggregates historical bitcoin prices across major exchanges, and the dataset is complemented in an OHLCV format. The most granular and continual public data was this minute-by-minute data that was closest to live conditions of the market and also computationally manageable for back tests. In contrast to hourly or daily datasets that smooth away short-lived spikes in volatility, minute-level data preserves Bitcoin's entire microstructure, including spikes in volatility and short-lived reversals back to averages, which are known to contain valuable information for high-frequency modelling (Andersen et al., 2000; Cont, 2001. The former is critical to high-frequency forecasting and trading.

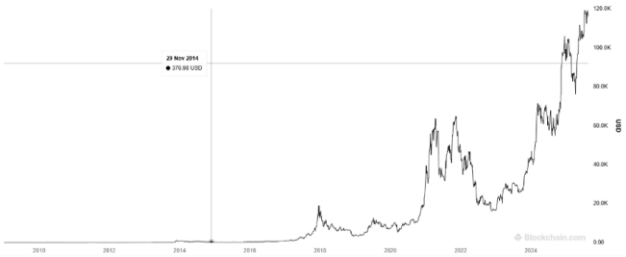


Figure 1 – Minute by minute bitcoin prices

To make trading simulations realistic, transaction costs were added to the data set. This is because realistic trading simulations must incorporate transaction and brokerage fees, since neglecting them leads to false back test profitability (Bailey et al., 2014) Two distinct fee types were modelled. To begin with, network fees are payments made to bitcoin miners to get transaction confirmations. The price of transaction fees depends on the demand and congestion surrounding blockchain and has no relation with the value of the transaction. For this element, we used the average transaction fees in USD from the public explorer of Blockchain.com. Brokerage fees are fees charged by the exchange based on percentage. It was assumed that a flat rate of 0.1% reflecting Binance's published spot trading costs. Although it is a fact that exchanges like Binance do not price network fees for off-chain internal transfers, both costs were embodied in the simulation to create stricter conditions of profitability. This conservative assumption ensures that any profitability demonstrated by the models cannot be attributed to overly optimistic cost assumptions.

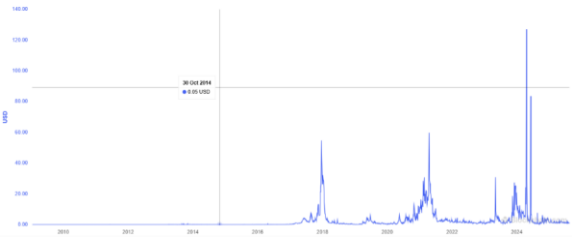


Figure 2 – Average daily transaction costs USD

Feature Selection

Log return calculated from closing prices is the key predictive variable, consistent with established practices in financial econometrics (Cont, 2001; Tsay, 2010). While widely used, this transformation can under-report the tail events that are most important for profitability. However, this traditional selection subsequently compresses the extreme price movements, which are exactly those events that dominate trading risk. So, while a standard method of estimating volatility, it may under-report the tail events most important for their algorithm profitability. Along with returns, variables related to fees were also included to ensure that the trading simulations incorporated realistic frictions. According to the work of Bailey et al. (2014), neglecting transaction cost would yield false back test profitability but they consider static frictions present in equity markets. Costs on exchanges in crypto markets are exchange-specific and dynamically affected by network congestion. Therefore, even their corrective does not account for the drag of actual execution costs. Although many studies engineer indicators such as RSI or moving averages, their incremental value is often limited once modern deep learning models are applied, since these networks can extract nonlinear temporal features directly from raw returns (Nelson et al., 2017; Fischer & Krauss, 2018). . Some say that these models increase the risk of overfitting without helping accuracy. To ensure the limiting factors are methodological and not merely pragmatic, the study

researchers limit the features in these models. By doing so, the impact of model architecture is assimilated without the confounding effect of indicator engineering. By examining forecasts from two different underlying models, one can more directly test an added forecasting skill of either the LSTM or the Transformer architectures, rather than relying on redundancy in the technical signals.

Data Cleaning

The dataset was cleaned before the training of the model. The duplicate timestamps were deleted to prevent distortions in the sequence continuity. Although scarce, the missing values were fill in using forward interpolation, to keep with time series without any exogenous information. For high-frequency series, continuity is critical, as irregular gaps distort temporal dependence and degrade model performance (Hyndman & Athanasopoulos, 2021). Data type consistency was also ensured. Financial time series models are known to be really sensitive to numerical instability, particularly when gradient-based optimisation is used. All price and return variables were converted into floating point 64 format for precision while categorical values (if they were present, like timestamps) were encoded into sequence modelling compatible format.

Scaling

Before training, features were standardised by z-score scaling so the result has zero mean and unit variance. This decision was motivated by two considerations. Ioffe and Szegedy (2015) show that normalisation mitigates internal covariate shift and stabilises training process. Another thing is, financial returns are unbounded and MinMax scaling does not work. Brownlee (2017) also suggests applying z-score scaling to returns to ensure that features with larger variance do not dominate learning. In order to prevent any information leakage, standardisation was applied separately to both training and test sets. We transformed everything back for interpretability during evaluation and trading simulations.

Data Splitting

A rolling walk-forward validation procedure was used to assess forecasting performance under live trading conditions. Following the approach developed by Bailey et al. (2014), the data was partitioned into training windows of three months, followed by test windows of one month. This rolling scheme avoids lookahead bias by making sure that all forecasts are produced from past data, not revelation of future data. Bitcoin markets are non-stationary, so models must be retrained frequently in order to account for changes in regime, for example, from bull rallies to bear corrections. Even though we tried longer training windows, they did not provide a lot of extra performance compared to the much greater cost of computing power. So, we just pragmatically decided on a 3:1 ratio.

Sequence Length Choice

Input sequences for the models were constructed using a 1-day lookback window (1,440 minutes). According to Wen and Ling (2023), a time frame of 24-hours was able to capture short-term dependencies in the crypto market and remove the noise of longer horizons. Using a shorter time frame can overlook relevant intraday cycles. In contrast, a longer time frame can dilute the high-frequency structure of price action with extraneous information. The models learned rapid movements in trading activity while also capturing daily changes as the input length was fixed to 1,440 timesteps.

K-Pooling for Sequence Reduction

To reduce computational costs, the raw input sequence of 1,440 timesteps was averaged pooled over non-overlapping intervals of 12 minutes to yield 120. Andersen et al. (2000) argue that de facto filtering of microstructure noise occurred without destroying the macro behaviour. Bai et al. (2018) show how pooling layers can help the generalisation of a sequence model by removing redundancies. The selection of pooling factors centred on a reasonably high pooling factor of 12 so that enough detail is retained for the short-horizon forecasts but not so high as to make estimation infeasible in a model simulation.

Software Environment

Experiments were carried out in Python version 3.10.18 on Visual Studio Code using a Windows workstation equipped with CUDA-enabled graphics processing units (GPUs) for fast training. We selected PyTorch 2.7.1+cu128 as the principle deep learning framework for its working flexibility with custom sequence models. For data manipulation, Matplotlib (3.10.5) was used for visualisation, and scikit-learn (1.15.3) were the preprocessing utilities such as scaling. We used Numba (0.61.2) for just-in-time compilation to speed up the trading simulator and other important parts of the project. The mixed precision training, conducted through torch.cuda.amp followed from Micikevicius et al. (2018), which demonstrated that reduced-precision arithmetic may significantly increase training speed and lower memory usage without decreasing predictive power. This arrangement allowed the experiments to be conducted on full-resolution minute-level data over a multi-year horizon in a reasonable time frame.

Part 1 – Forecasting with LSTM

Context

The LSTM is the first model we are going to look into in this research; this is a variant of recurrent neural networks which can capture the temporal dependencies in sequences of data. The objective was to predict the absolute price of Bitcoin over twelve horizons ranging from one minute to seven days ahead based on minute-level OHLCV data with transaction fees. Under a multi-horizon setting, forecasting was conducted such that the model generates predictions for all horizons simultaneously instead of training a model for each. A rolling walk-forward validation scheme was used to decrease lookahead bias and mimic realistic deployment. Each training window comprised three months of historical minutely data, plus one month of testing. This ensured that models adapted to changing market scenarios while being strictly evaluated on new data. Rolling designs are recommended in financial forecast, especially for the non-stationary asset (like Bitcoin) (Bailey et al., 2014).

What is LSTM

Hochreiter and Schmidhuber (1997) introduced LSTMs to tackle the vanishing gradient problem found in ordinary recurrent systems. The gating mechanism through input-forget-output of LSTM network is its key innovation that helps to control information flow. As a result, LSTMs can maintain and update a memory cell for long sequences. Consequently, LSTMs are able to handle short-term and long-term dependencies. LSTMs are a popular choice for modelling non-linear dynamics, volatility clustering, and regime shifts in finance applications. The studies of Nelson et al. (2017) and Fischer & Krauss (2018) document the power of LSTMs in capturing short-term financial dynamics, but both are restricted to

equities or daily horizons where liquidity and noise structures differ markedly from that of Bitcoin at minute-level. As such, their findings show promise, but the question remains whether LSTMs would still dominate in the extreme volatility and non-stationarity of cryptocurrency markets.

Implementation in This Study

The input sequence for one day of historical data (1,440 minutes) for an hour's price movement, is in concurrence with the previous researcher to be sufficient to retain maximum signal and minimise noise (Wen & Ling, 2023). Before entering the network, the data is normalised at the layer, which was found to stabilise training in recurrent architectures (Ba et al., 2016) In order to decrease noise and lessen the computational load, 12-minute average pooling was performed on the sequences to downsample them, giving 120 timesteps per sequence. This aggregation filtered out fluctuations on the microstructure level but not on a broader scale. (Bai et al., 2018) The model had three LSTM layers of 128 units each with a dropout of 0.2 between the layers. Stacked designs (Zhang et al., 2020) increases representational capacity for complex temporal patterns while dropout (Gal & Ghahramani, 2016) mitigates overfitting through stochastic regularisation. The last hidden state underwent a LayerNorm operation, followed by a fully connected output layer that yielded a dozen simultaneous forecasts of log returns for the fixed horizons.

Training Configuration

Training was done using Adam optimiser with a learning rate of 0.001. This seems to be a standard setting for non-stationary time series (Kingma & Ba, 2014). The log returns have a loss function of MSE (mean square error) as this loss function penalizes large deviations more a like error. Also, this loss function is consistent with continuous-valued regression tasks (Chai & Draxler, 2014). We maintain the batch size as 256 for gradient stability and GPU memory limitations. The training was limited to at most 15 epochs while early stopping (patience = 1) was used to prevent overfitting (Prechelt 1998). To reduce computational costs, we trained our framework using mixed precision which did not sacrifice speed nor accuracy (Micikevicius et al., 2018) The forecasts outputted, which were expressed in log-return space, were transformed back to absolute price forecasts by means of the exponential mapping generally used in financial econometrics (Tsay, 2010). Lastly, after post-processing, by applying an exponential moving average ($\alpha = 0.5$) to the forecasts, the short-term fluctuations were smoothed out as shown by tests carried out for the 2025 evaluation period.

PART 1.5 Forecasting - Transformer

Context

This study also employed a Transformer as a forecasting model, which is based on self-attention neural architecture. The transformer imposed standardization in NLP and more recently by some researchers in time series forecast due to the ability to capture long-range dependencies more efficiently than those of recurring architectures (Vaswani et al., 2017; Wu et al., 2021; Zhou et al., 2021). Transformers process all timesteps together instead of LSTMs that process them one by one. Thus, due to its better computational efficiency, it is theoretically more powerful than LSTMs at dealing with longer historical contexts. For the purpose of forecasting Bitcoin values at different horizons, the Transformer was implemented in this dissertation under the same experimental framework as the LSTM. The inputs comprised one day of

minute-level OHLCV data (with transaction cost) that had been pre-processed in the same manner for easy comparability. The evaluation loop was also designed to roll forward by a walk of three months training and one month testing. We choose this set-up to keep the experimental conditions consistent over time. It also served to avoid lookahead bias.

What is the Transformer

The transformer architecture was presented by Vaswani et al. in Attention is all you need (2017). The self-attention mechanism lets the model to weigh input positions by their importance while making a prediction. Compared with recurrent models that rely on sequential memory, through self-attention, the Transformer can connect with other distant timesteps directly. This idea is particularly attractive in domains that require remembering long-term dependencies. After research has maintained the transformer towards the time sequence forecasting. Wu and Zhou point to the use of Transformer variants (Informer, Autoformer) for multi-horizon forecasting. However, the gains reported by Wu and Zhou are mainly limited to structured, largely stationary datasets (electricity demand). As warned by Zhang et al. (2023), these results do not directly transfer to high-noise regimes such as finance, as the volatility regime is more severe and less known. However, the use of Transformers to high-frequency financial series is not popular. As such, the study offers an empirical test of whether these theoretical advantages actually translate into a practical improvement over LSTM in the volatile non-stationarity of Bitcoin market.

Implementation in This Study

This project implemented a custom encoder-only Transformer in PyTorch for multi-horizon forecasting. The data was taken for 1 day, or 1440 minutes, and was average pooled with a factor of 12, resulting in a sequence length of 120 timesteps. Similar to the LSTM, the pooling altered the noise reduction while preserving trend information (Bai et al., 2018). Initially, a linear embedding layer was used to project the input features into the model dimension 256, and positional encodings were added after that, as self-attention contains no inherent positional information (Vaswani et al., 2017). The encoder takes the form of three stacked Transformer layers, each made up of four attention heads, followed by a feedforward sublayer of dimension 512 with dropout .2. After encoding, the sequence outputs undergo global average pooling followed by a linear projection layer. This process yields twelve simultaneous forecasts of log returns at the specified horizons.

Training Configuration

The conditions under which the Transformer was trained were similar to LSTM. An Adam optimizer with learning rate 0.001 was used to train the model with MSE on log returns as the loss function. We have made use of a smaller batch size of 256 in our training, owing to the attention layer's heavier memory use. Training of the model lasted 15 epochs with early stopping (patience = 1) to avoid overfitting (Prechelt, 1998). To speed up training and lessen GPU memory use (Micikevicius et al, 2018), we again utilized mixed precision training The log-return space used for predictions was transformed to absolute price forecasts through exponential transformation (Tsay, 2010) just like LSTM. The forecast trajectories, particularly at longer horizons where variance was high, were stabilised by empirical post-processing with exponential moving average smoothing ($\alpha = 0.5$).

Data Handling and Evaluation

To maintain comparability with the LSTM, the Transformer adhered to the same rolling window scheme of 3:1, which is three months training and one month testing. Despite early tests indicating smaller forecast errors when utilizing longer training horizons for the size ratio, the costs involved and the limited benefits led to the adoption of the 3:1 ratio.

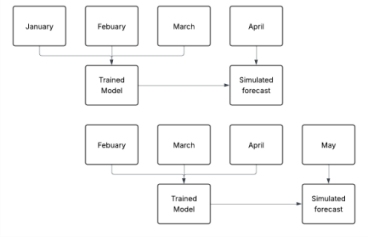


Figure 3 – 3:1 rolling split example

The evaluation metrics were the same: MSE, RMSE, NRMSE, and MAPE. This ongoing assessment permitted a straightforward architectural comparison, disentangling whether the Transformer's self-attention mechanism affords practical predictive advantages over the sequential memory-based configuration of the LSTM.

Aggregate Performance

Metric	LSTM	Transformer	Relative Difference
MSE	1,716,567	24,420,155	Transformer ~14× higher
RMSE	796.26	1,189.07	Transformer ~1.5× higher
NRMSE	0.05272	0.05102	Similar magnitude
MAPE	3.267%	3.290%	Nearly identical

Table 1

Both models' aggregate forecasting performance over the January–June 2025 evaluation period is summarized in Table 1. While the absolute error metrics indicate an MSE of 1,716,567 for LSTM and 24,420,155 for the Transformer, the superior performance of the LSTM is obvious. Similarly, the RMSE of LSTM (796.26) was about 1.5× lower than the Transformer (1,189.07). On the contrary, the normalised metrics (NRMSE and MAPE) yield similarly low values for both models. The NRMSE values are 0.05272 and 0.05102 for the LSTM and Transformer, respectively. The MAPE values are 3.267% and 3.290%. The output indicates an age-old problem in the evaluation of financial forecasts (Bailey et al. 2014) – normalized error measures can mask economically significant differences in absolute predictive accuracy, especially at high price levels and with large variance. In a real trading situation, the absolute error gap found here will lead to very different economic outcomes.

Monthly Observations

LSTM

(Figures 4 – 9 in appendix)

The LSTM model maintained a strong short-horizon accuracy, which consistently aligned with the true movements of the series within the 1–30 minute timeframe. For example, the forecasts for January and February kept pace with early price direction before deviating at longer horizons. At longer time scales, the model became noticeably biased. More specifically, it overshot significantly during prices' January to March upward run. Similarly, it under-forecasted during the May to June bullish reversal. This highlighted another fault in the April forecasts; their oscillatory swings fail to capture the smooth upward trend in the true series, signifying underfitting of longer-run structure. Even with these problems, the LSTM managed to adapt by flipping the direction of the bias, thanks to the recent adjustments. This is highly beneficial for cases of rolling retraining.

Transformer

(Figures 10-15 in appendix)

On the other hand, the Transformer showed instability patterns across horizons. Often, it would forecast these sharp swings in errors, with a U shape in January, large mid-horizon collapses and a late upturn in February. This behaviour represents difficulties in maintaining temporal alignment within non-stationary conditions. It was found to overpredict in March and April. Subsequently, in May and June it was found to underpredict. Thus, it does not adjust the direction of bias dynamically. Furthermore, in the short horizons, Transformer was less accurate than LSTM, thereby making it less useful in high-frequency trading situations, which require responsiveness.

Quantitative and Qualitative Comparison

Taken together, the results confirm that **the LSTM was superior in this high-frequency forecasting task**. Its lower MSE and RMSE, alongside stronger short-term alignment, underscore the advantage of sequential memory architectures for modelling Bitcoin's microstructure. These findings align with Fischer and Krauss (2018) and Nelson et al. (2017), both of whom found LSTMs effective in capturing non-linear dependencies and volatility patterns in financial data.

While the Transformer holds theoretical advantages in modelling long-range dependencies, these did not materialise under the present experimental setup. Several design factors contributed:

- **Chunked input sequences** limited continuity, weakening self-attention's ability to exploit long horizons.
- The absence of an **autoregressive decoder** meant the model lacked explicit mechanisms for sequential multi-step forecasting.
- Hyperparameters optimised for NLP or lower-frequency financial data were not well-tuned for minute-level cryptocurrency forecasting.

Consequently, although Transformers produced smoother outputs in some low-volatility periods — a potential benefit for risk-averse trading strategies — they were consistently inferior to LSTMs across both absolute and relative error measures.

Interpretation and Implications

The finding that LSTMs consistently outperform Transformers at minute resolution aligns with Nelson et al. (2017) and Fischer & Krauss (2018), who emphasise the importance of short-term memory in financial forecasting. In contrast, our results challenge the optimism of recent Transformer-based forecasting studies (e.g., Wu et al., 2021; Zhou et al., 2021), which show strong performance in structured, low-noise domains but do not generalise well to volatile assets like Bitcoin. This divergence suggests that the self-attention mechanism, while powerful in capturing long-range dependencies, may be ill-suited for markets where predictive signals are fleeting and dominated by noise. Rather than simply attributing Transformer underperformance to implementation, the evidence here indicates a deeper issue: architectural strengths in NLP and energy forecasting do not automatically transfer to financial time series. Future work must therefore ask not only how to adapt Transformers technically but whether they are conceptually appropriate for minute-level cryptocurrency prediction at all.

In summary, while both models achieved comparable relative error rates, only the LSTM delivered robust absolute accuracy. For practical applications such as algorithmic trading, this distinction is crucial: a model with lower RMSE translates directly into tighter forecast intervals, reduced slippage, and greater economic viability.

Part 2 – Algorithmic Trading (2500)

Context and Goal

To make the multi-horizon forecasts from the LSTM and Transformer transparent, we turn them into deterministic long/flat trading rules and evaluate these with a rolling "champion" procedure: we tune on month d_0 and trade the next month d_1. Our system uses feedback from LSTMs to fine-tune the positioning of trading rules. It uses the same active input space across both the LSTM and Transformer. It will help to lessen look-ahead as well as data/snooping bias; these are similar to walk forward out-of-sample practices that are commonly used in finance.

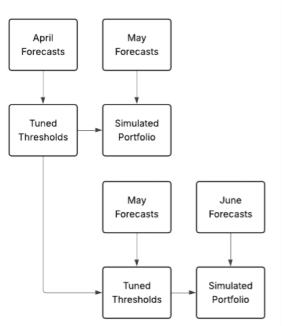


Figure 16 – Train on X run on X + 1 procedure

Signals and Decision Logic

Inputs (per minute i)

- Current mid-price: p_i
- Proportional fee (broker): f_i

- Fixed fee (network): c_i
- Forecast vector over 12 horizons: $\{\hat{P}_{i+h}\}$

Define forecast-based "gain" signals relative to current price:

$g_0(i) = \max_{(h \in Hshort)} (\hat{P}_{i+h} - p_i)$
$g_1(i) = \max_{(h \in Hmid)} (\hat{P}_{i+h} - p_i)$
$g_2(i) = \max_{(h \in Hlong)} (\hat{P}_{i+h} - p_i)$

where Hshort = first 4 horizons, Hmid = first 7 horizons, Hlong = all 12 horizons.

The round-trip cost to overcome at minute i is:

$cost_i = 2 f_i p_i + c_i$

which explicitly prices broker percentage fees (both sides) plus the fixed network fee.

Trend (slope) filters

We compute normalised slope measures over local windows (indices relative to i):

- Short slope S_s over past 3 and future 4 points
- Mid slope S_m over past 6 and future 7 points
- Long slope S_l over past 11 and future 12 points

These serve as regime filters to avoid whipsaws; windows were chosen for simplicity and interpretability rather than exhaustive tuning.

Entry/Exit Rules with Cooldowns

Let short_gain_mult, mid_gain_mult, and long_gain_mult be buy thresholds; and short_gain_mult_sell, mid_gain_mult_sell, and long_gain_mult_sell be sell thresholds.
Let bullish_slope_* and bearish_slope_* be slope cut-offs.
Trades are allowed only if the relevant buy/sell cooldown has elapsed.

Buy condition (when flat): trigger if any hold

$g_0 >$ short_gain_mult $\times cost_i$ and $S_s >$ bullish_slope_short
$g_1 >$ mid_gain_mult $\times cost_i$ and $S_m >$ bullish_slope_mid
$g_2 >$ long_gain_mult $\times cost_i$ and $S_l >$ bullish_slope_long

Sell condition (when long): trigger if any hold

$g_0 <$ – short_gain_mult_sell $\times cost_i$ and $S_s <$ – bearish_slope_short
$g_1 <$ – mid_gain_mult_sell $\times cost_i$ and $S_m <$ – bearish_slope_mid
$g_2 <$ – long_gain_mult_sell $\times cost_i$ and $S_l <$ – bearish_slope_long

The rules are strictly deterministic and cost-aware; no trades occur unless the forecasted move clears round-trip costs and the trend filter agrees.

Execution Model (Fees)

Trade prices include proportional and fixed fees:

- Buy fill: $p_i (1 + f_i) + c_i$
- Sell fill: $p_i (1 – f_i) – c_i$

This conservative assumption avoids overstating profits.

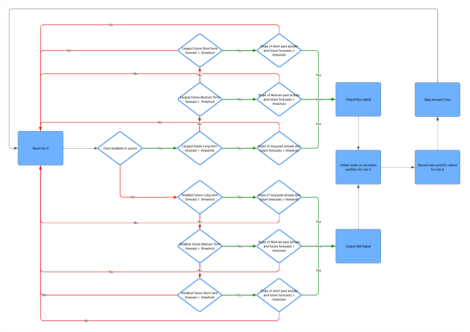


Figure 17 - Rules

Cooldowns

We enforce buy_cooldown_minutes and sell_cooldown_minutes to limit trade frequency, with typical ranges 20–50 minutes. This is to accommodate the fact that bitcoin transactions can take on average 10 minutes to process.

Parameter Space and Random Search

Tuned parameters include:

{short_gain_mult, mid_gain_mult, long_gain_mult, short_gain_mult_sell, mid_gain_mult_sell, long_gain_mult_sell, bullish_slope_short, bullish_slope_mid, bullish_slope_long, bearish_slope_short, bearish_slope_mid, bearish_slope_long, buy_cooldown_minutes, sell_cooldown_minutes}

Discrete grids: multipliers $\in$ {0, 0.2, 0.4, 0.6, 0.8, 1,1.2,1.4,1.6,1.8,2}; cooldowns $\in$ {20, 30, 40, 50}.
We sample 3,000 combinations per tuning month and score with a Numba-JIT month simulator.
Random search is preferred over exhaustive grids for efficiency and coverage.

Rolling “Champion” Flow (tune d_0 → trade d_1)

For each adjacent month pair (d_0, d_1):

1. Random-grid on d_0 → select best by final portfolio value (grid_best).
2. Compare to current global champion on d_0; keep the better one.
3. Deploy the champion (frozen parameters) to trade d_1 (no further tuning).
4. Save artefacts: champion parameters, equity curve, and Buy-and-Hold baseline.

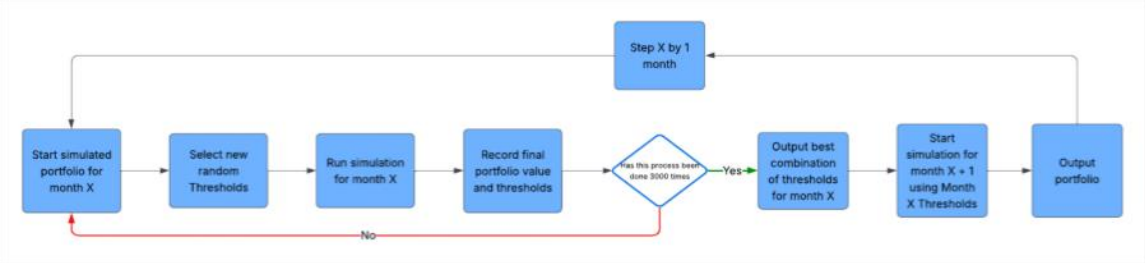


Figure 18 – Rolling champion flow

Performance Measurement and Outputs

Final portfolio value determined the main metric: cash values at the start. A baseline strategy is a buy-and-hold over the same month but with fees applied. We also keep track of trade counts, transaction costs per trade and export the artefacts portfolio.csv, portfolio.png, best_params.csv, best_params_live.csv.

Assumptions and Limitations

Trades are assumed to fill immediately at fee-adjusted prices. This is conservative, as real blockchain confirmation averages ~10 minutes, but exchange executions can be instantaneous. Rolling champions mitigate, but cannot eliminate, parameter instability. The strategy's effectiveness is ultimately upper-bounded by forecast quality.

LSTM Algorithmic-Based Trading Performance (Standalone months)

How the Simulation Was Run

A walk-forward rolling champion approach was again applied:

- A randomised grid search tuned thresholds on month d_0.
- The best parameter set was frozen and deployed to trade month d_1.
- Trading decisions were made minute-by-minute using only current and past information, ensuring no look-ahead bias.
- Performance was measured against a Buy & Hold baseline, with both portfolios starting from the same initial capital.

Two simulation variants were run:

1. **Standalone months** – resetting capital each month to isolate performance.
2. **Continued portfolio** – starting in January and compounding across months.

Note: due to the randomness of champion selection, continued-portfolio results can diverge from the standalone profile.

The first layer of evaluation examined model-driven trading on a **standalone monthly basis**, with the champion parameter set tuned on the immediately preceding month and then deployed out-of-sample. This provides a strict, rolling-window test of generalisation. Results are reported for January–June 2025, capturing a range of market conditions.

January 2025 (trained on October–December 2024)

Figure 357

Perfectly mimicked the buy and hold portfolio indicating the thresholds were to high to justify any sell actions.

February 2025 (trained on November 2024–January 2025)

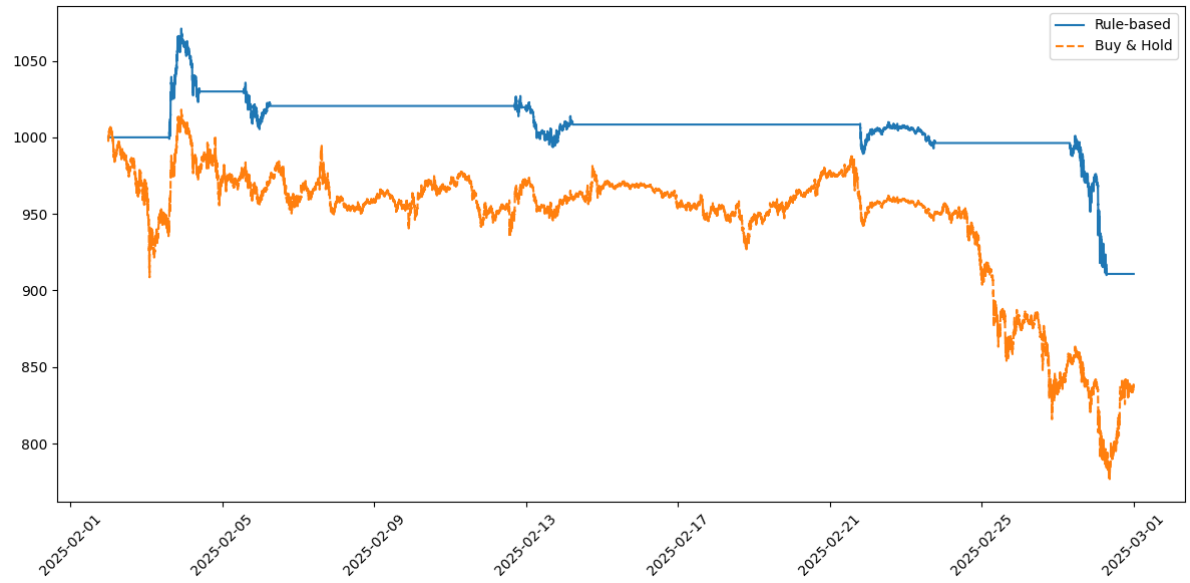


Figure 358

Accurately waited top enter the market until a good time, and then tried to hold cash to maintain value, however re-entered at a poor time near the end showing forecasts might not have captured the downwards trend.

March 2025 (trained on December 2024–February 2025)

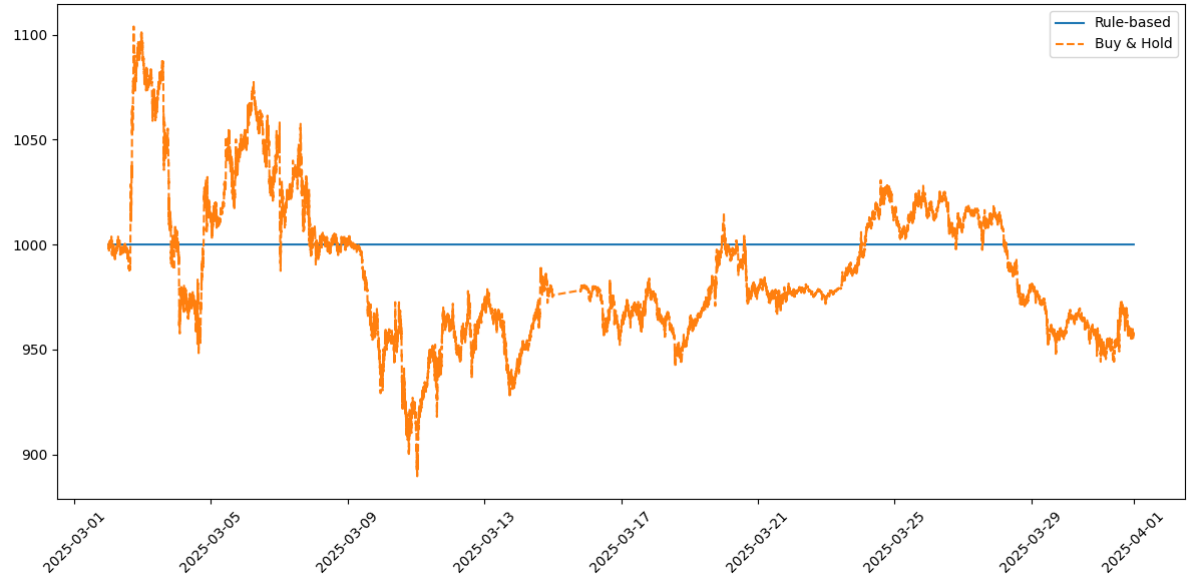


Figure 359

A flat line is indicative that the algorithm justified no buy or sell actions, on a month by month basis this causes for the portfolio to constantly hold its $1000 of cash, however on a continuous portfolio this would more likely be a constant hold of btc.

April 2025 (trained on January–March 2025)

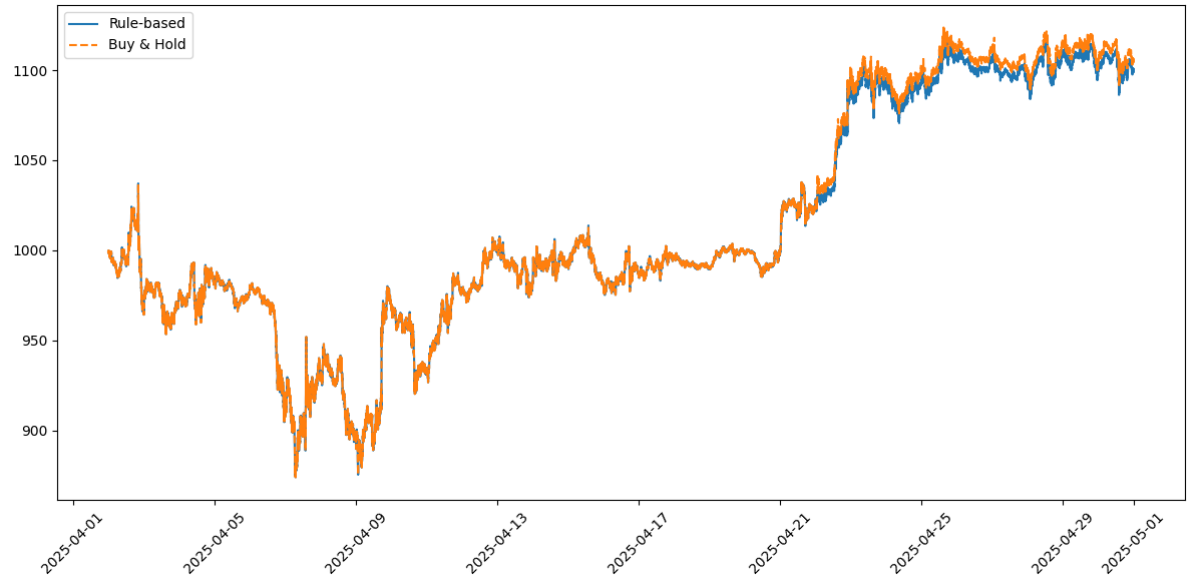


Figure 360

Aligned itself to the Buy/Hold metric indicating no trades for the most part. However at the point where the price of bitcoin rose sharply it missed the price rise by a small margin causing for it to lose a small amount of potential value.

May 2025 (trained on February–April 2025)

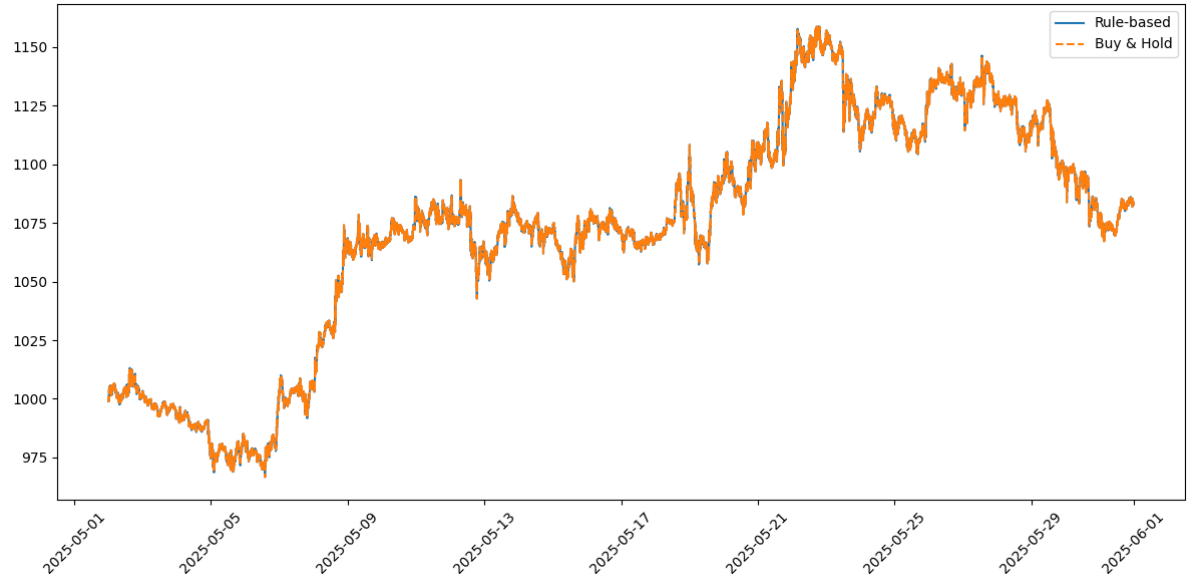


Figure 361

Perfectly mimicked the buy and hold portfolio indicating the thresholds were to high to justify any sell actions.

June 2025 (trained on March–May 2025)

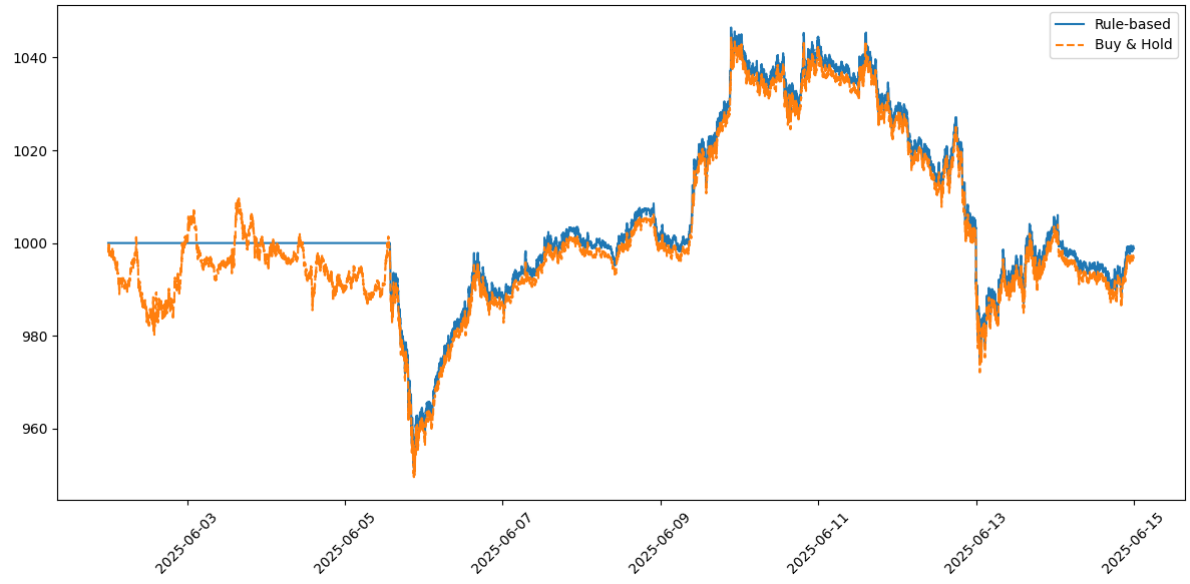


Figure 362

Held Through a period of instability and then entered just before after a drop in price making a small gain, however had it held for a short period longer the gain would've been far larger. This was most likely due to an issue with the buy / sell logic.

Summary of Standalone Results

Strengths

- Rarely made any minor losses
- Was very safe in its actions mimicking the buy / hold portfolio on most cases

Weaknesses

- Missed out on large points of potential profit due to its riosk averse nature.

Trading Performance – Transformer (standalone months)

Monthly Performance Results (Jan–Jun 2025)

January 2025 (trained on October–December 2024)

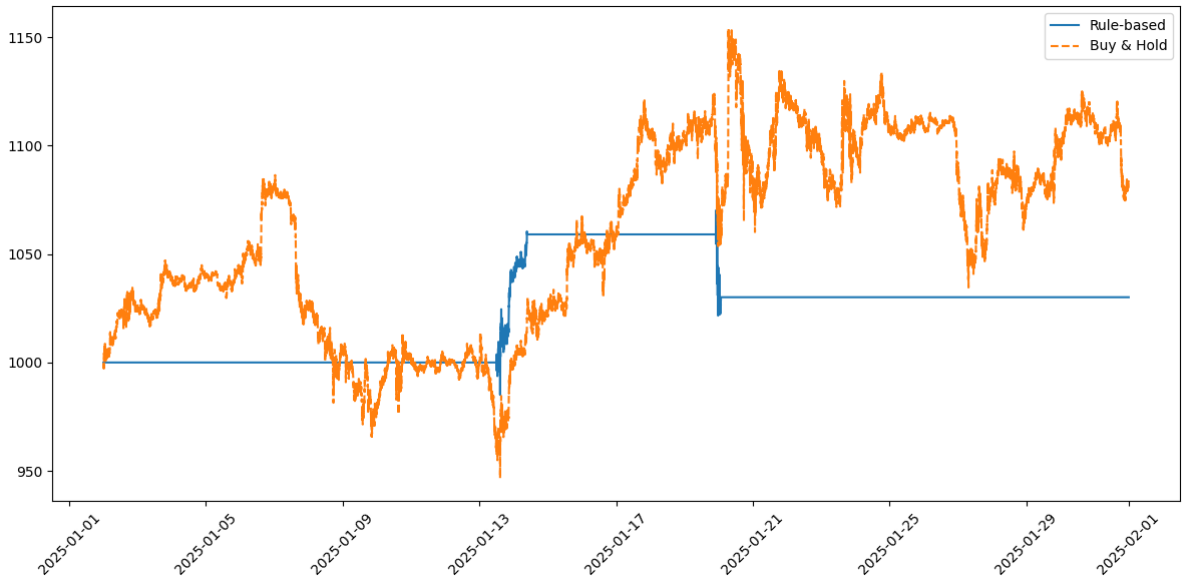


Figure 185

Hesitant to take any actions until it was sure of a good buy caused for it to make a gain initially, however it exited to early and the brought back at a poor time which significantly mitigated the potential gain it could've had. This is indicative of the algorithm not trusting the forecasts and so training was poor

February 2025 (trained on November 2024–January 2025)

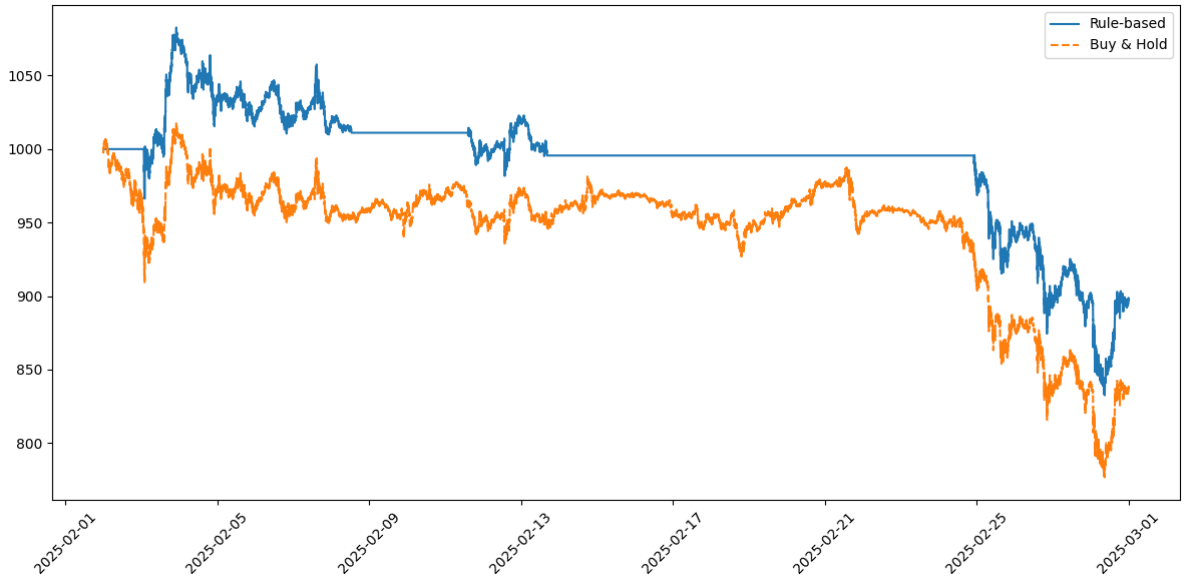


Figure 186

Nearly identical to the lstm in that it waited to enter until a good time, held as cash due to horizontal nature of btc value. However it re-entered at a poor time also indicating forecasts not predicting downwards trend.

March 2025 (trained on December 2024–February 2025).

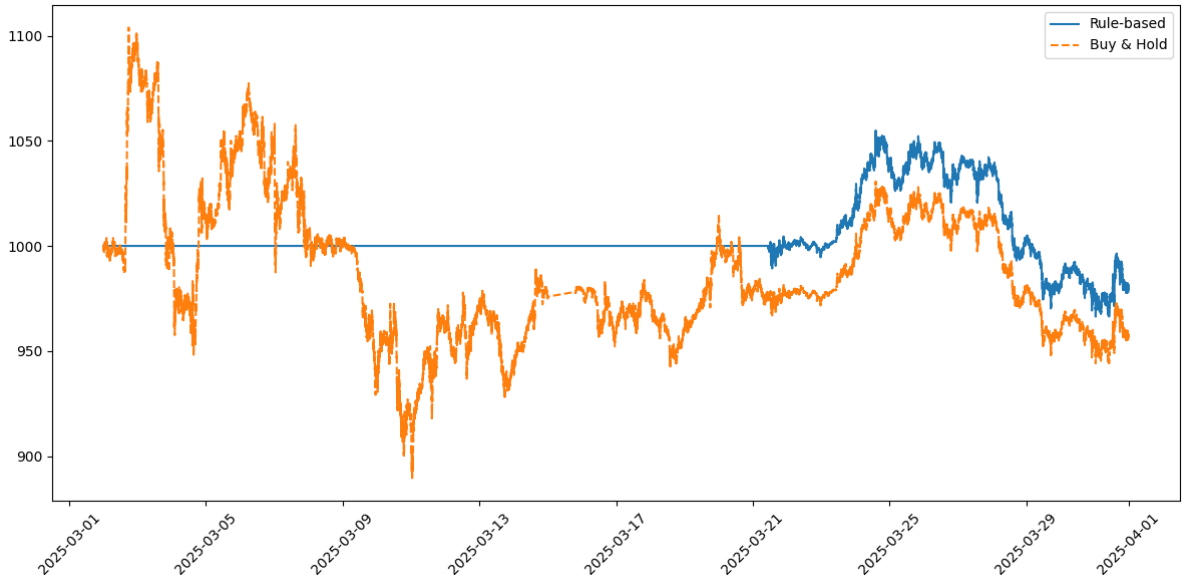


Figure 187

Held as cash through a period of instability showing forecasts might have been unstable. But correctly rendered at a point where it could capture value.

April 2025 (trained on January–March 2025)

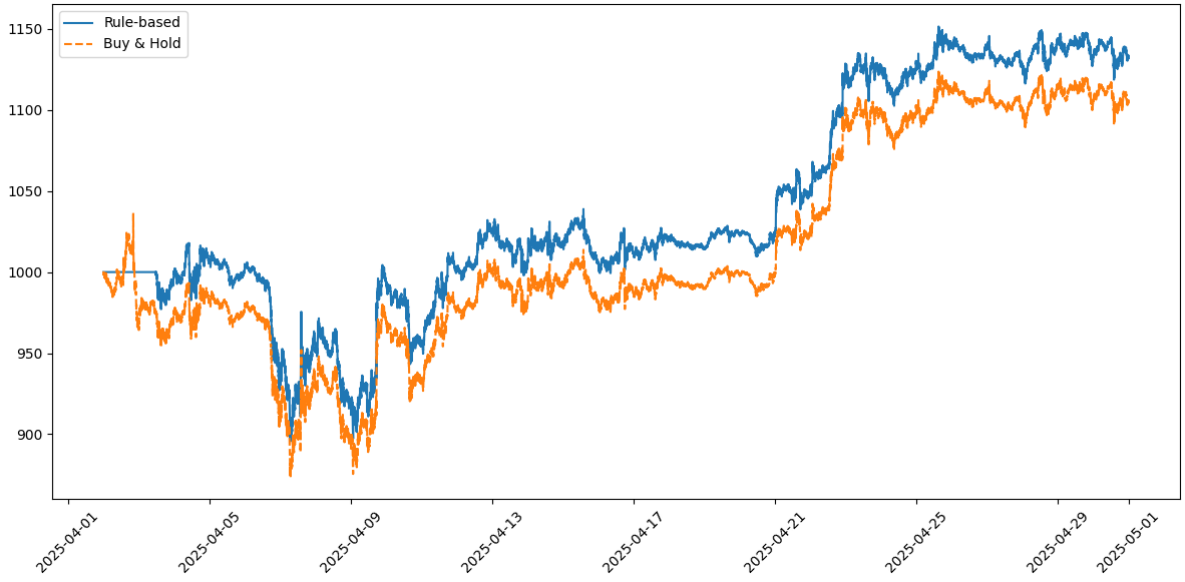


Figure 188

A good entry point caused for all the gain over the buy and hold model here, however it failed to forecast the large drop in price causing for large opportunity costs

May 2025 (trained on February–April 2025)

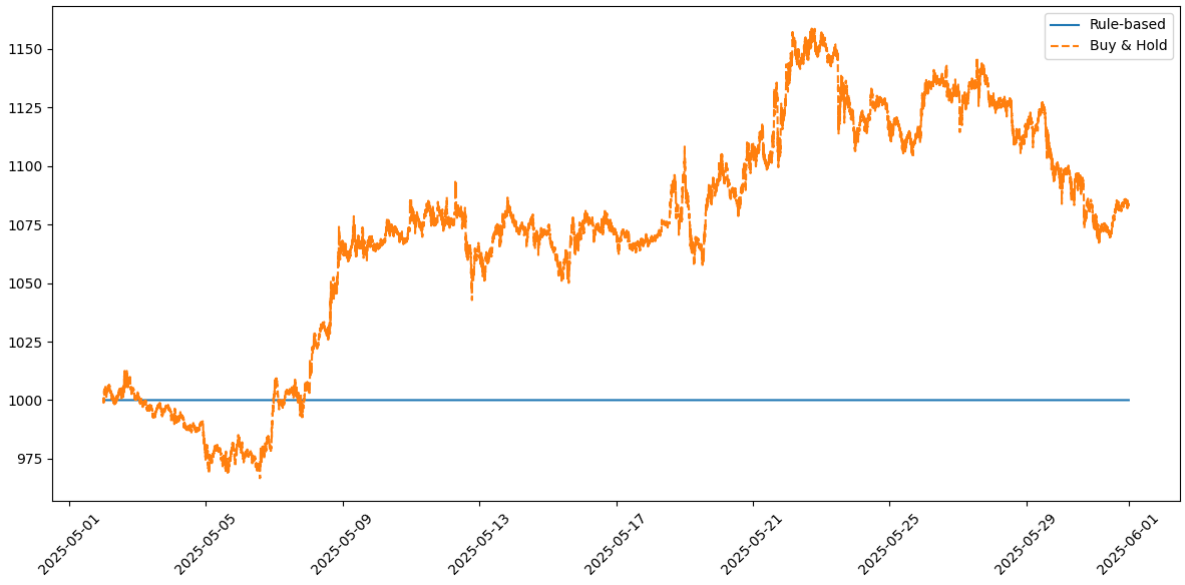


Figure 189

A flat line is indicative that the algorithm justified no buy or sell actions, on a month by month basis this causes for the portfolio to constantly hold its $1000 of cash, however on a continuous portfolio this would more likely be a constant hold of btc.

June 2025 (trained on March–May 2025)

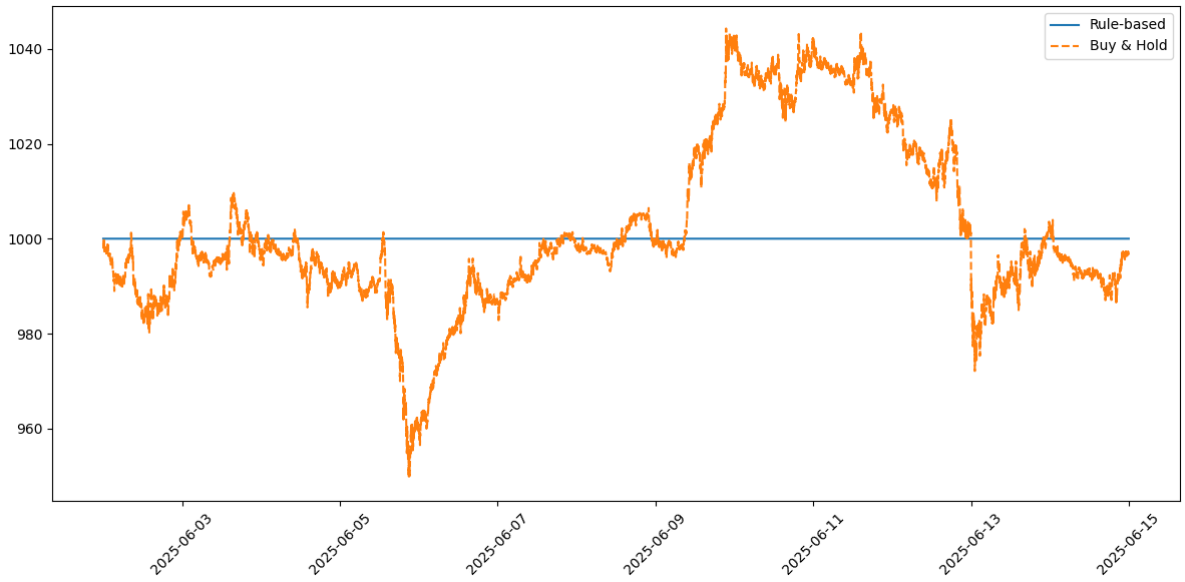


Figure 190

A flat line is indicative that the algorithm justified no buy or sell actions, on a month by month basis this causes for the portfolio to constantly hold its $1000 of cash, however on a continuous portfolio this would more likely be a constant hold of btc.

Strengths & Weaknesses

Strengths

- Correctly predicted where to enter to make a gain on multiple occasions

Weaknesses

- Failed to act and forecast large drops in price causing for large opportunity costs

Continuation Portfolio Performance (Jan–Jun 2025)

In addition to evaluating each month in isolation, a second layer of analysis tracked **continued portfolios** across the first half of 2025. Here, capital was initialised in February (January would've been ideal

however flaw in code only noticed at late point meant February start instead) and allowed to roll forward, with no resets between months. This structure highlights compounding effects, longer-term capital preservation, and the cumulative impact of both favourable and unfavourable months.

LSTM Continuation Portfolio

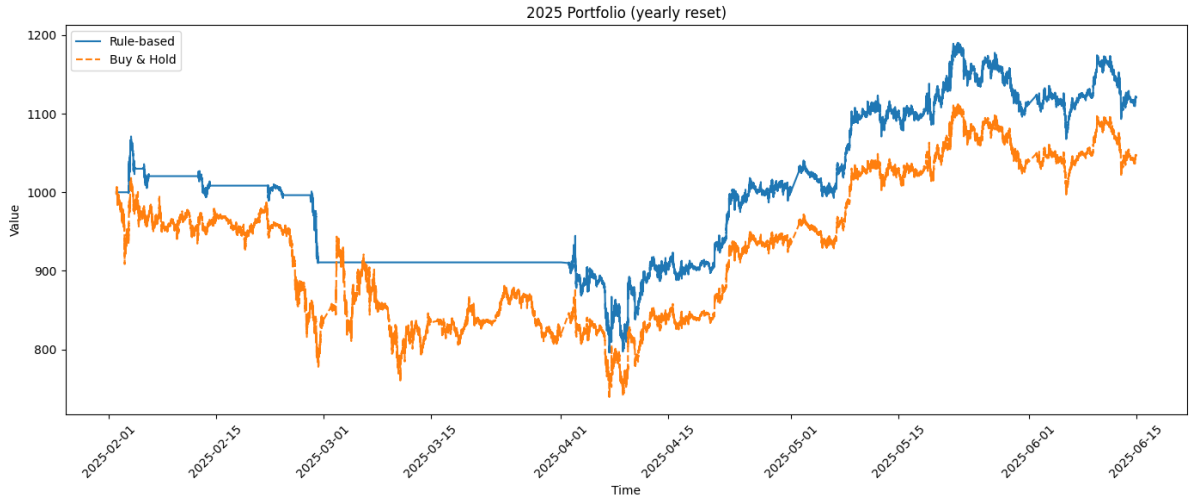


Figure 356

The strategy main an early gain in February with a good entry to the market, however it failed to forecast the large dip and entered at a poor time causing for a large loss of value. Due to potential instability in the forecasts and uncertainty it held as cash losing out on potential gain before re-entering at a gain over the buy and hold. A Gain which it held through inactivity till the end of the period.

Transformer Continuation Portfolio

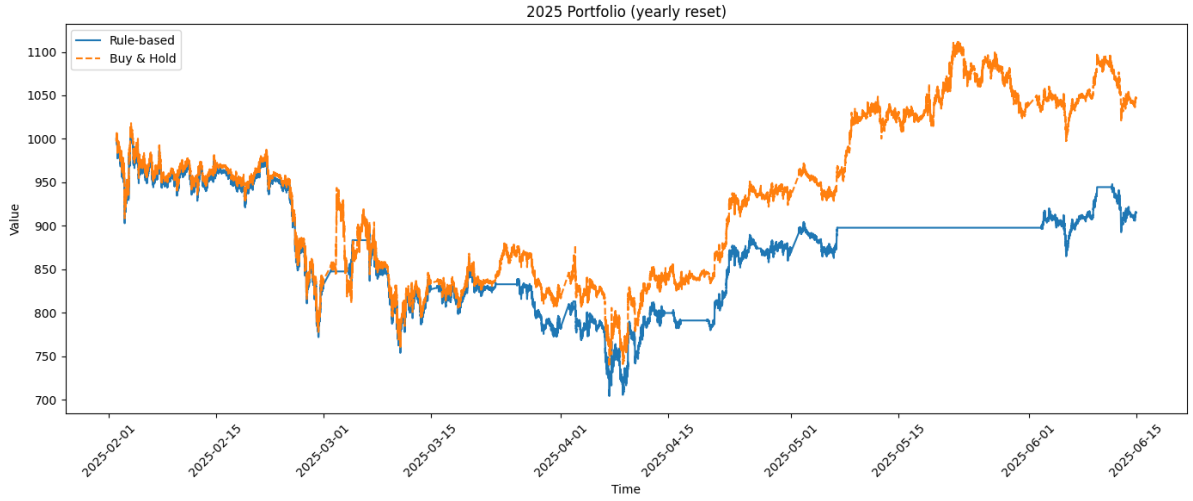


Figure 184

Unlike the LSTM the transformer was performed much worse. It entered the market poorly and continued to exit the market before time where the price recovered showing ill forecasting. It subsequently lost value compared to its stat point and performed much worse than the Buy / hold benchmark

Comparative Analysis of LSTM and Transformer Trading Performance

The algorithmic trading simulations revealed a consistent performance gap between the LSTM and Transformer approaches across both standalone months and continuation portfolios. While both models were subject to forecast biases that limited profitability, the LSTM demonstrated more practical utility in live trading scenarios, particularly in terms of risk control and capital preservation.

Standalone Months (January–June 2025)

The LSTM strategy had mixed but generally more stable performance. The protection strategy exhibited its strengths in February, when it successfully dampened capital losses during periods of market stress and outperformed Buy & Hold on a risk-adjusted basis. In the low signal or sideways market (April and May), it stayed closely with Buy & Hold without adding niggle volatility thus showing conservative bias. In fact, the LSTM would often manage to miss some upside potential in the months of Jan and Mar as it completely missed rallies and would often just do nothing or consequently get misaligned with the momentum on the upside quite mechanically.

The transformer, by contrast, always did worse in stand a lone months. Whipsawed trades, cost damage, and systematic loss were created by forecast instability. In April, there was little stabilisation. The general trend was persistent misaligned long position during downtrends (March) or premature exit in rallies (January, February, June). As a result, these strategies underperform, compared to Buy & Hold, and offer little evidence of generalisation under the rolling champion framework.

Continuation Portfolios (January–June 2025)

A divergence in models became apparent when results were compounded across months. The continuation LSTM portfolio produced slight cumulative outperformance, with its main benefit coming from successful drawdown protection in February. Between March and May, little value was added, but having the buffer that was still there meant that the LSTM equity curve was slightly in front of Buy & Hold by the end of June. As the LSTM failed to capture sustained rallies, this limited total return, but it did not lead to excessive downside risk. In stark contrast, the Transformer forward strip declined over time. The first few setbacks in the months of January and February were never recovered as systematic forecasting bias and high turnover in March and May led to further erosion of capital. The performance couldn't close the gap even in relatively calm April conditions. By the mid-year point, the Transformer portfolio had underperformed Buy & Hold and showed no sign of any compounding advantage. Thus, it does not qualify as a reliable trading backbone under the tested framework.

Synthesis

All in all, the comparative evidence reinforces that the LSTM was relatively better in high-frequency Bitcoin trading in this setup. Though it missed upside momentum, the LSTM provided decent capital preservation and acceptable risk exposure. Thus, LSTM looks like the more robust architecture of the two.

The reason the transformer was unsuccessful in practice though its theory says it can model long-range dependencies is due to forecast instability, weak short-horizon accuracy and systematic bias. Accordingly, its implementation of algorithmic trading generated losses instead of defensible risk-adjusted returns.

In short, the LSTM is more in line with the objectives of algorithmic trading, which include capital preservation, drawdown mitigation, and stability. On the other hand, the Transformer did not generalise well across months and underperformed against even the simplest baseline.

Outcomes (2025)

The analysis of outcomes is limited to the year 2025 due to market conditions. Over the life of bitcoin, there has been a change in volatility and liquidity structure. Further, testing models on recent data ensures that results are pertinent to current behaviour (Cheah & Fry, 2015; Corbet et al., 2019).

Forecasting Outcomes

The LSTM performed better than the Transformer, as seen in comparative forecasting results. An LSTM model performed significantly better than a Transformer model as the mean square error was around 14 times lower for the LSTM model than the transformer. Moreover, absolute metrics showed that a transformer's root mean square error was around 1.5 times as bad as the LSTM. At first glance, relative measures such as NRMSE and MAPE appeared to be similar in size (≈0.05 and 3.2%, respectively).Since these are normalised statistics, they don't capture the large difference in the accumulation of error at the lower level. According to Nelson et al. (2017) and Fischer & Krauss (2018), the LSTM's reduction of absolute error is very important in a high-frequency market where small absolute error can result in a large impact on trading.

The superiority was most prominent at short forecast horizons (1-30 minutes), which are predominantly responsible for intraday algorithmic trading profitability. The LSTM consistently captured quick price movements at a higher accuracy level than other models. In contrast, the study's limitations undermined the theoretical strength of the transformer in modelling long-range dependencies (Vaswani et al., 2017). The use of chunked training windows inhibited temporal coherence, while not using an autoregressive decoder meant the model did not condition its predictions on outputs generated sequentially.

The consequence was systematic bias as forecasts would overshoot/undershoot true trajectories giving rise to U-shaped error profiles or exaggerated oscillations, particularly pronounced in February as well as March and then also in May. The evidence is consistent with earlier evidence about similar findings with regard to forecasts concerning financial matters. Nelson et al. (2017) demonstrated that LSTMs work better than traditional machine learning techniques for stock price prediction by efficiently capturing local temporal dependency. According to Fischer and Krauss (2018), LSTMs outperformed random forests and logistic regression in selecting S&P 500 stocks. Translators are a valuable tool for the translation of texts, documents, etc. If you wish to have a French document translated into English, it might be easier for the translator to understand your French document. But the translator also needs to understand the subtle meanings of the text. This involves understanding the subtext, cadence, and many other things. Similarly, what is the humidity of the translator? The above-mentioned person needs to be the right translator for the translation. Let's go through some points to consider while hiring a translator for your academic texts. These results strengthen the notion that LSTMs are still very competitive in financial forecasting, particularly at short horizons, where fine temporal dynamics prevail.

Trading Outcomes

Forecasting quality translated directly into **differences in algorithmic trading performance**. The LSTM-driven strategy consistently demonstrated **superior cumulative performance** when evaluated both on a standalone monthly basis and under a continuation portfolio framework.

- in February 2025, there were significant drawdowns, however. The LSTM strategy did a decent job of preserving capital while avoiding the worst of the drawdown and in the April-May months. It very closely matched the Buy & Hold strategy. The missed upside opportunities in January and March proved to be a weakness as the forecasts were unable to account for sustained rallies and ended up being inactive for a longer period of time. However, the model was cautious in uncertain conditions, which ensured it did not suffer catastrophic losses.
- Continuation portfolio: LSTM maintained a small but significant lead over Buy & Hold by mid-year when returns were compounded over months. The principal benefit came from downside protection in February, a buffer for performance that carried forward. The strategy ceased to prosper in March–May, but by avoiding a severe pullback in June, it was able to add value again. In summary, the LSTM's continued equity curve showed through the growth of "risk-adjusted returns" rather than maximum exposure.

The Transformer strategy, by contrast, struggled to convert forecasts into profitable trading outcomes.

- Standalone months: It showed poor performance against Buy & Hold due to forecast bias and instability. The months of January and February reveal limited effectiveness for premature exits and whipsaw trades, while March and June suggest excessive lock-ins to losing trades due to over- or under-prediction. Even though the markets were quiet in April, performance was still negative as systematic optimism left it out of line with what was happening.
- Compounding structure increased continuation portfolio weaknesses. Due to underperformance in January–February, a deficit was created, which was never recovered. In the following months, the situation did not improve, with May being marked by instability and June seeing a significant underprediction, which led to a further loss of capital. By the middle of 2025, the Transformer portfolio was significantly below both LSTM and Buy & Hold, suggesting no compounding advantage and not demonstrating the theoretical strengths of long-range modelling in practice.

The claim is that similar results come about from broader research work in algorithmic trading, that high likelihood of downside protection will result in successful strategy, as will consistent short-horizon alignments over noisier long-horizon alignments. The LSTM worked robustly and adapted easily even if it missed some opportunities, and the Transformer's instability made it unsuitable for a high-frequency environment.

Synthesis

Overall, the results demonstrate that in 2025 the LSTM offered better forecasting accuracy and trading performance than the benchmark in high-frequency Bitcoin markets. It reduced raw forecast error, adapted to changing regimes and preserved capital under drawdowns while its practical advantages manifested when deployed in a trading context. Although the Transformer architecture excels in many situations, the Transformer does not generalise under the walk-forward structure, causing systematic biases and cumulative underperformance.

The difference here shows the more general principle in financial machine learning that in high frequency, accuracy, and flexibility at short horizons will offer more value in practice than stable conservativeness without participation. The LSTM accomplished this by displaying strong control over errors and providing low but consistent trading gains while the economic Utility of the Transformer deteriorated as a result of a defensive but inactive posture.

What's next

The current study set a comparative framework for forecasting based on LSTM and Transformers and algorithmic trading of Bitcoin. These results support LSTM superiority for short-horizon forecasting and trading adaptability. Nevertheless, there exists many interesting opportunities for extension. Future work can be organized around five mutually-reinforcing themes: model expansion, value innovation, enhanced evaluation, real-world application, and operational enhancement.

Model Expansion

To strengthen the empirical foundation, the evaluation should be broadened to encompass a wider set of forecasting architectures. In particular:

- **Advanced sequence models:** Temporal Convolutional Networks (Bai et al., 2018), Temporal Fusion Transformers (Lim et al., 2021), and hybrid CNN–LSTM designs (Borovykh et al., 2017) offer richer ways of capturing local and global dependencies in high-frequency financial data.
- **Classical statistical baselines:** Including ARIMA, exponential smoothing, and Prophet (Hyndman & Athanasopoulos, 2021) would benchmark deep learning results against established time series tools, providing a clearer picture of both accuracy and stability advantages.
- **Hybrid ensembles:** Combining statistical and neural approaches may help balance robustness and adaptability across different market conditions.

Trading Strategy Innovation

Beyond forecasting, the trading framework itself should evolve toward more adaptive and autonomous mechanisms:

- Reinforcement learning (RL): Approaches like policy-gradient actor–critic algorithms (Deng et al., 2016; Spooner et al., 2018) can directly optimise buy/sell policies by maximising cumulative returns and avoiding predefined rules.
- Fundamental design changes would allow a portfolio to adjust its exposure more flexibly; for example, by introducing position sizing, dynamic leverage and stop-loss structures (Chan, 2013).

- Expanding the framework to BTC–altcoin pairs or broader cryptocurrency baskets could allow us to exploit cross-asset correlations to generate more profitable signals.

Extended Trading Evaluation

Evaluation should extend beyond the single-year window.

- Longer horizon testing ensures survivorship and reduces regime-dependence.

- We can put the models and strategies under stress-testing of the market regime by simulating, bull, bear and sideways environment.
- Resilience to extreme events: Events, such as the 2020 crash of the BTC, or the 2021 liquidity squeezes form natural laboratories for testing model fragility and stress behaviour.

Real-World Integration

Moving from simulation toward live deployment represents the ultimate challenge:

- **Execution layer:** Connecting forecasts to exchange APIs (e.g., Binance, Coinbase Pro) allows real-time order placement, where latency and slippage must be explicitly modelled.
- **Microstructural frictions:** Spread, order-book depth, and transaction costs must be integrated into evaluation to bridge the gap between theoretical and realised returns.
- **Risk controls:** Practical implementations require automatic halts under excessive drawdown, trade throttling, and capital preservation mechanisms to guard against tail-risk failures.
- **Live vs simulated decay:** Comparing paper-traded performance against live executions would quantify the real-world degradation of algorithmic models.

Operational Improvements

Finally, several pipeline-level refinements could improve efficiency and adaptivity:

- Changing preprocessing: Changing input window lengths and/or applying temporal pooling, noise rearrangement with data loss balancing.
- Automatically searching for hyperparameters: Techniques such as Bayesian optimisation and population-based training (Jaderberg et al., 2017) could hasten convergence and find more resilient sets of parameters.
- Regular feedback loops with automatic retraining based on newly available data will prevent model drift and sustain performance.

Ultimate Goal

The ultimate goal is to advance from controlled backtests into a production-level live crypto trading system. A system incorporating state-of-the-art forecasting models, adaptive trading policies, robust risk management and continuous self-tuning. The focus should be on “profitability”, “robustness under uncertainty”, and “safety in practice” rather than static accuracy measures.

References And Acknowledgments

I would like to thank my project supervisor for his assistance and guidance throughout this project. He helped bridge the gap between what is expected in an academic setting such as what I'm used to, and what is expected of a professional job setting. And as such i have walked away from this with knowledge and experience on how to-do similar projects professionally.

Thank you Doug.

- **Core Cryptocurrency & Financial Market Studies**
- Cheah, E.-T., & Fry, J. (2015). Speculative bubbles in Bitcoin markets? *Economics Letters, 130*, 32–36.
- Corbet, S., Lucey, B., Urquhart, A., & Yarovaya, L. (2019). Cryptocurrencies as a financial asset: A systematic analysis. *International Review of Financial Analysis, 62*, 182–199.
- Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies, 33*(5), 2223–2273.
- Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research, 259*(2), 689–702.
- Nelson, D. M. Q., Pereira, A. C. M., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 1419–1426.

Deep Learning & Forecasting Methods

- Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.
- Borovykh, A., Bohte, S., & Oosterlee, C. W. (2017). Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT*, 4171–4186.
- Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research, 270*(2), 654–669.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.
- Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting, 37*(4), 1748–1764.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*, 5998–6008.
- Zhang, Y., Liu, Y., & He, X. (2023). Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.

Statistical & Econometric Foundations

- Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2000). Great realizations. *Risk, 13*(3), 105–108.
- Bailey, D. H., Borwein, J., de Prado, M. L., & Zhu, Q. J. (2014). Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance. *Notices of the American Mathematical Society, 61*(5), 458–471.
- Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in model evaluation. *Geoscientific Model Development, 7*(3), 1247–1250.
- Cont, R. (2001). Empirical properties of asset returns: Stylized facts and statistical issues. *Quantitative Finance, 1*(2), 223–236.
- Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice* (3rd ed.). OTexts.
- Lo, A. W. (2002). The statistics of Sharpe ratios. *Financial Analysts Journal, 58*(4), 36–52.
- Magdon-Ismail, M., Atiya, A. F., Pratap, A., & Abu-Mostafa, Y. S. (2004). On the maximum drawdown of a Brownian motion. *Journal of Applied Probability, 41*(1), 147–161.
- Neely, C. J., Rapach, D. E., Tu, J., & Zhou, G. (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science, 60*(7), 1772–1791.
- Tsay, R. S. (2010). *Analysis of financial time series* (3rd ed.). John Wiley & Sons.

Machine Learning Optimisation & Techniques

- Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research, 13*, 281–305.
- Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 1050–1059.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 448–456.
- Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., & Kavukcuoglu, K. (2017). Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., ... & Le, Q. V. (2018). Mixed precision training. *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Zhang, X., Zhou, Y., & Lin, W. (2020). A deep recurrent neural network for financial time-series prediction. *Procedia Computer Science, 177*, 280–285.

Algorithmic Trading & Reinforcement Learning

- Achelis, S. B. (2013). *Technical analysis from A to Z* (2nd ed.). McGraw-Hill.
- Chan, E. P. (2013). *Algorithmic trading: Winning strategies and their rationale*. John Wiley & Sons.

- Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2016). Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems, 28*(3), 653–664.
- Pardo, R. (2008). *The evaluation and optimization of trading strategies* (2nd ed.). John Wiley & Sons.
- Spooner, T., Fearnley, J., Savani, R., & Koutroumpis, P. (2018). Market making via reinforcement learning. *arXiv preprint arXiv:1804.04216*.
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.

Datasets & Market Data Sources

- McZielinski. (2025). *Bitcoin historical data (minute-level, 2012–2025)* [Data set]. Kaggle. https://www.kaggle.com/datasets/mczielinski/bitcoin-historical-data
- Binance Data Collection. (2025). *Historical Bitcoin market data*. Retrieved from https://data.binance.vision/
- Blockchain.com Explorer. (2025). *Market price (USD) chart*. Retrieved from https://www.blockchain.com/explorer/charts/market-price

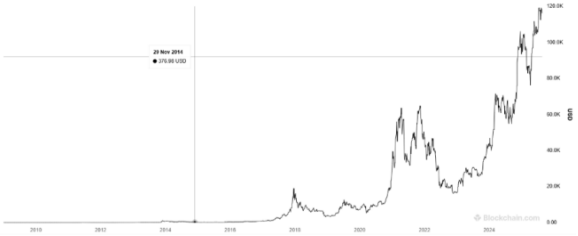
Appendices

Note – The code was optimised for 2025 conditions, all years have been included however trading may not be optimal and as such to not take as full reflection of effectiveness.

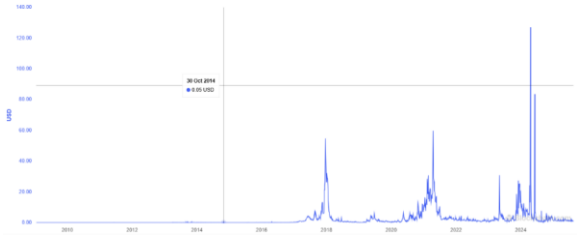
Portfolio Equity Graphs – transformer

2012

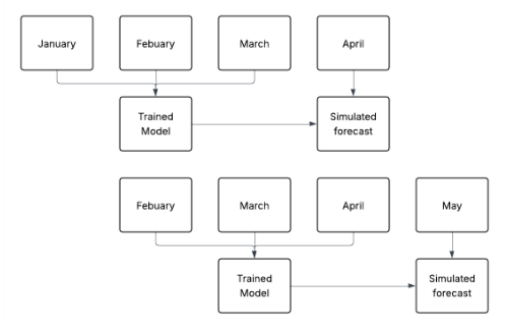
1) Minute by minute bitcoin prices



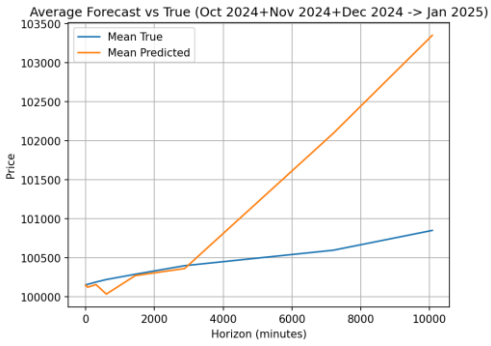
2) Average daily transaction costs USD



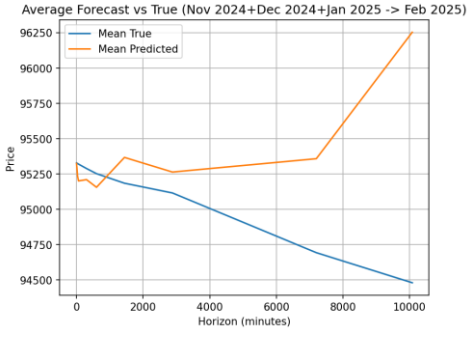
3) 3-1 rolling split



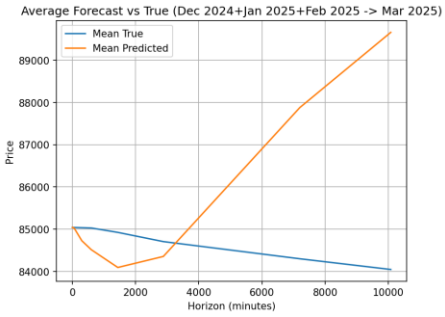
4)



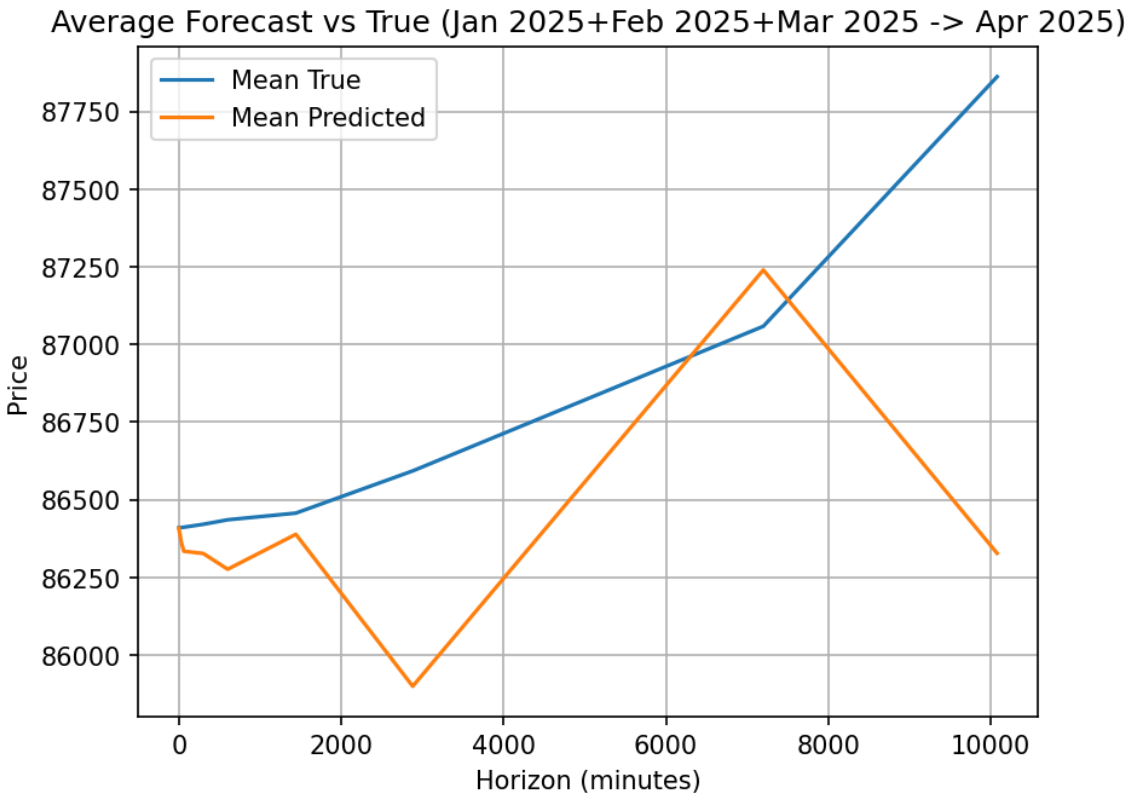
5)

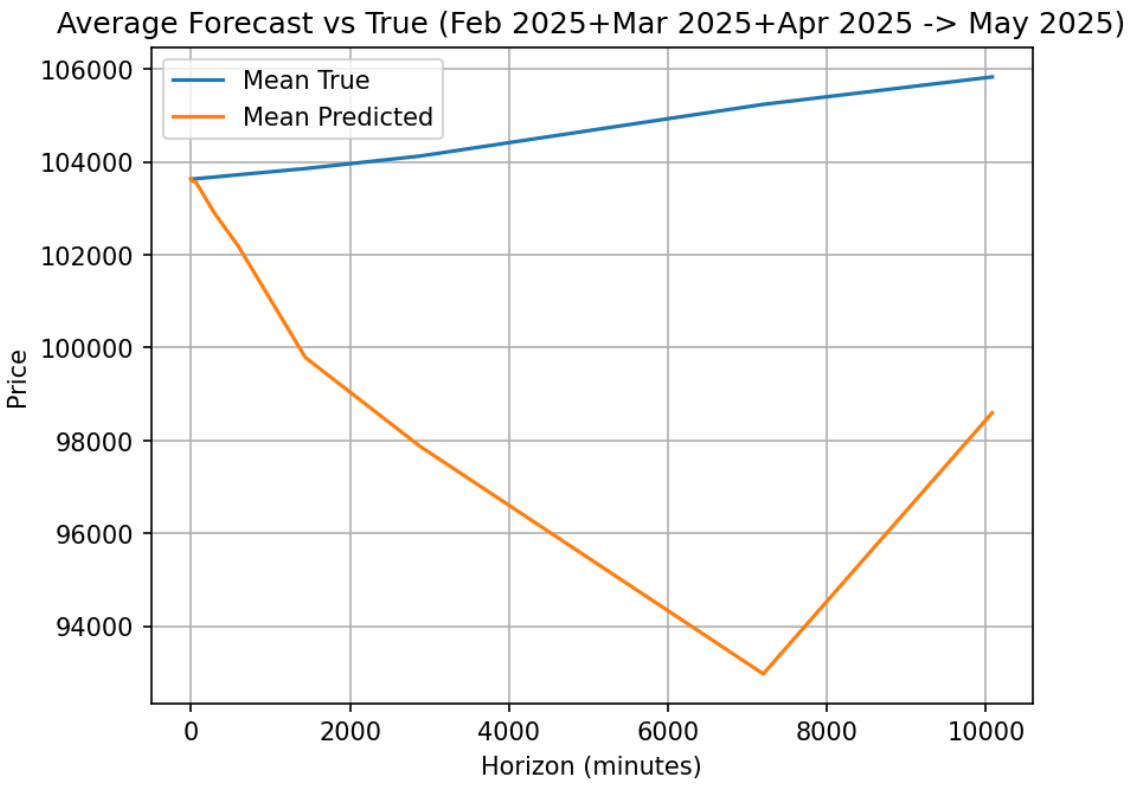


6)

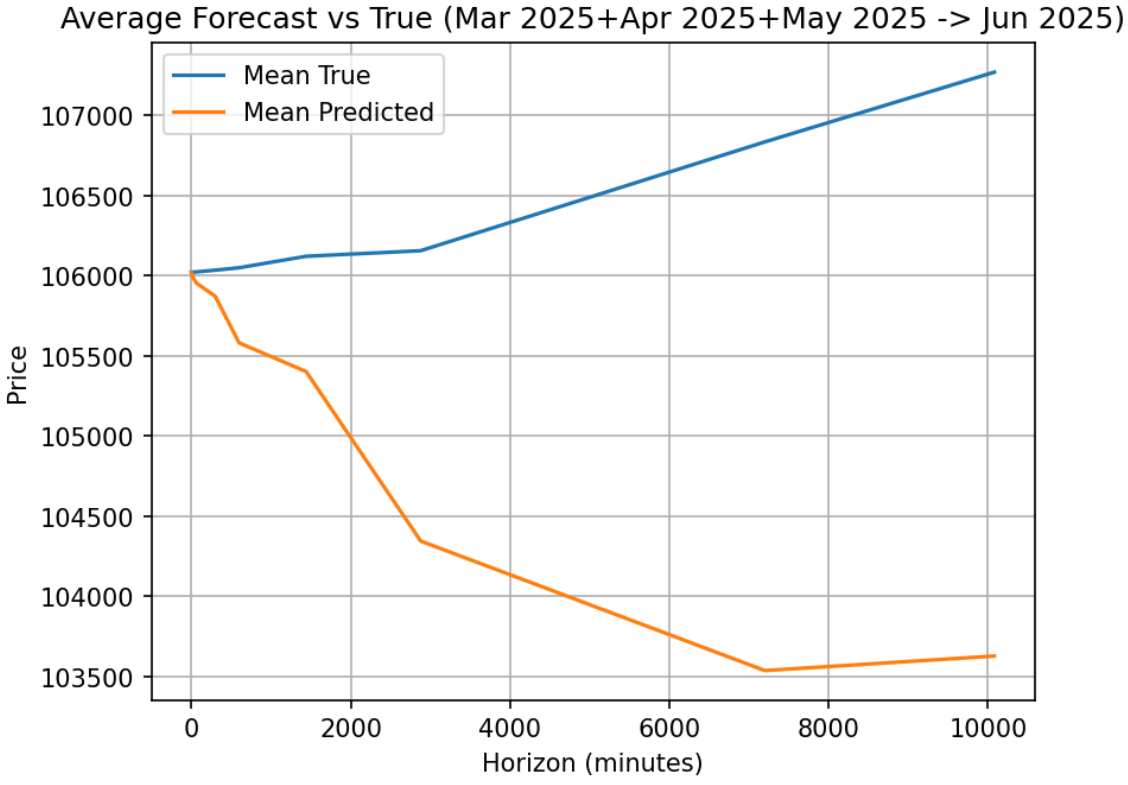

Average Forecast vs True (Dec 2024+Jan 2025+Feb 2025 -> Mar 2025)
Mean True
Mean Predicted
Price
89000
88000
87000
86000
85000
84000
0
2000
4000
6000
8000
10000
Horizon (minutes)

7)

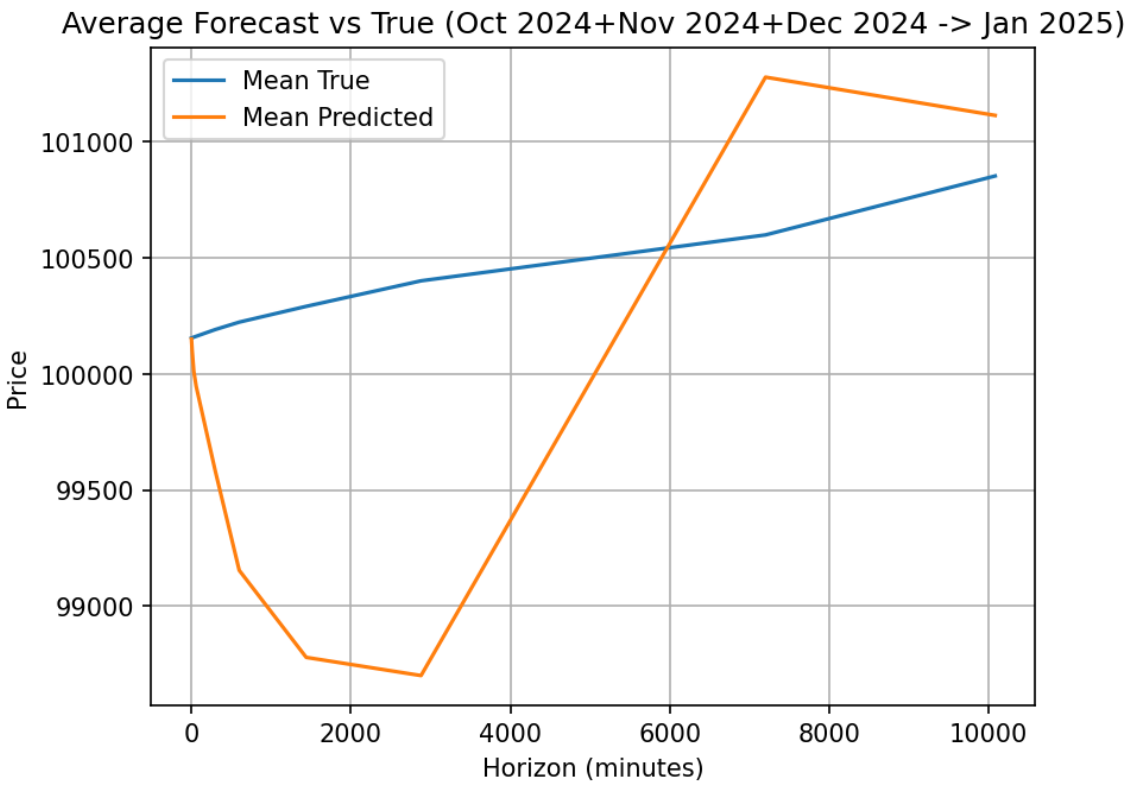

Average Forecast vs True (Jan 2025+Feb 2025+Mar 2025 -> Apr 2025)
Mean True
Mean Predicted
Price
87750
87500
87250
87000
86750
86500
86250
86000
0
2000
4000
6000
8000
10000
Horizon (minutes)


Average Forecast vs True (Feb 2025+Mar 2025+Apr 2025 -> May 2025)
Mean True
Mean Predicted
Price
Horizon (minutes)
106000
104000
102000
100000
98000
96000
94000
0
2000
4000
6000
8000
10000

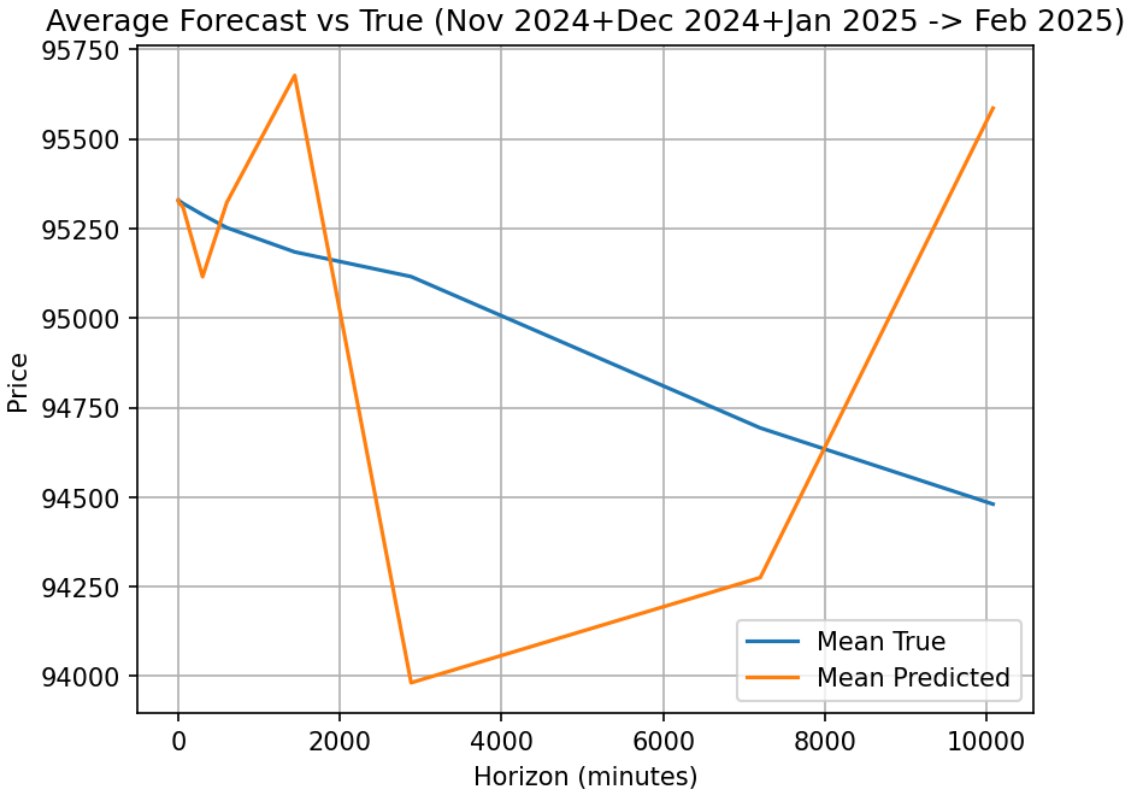
8)


Average Forecast vs True (Mar 2025+Apr 2025+May 2025 -> Jun 2025)
Mean True
Mean Predicted
Price
Horizon (minutes)
107000
106500
106000
105500
105000
104500
104000
103500
0
2000
4000
6000
8000
10000

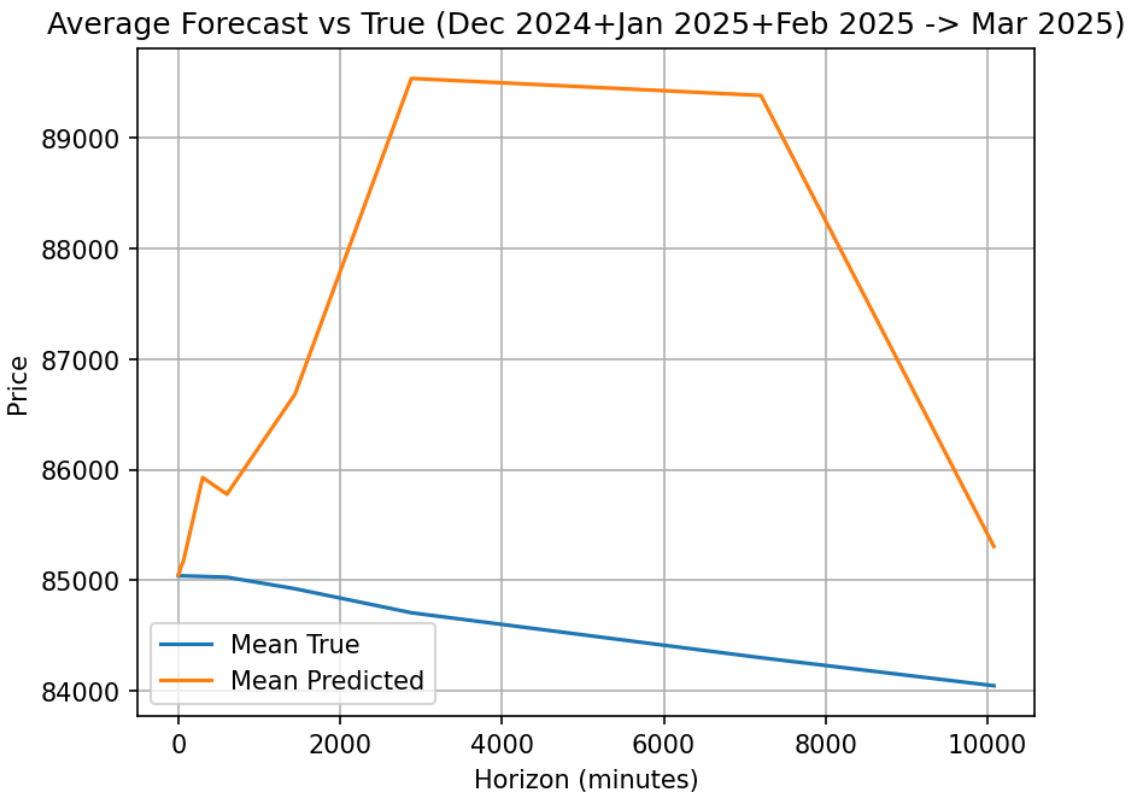
9)


Average Forecast vs True (Oct 2024+Nov 2024+Dec 2024 -> Jan 2025)
Mean True
Mean Predicted
101000
100500
100000
99500
99000
Price
0
2000
4000
6000
8000
10000
Horizon (minutes)

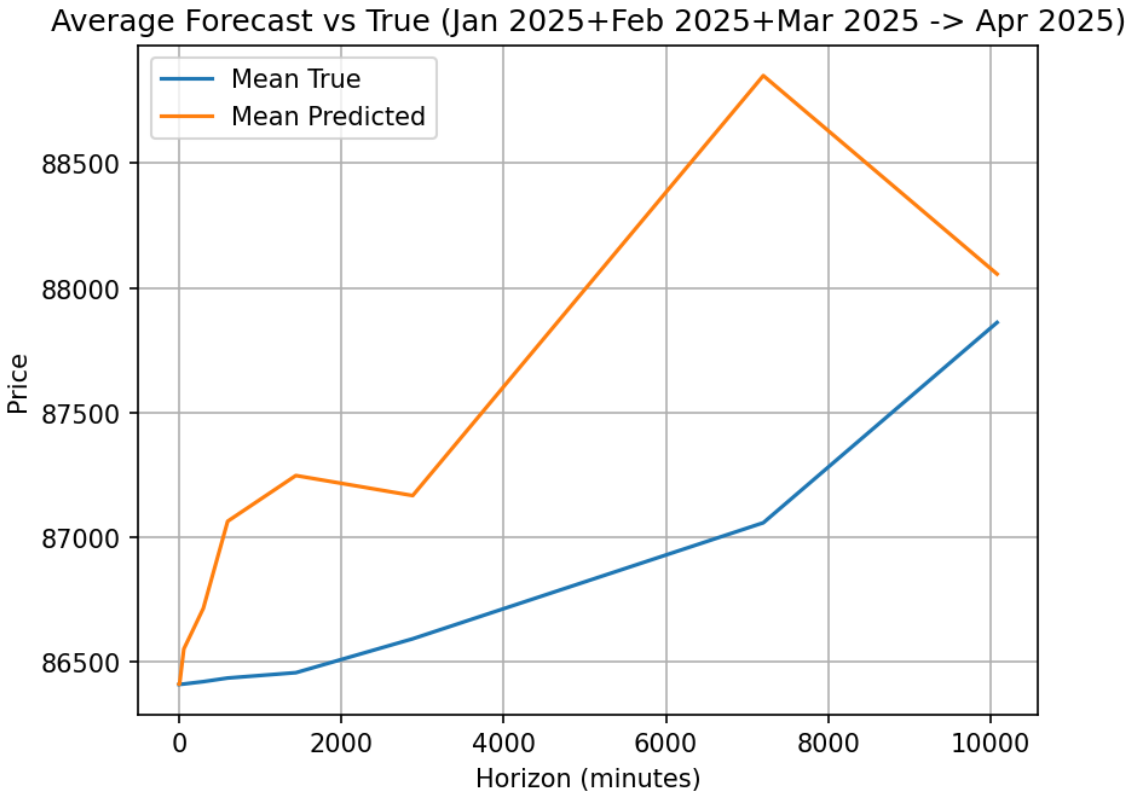
10)


Average Forecast vs True (Nov 2024+Dec 2024+Jan 2025 -> Feb 2025)
95750
95500
95250
95000
94750
94500
94250
94000
Price
Mean True
Mean Predicted
0
2000
4000
6000
8000
10000
Horizon (minutes)

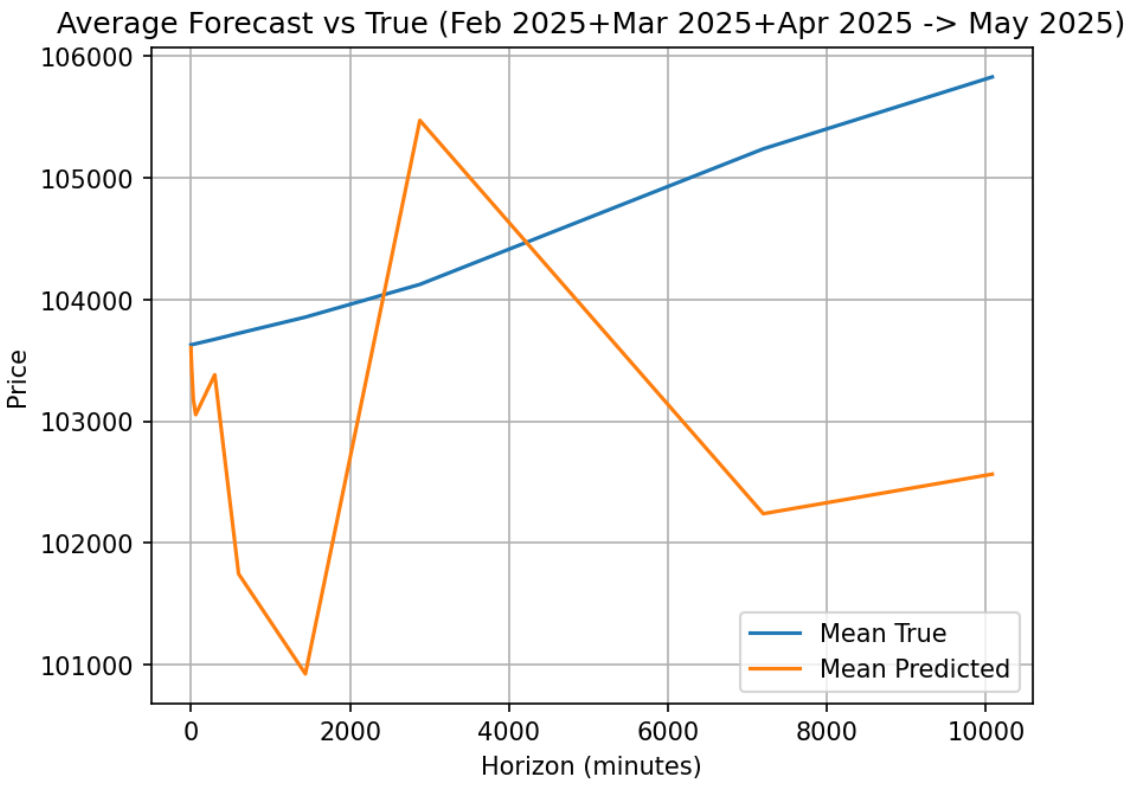
11)


Average Forecast vs True (Dec 2024+Jan 2025+Feb 2025 -> Mar 2025)
Price
Horizon (minutes)
Mean True
Mean Predicted
89000
88000
87000
86000
85000
84000
0
2000
4000
6000
8000
10000

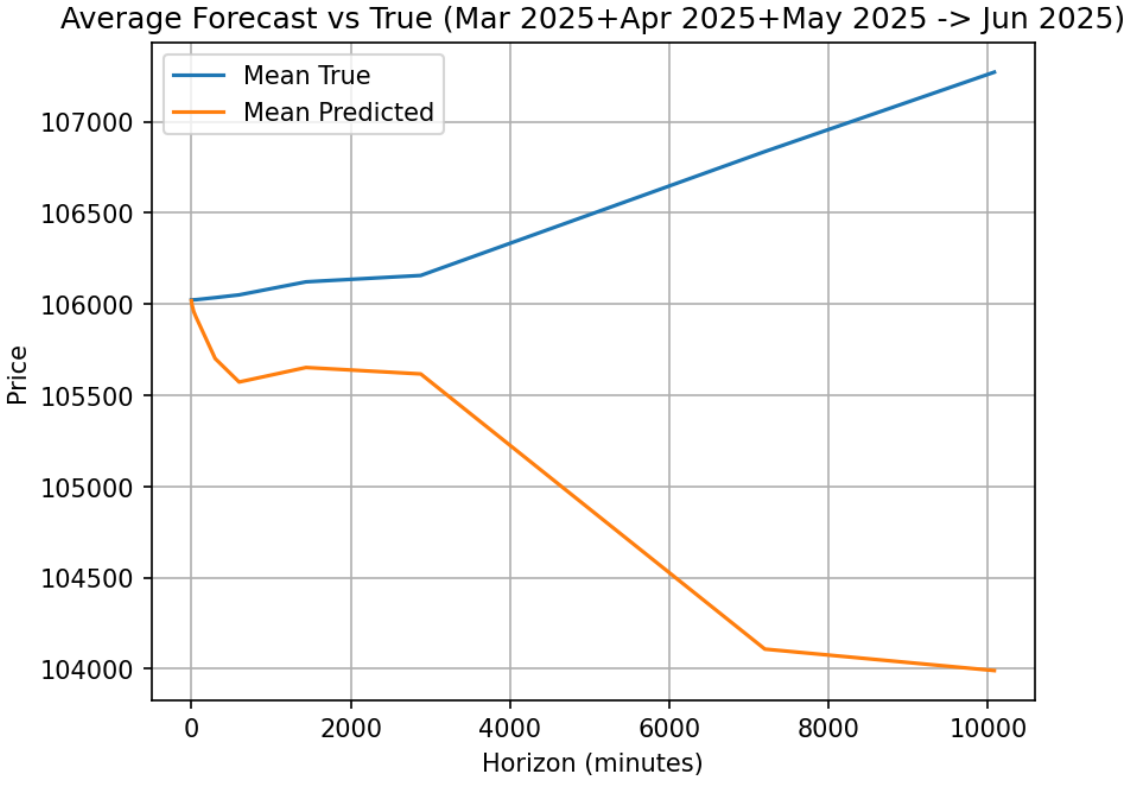
12)


Average Forecast vs True (Jan 2025+Feb 2025+Mar 2025 -> Apr 2025)
Price
Horizon (minutes)
Mean True
Mean Predicted
88500
88000
87500
87000
86500
0
2000
4000
6000
8000
10000

13)

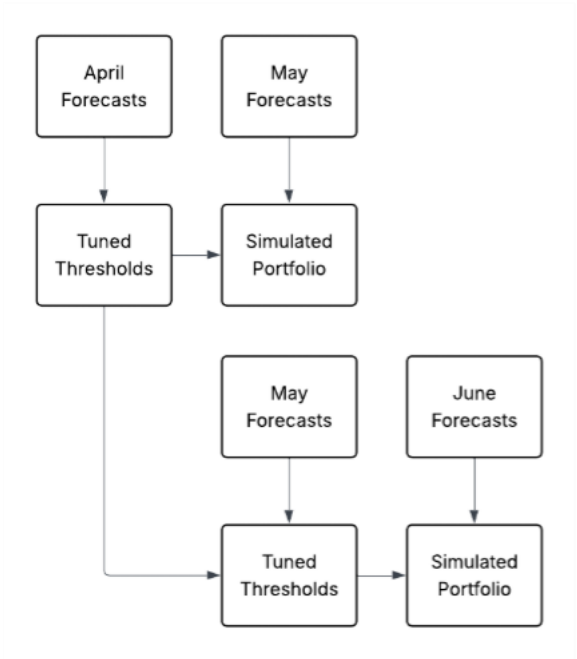

Average Forecast vs True (Feb 2025+Mar 2025+Apr 2025 -> May 2025)
Price
Horizon (minutes)
Mean True
Mean Predicted
106000
105000
104000
103000
102000
101000
0
2000
4000
6000
8000
10000

14)

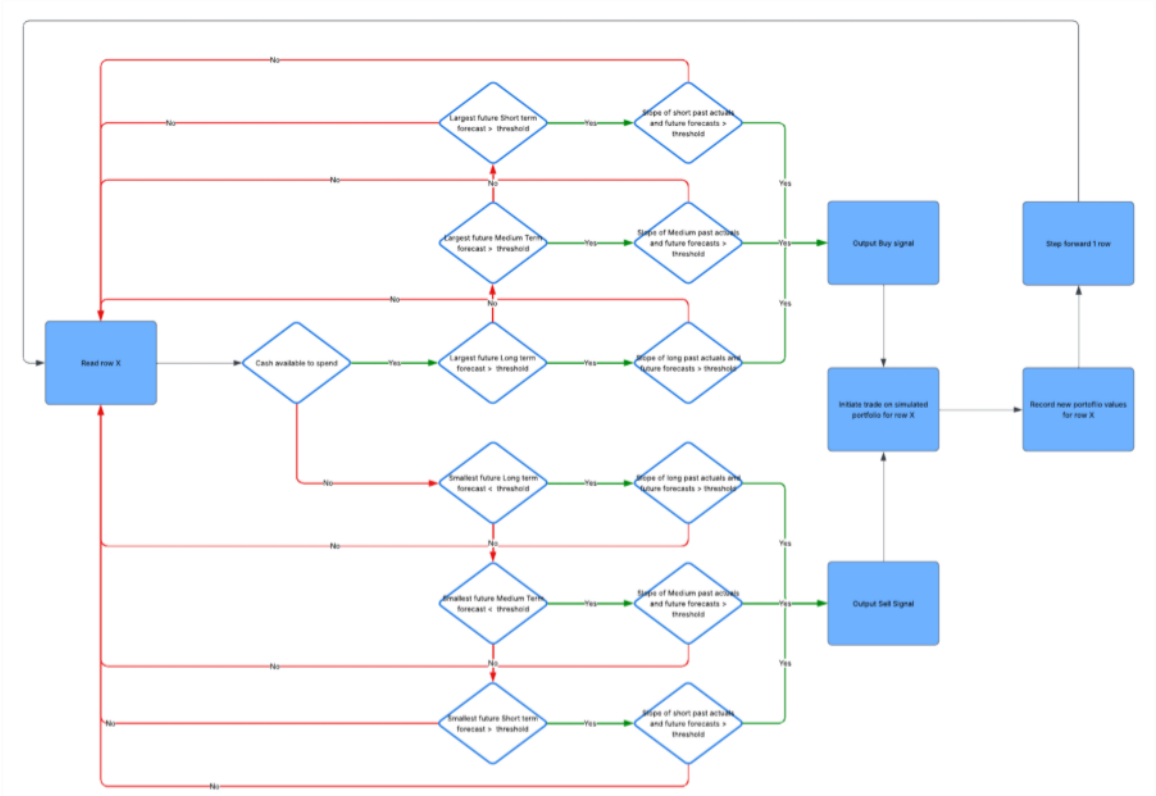

Average Forecast vs True (Mar 2025+Apr 2025+May 2025 -> Jun 2025)
Price
Horizon (minutes)
Mean True
Mean Predicted
107000
106500
106000
105500
105000
104500
104000
0
2000
4000
6000
8000
10000

15)

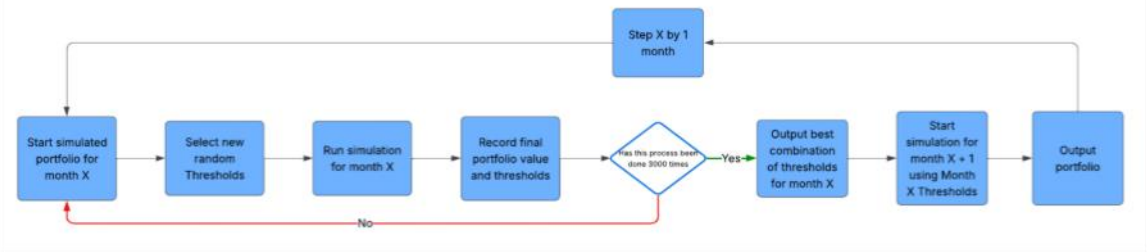
16)


April Forecasts
May Forecasts
Tuned Thresholds
Simulated Portfolio
May Forecasts
June Forecasts
Tuned Thresholds
Simulated Portfolio

17)


Read row X
Cash available to spend
Largest future Short term forecast > threshold
Largest future Medium Term forecast > threshold
Largest future Long term forecast > threshold
Smallest future Long term forecast < threshold
Smallest future Medium Term forecast < threshold
Smallest future Short term forecast > threshold
Slope of short past actuals and future forecasts > threshold
Slope of Medium past actuals and future forecasts > threshold
Slope of long past actuals and future forecasts > threshold
Output Buy signal
Output Sell Signal
Initiate trade on simulated portfolio for row X
Record new portoflio values for row X
Step forward 1 row
No
Yes

18)

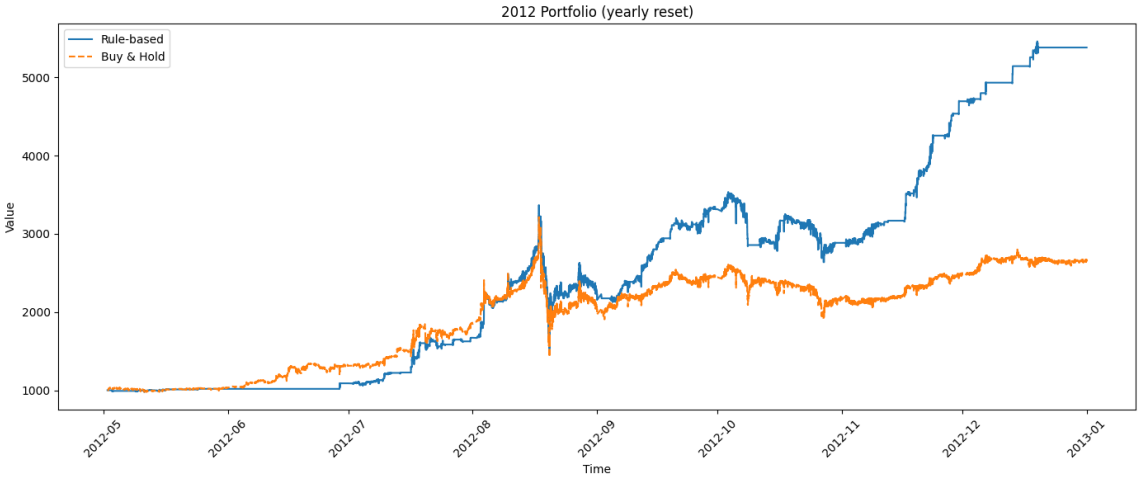

Start simulated portfolio for month X
Select new random Thresholds
Run simulation for month X
Record final portfolio value and thresholds
Was this process been done 3000 times
Yes
No
Output best combination of thresholds for month X
Start simulation for month X + 1 using Month X Thresholds
Output portfolio
Step X by 1 month

Note – The code was optimised for 2025 conditions, all years have been included however trading may not be optimal and as such to not take as full reflection of effectiveness.

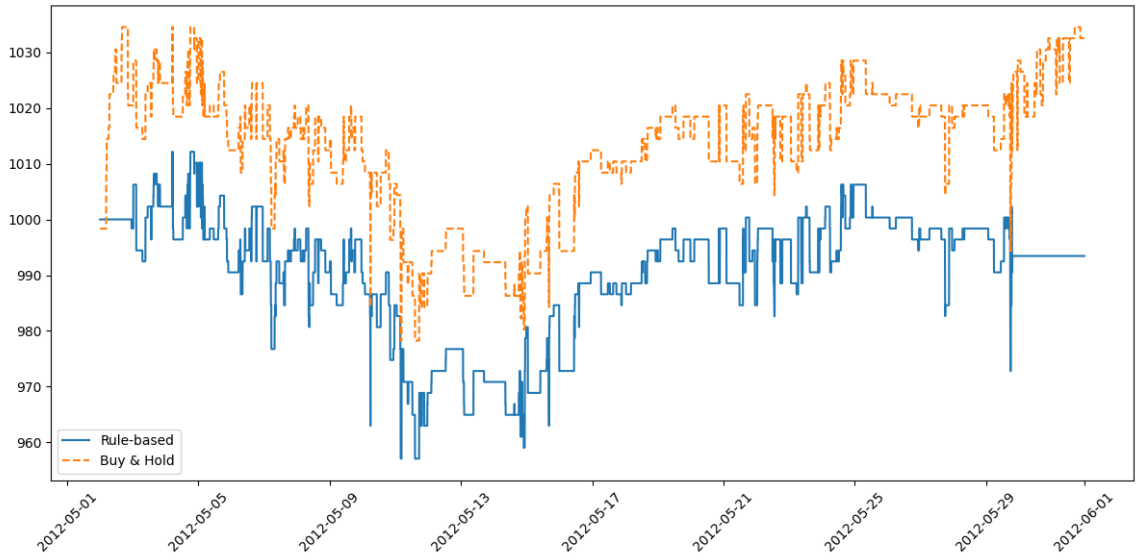
Portfolio Equity Graphs – transformer

2012

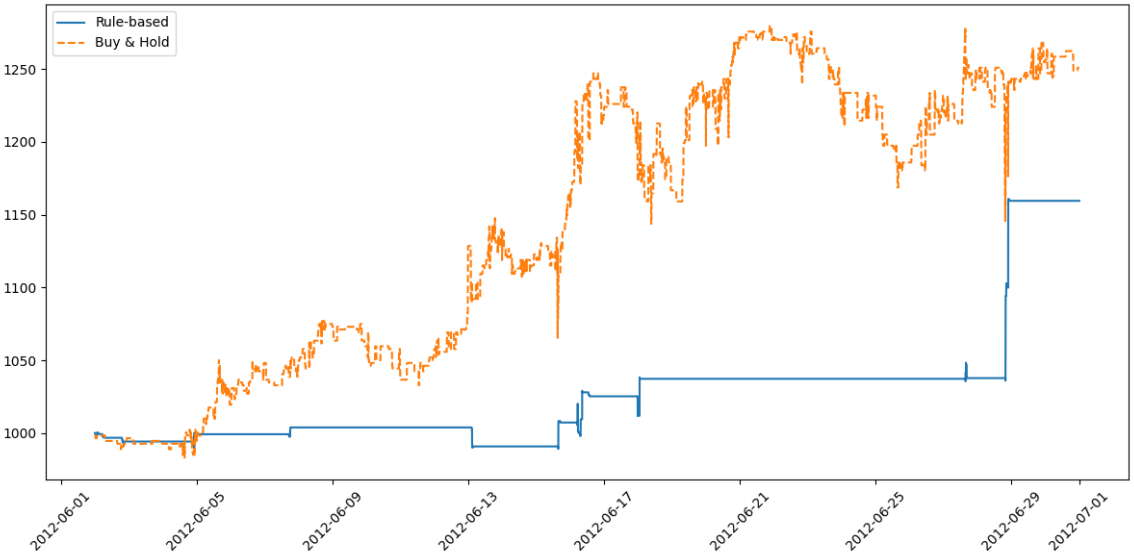
19) Full Year



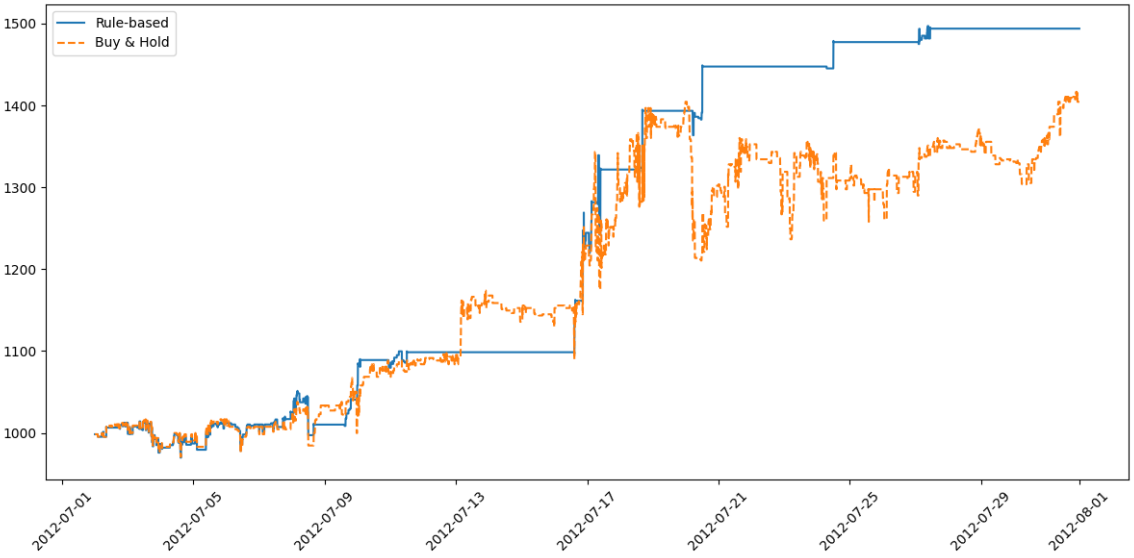
20) May



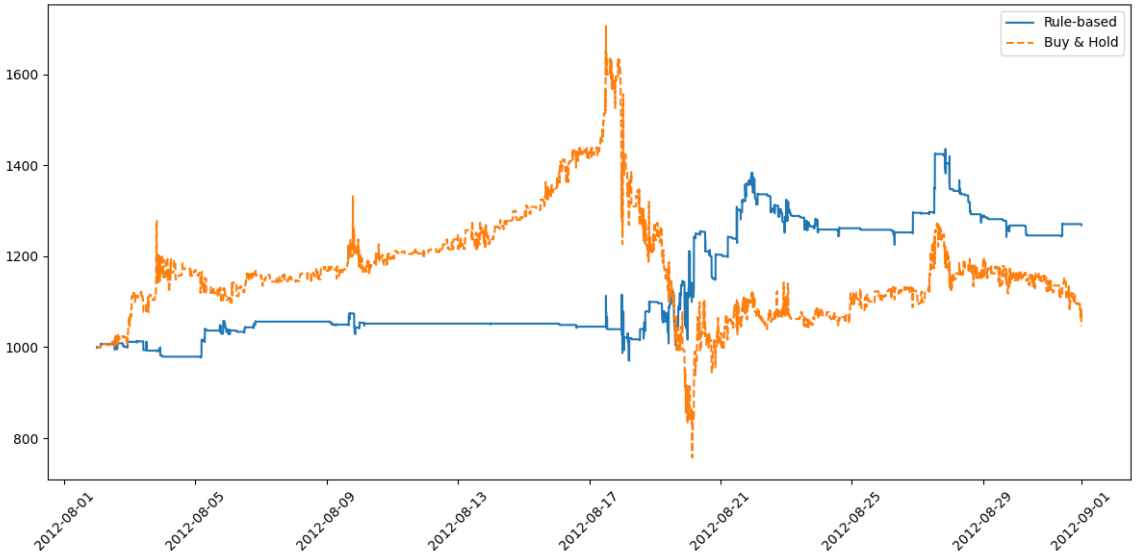
21) June



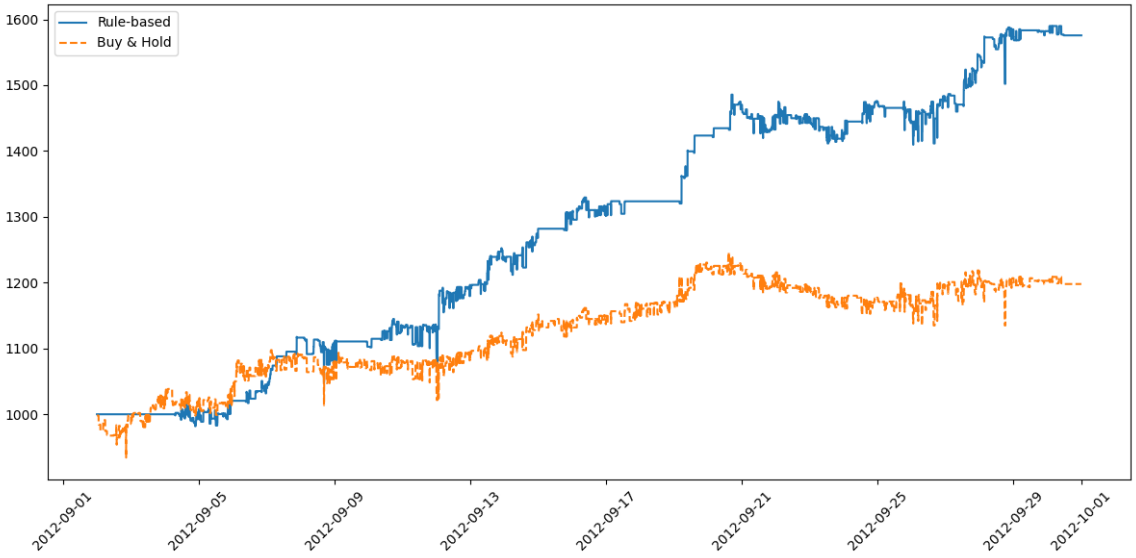
22) July



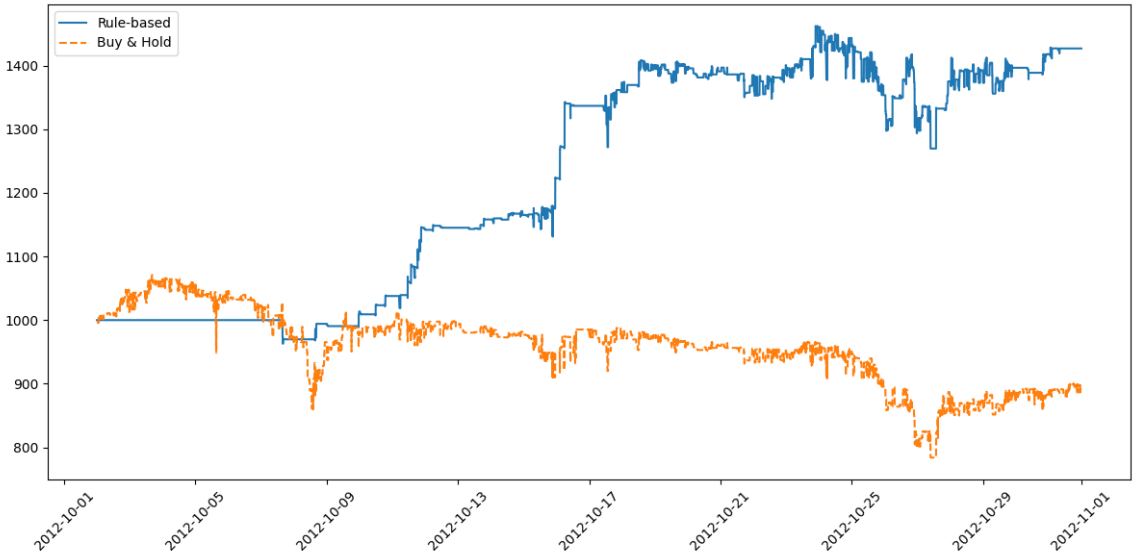
23) August



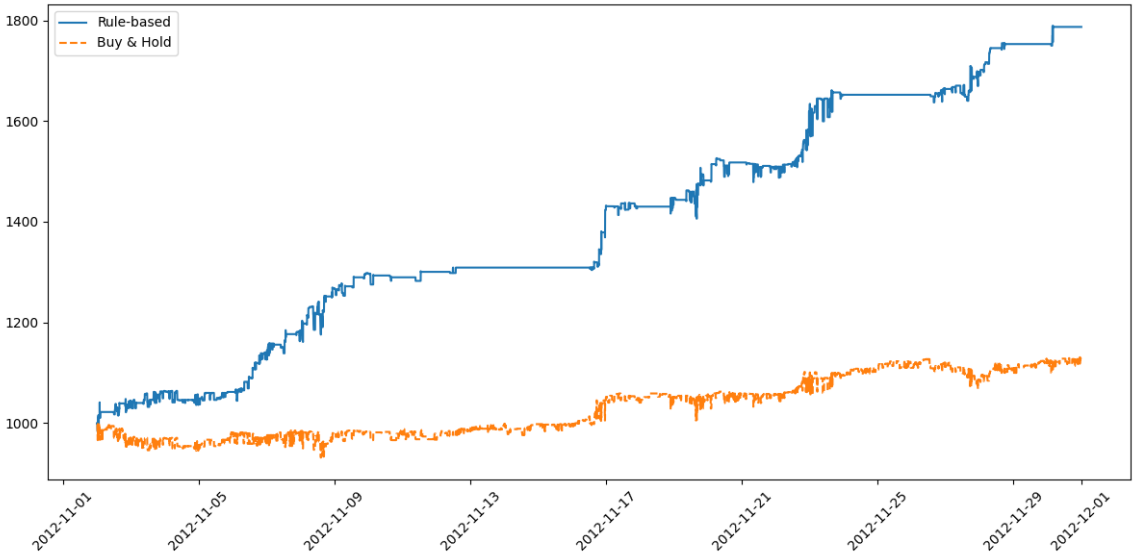
24) September



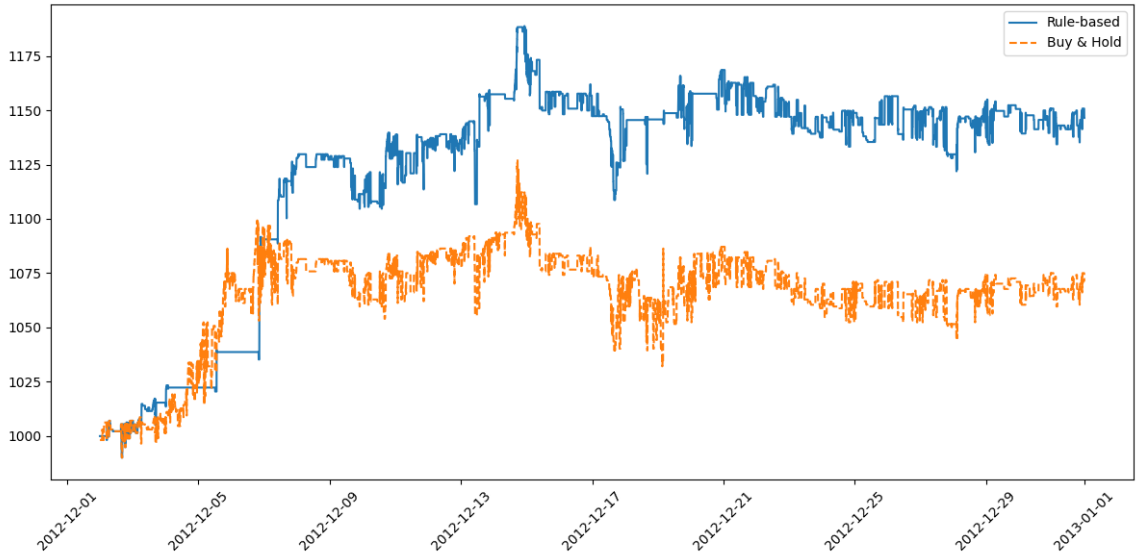
25) October



26) November

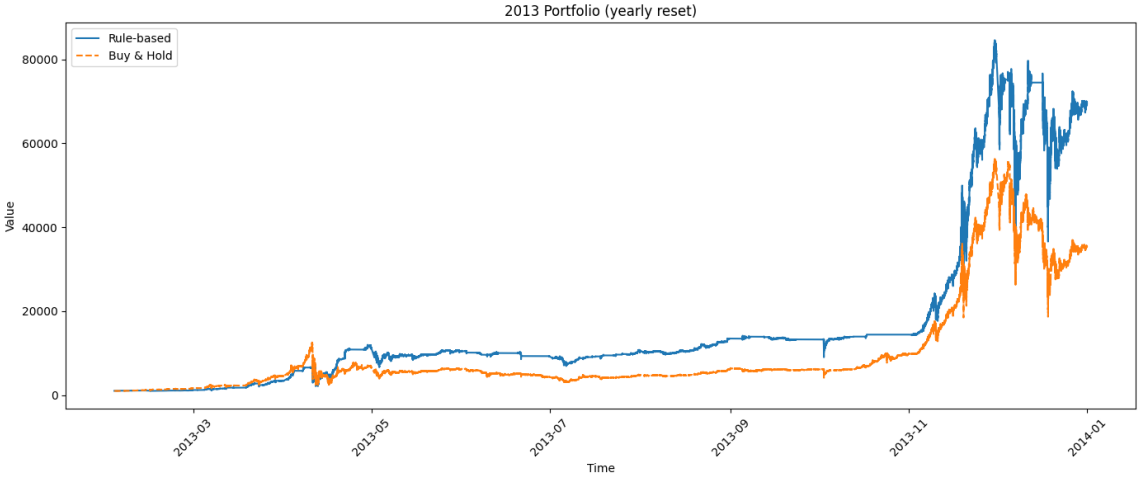


27) December

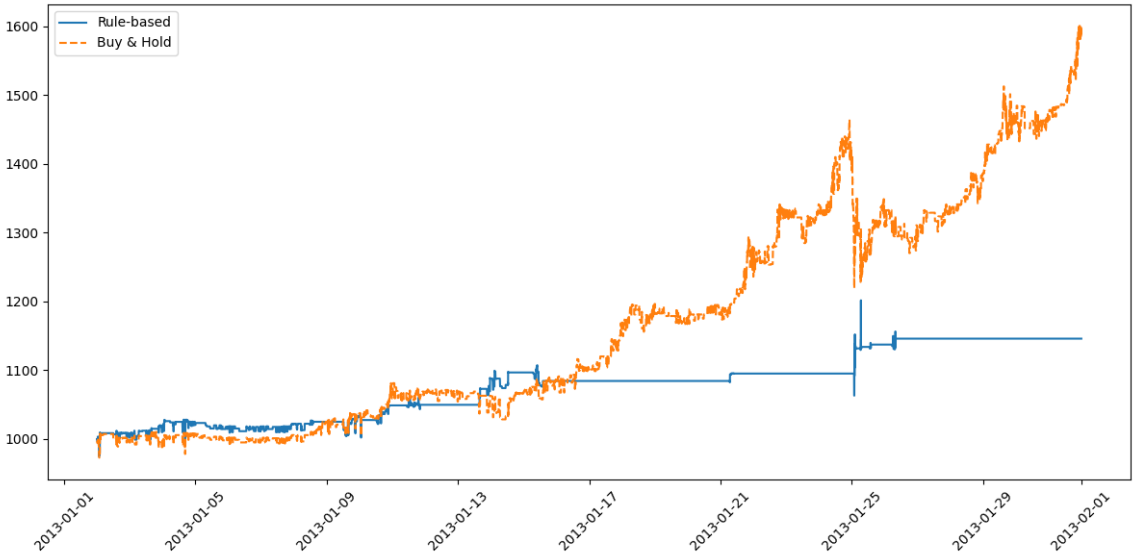


2013

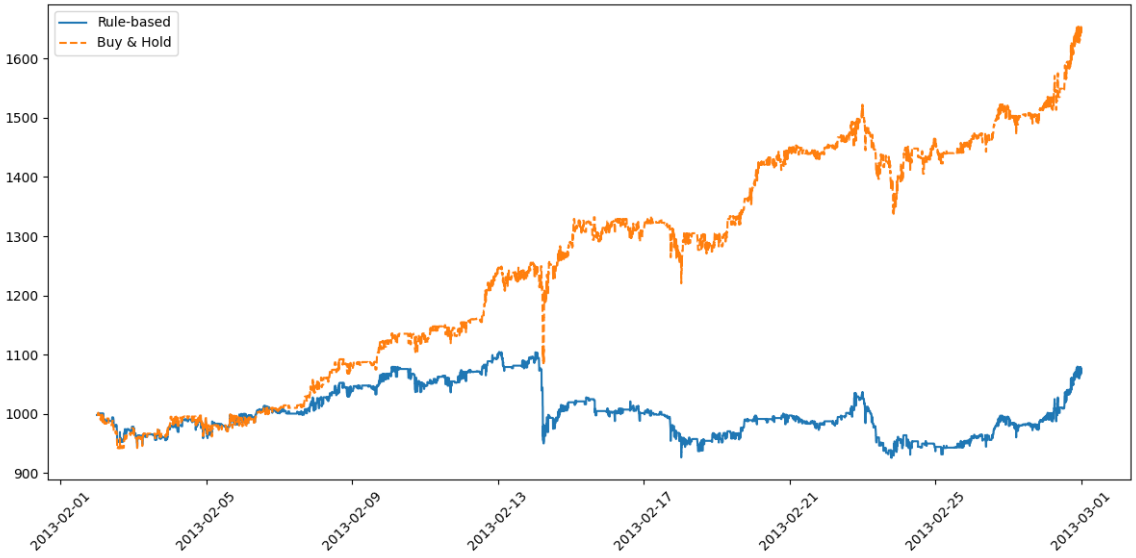
28) Full Year



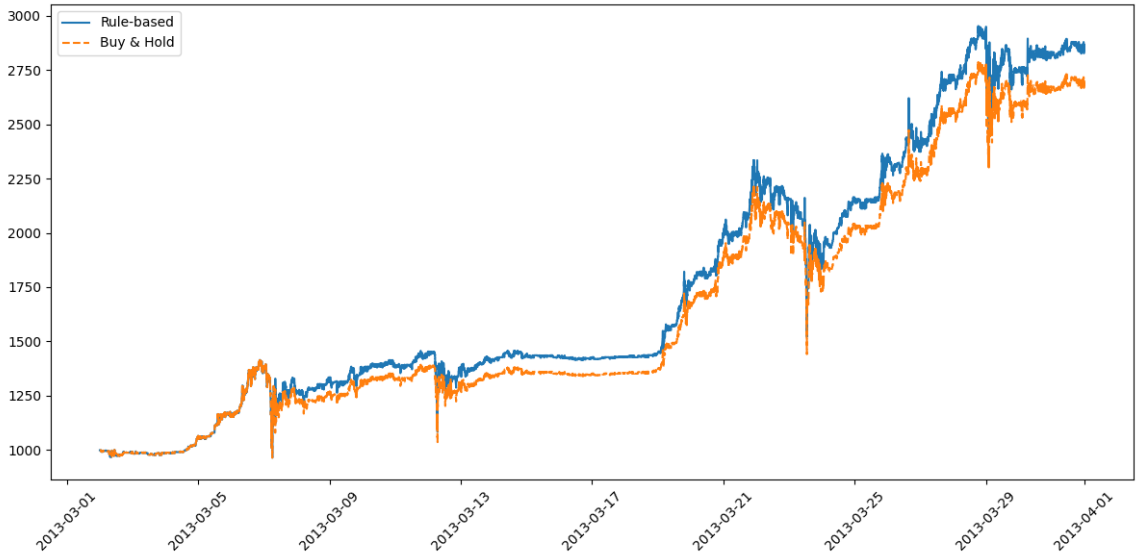
29) January



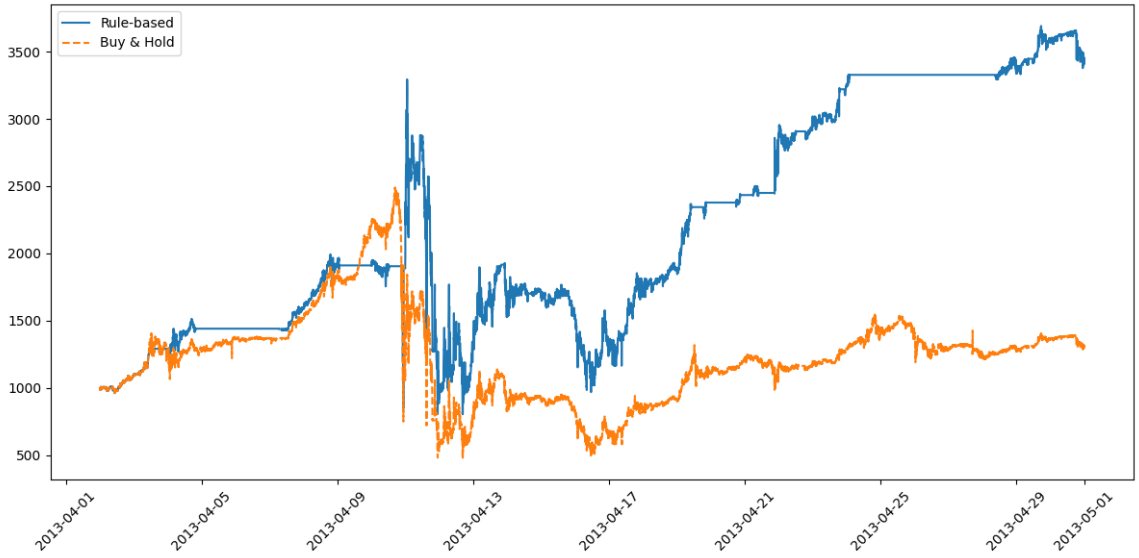
30) February



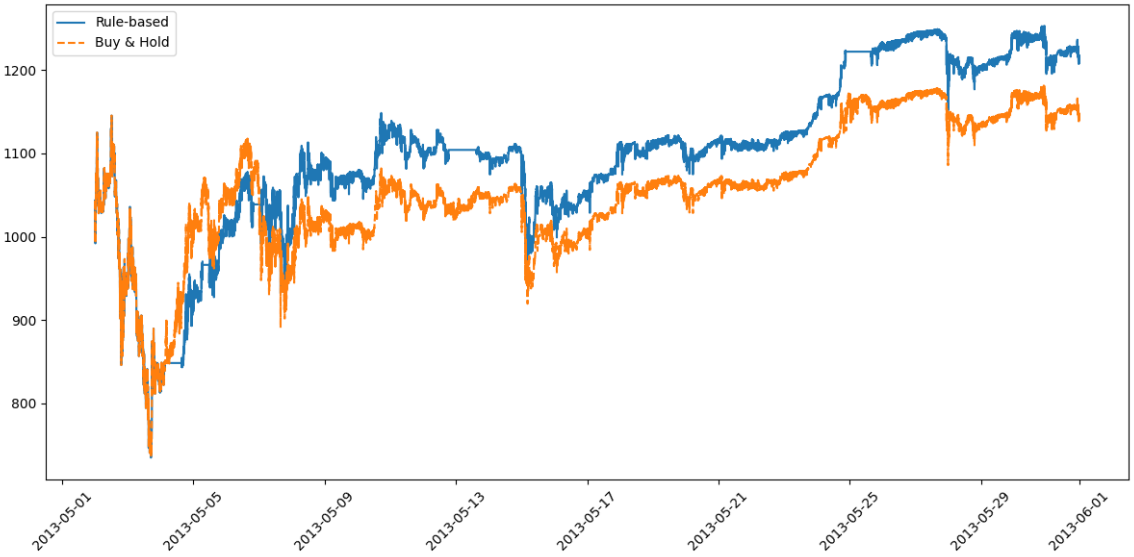
31) March



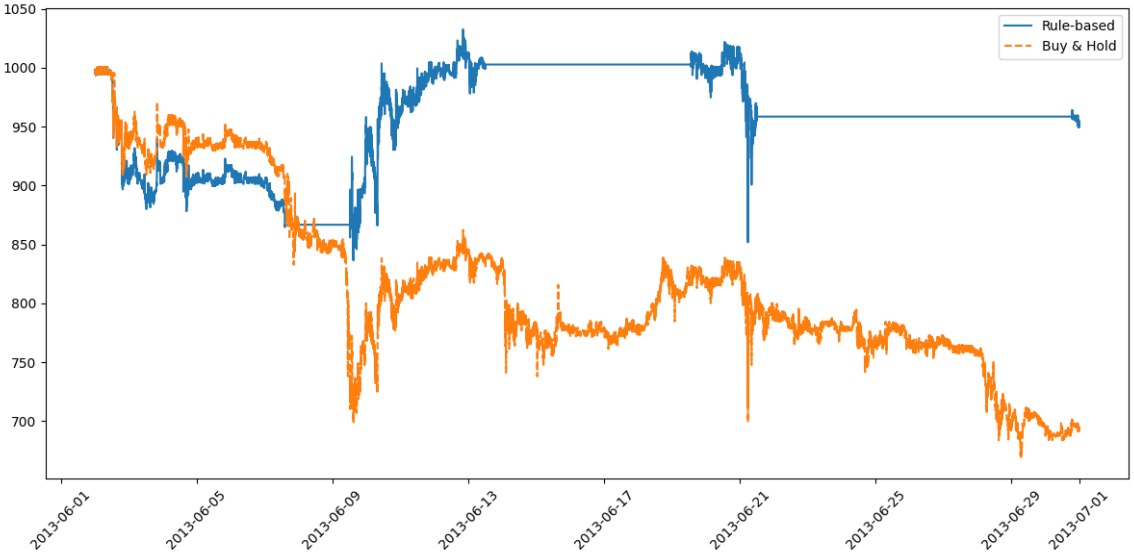
32) April



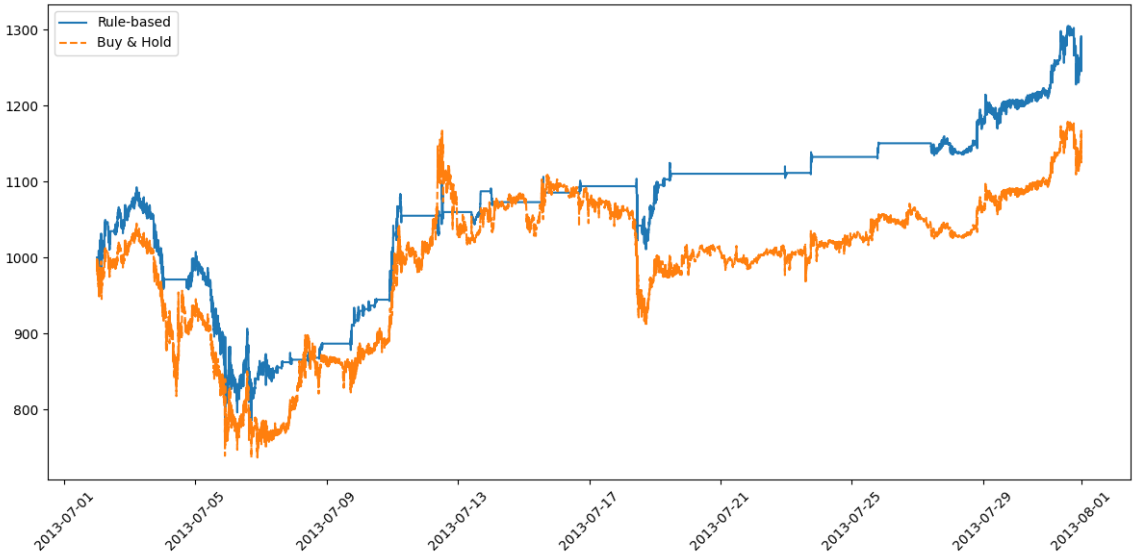
33) May



34) June

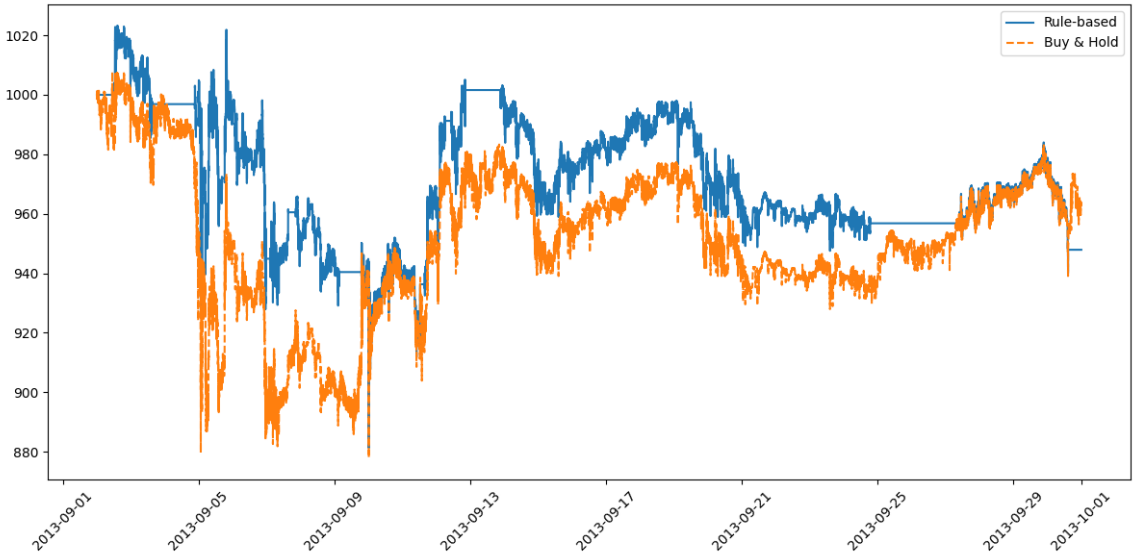


35) July

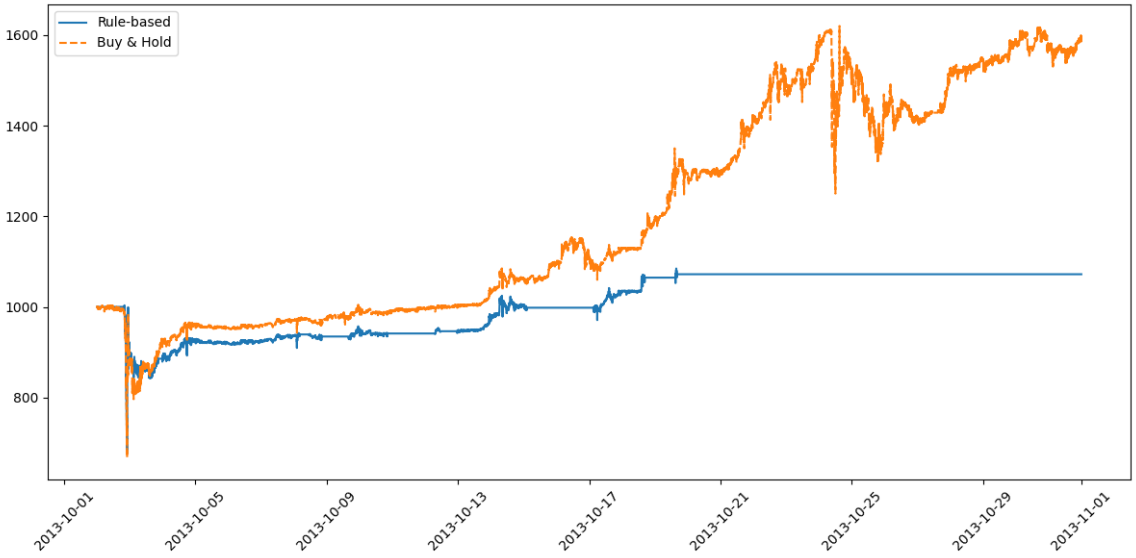


36) August

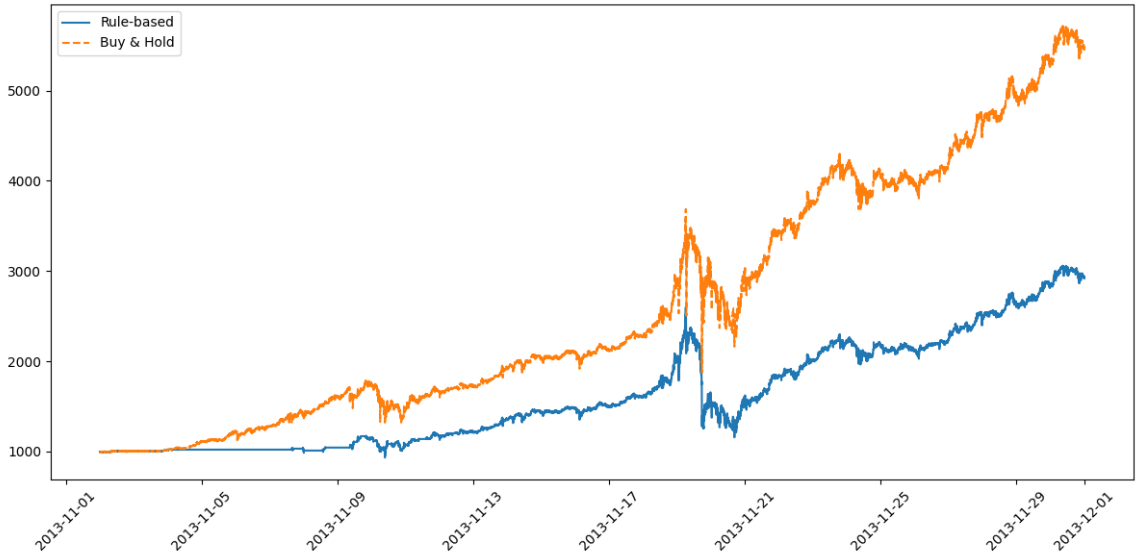
37) September



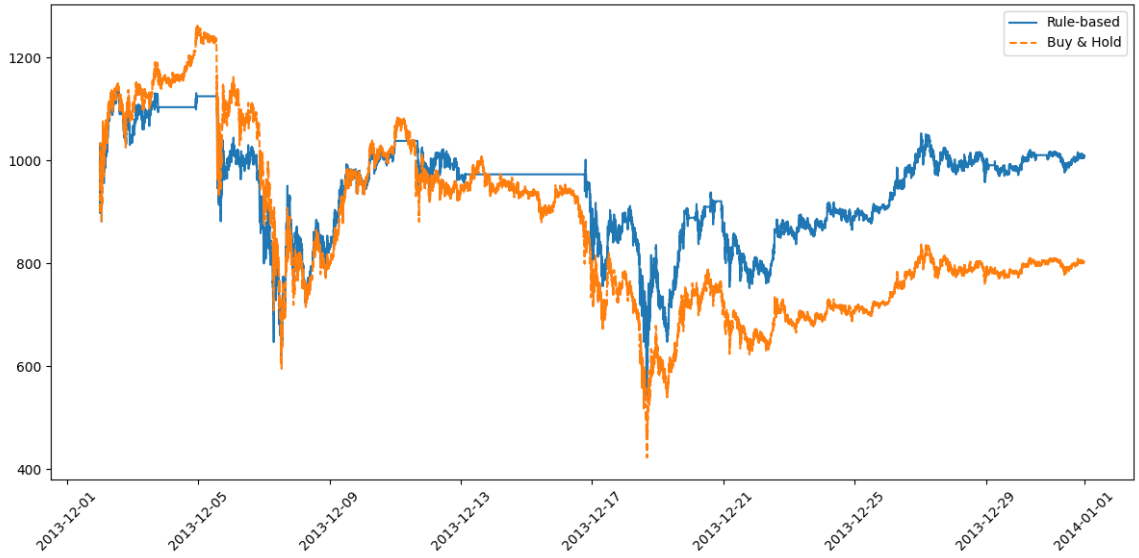
38) October



39) November

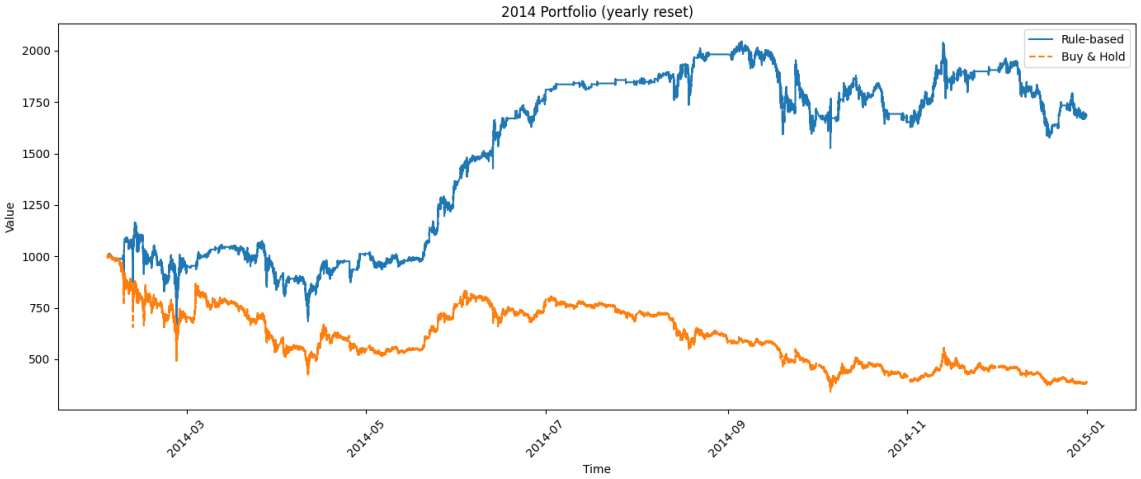


40) December

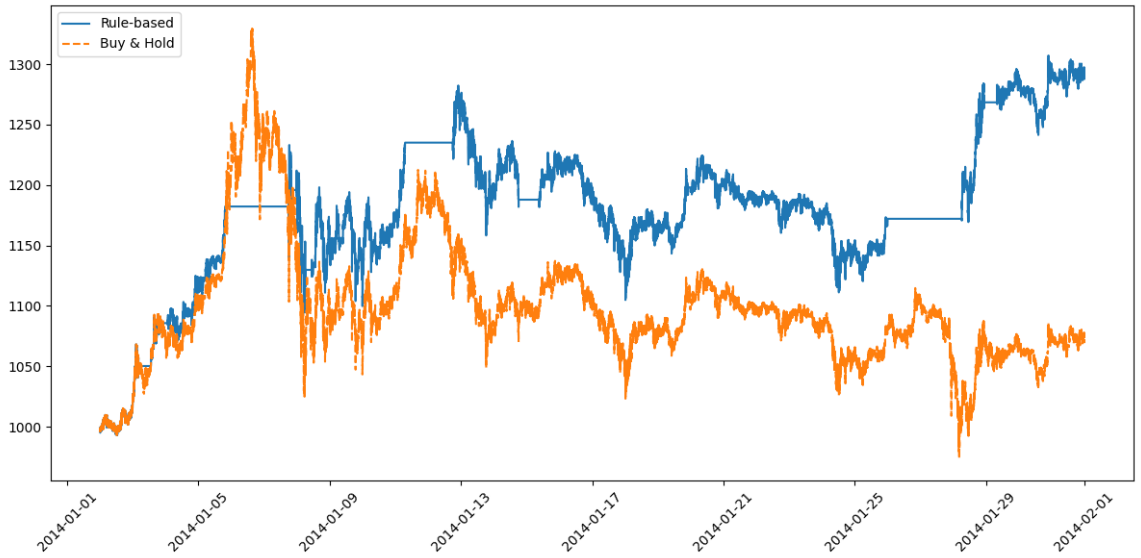


2014

41) Full Year

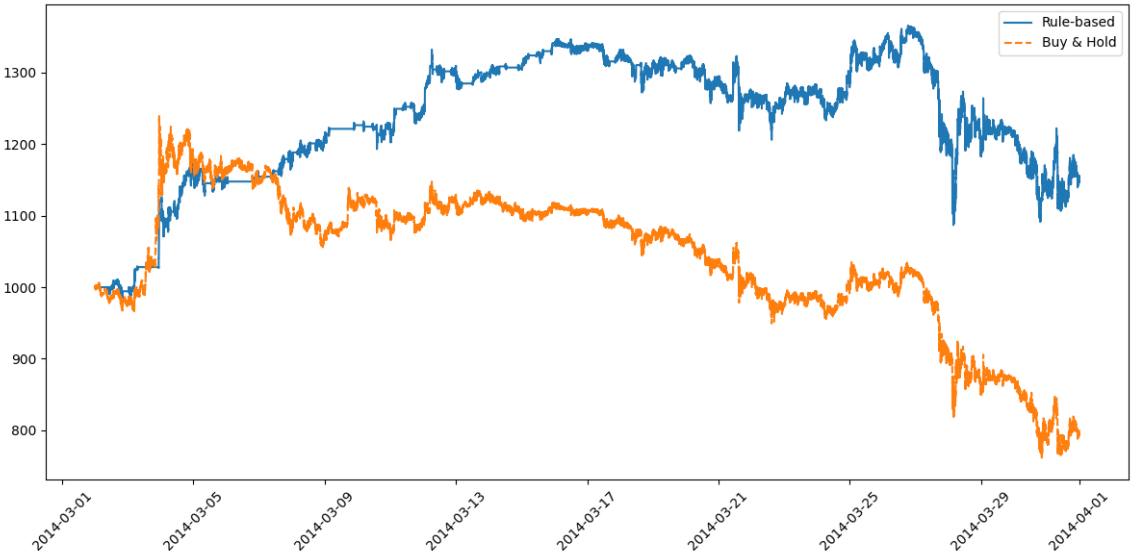


42) January

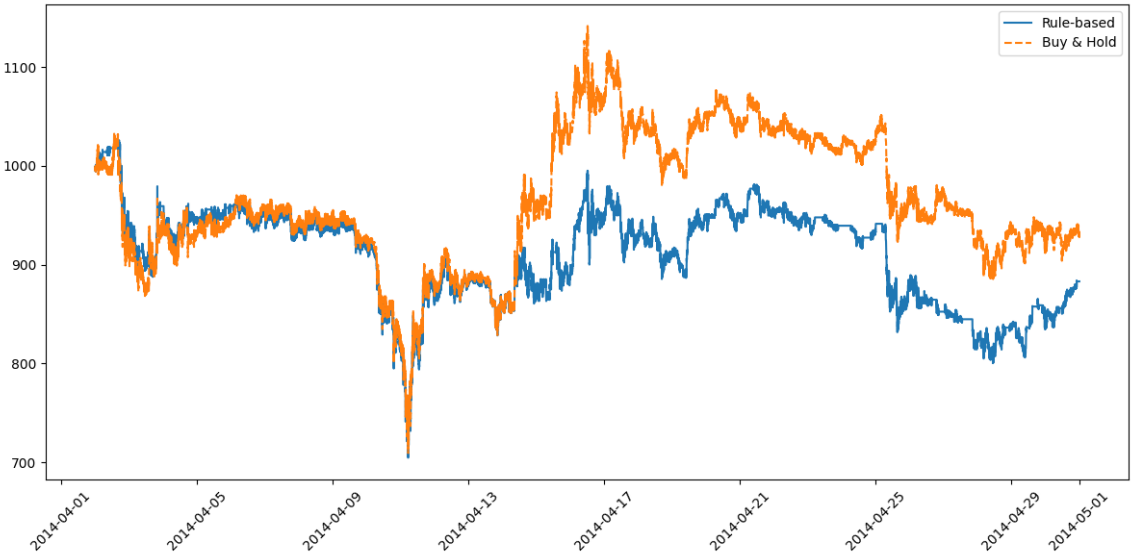


43) February

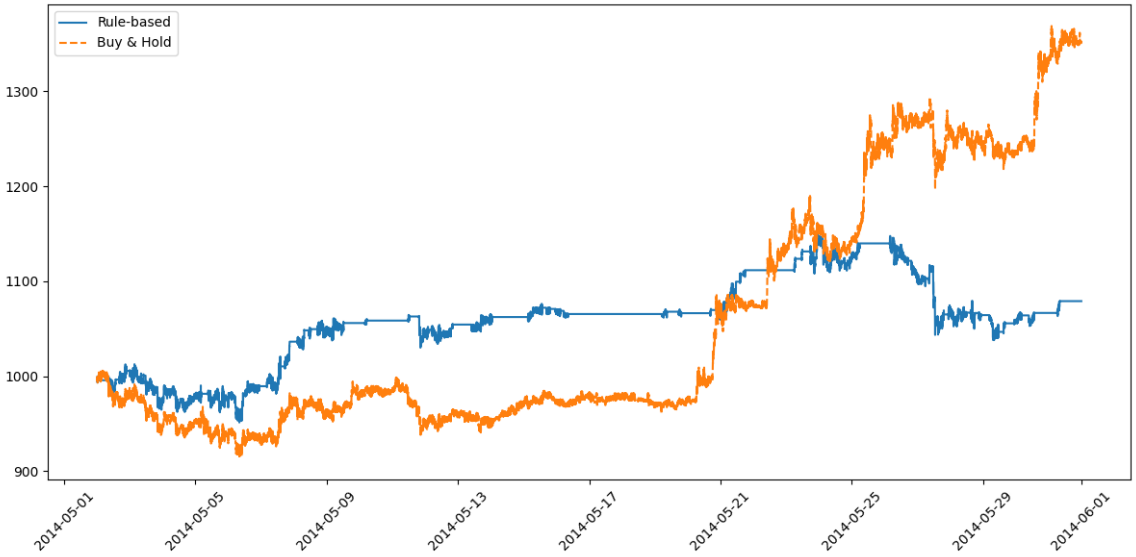
44) March



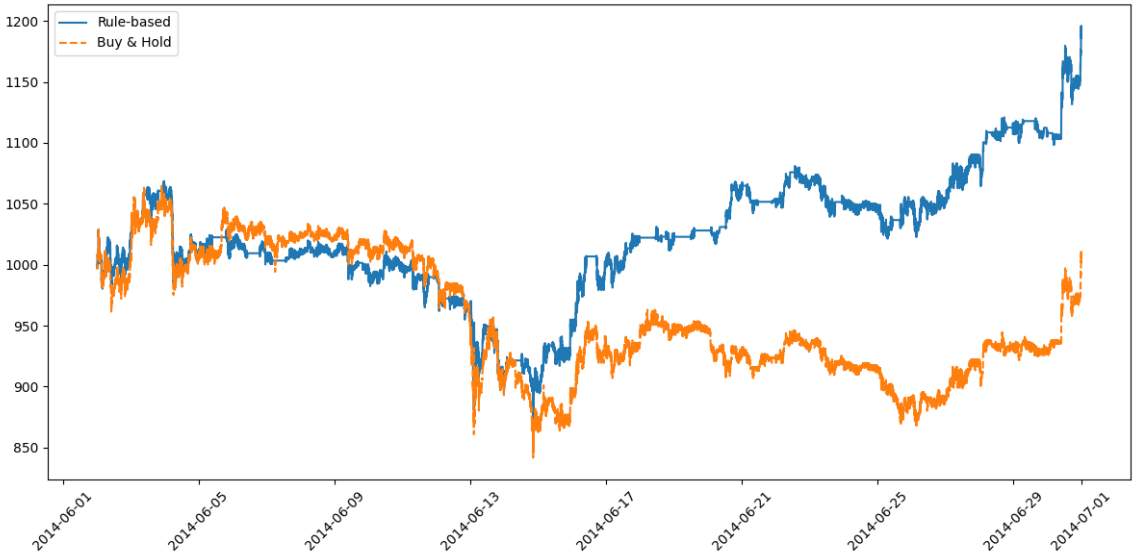
45) April



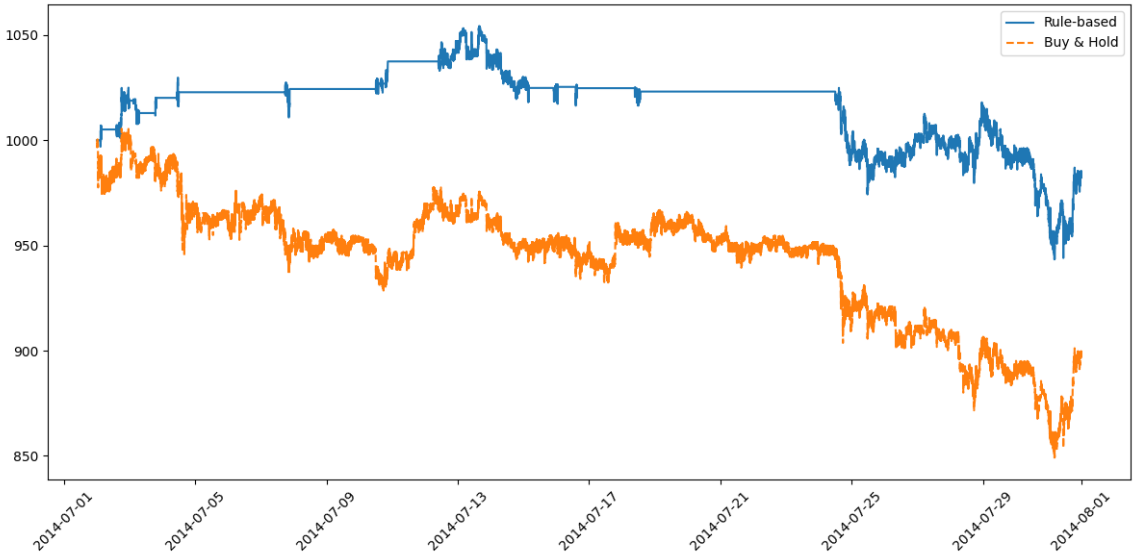
46) May



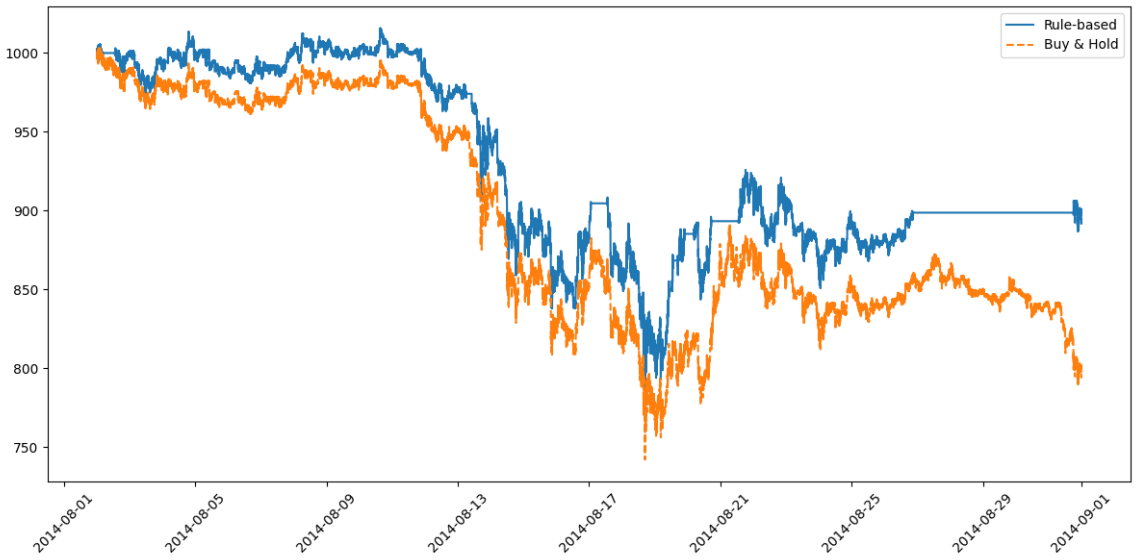
47) June



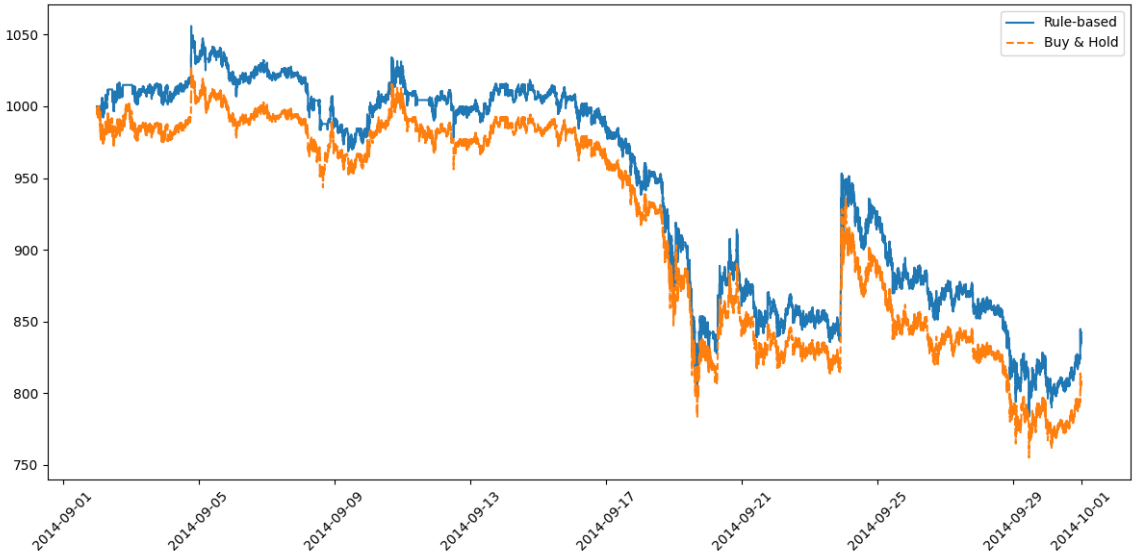
48) July



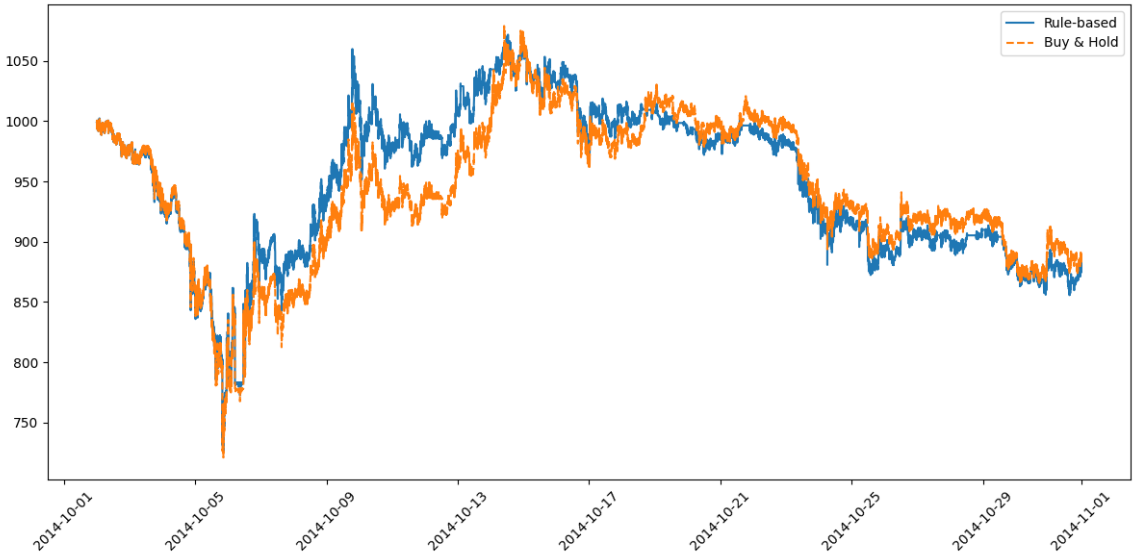
49) August



50) September

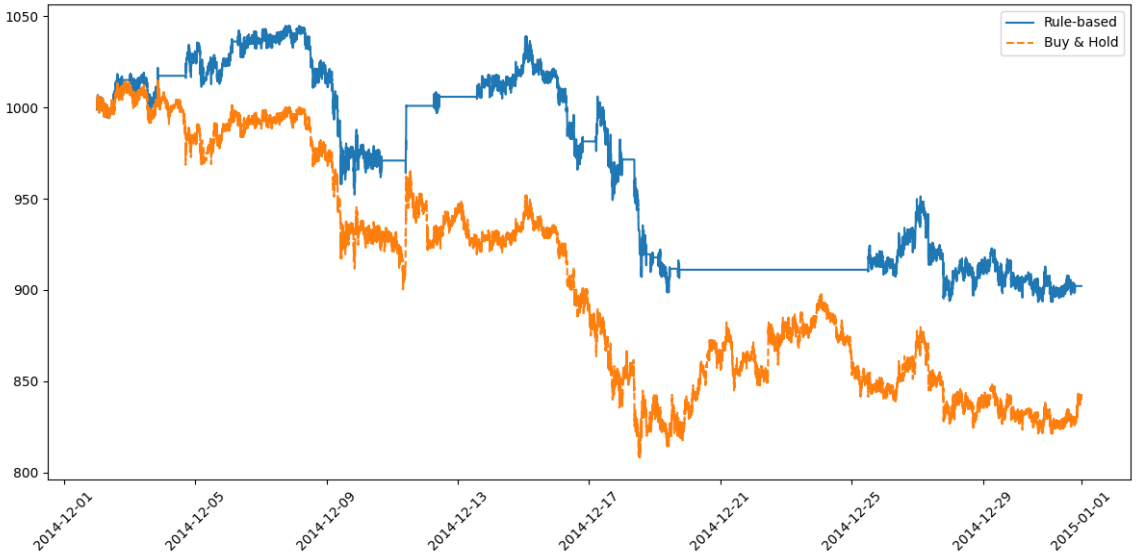


51) October



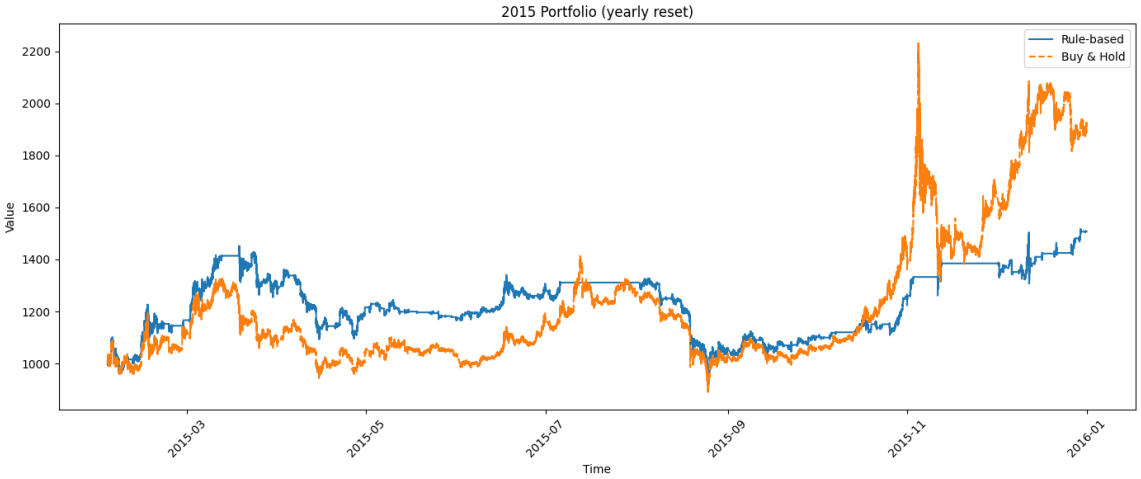
52) November

53) December

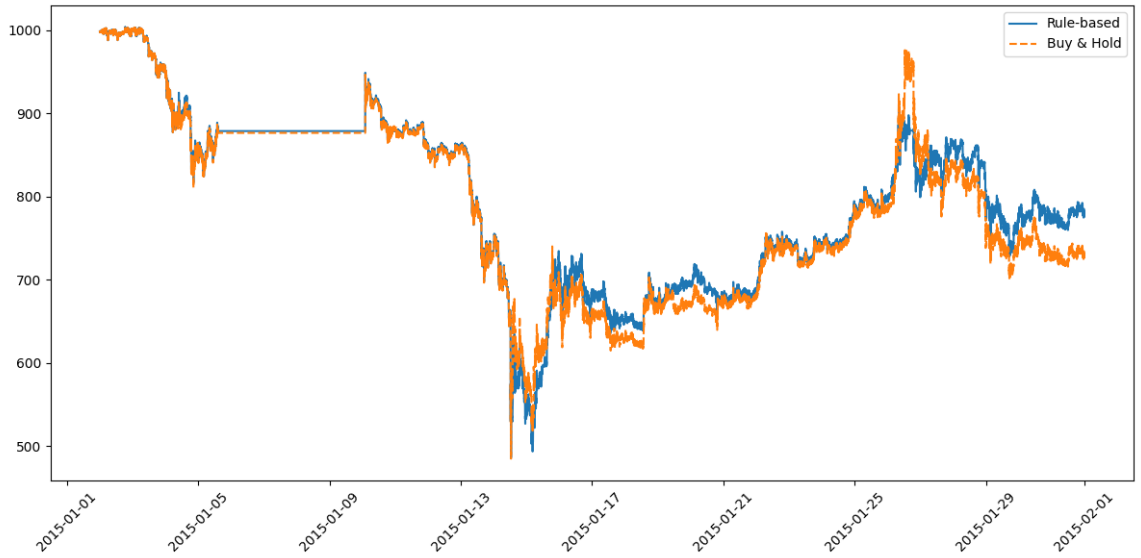


2015

54) Full year

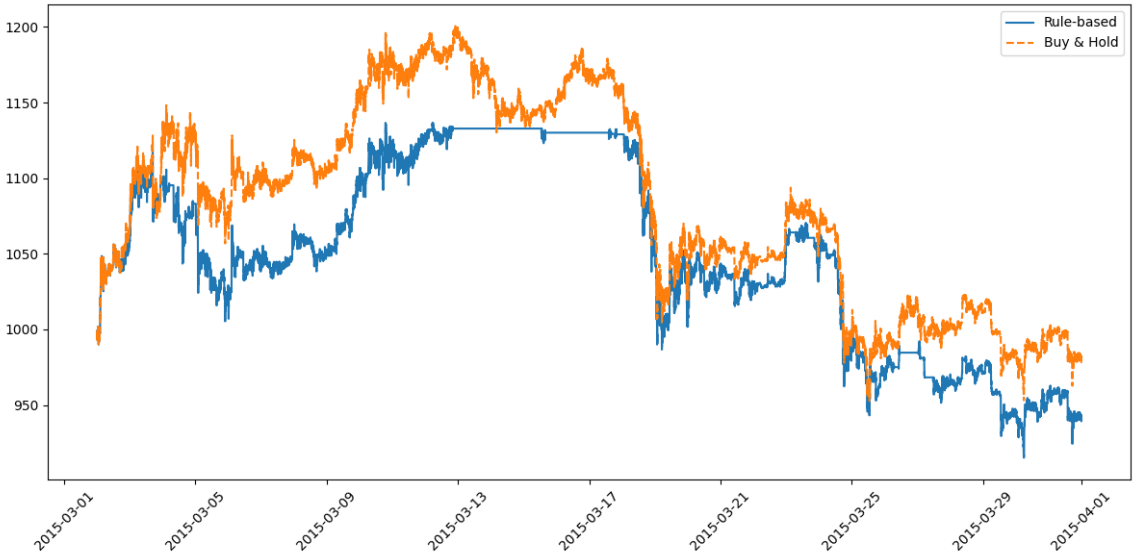


55) January



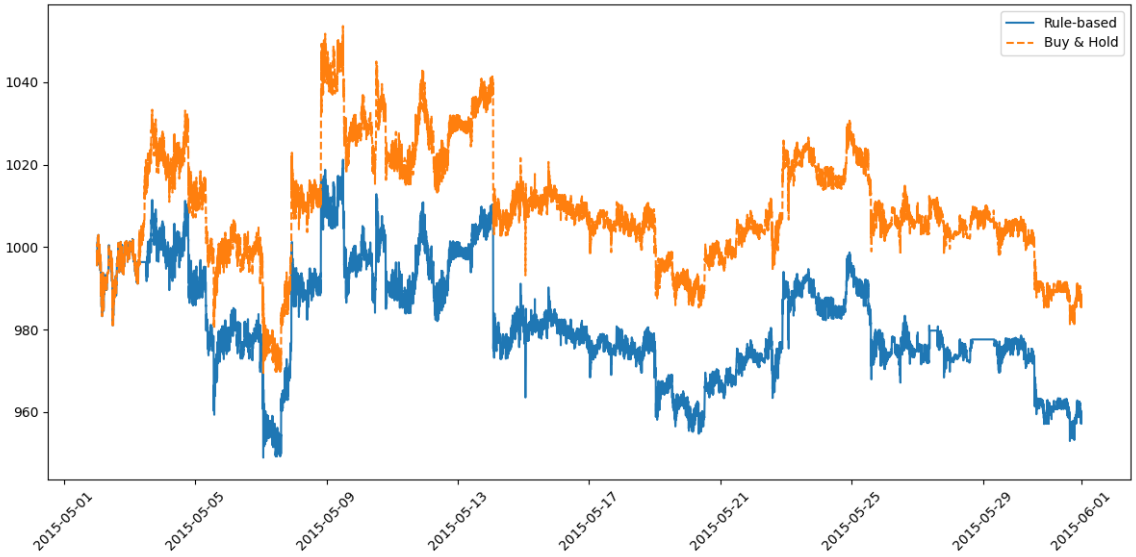
56) February

57) March



58) April

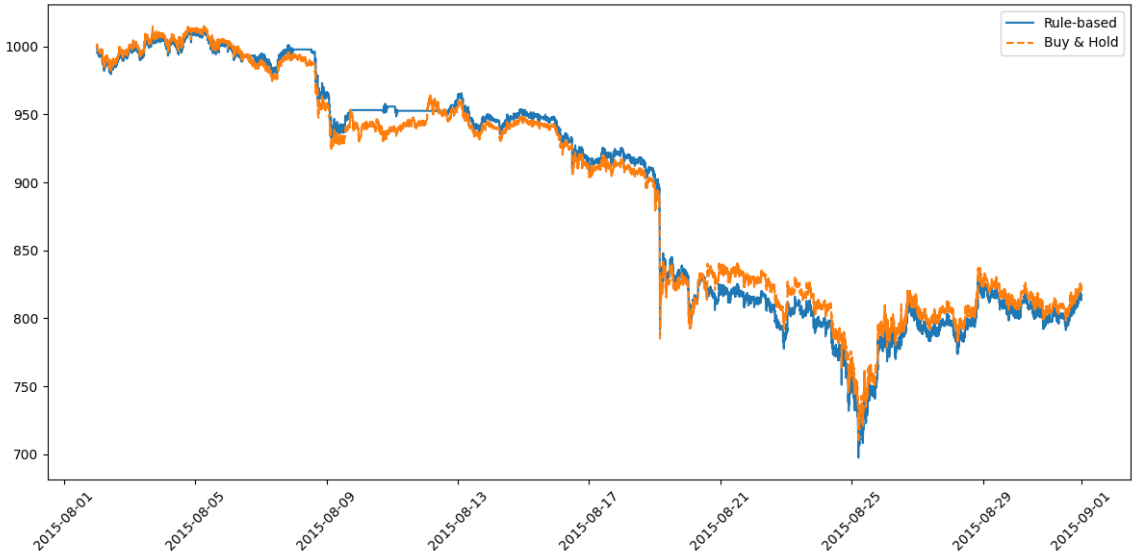
59) May



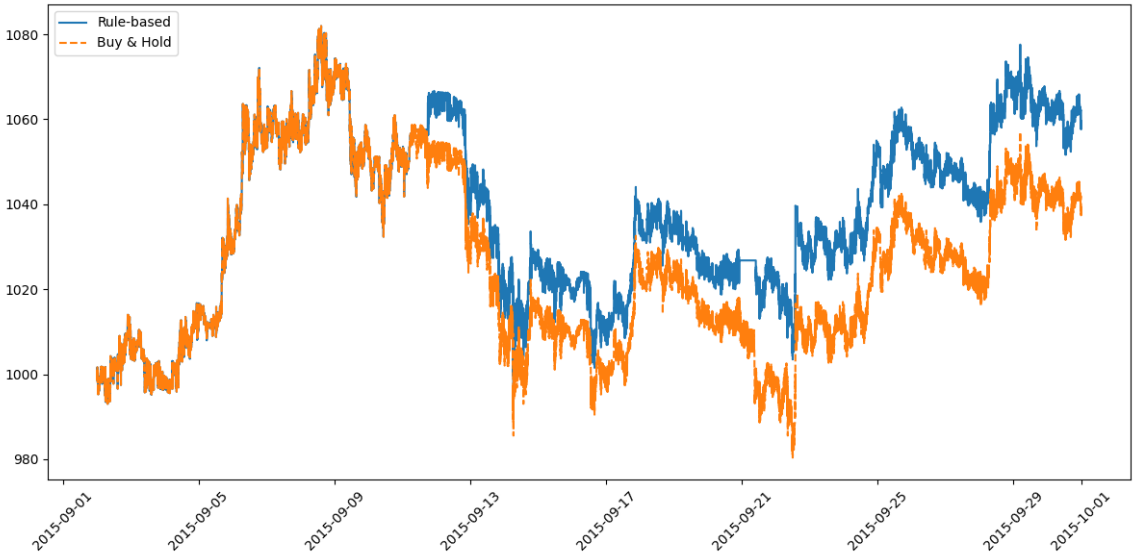
60) June

61) July

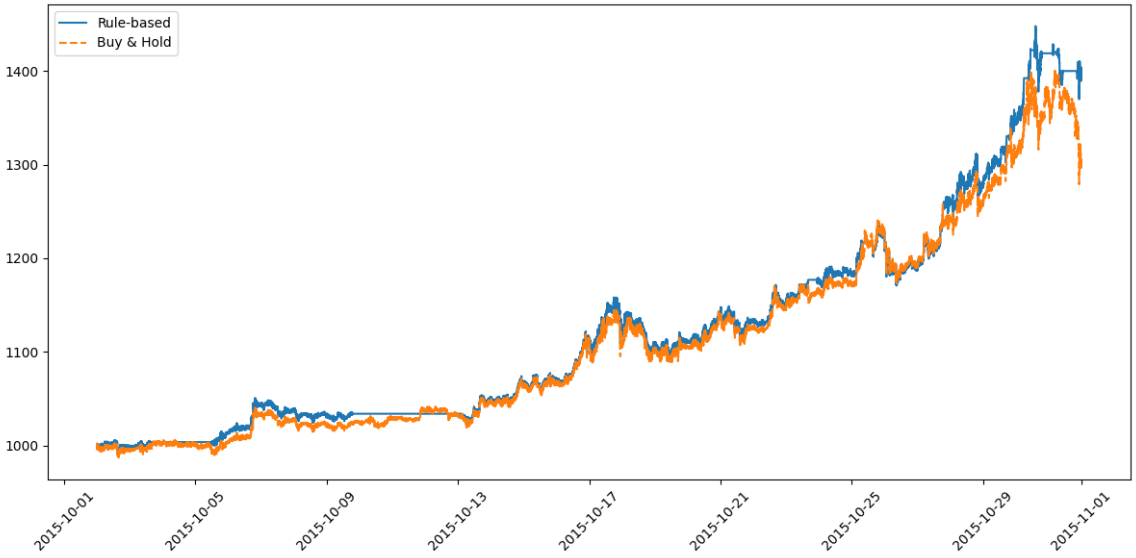
62) August



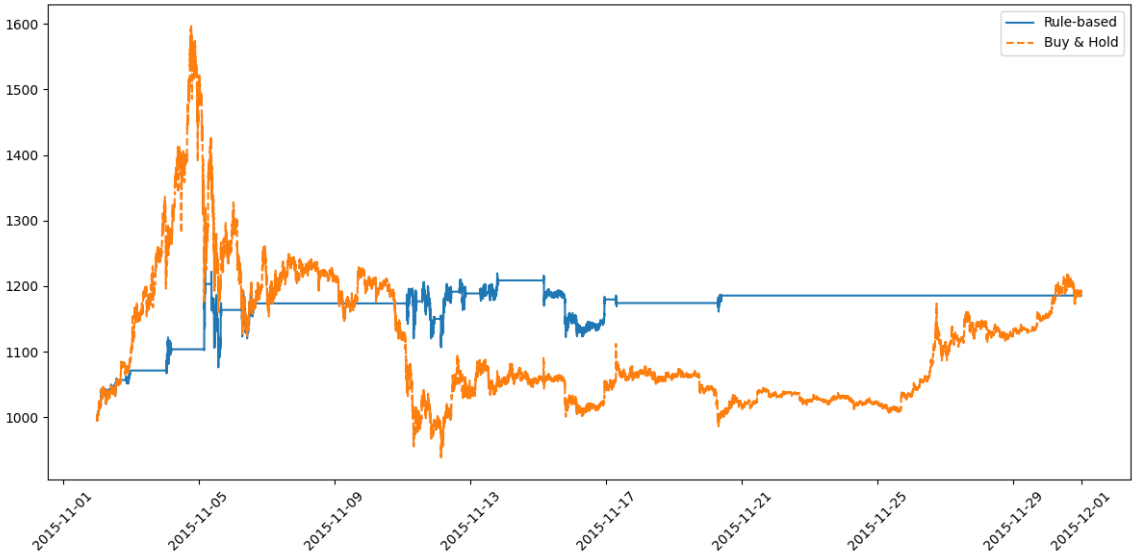
63) September



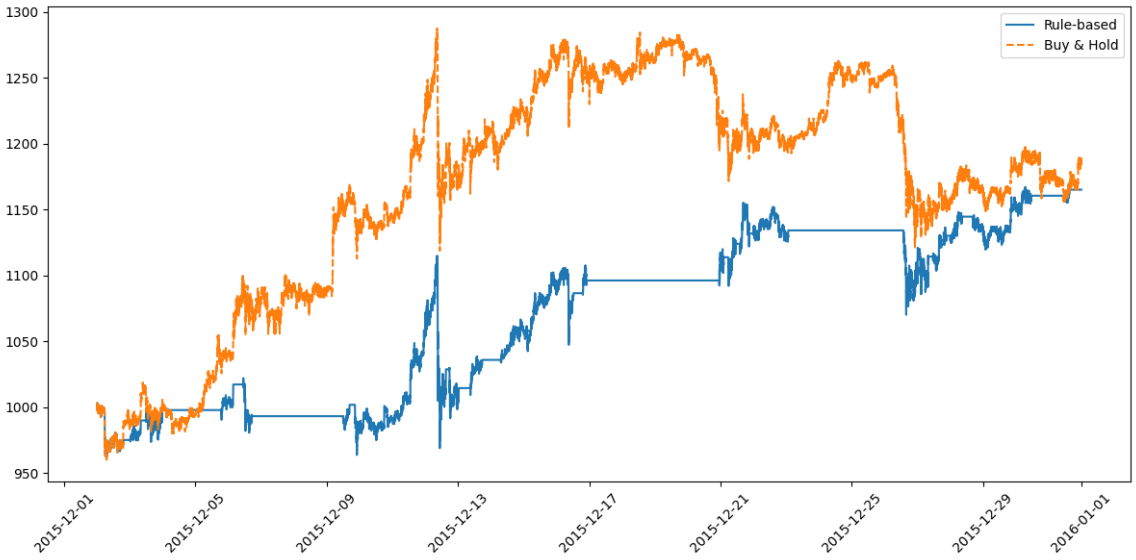
64) October



65) November

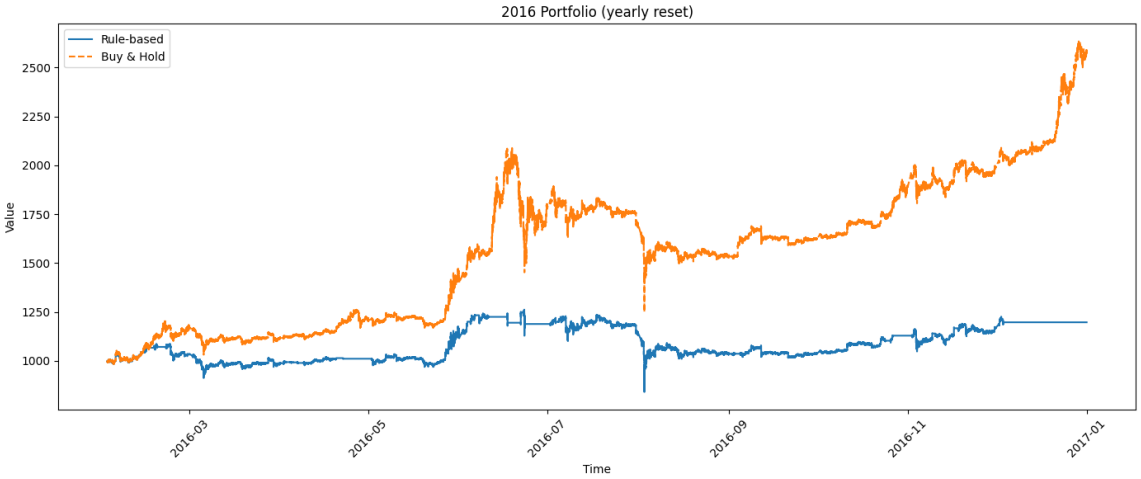


66) December

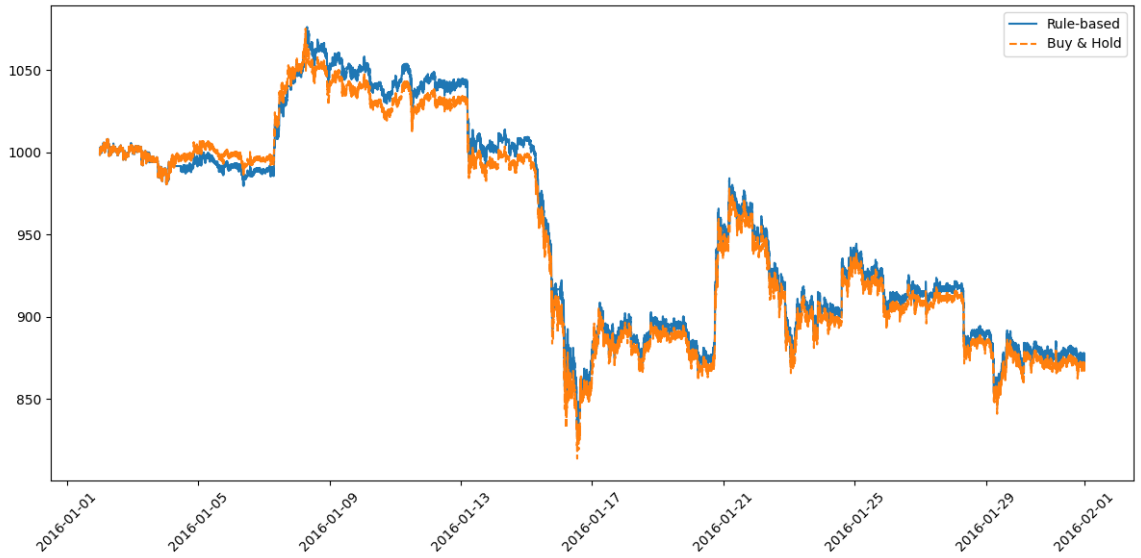


2016

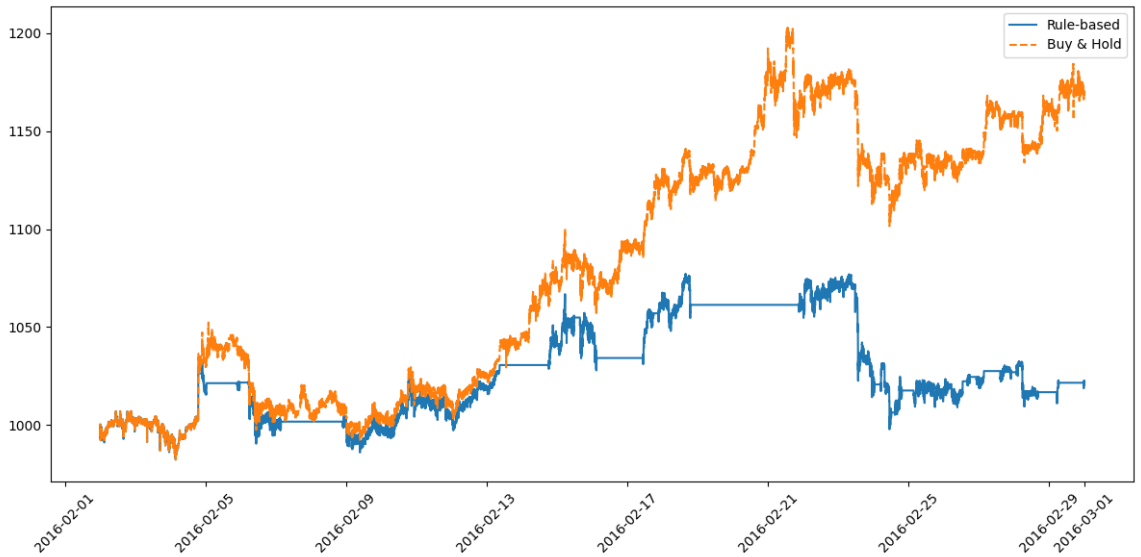
67) Full year



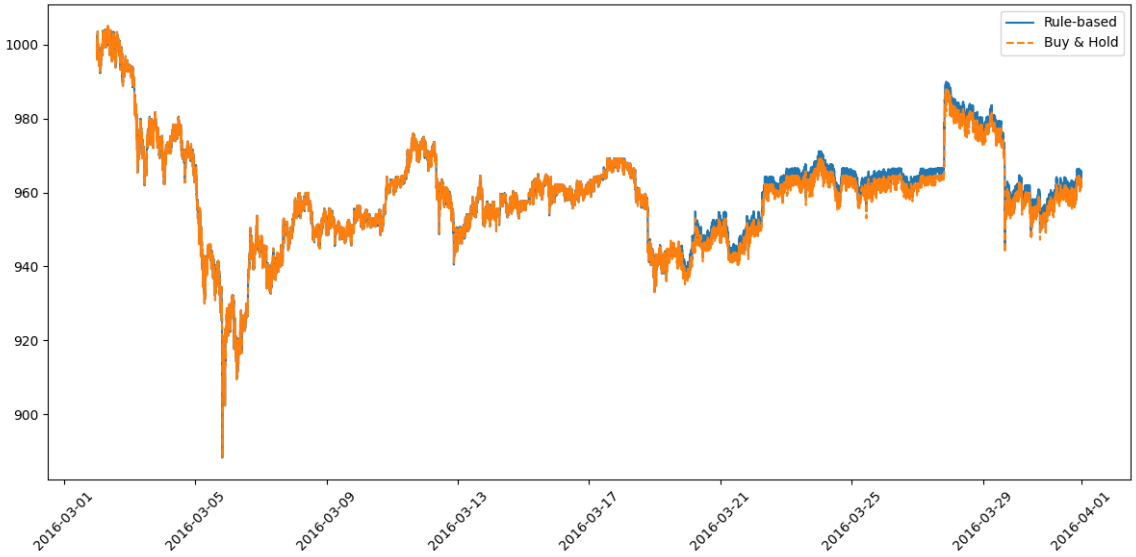
68) January



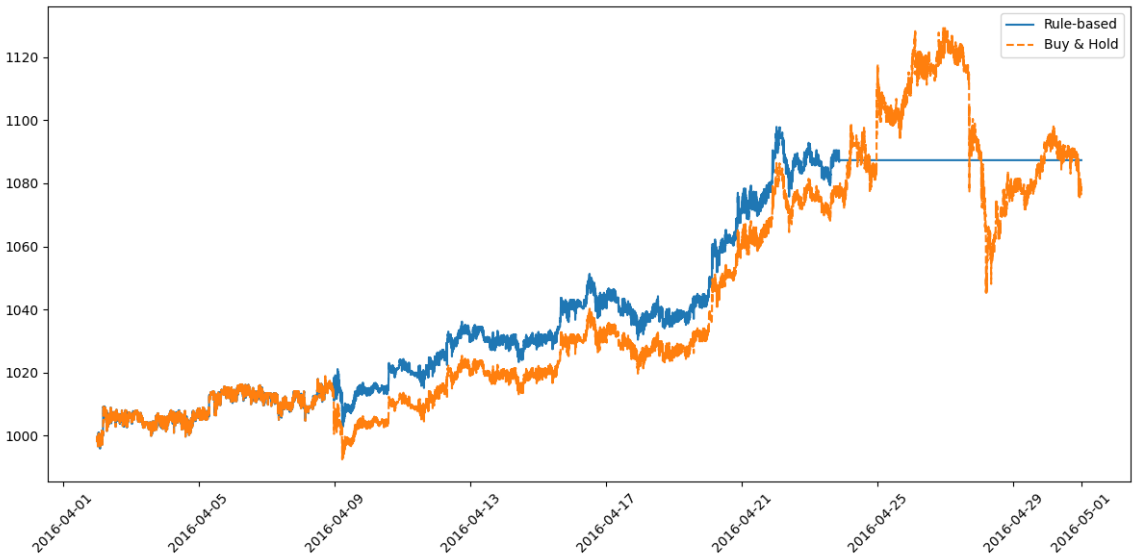
69) February



70) March



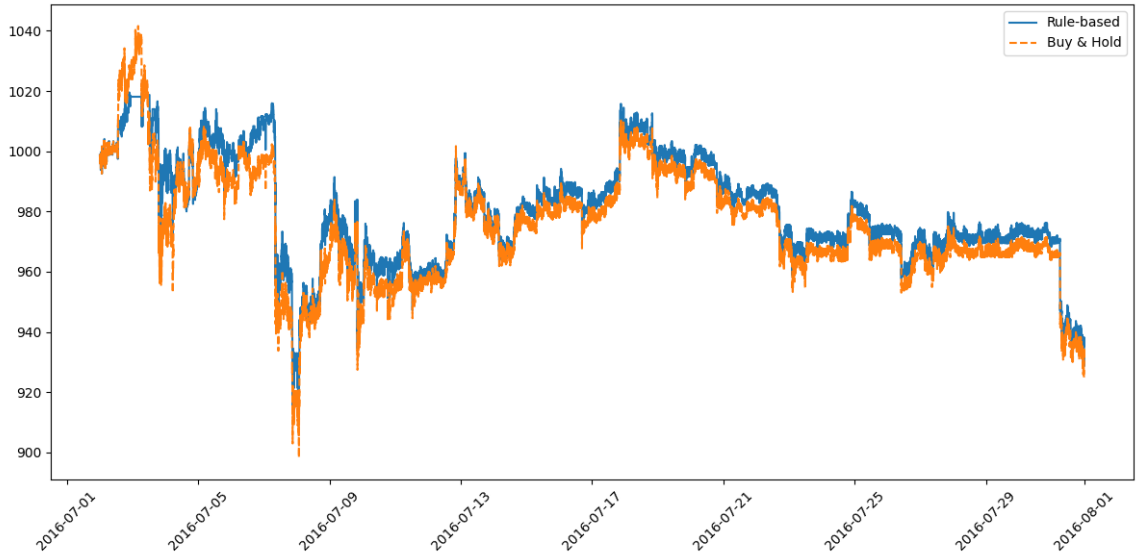
71) April



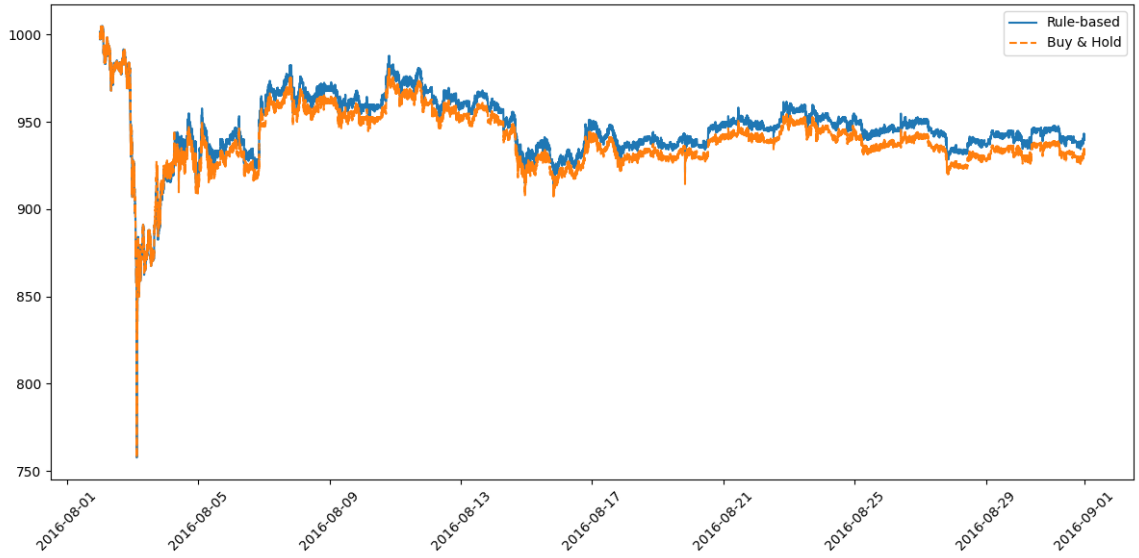
72) May

73) June

74) July



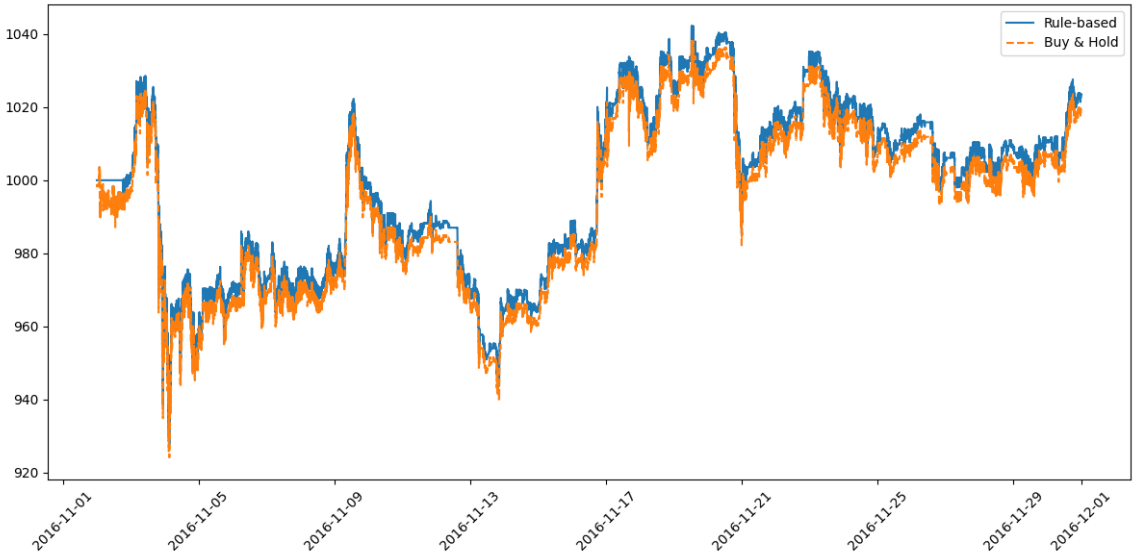
75) August



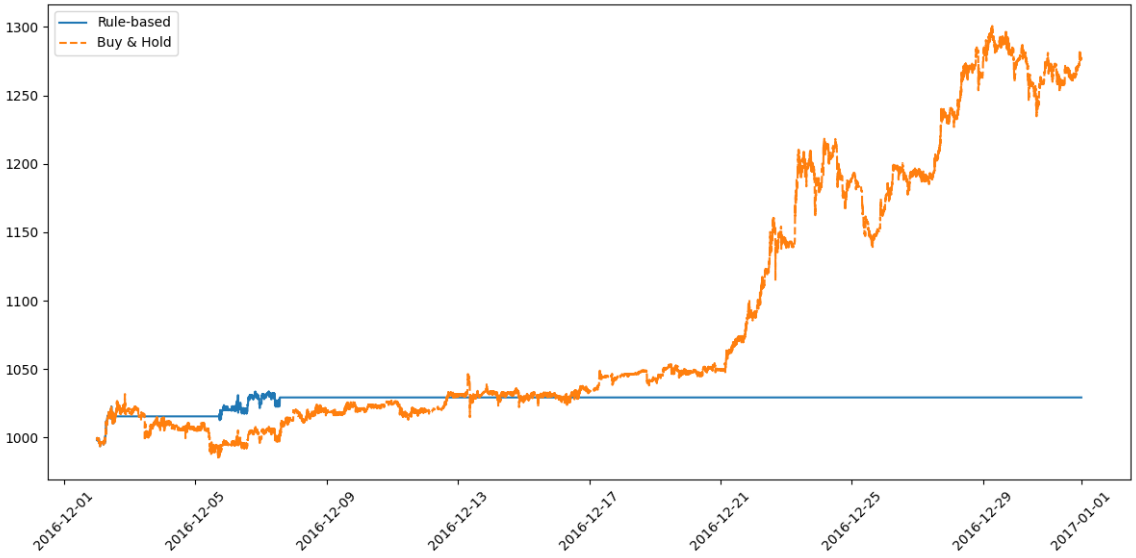
76) September

77) October

78) November

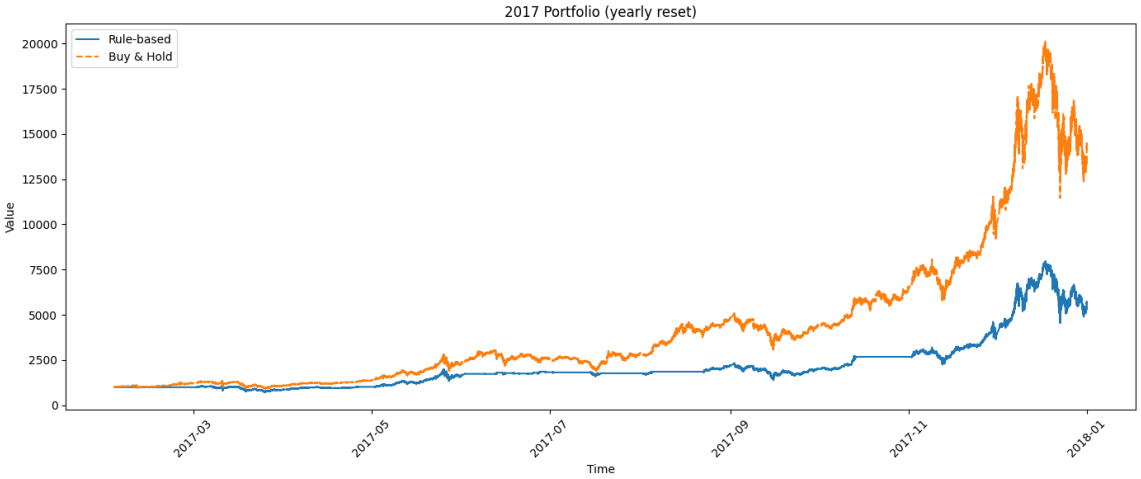


79) December



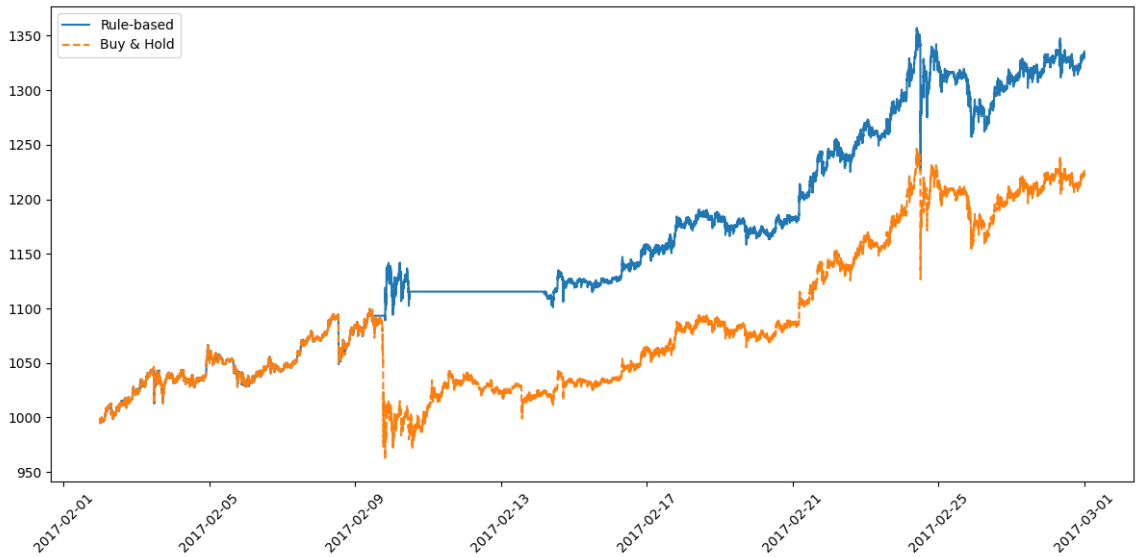
2017

80) Full Year

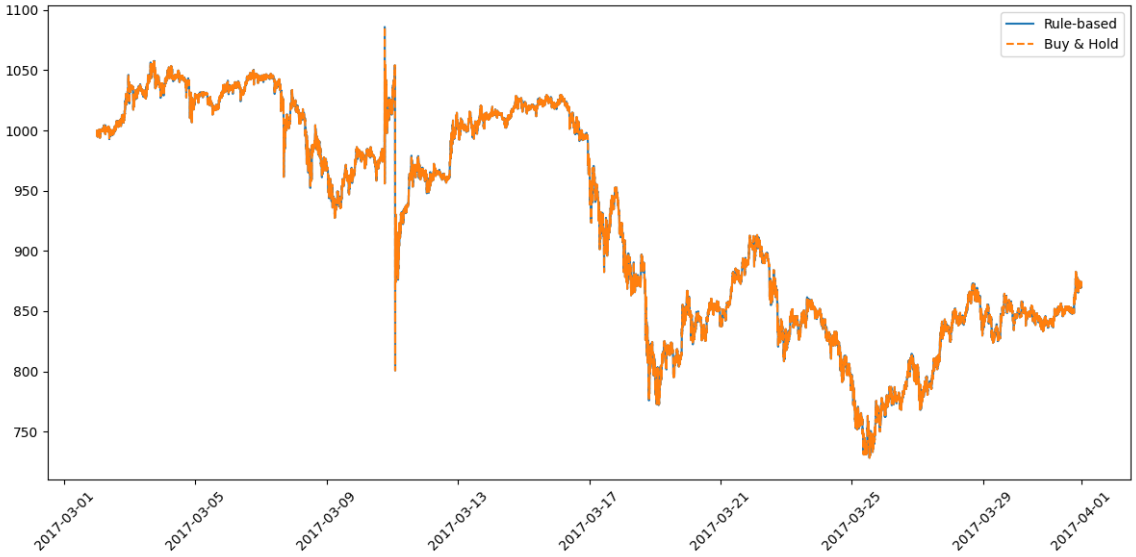


81) January

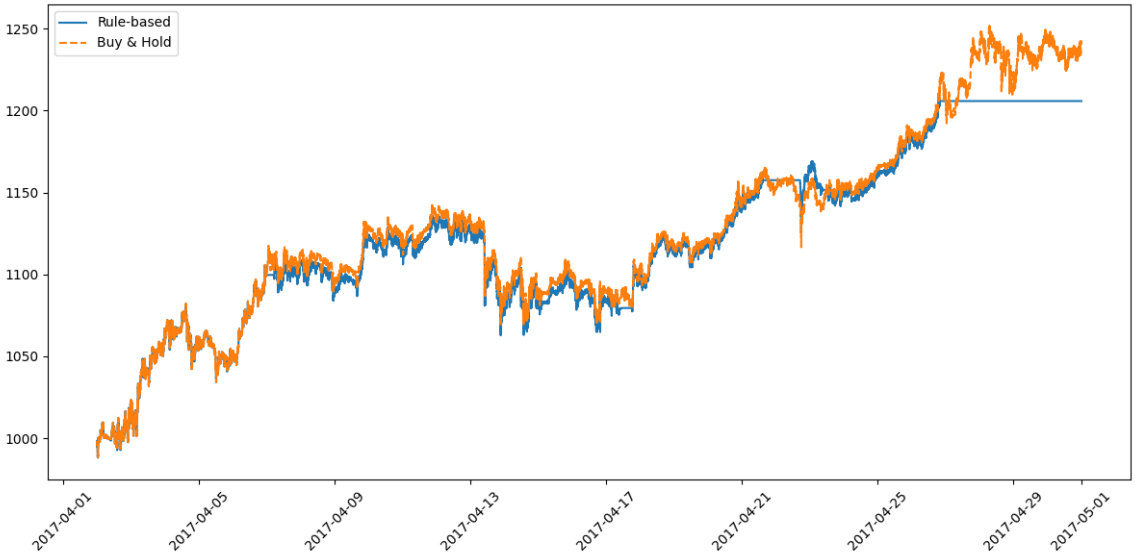
82) February



83) March



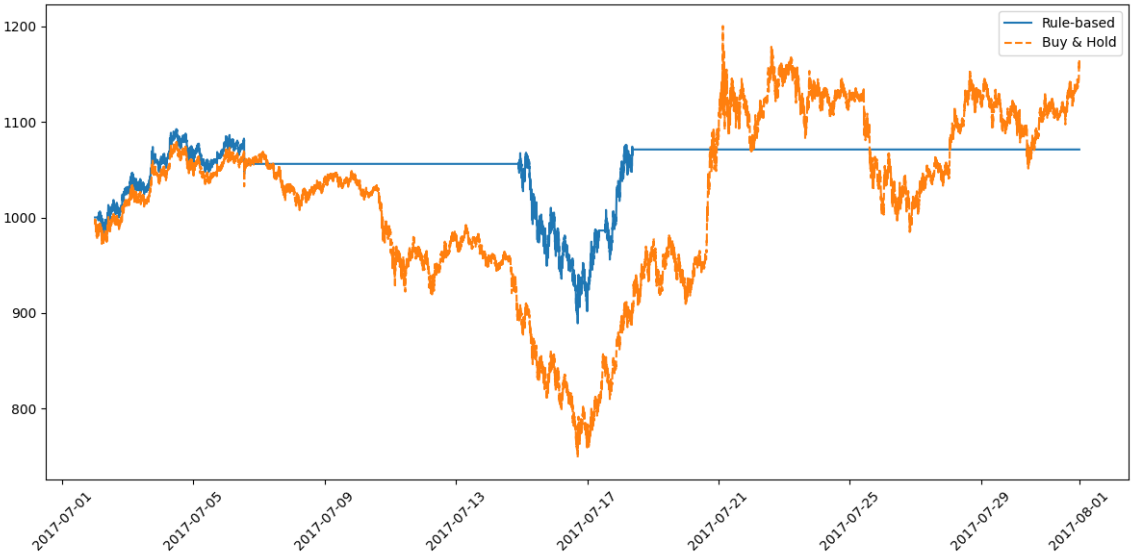
84) April



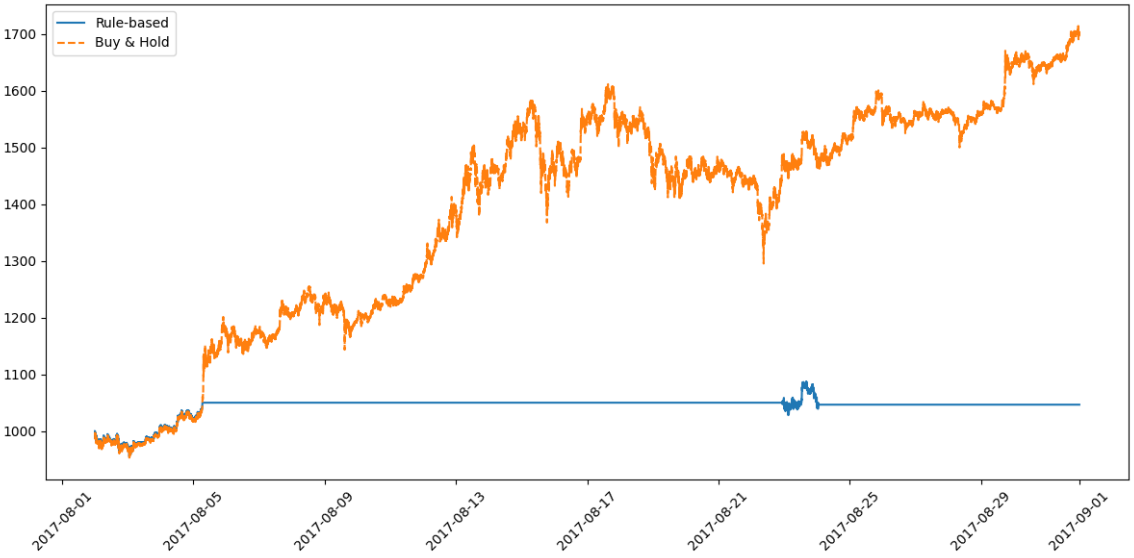
85) May

86) June

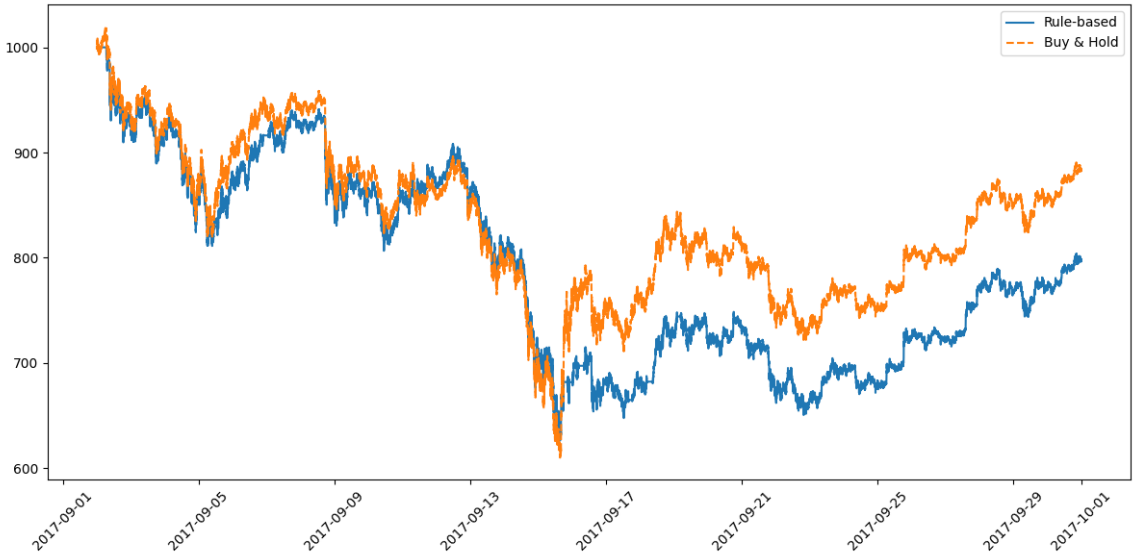
87) July



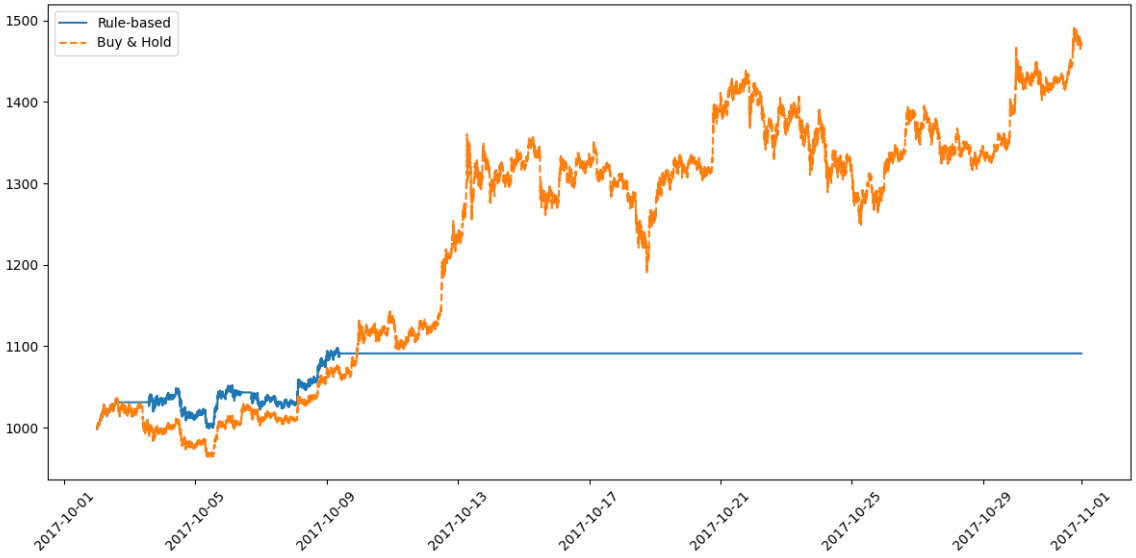
88) August



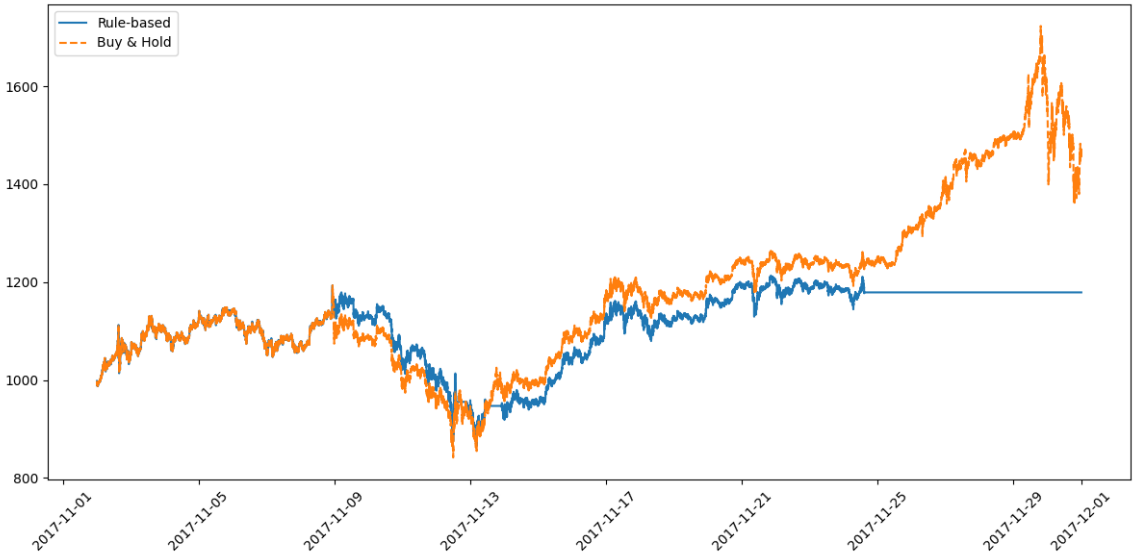
89) September



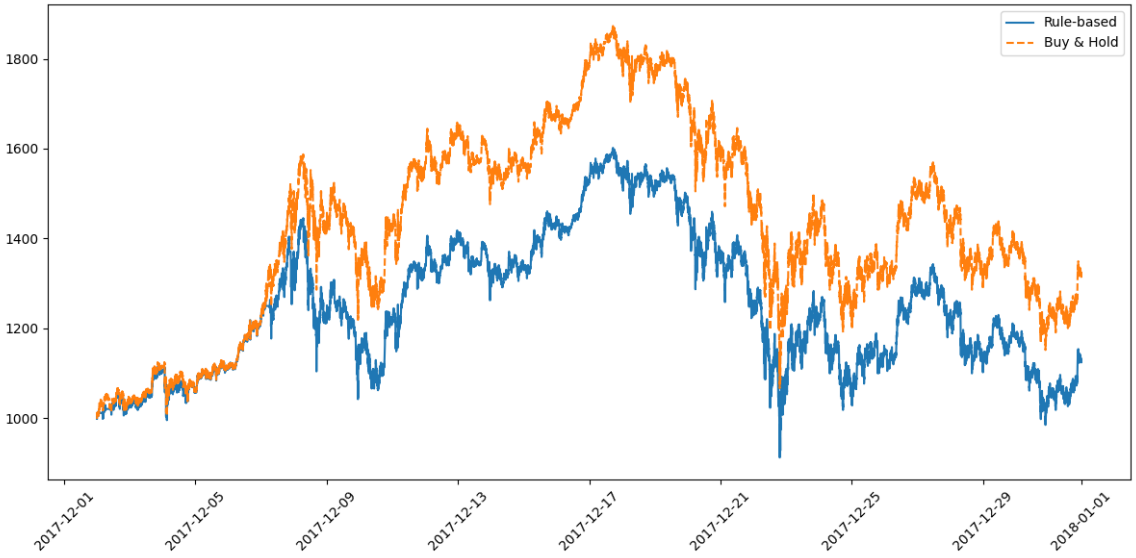
90) October



91) November

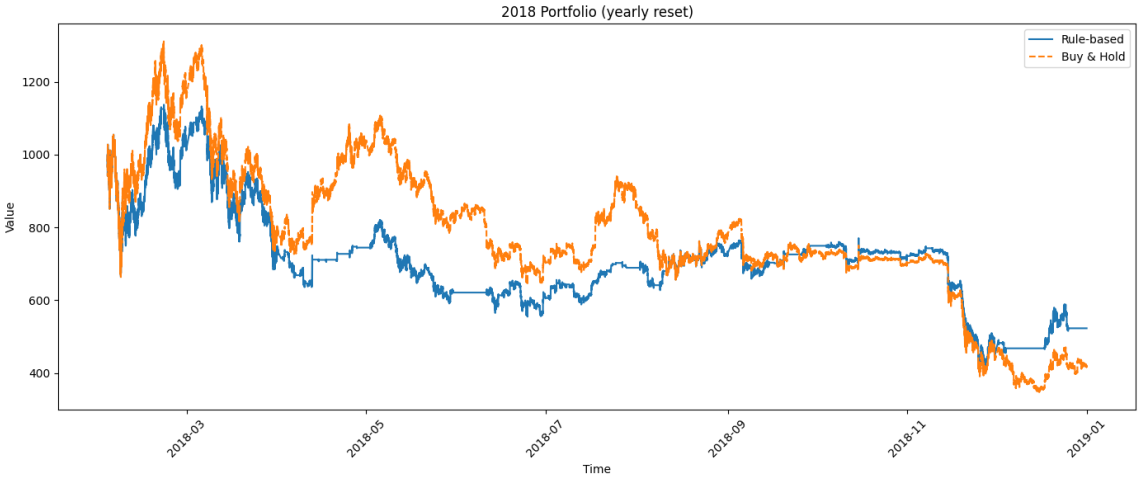


92) December

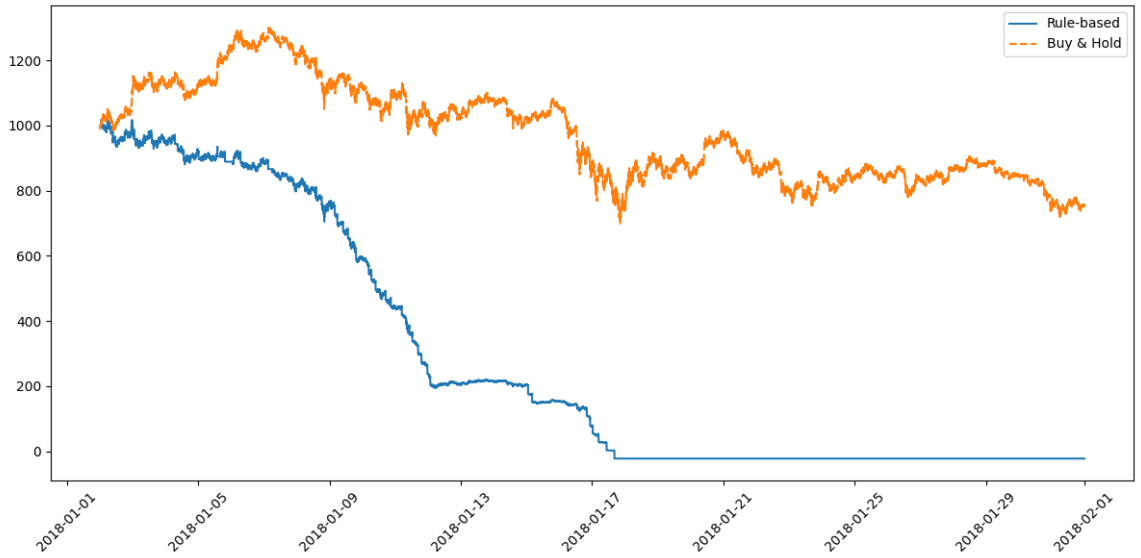


2018

93) Full Year

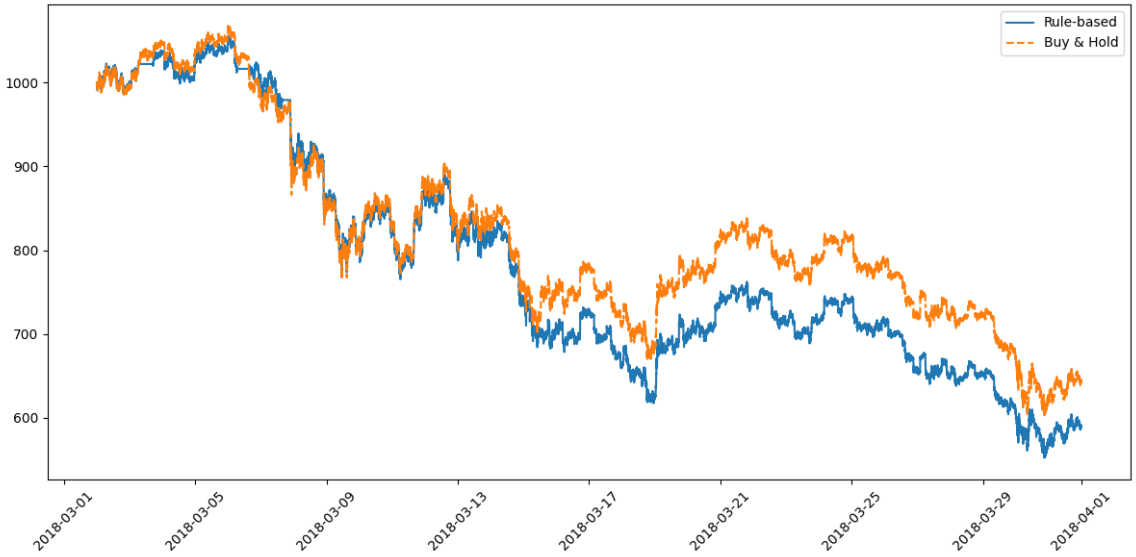


94) January

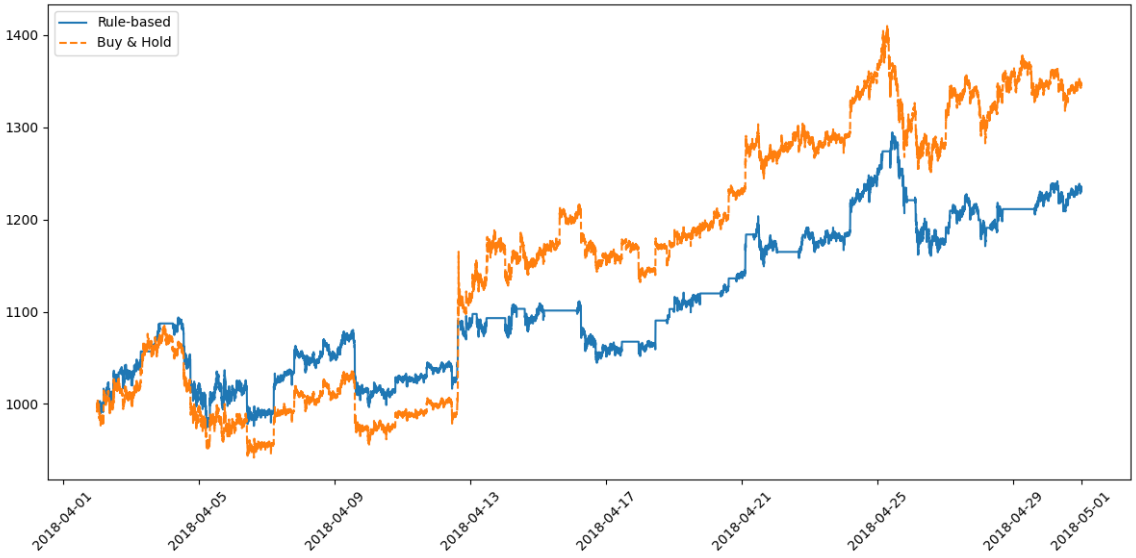


95) February

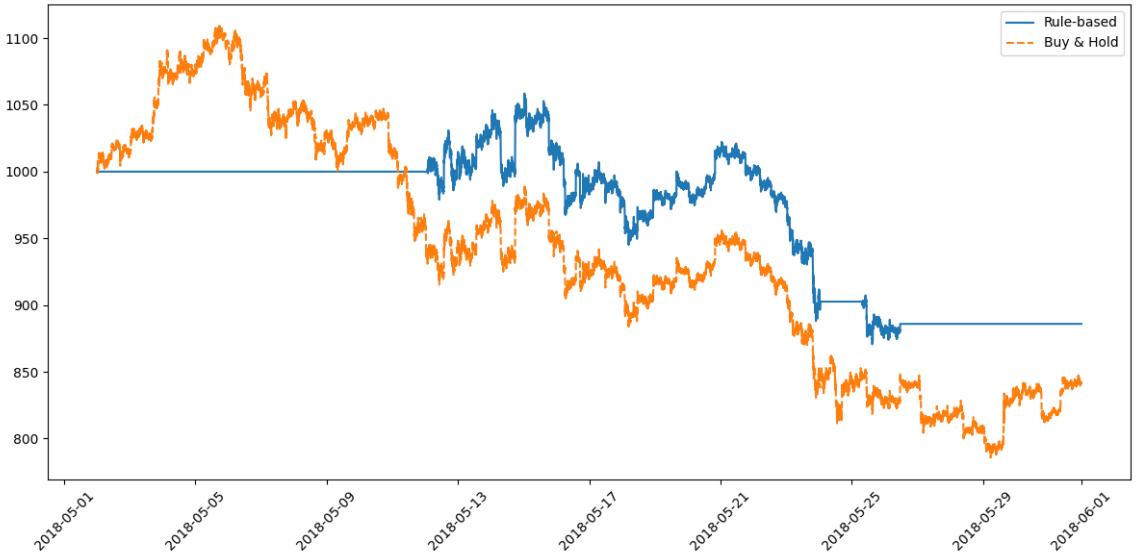
96) March



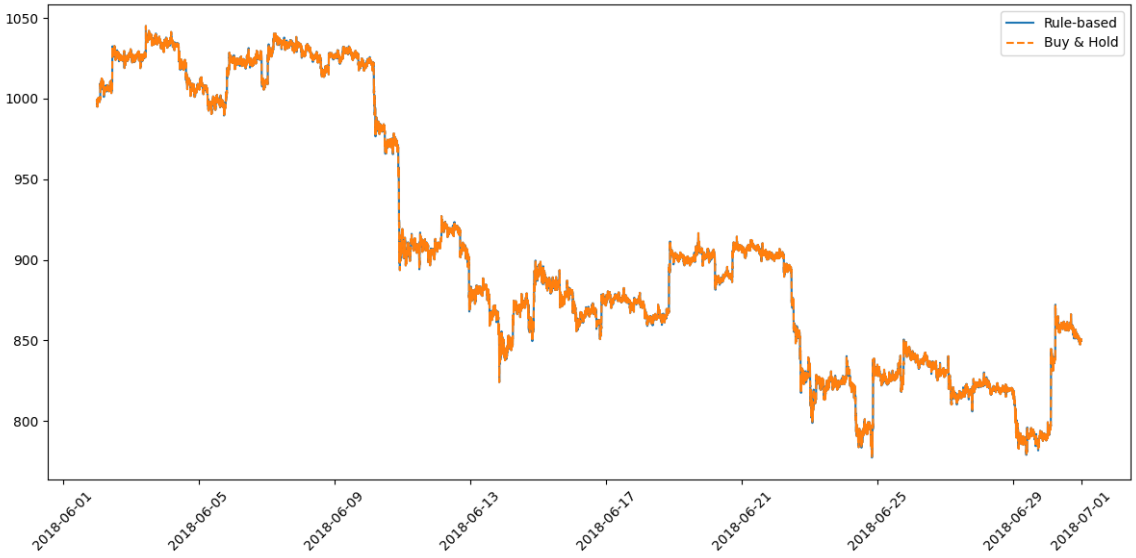
97) April



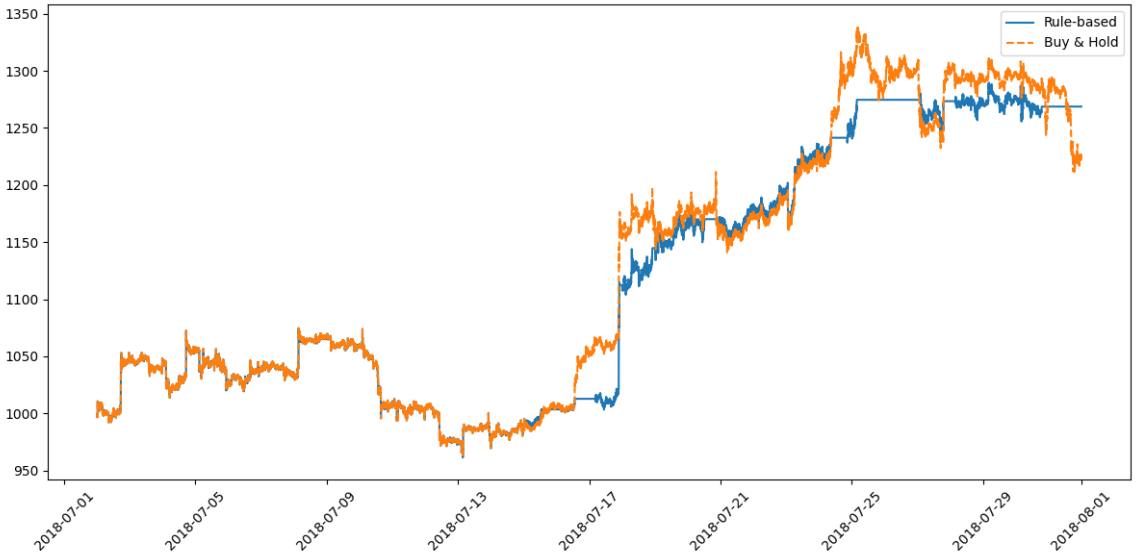
98) May



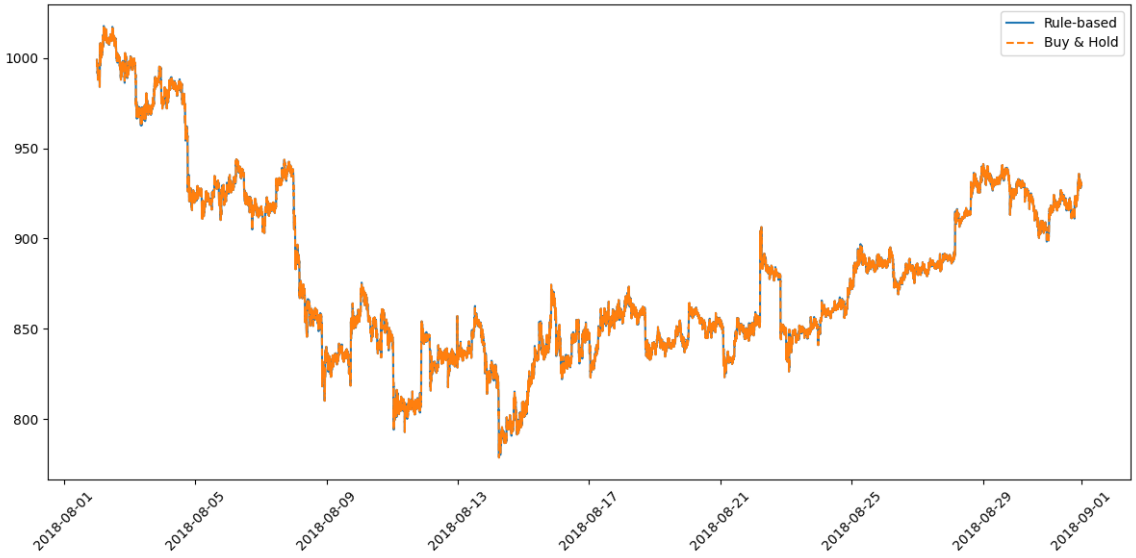
99) June



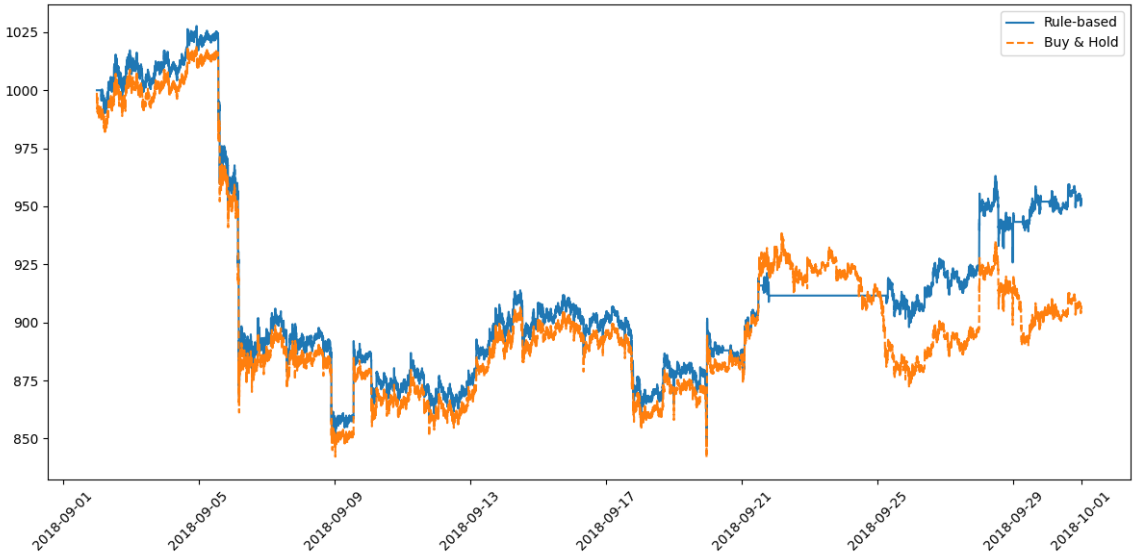
100) July



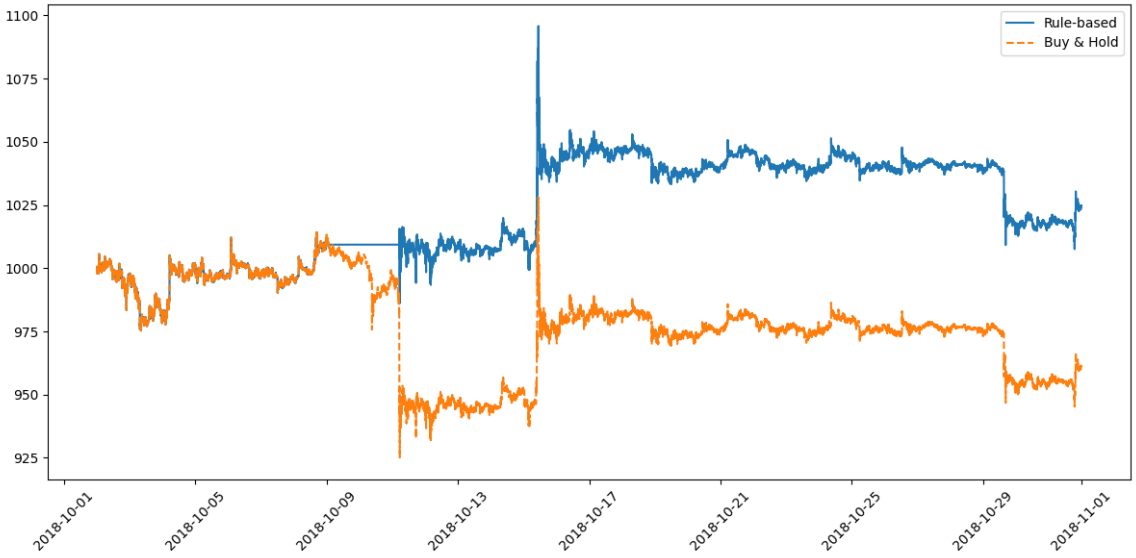
101) August



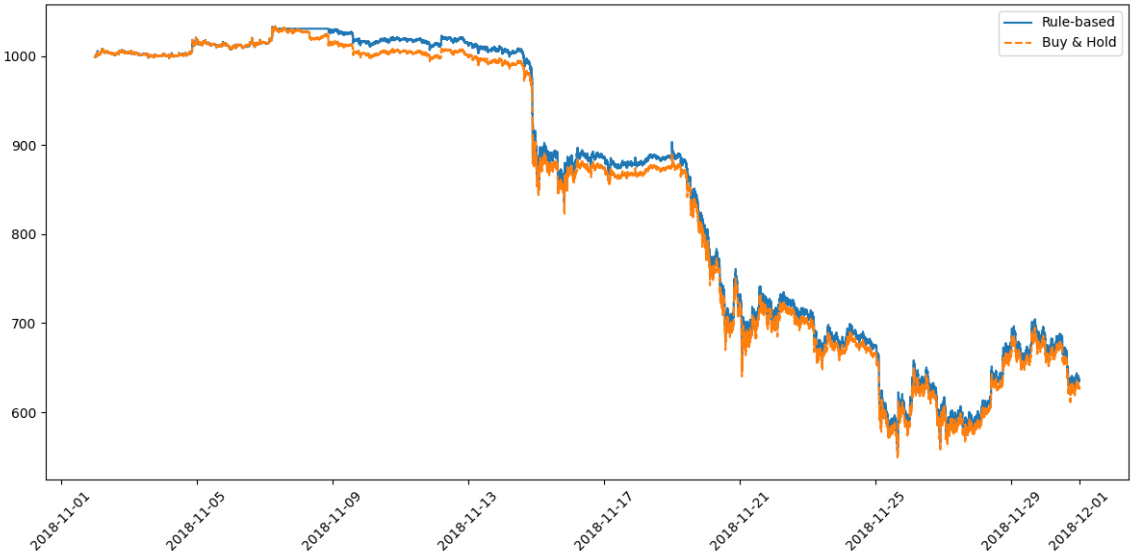
102) September



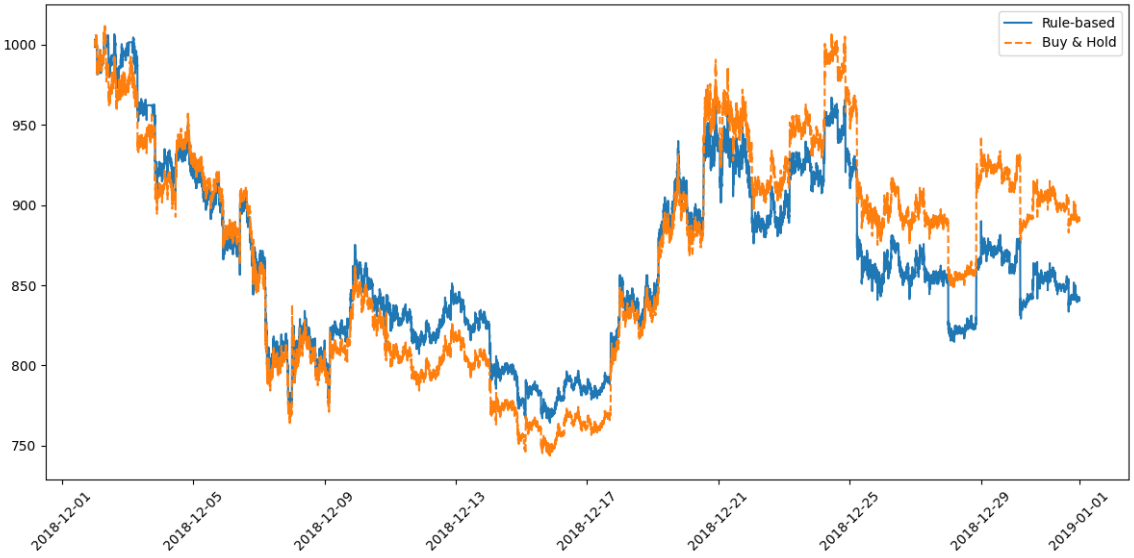
103) October



104) November

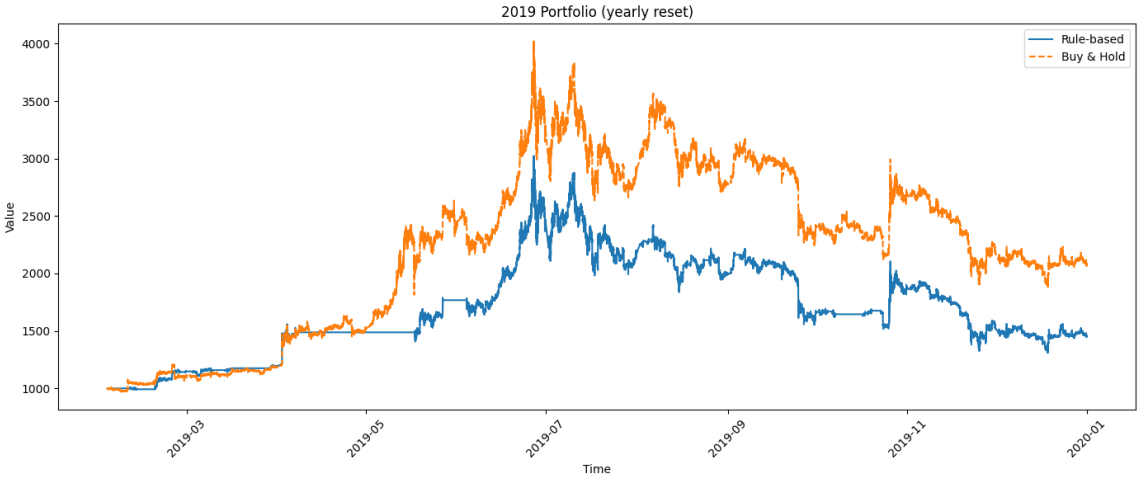


105) December

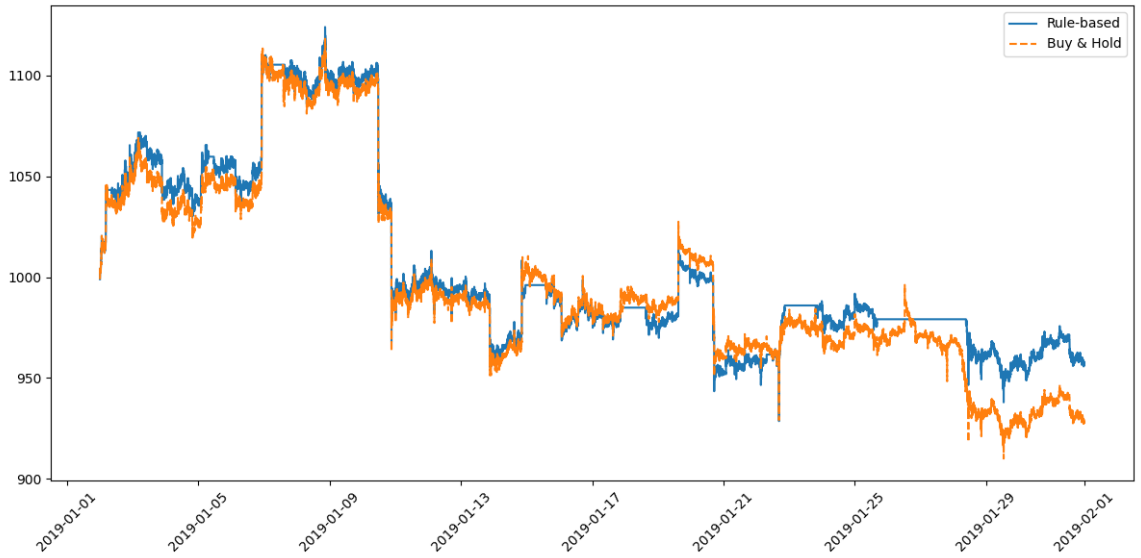


2019

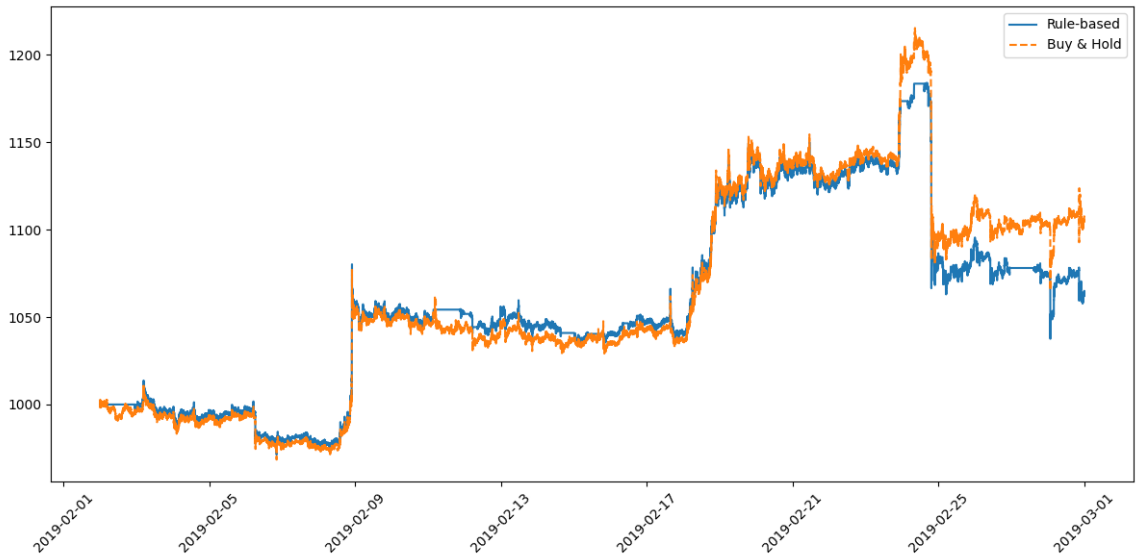
106) Full Year



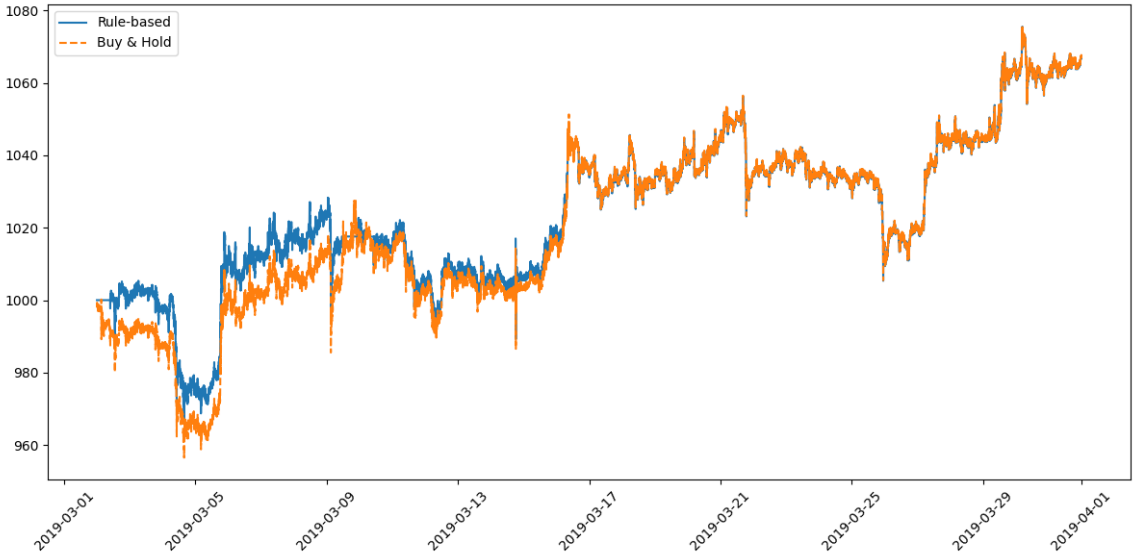
107) January



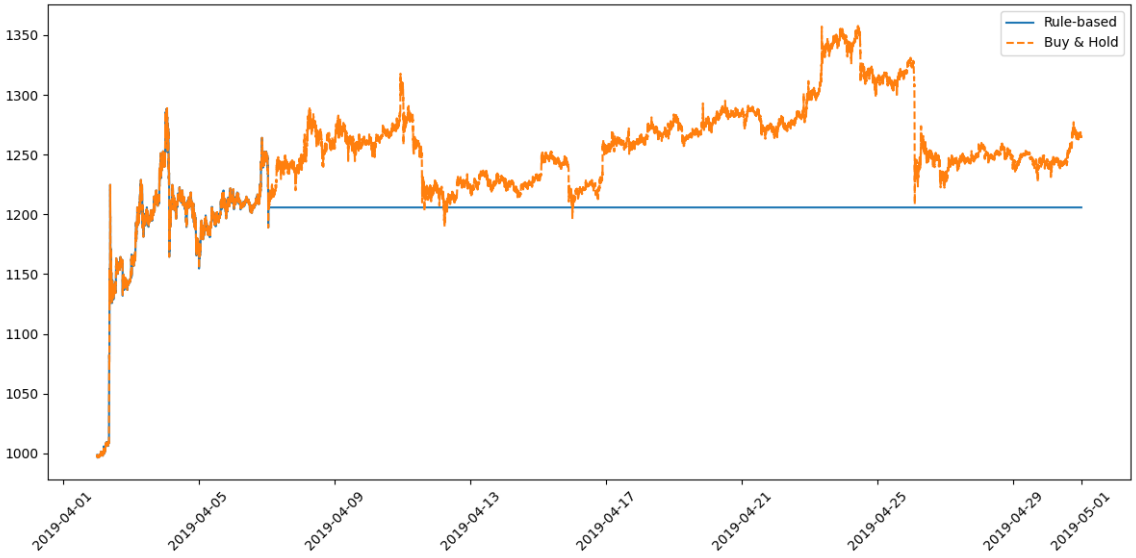
108) February



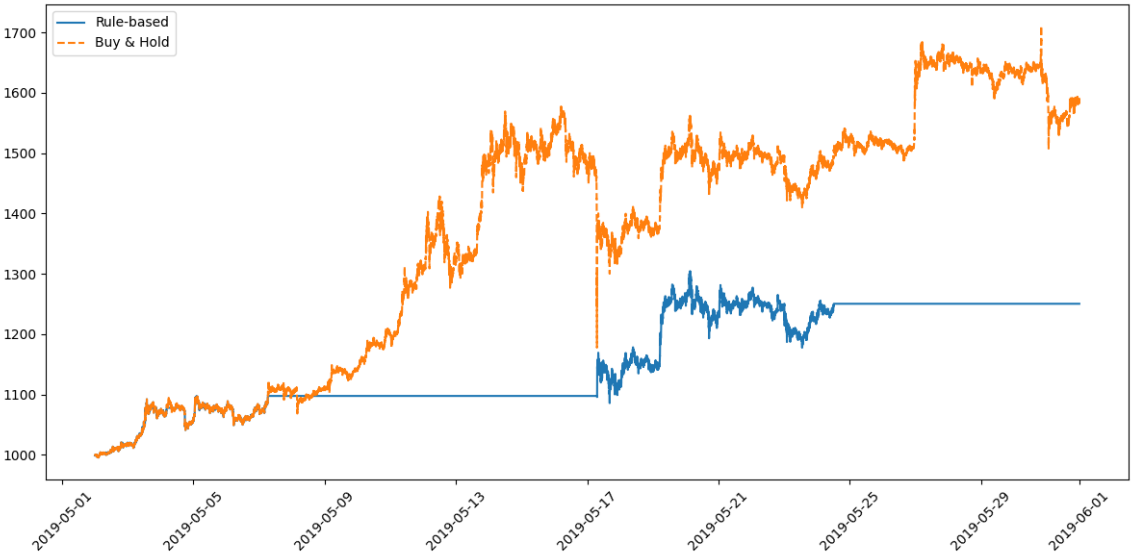
109) March



110) April

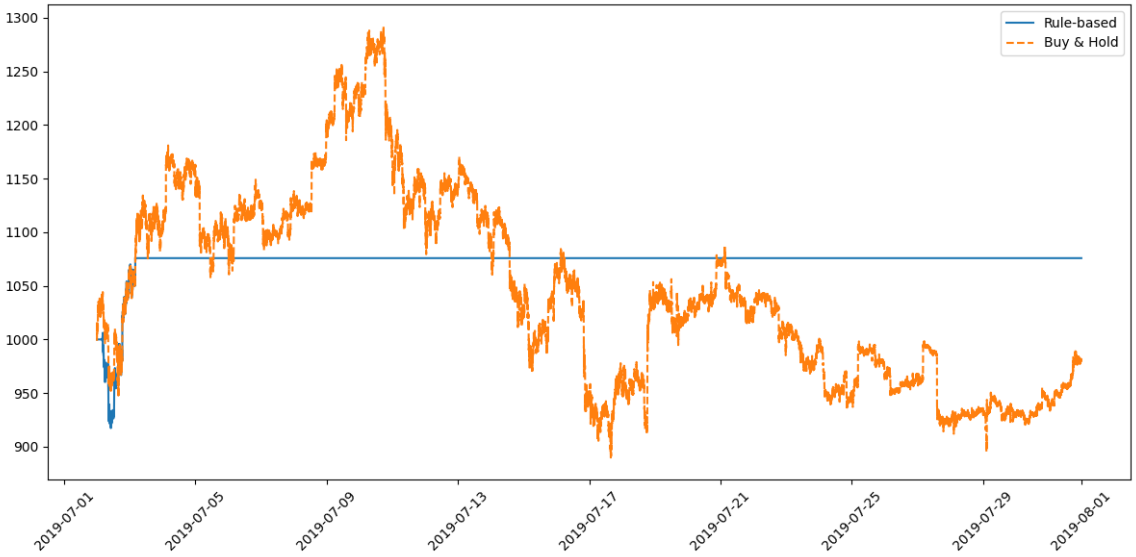


111) May



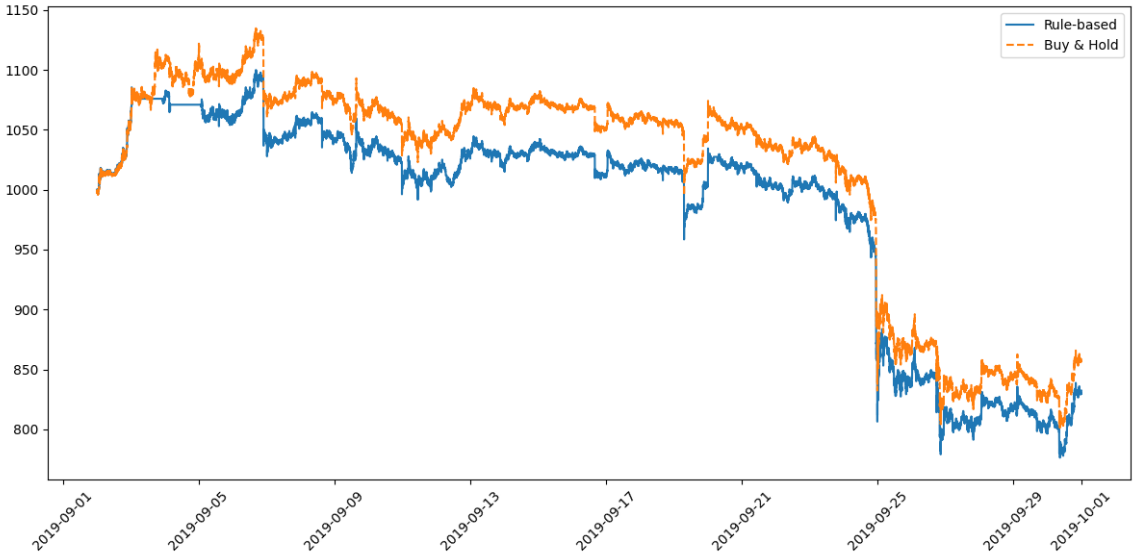
112) June

113) July

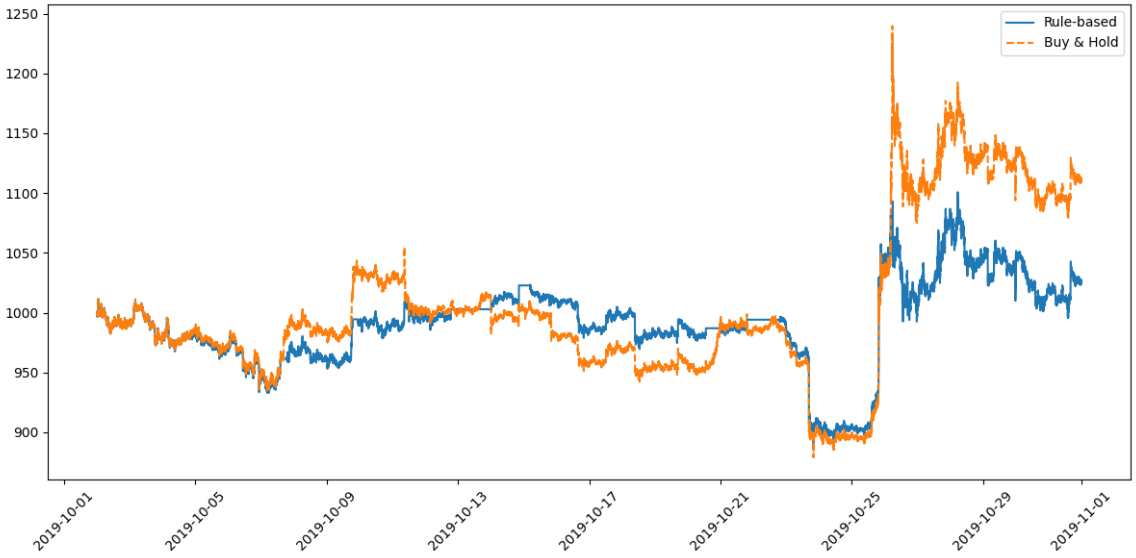


114) August

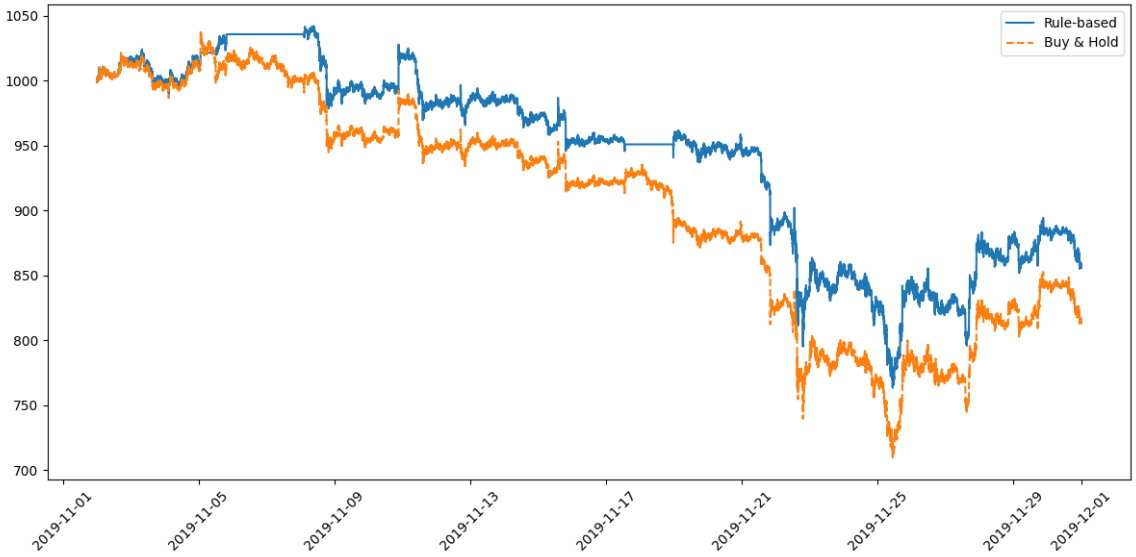
115) September



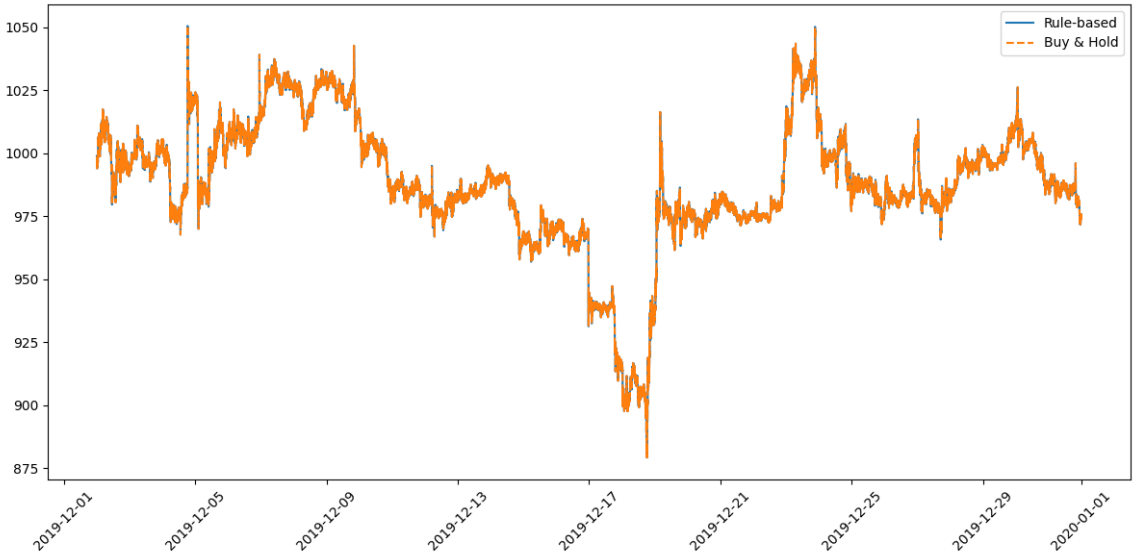
116) October



117) November

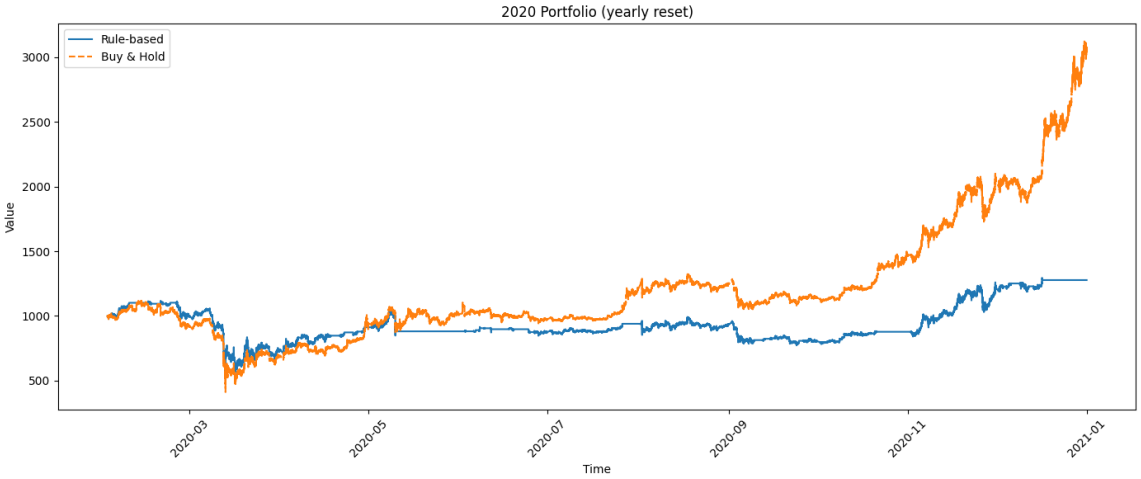


118) December

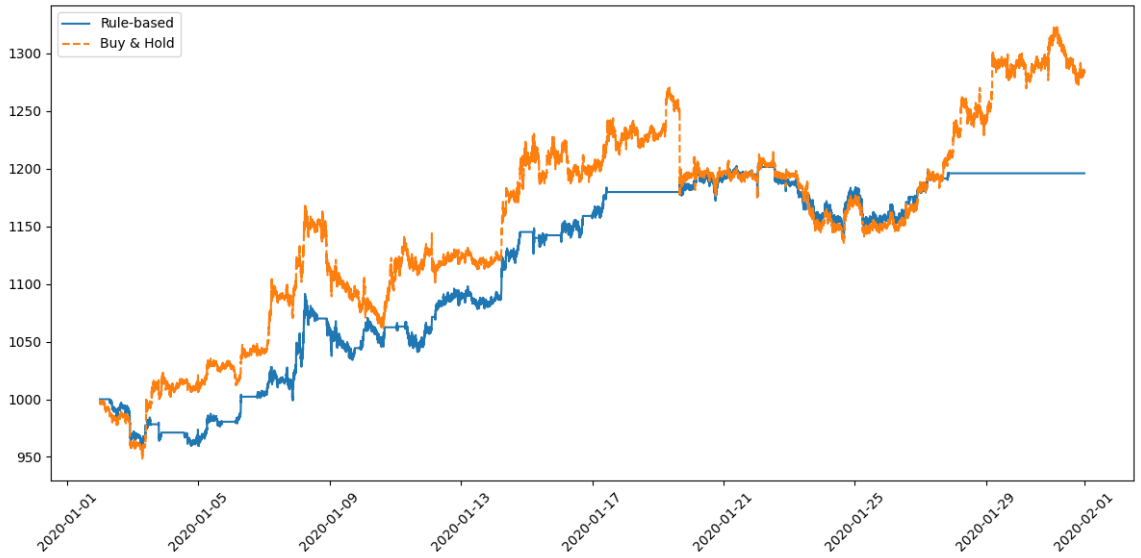


2020

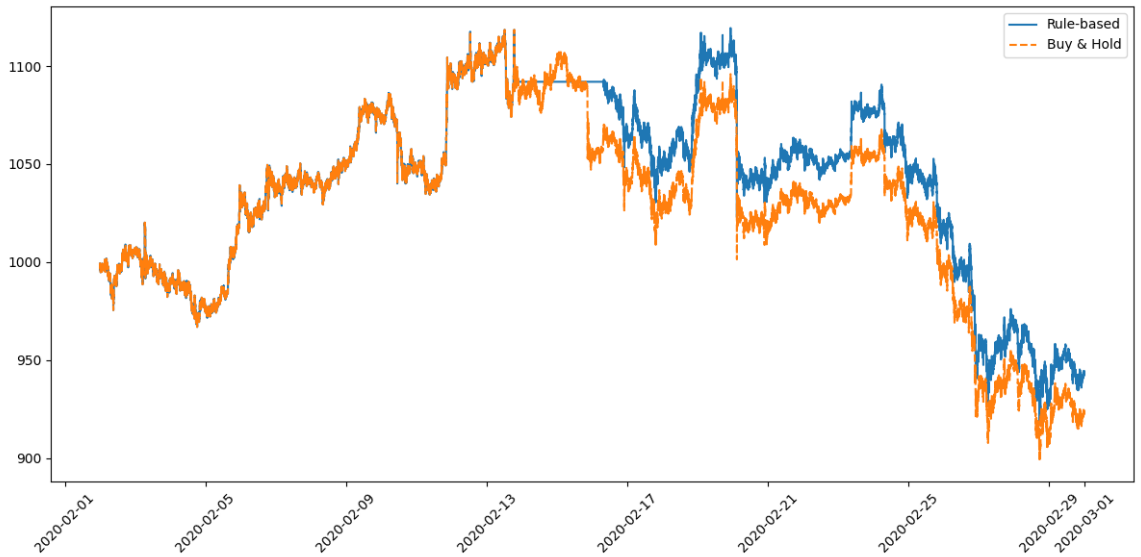
119) Full Year



120) January

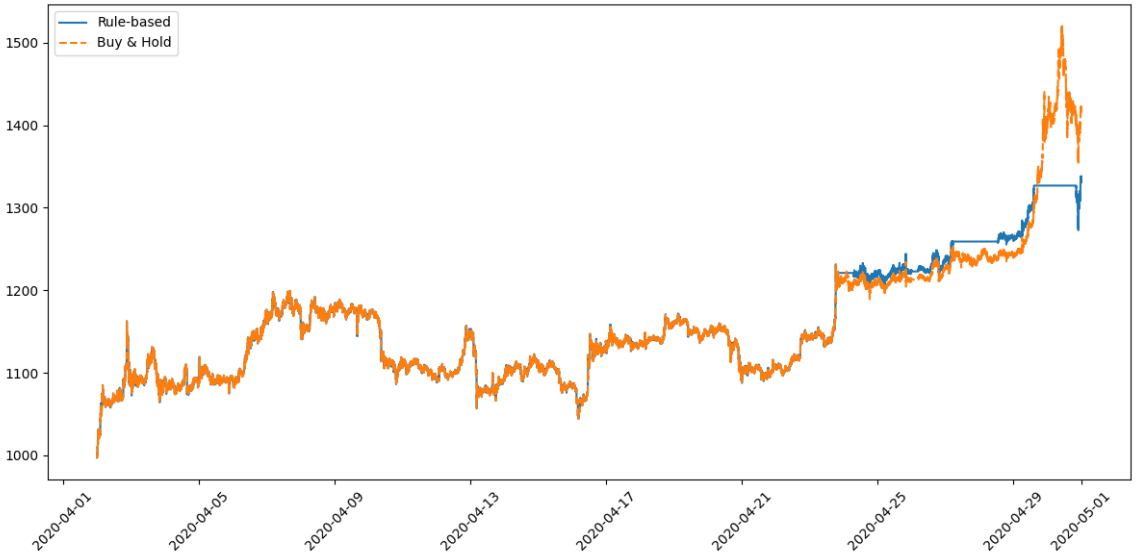


121) February

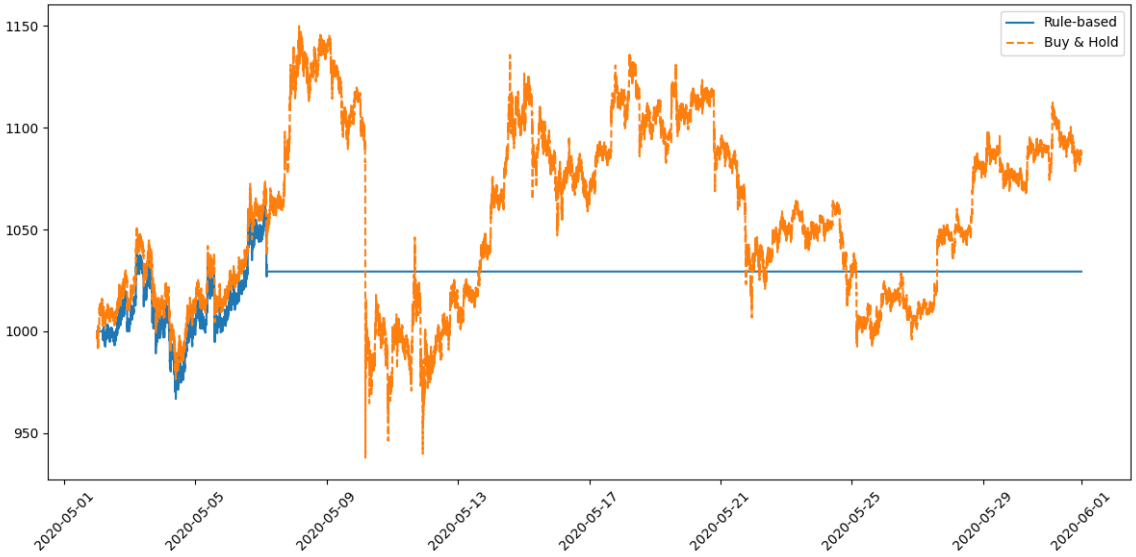


122) March

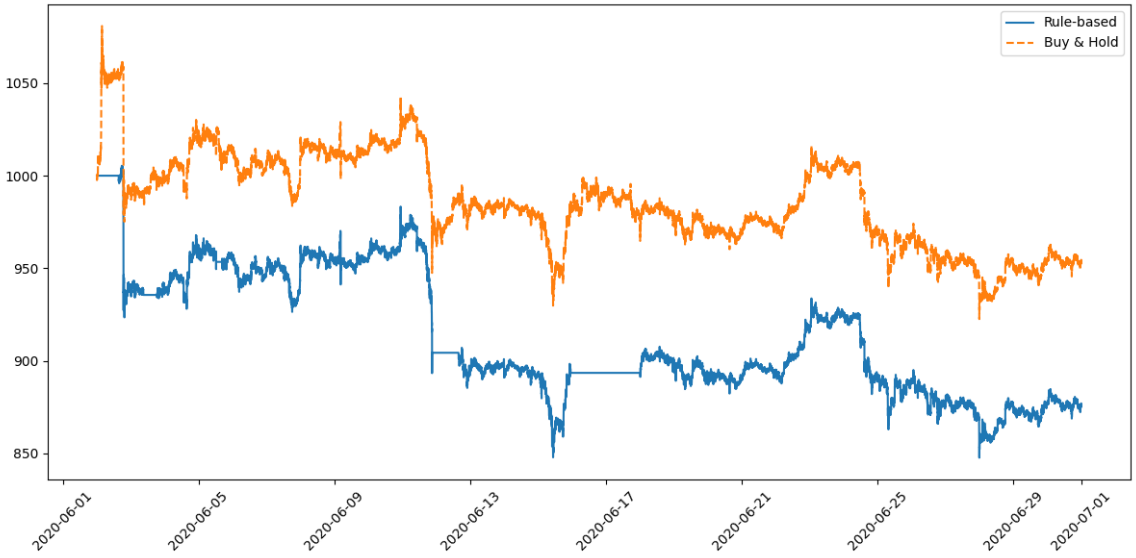
123) April



124) May



125) June

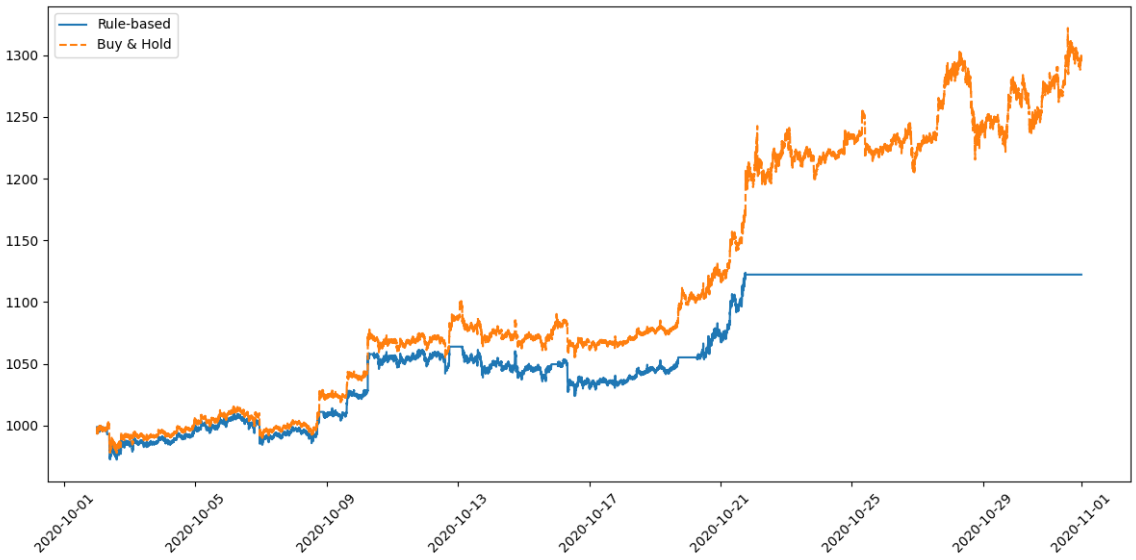


126) July

127) August

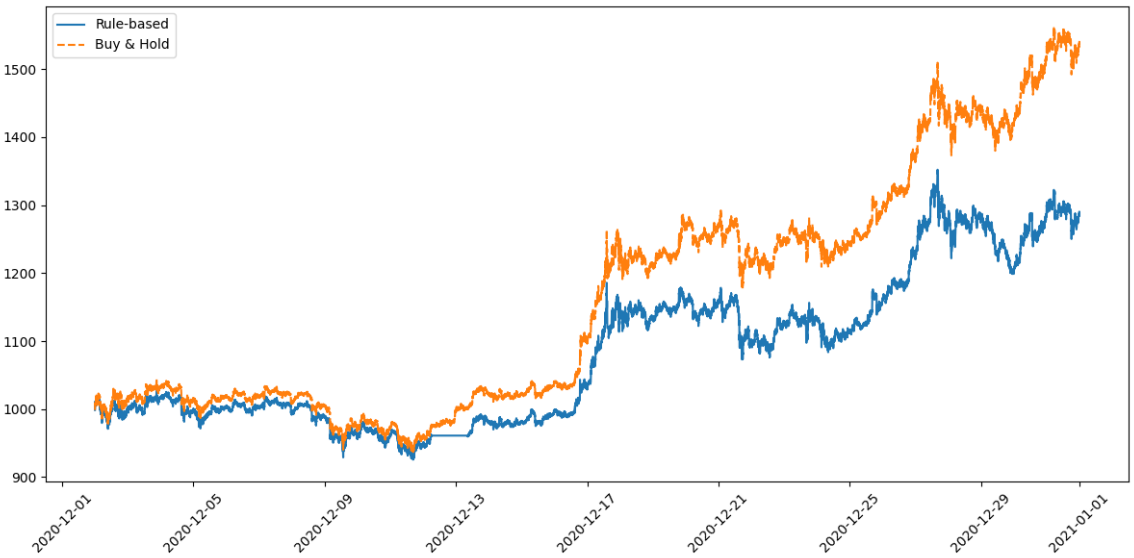
128) September

129) October



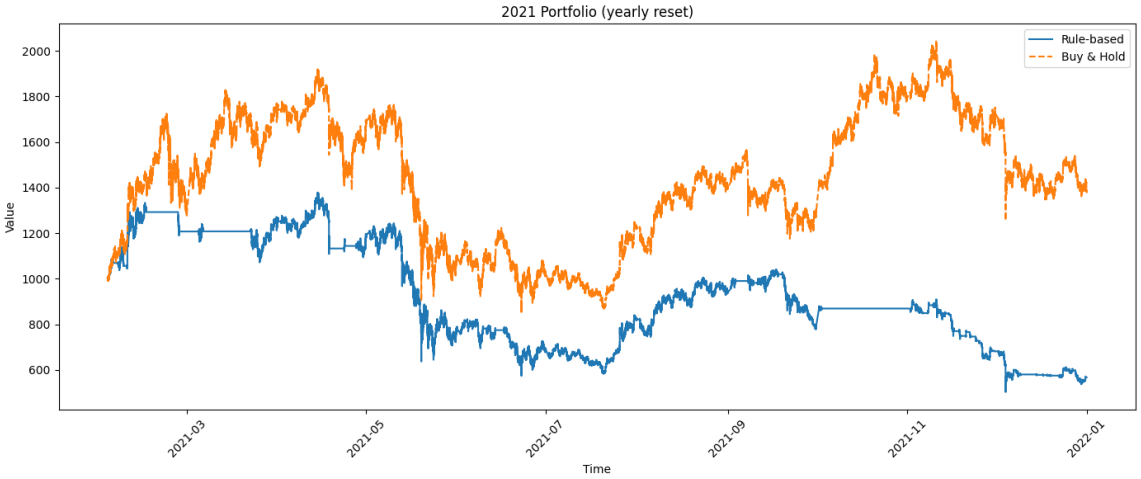
130) November

131) December



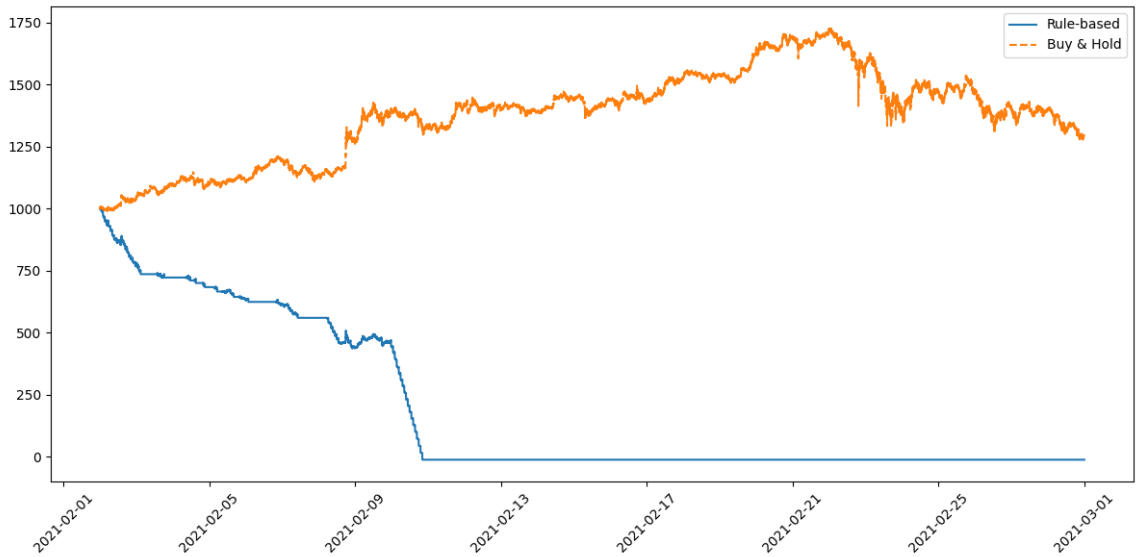
2021

132) Full Year

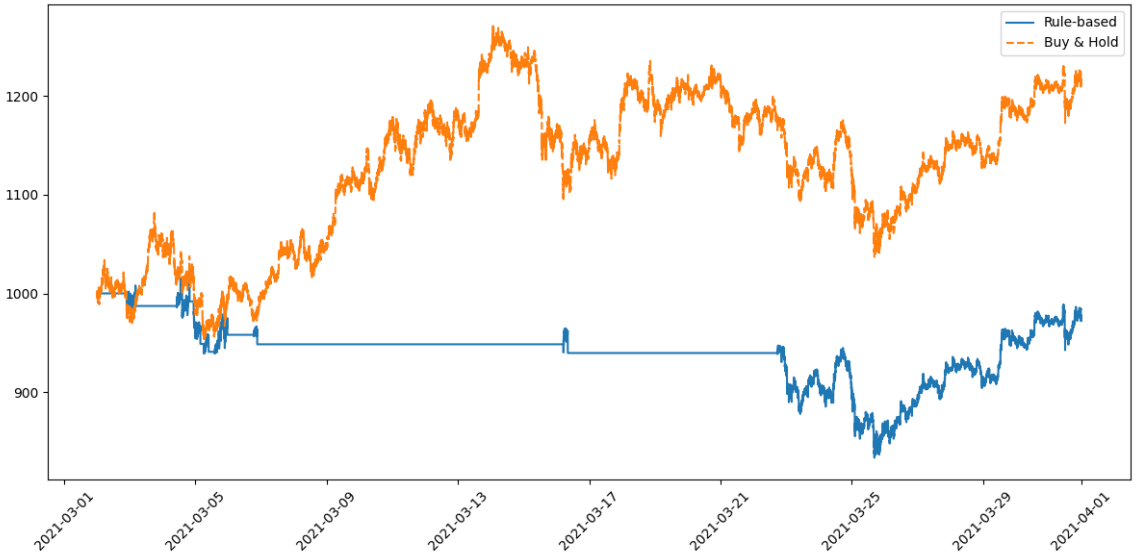


133) January

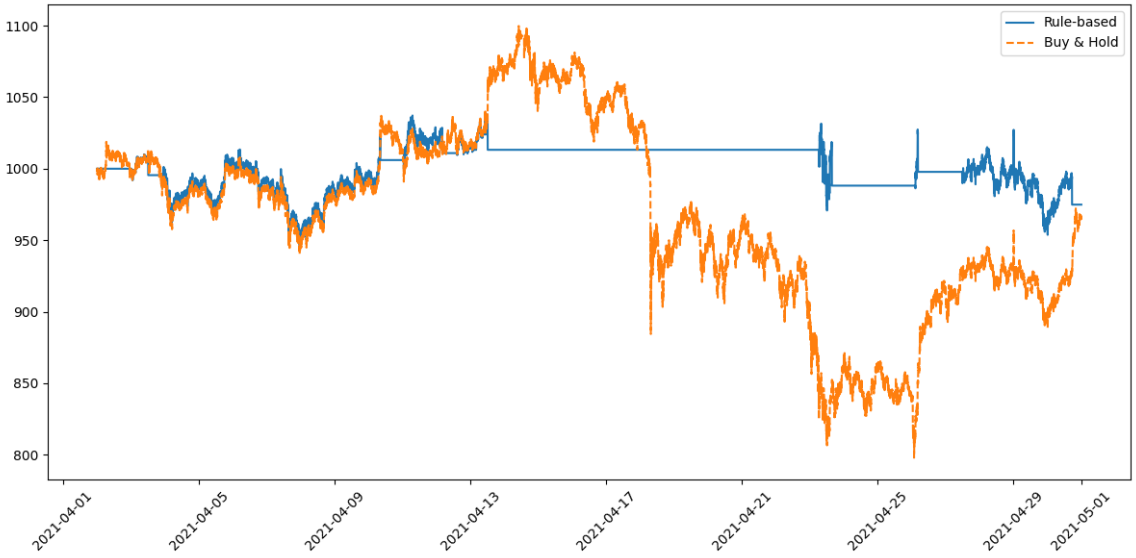
134) February



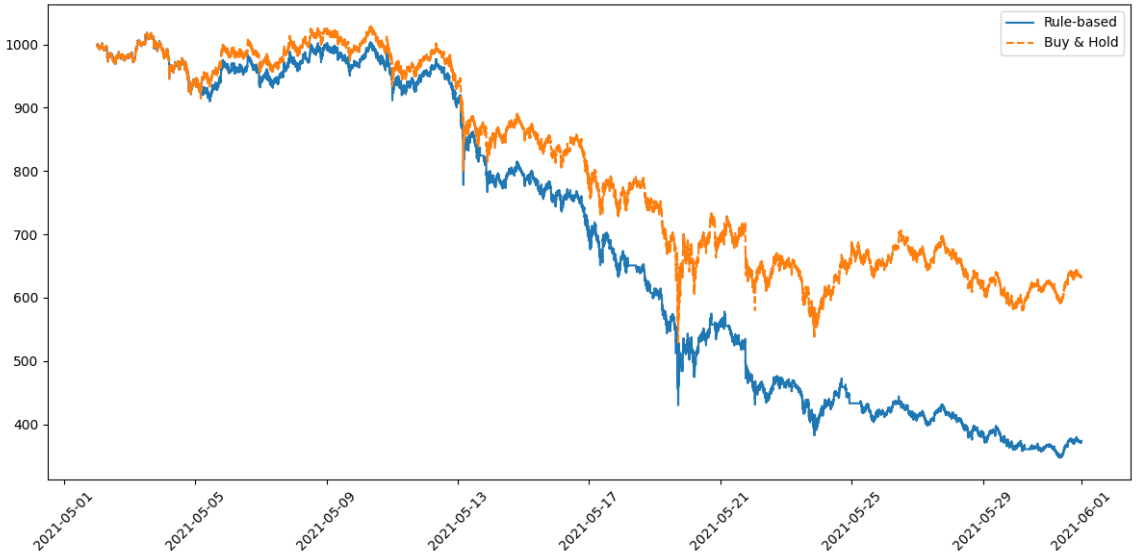
135) March



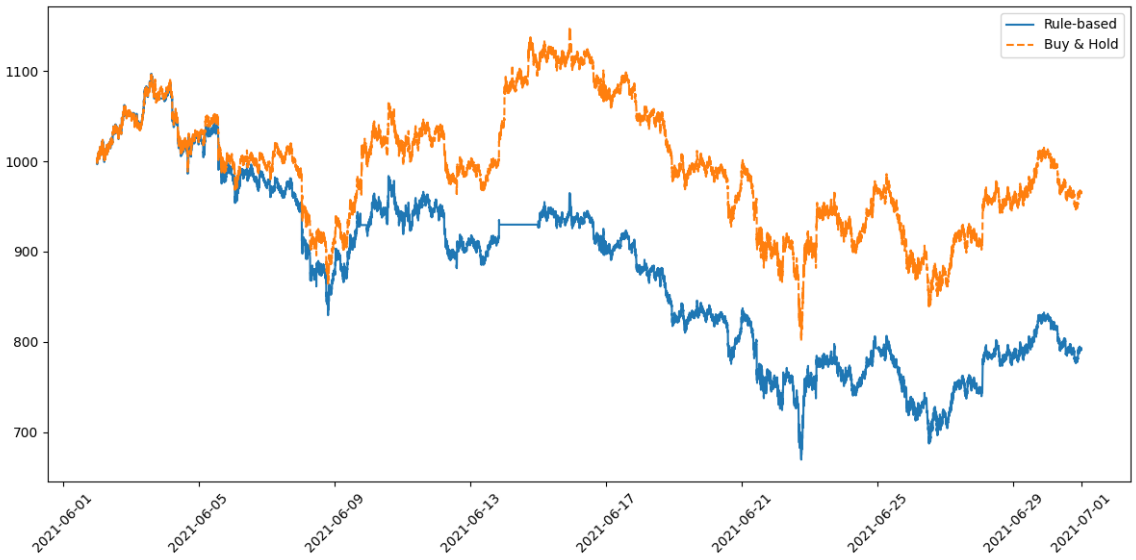
136) April



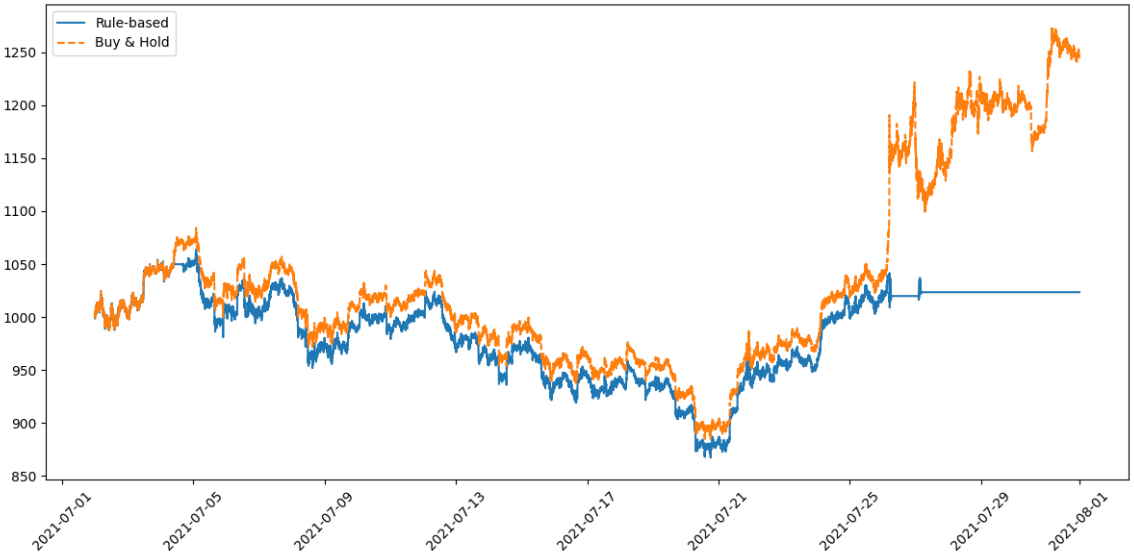
137) May



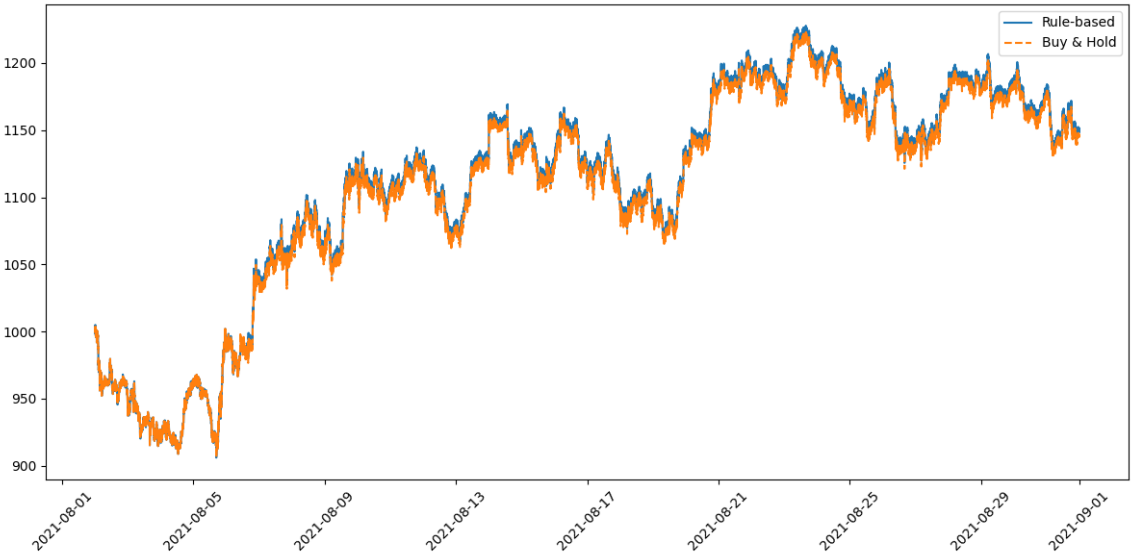
138) June



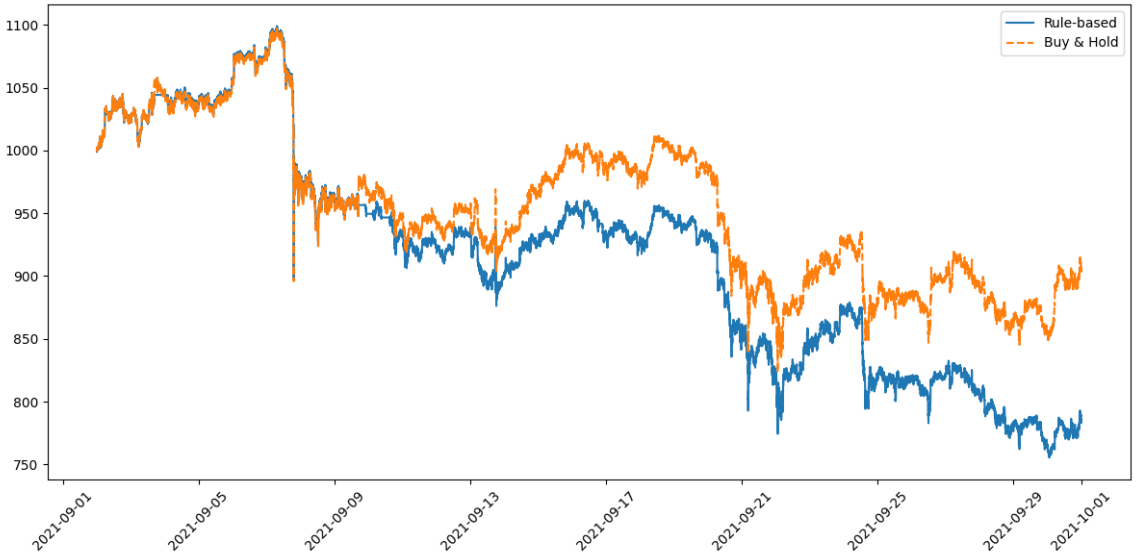
139) July



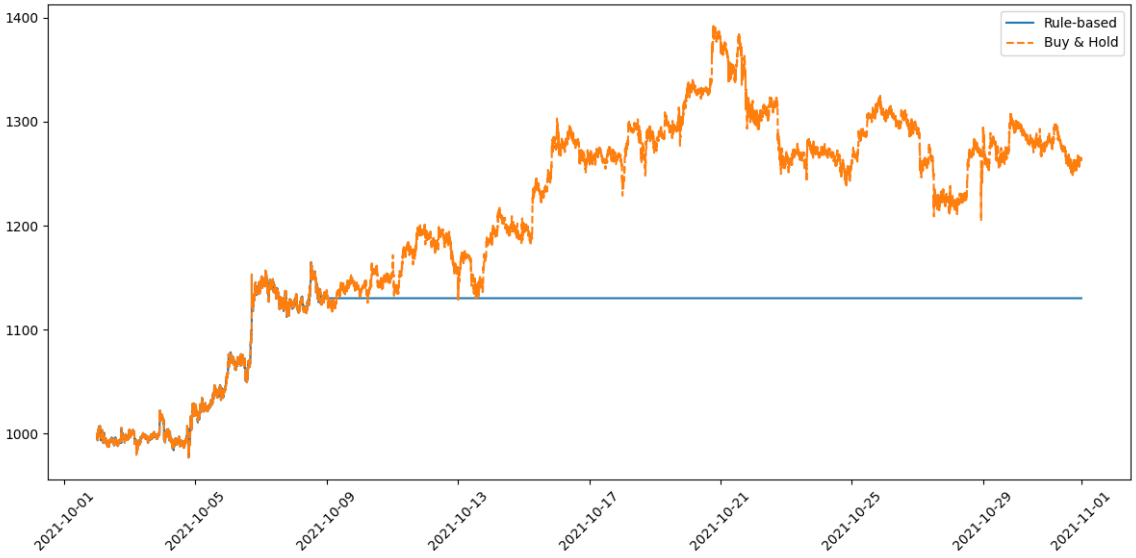
140) August



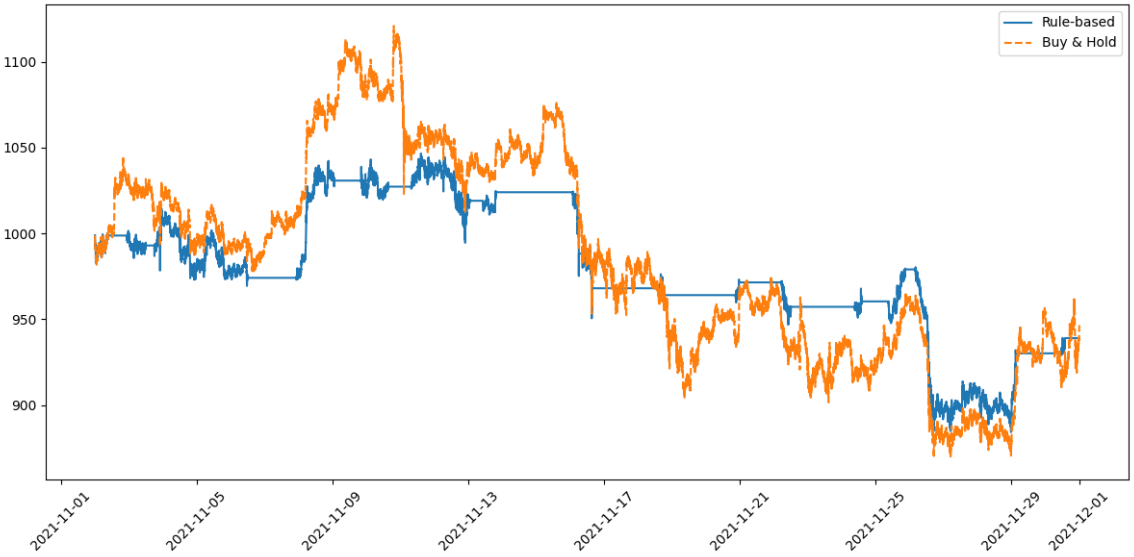
141) September



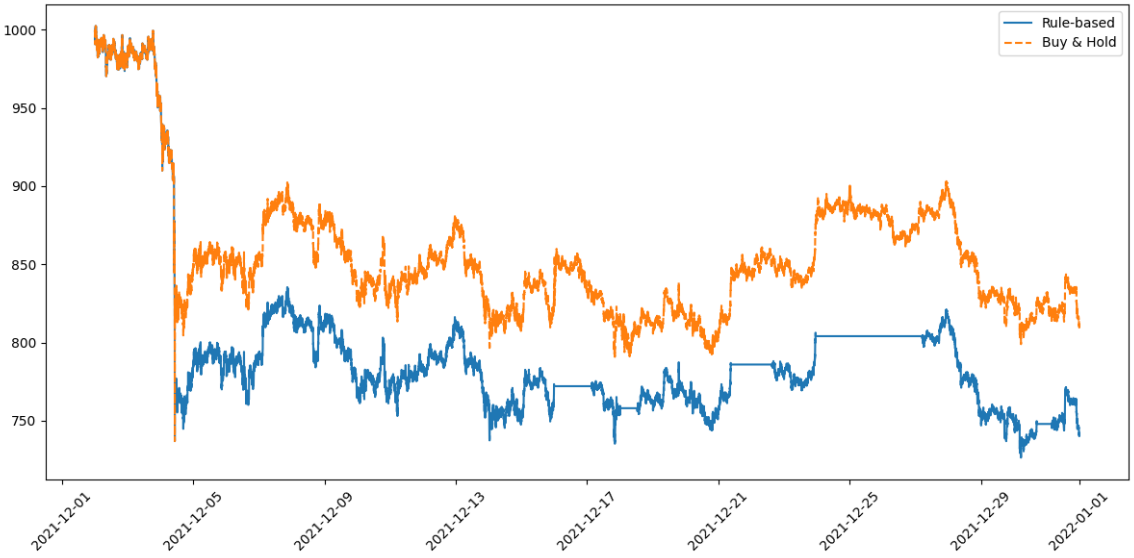
142) October



143) November

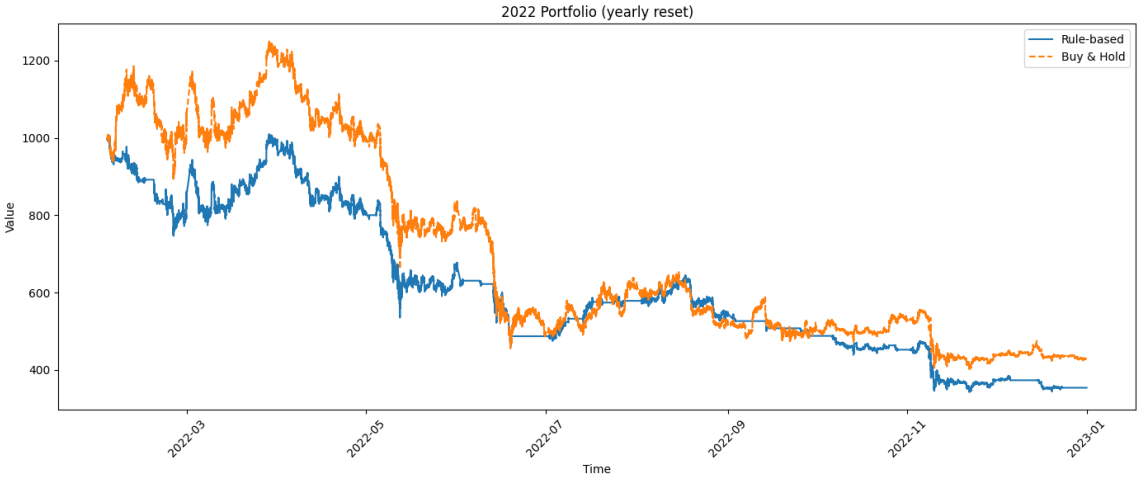


144) December

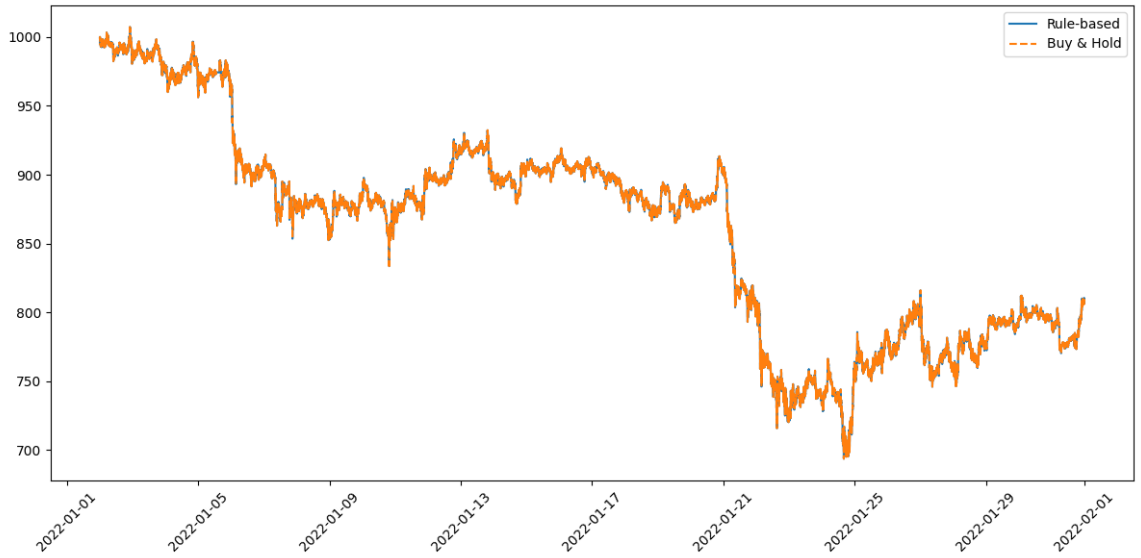


2022

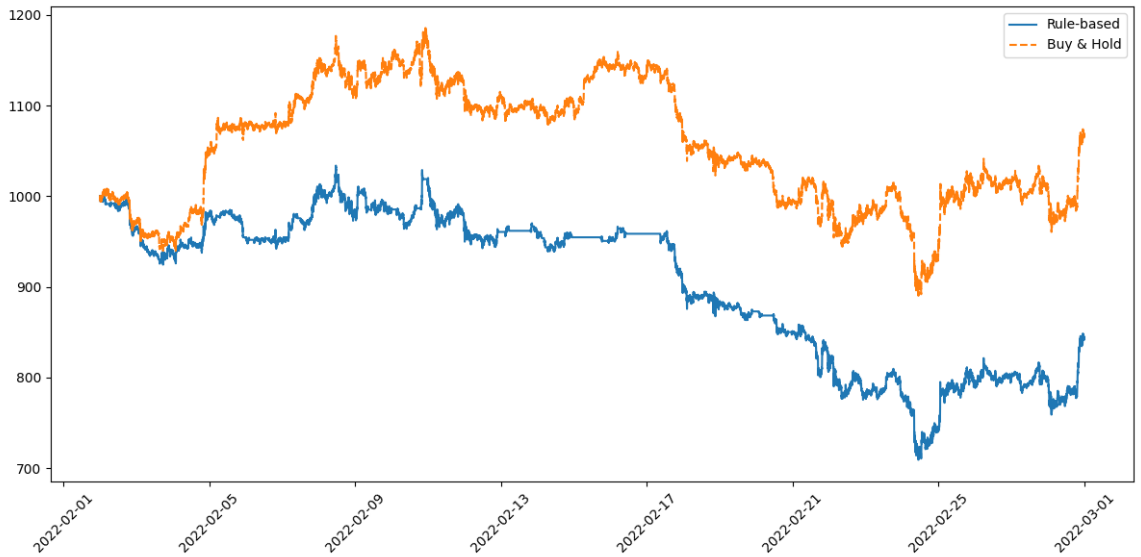
145) Full Year



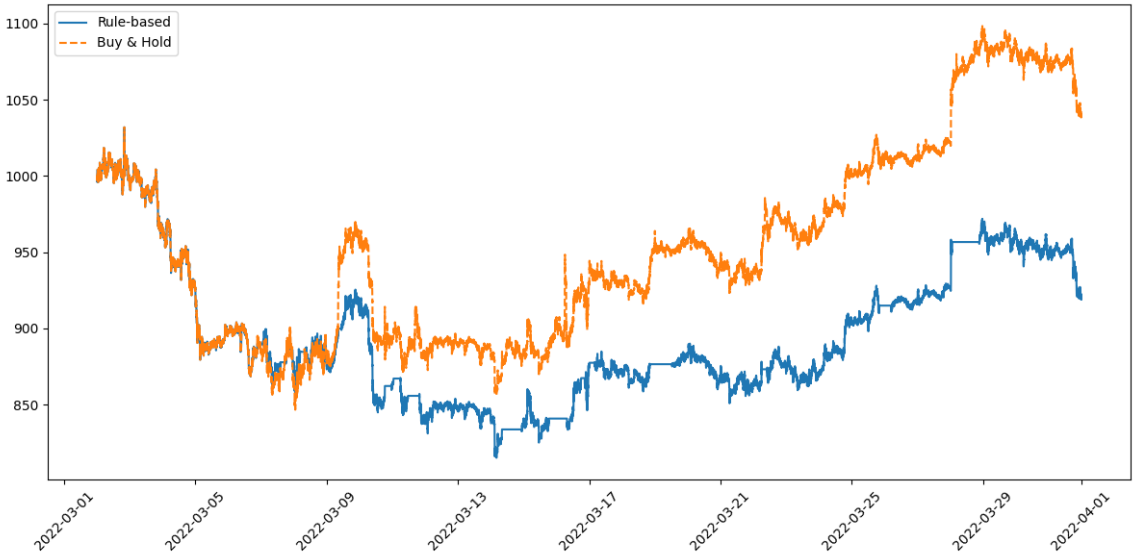
146) January



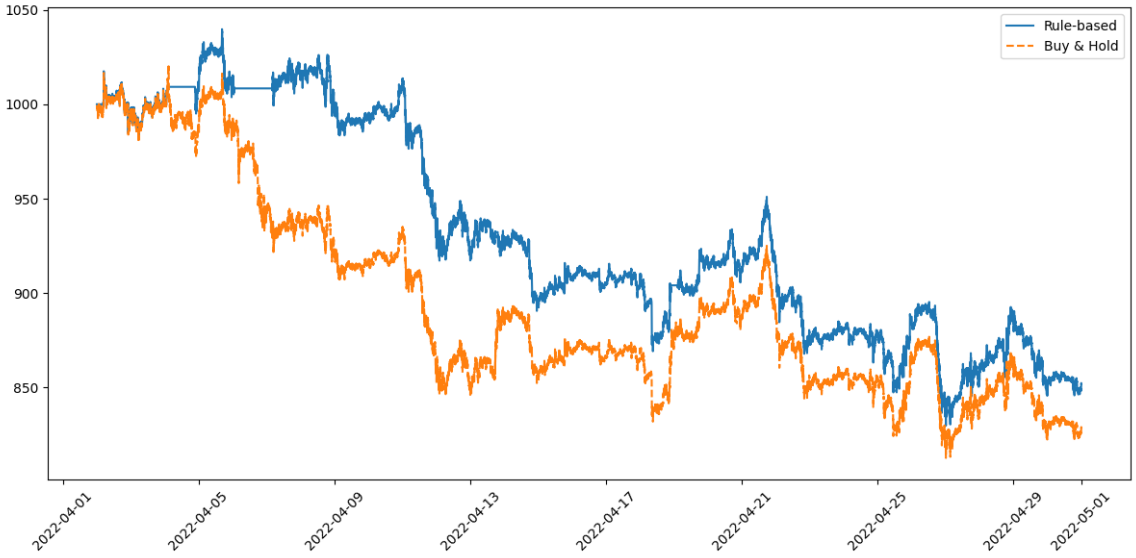
147) February



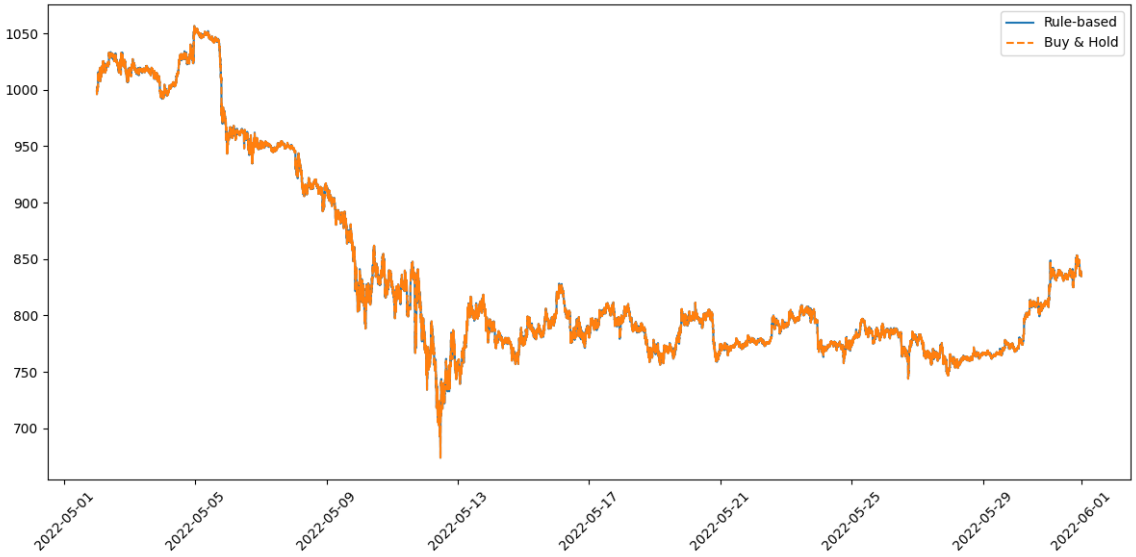
148) March



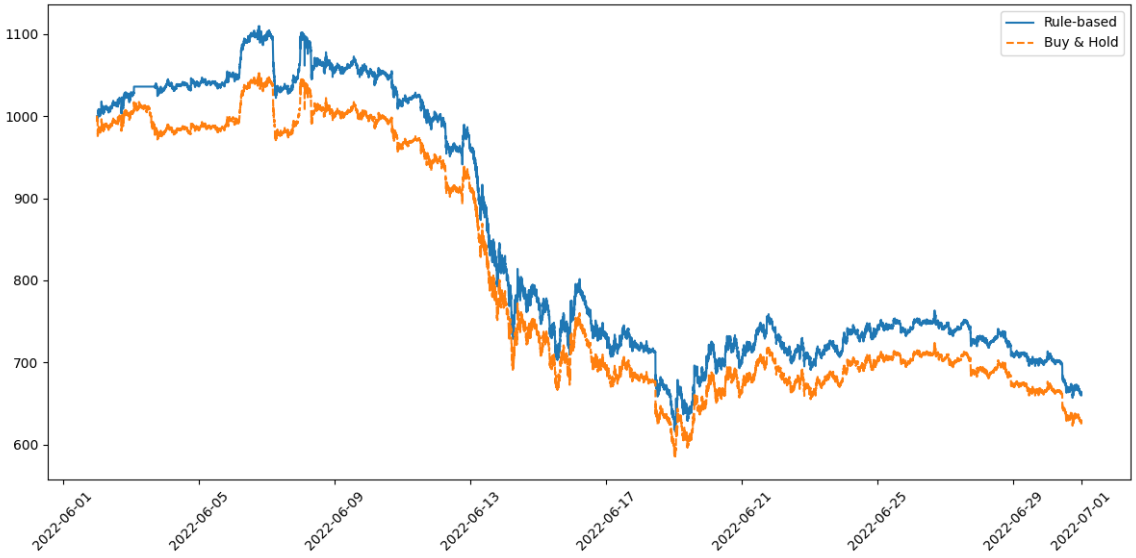
149) April



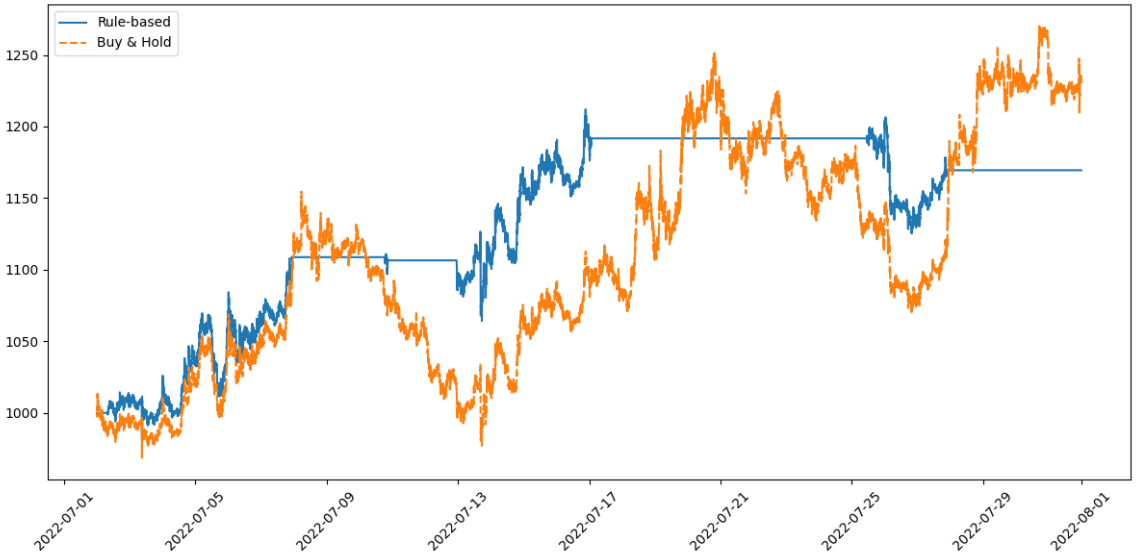
150) May



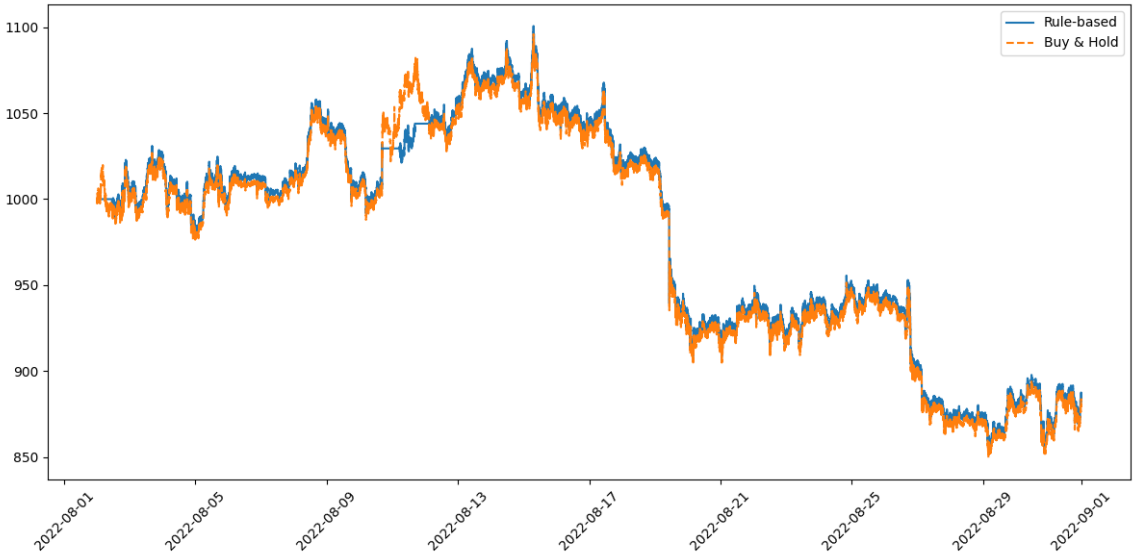
151) June



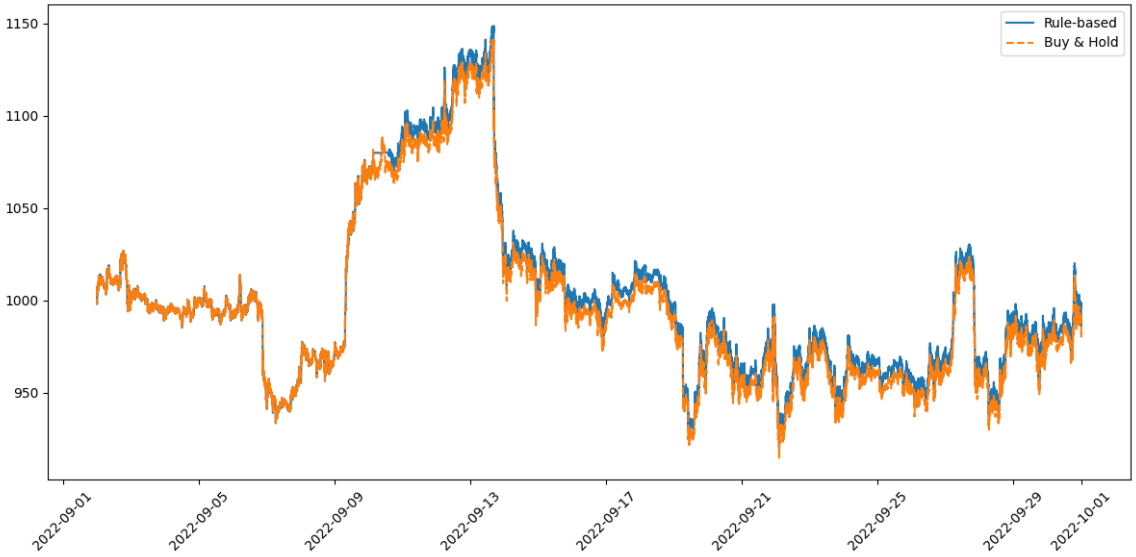
152) July



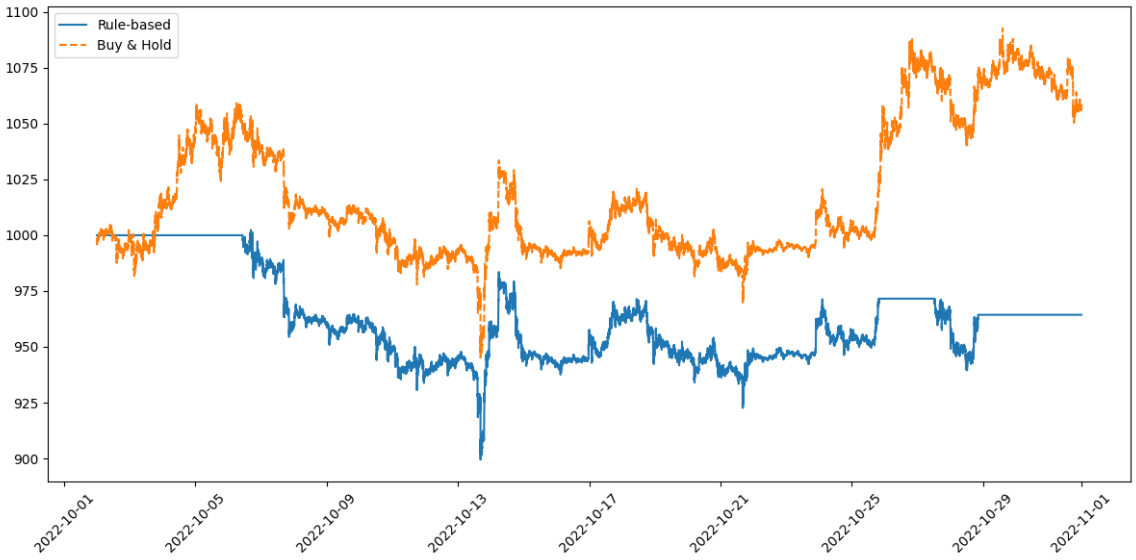
153) August



154) September

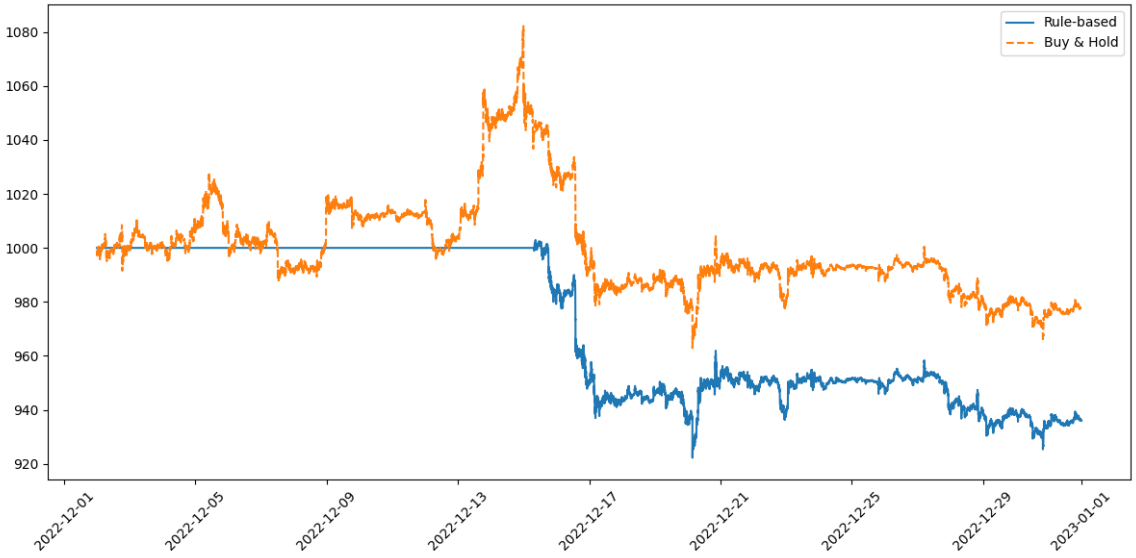


155) October



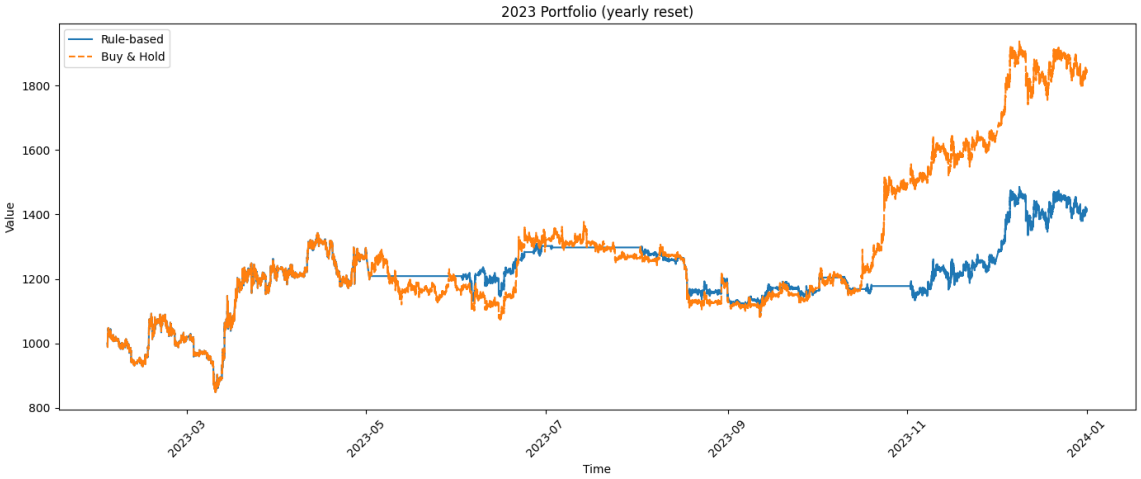
156) November

157) December

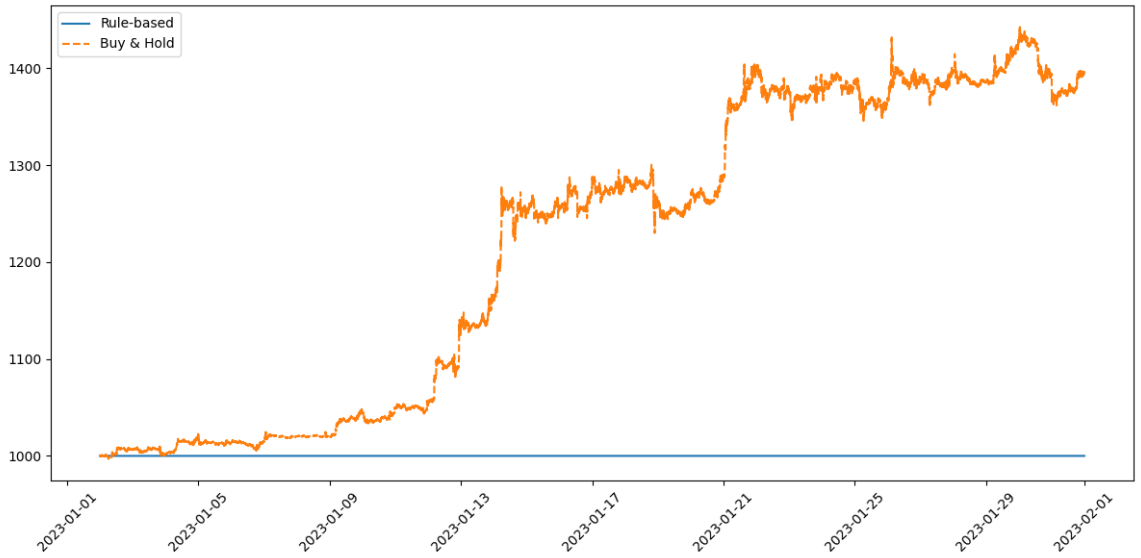


2023

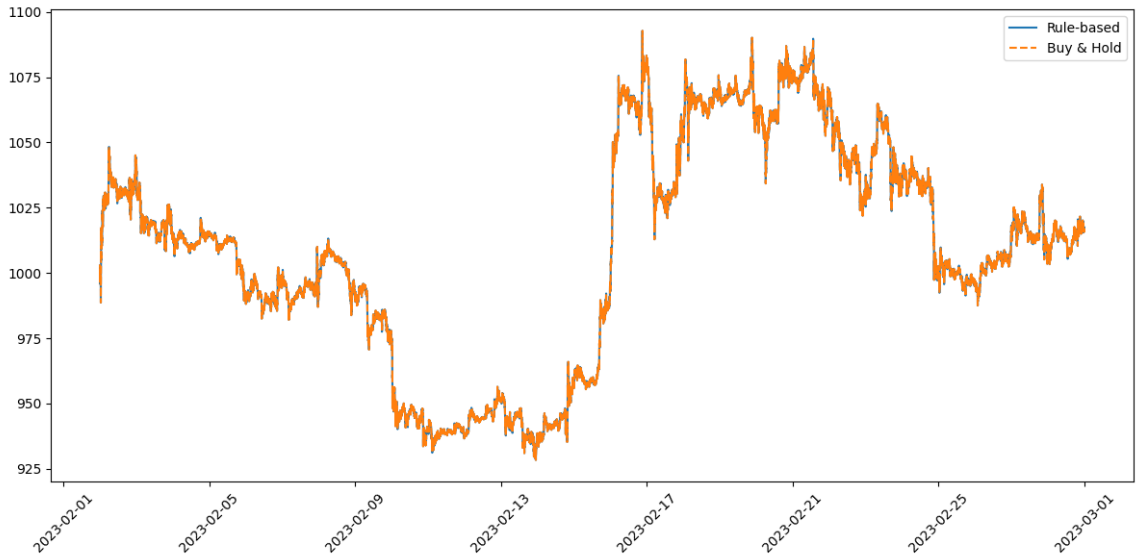
158) Full Year



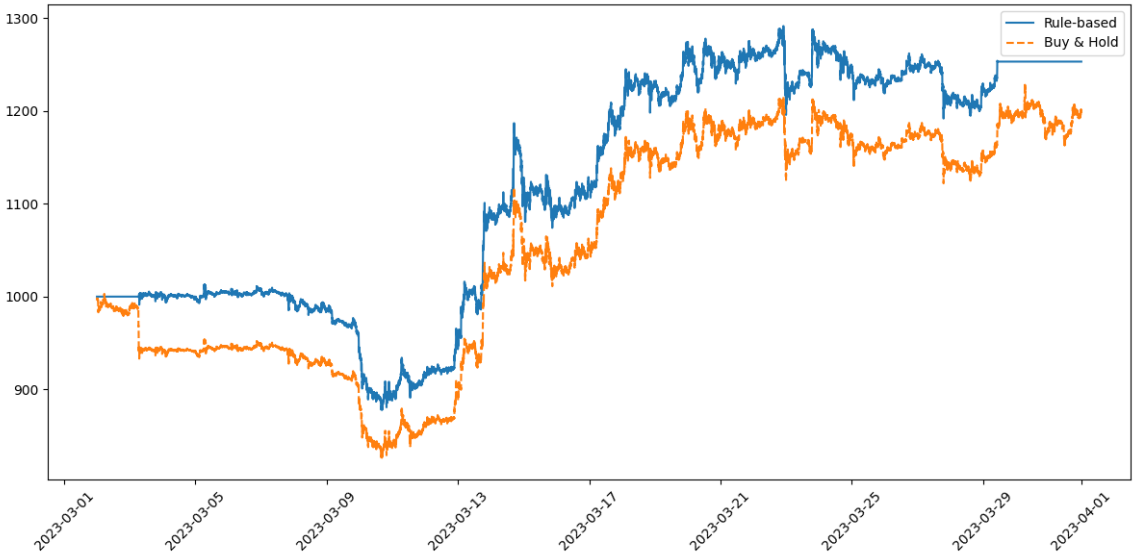
159) January



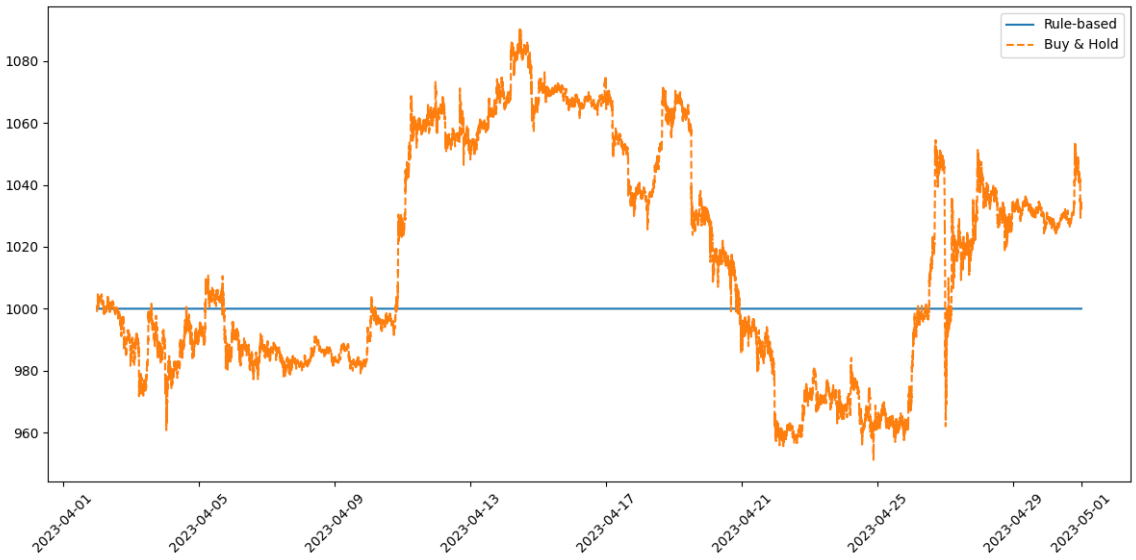
160) February



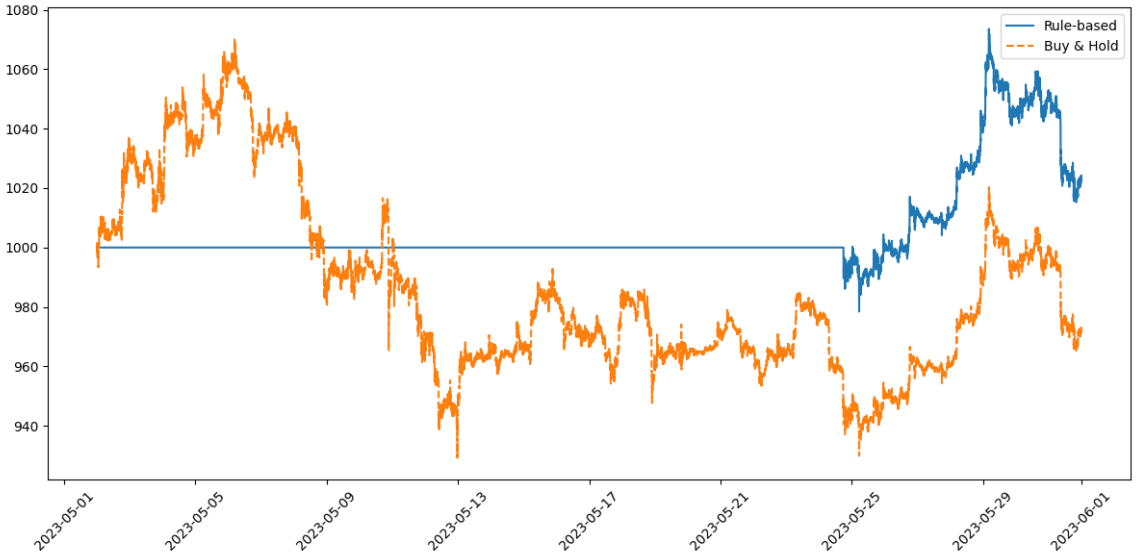
161) March



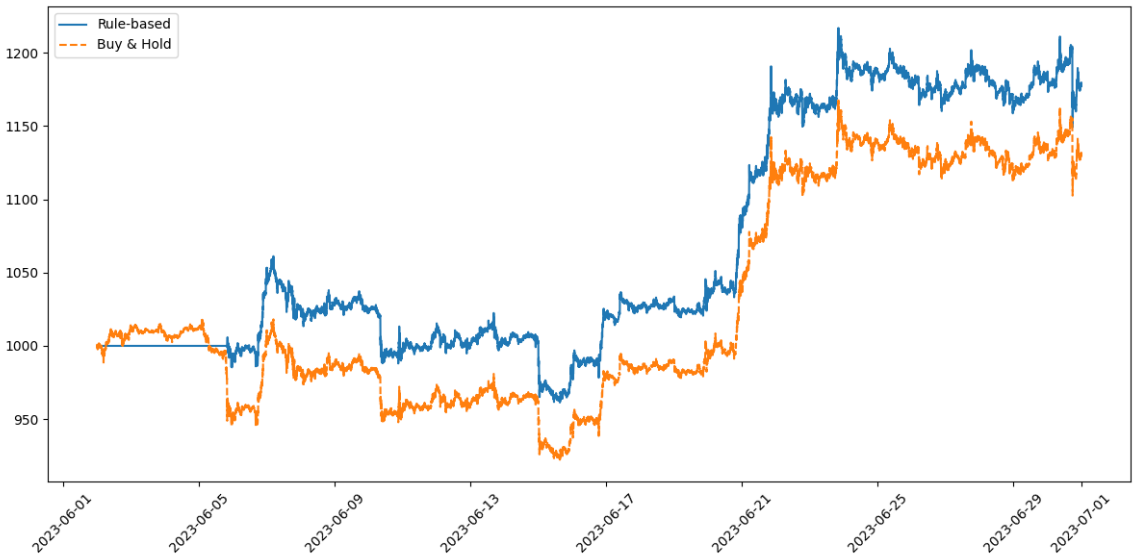
162) April



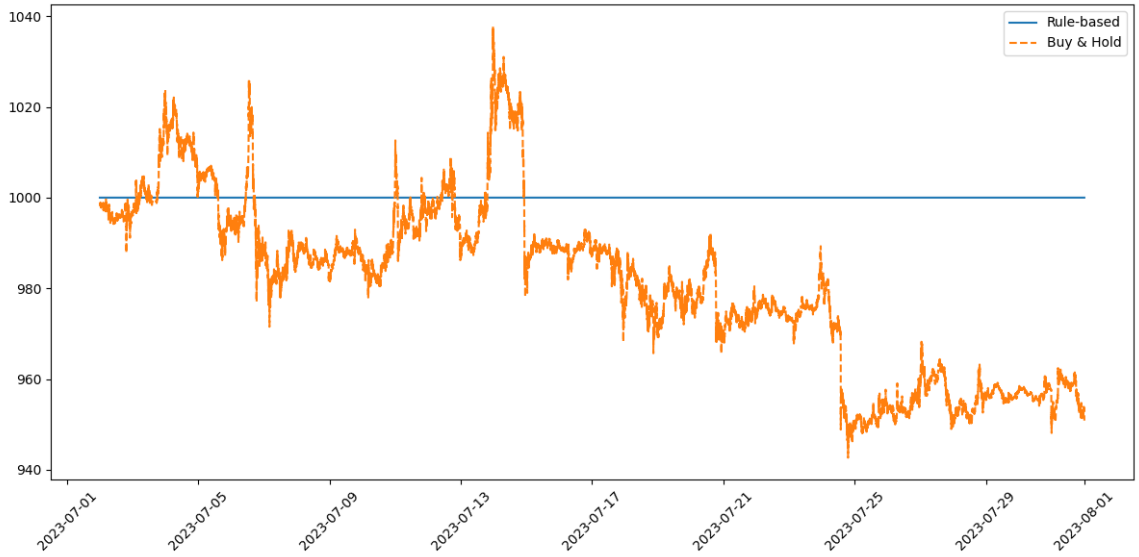
163) May



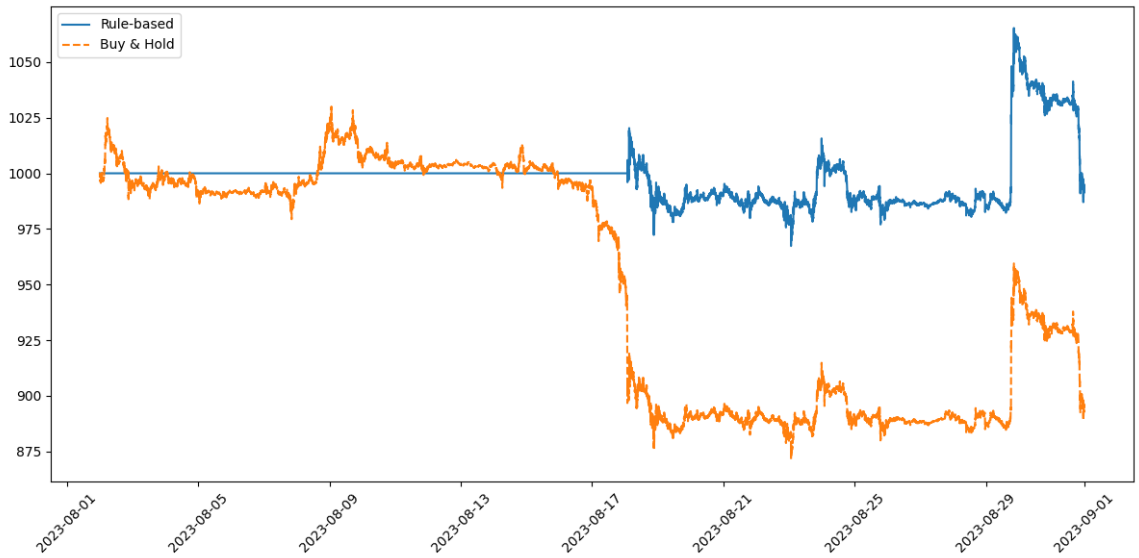
164) June



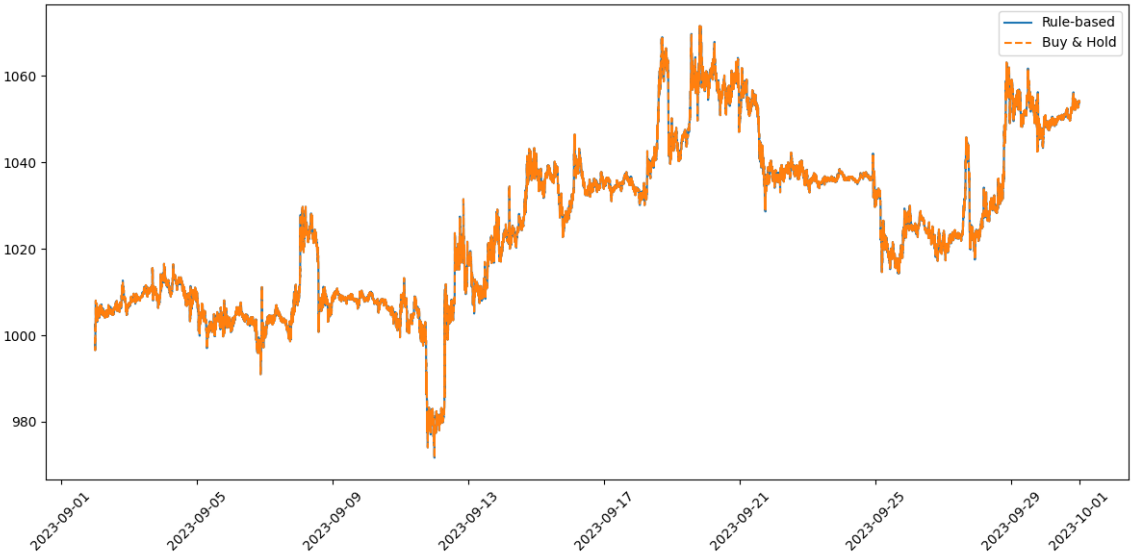
165) July



166) August

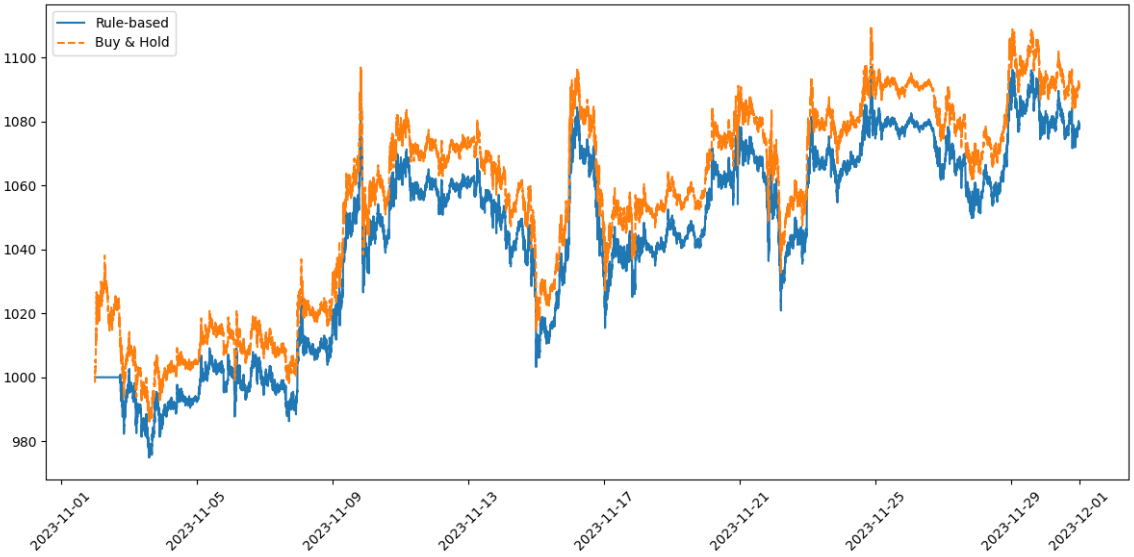


167) September

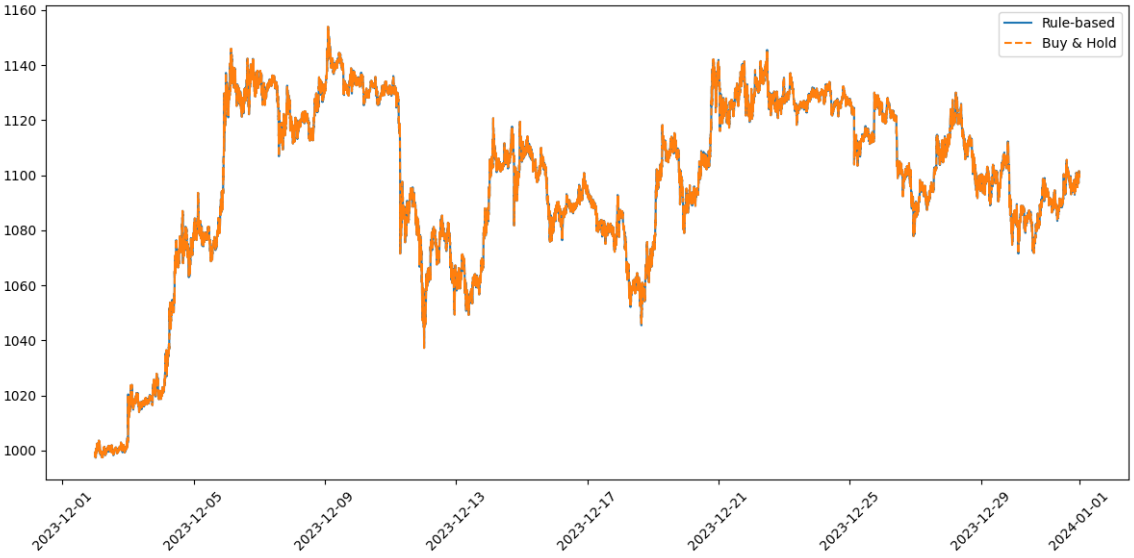


168) October

169) November

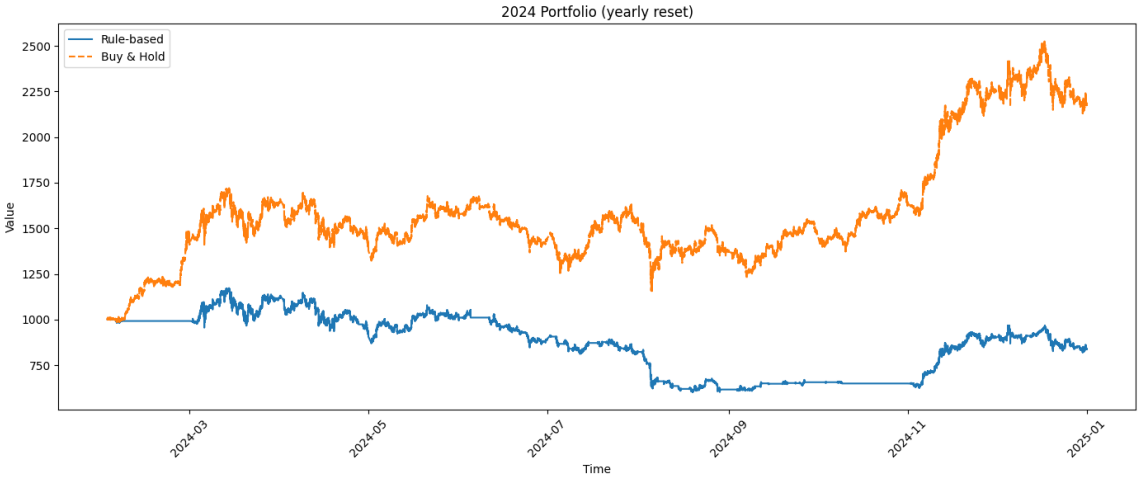


170) December



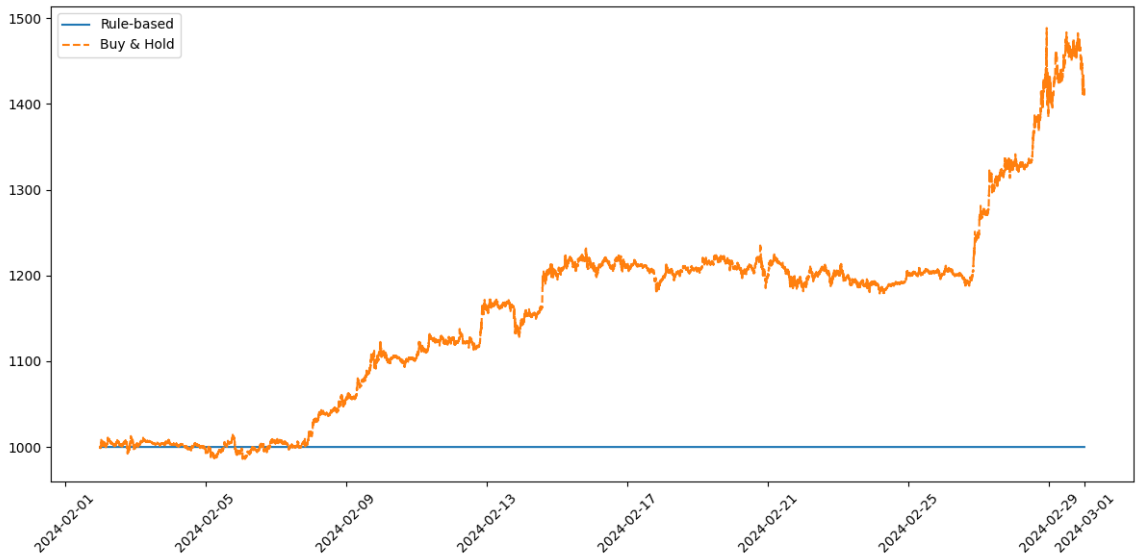
2024

171) Full Year



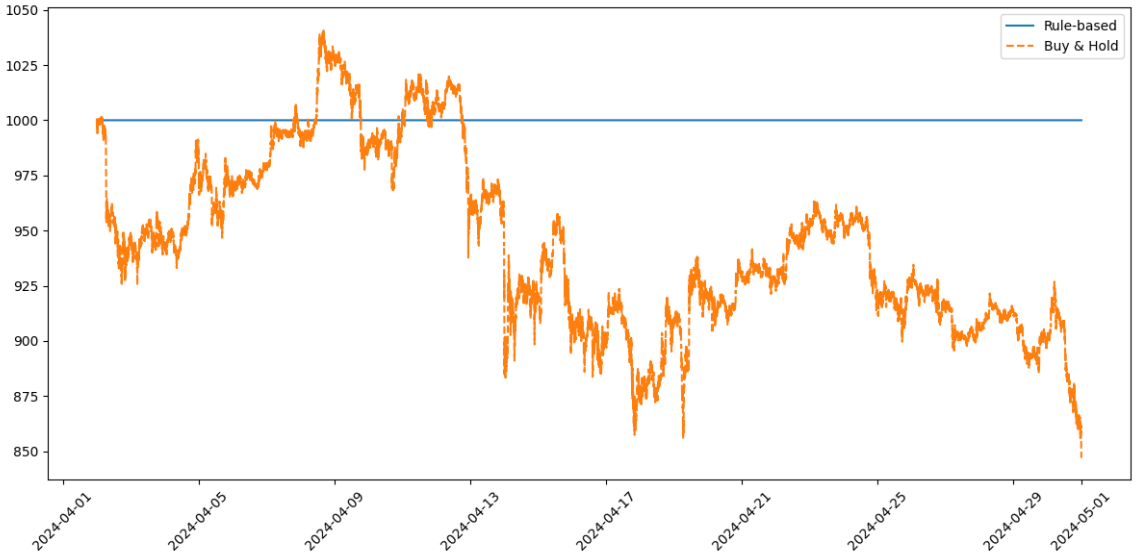
172) January

173) February



174) March

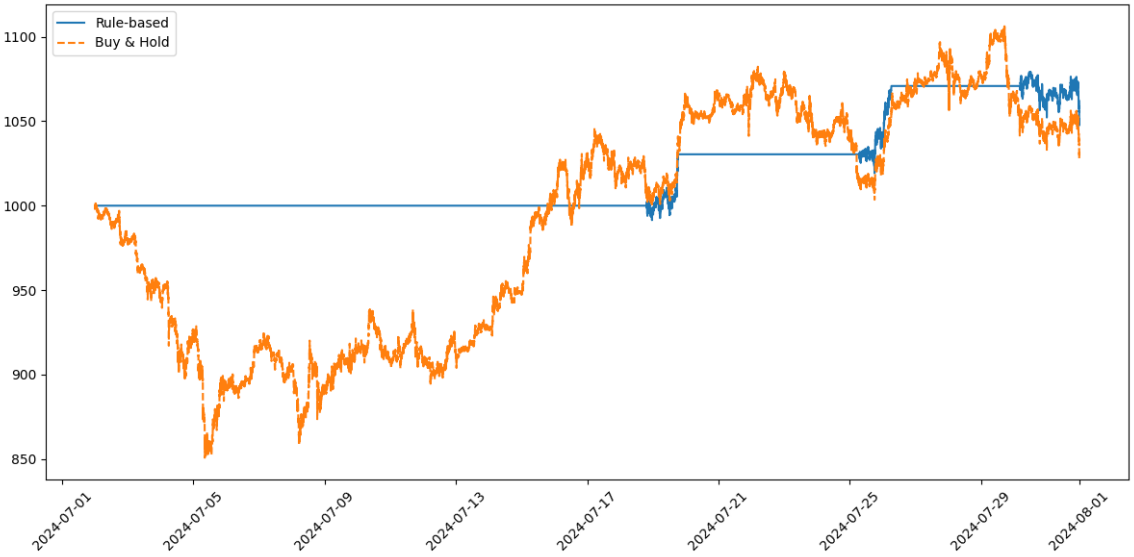
175) April



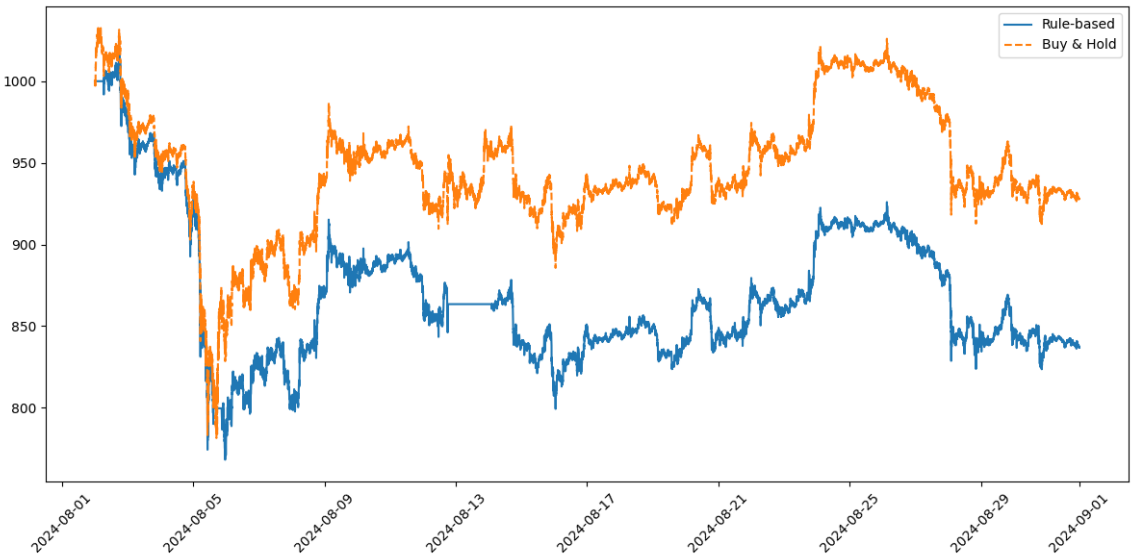
176) May

177) June

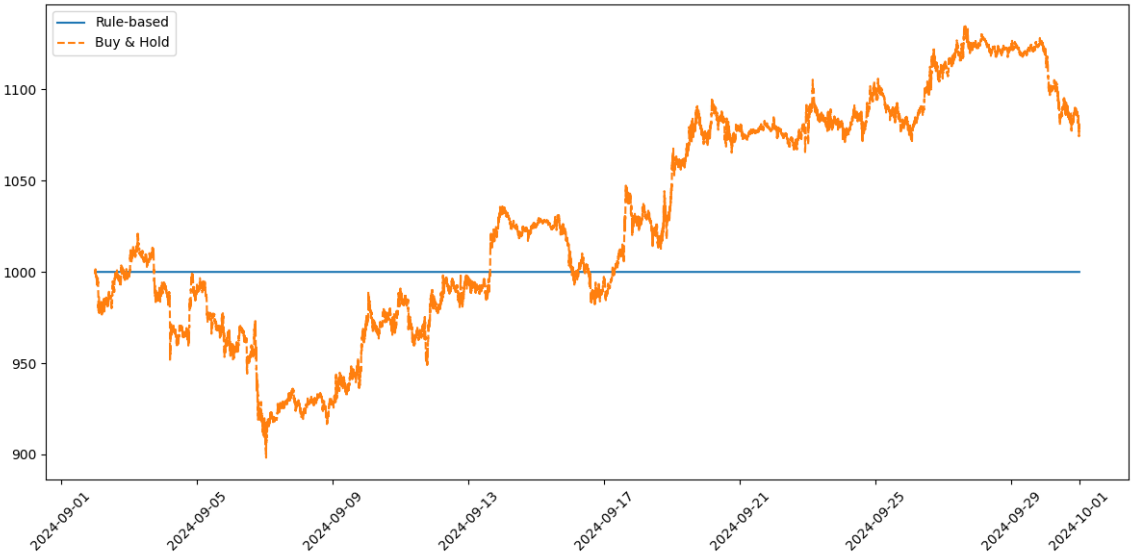
178) July



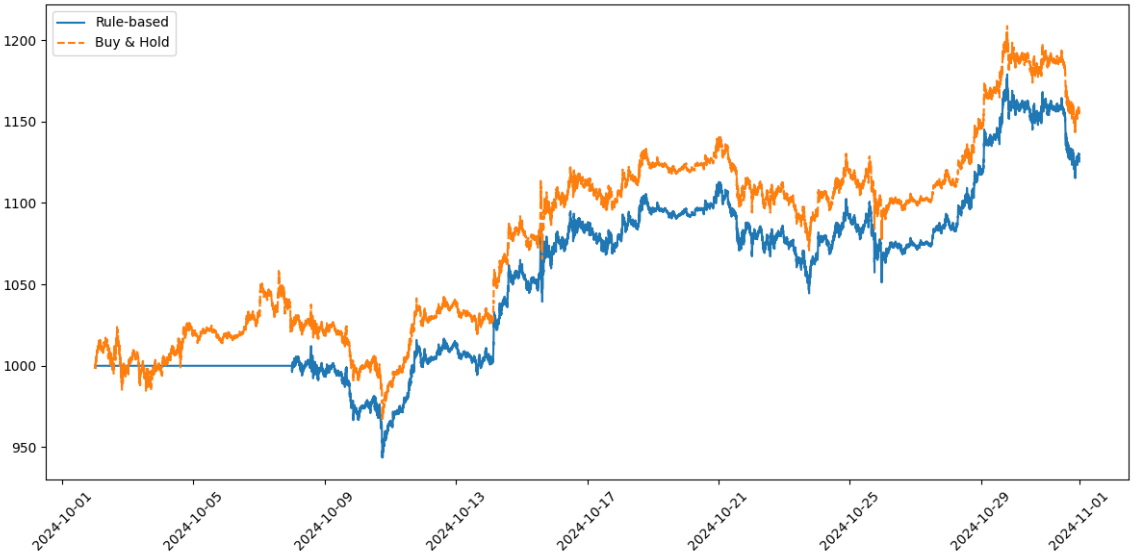
179) August



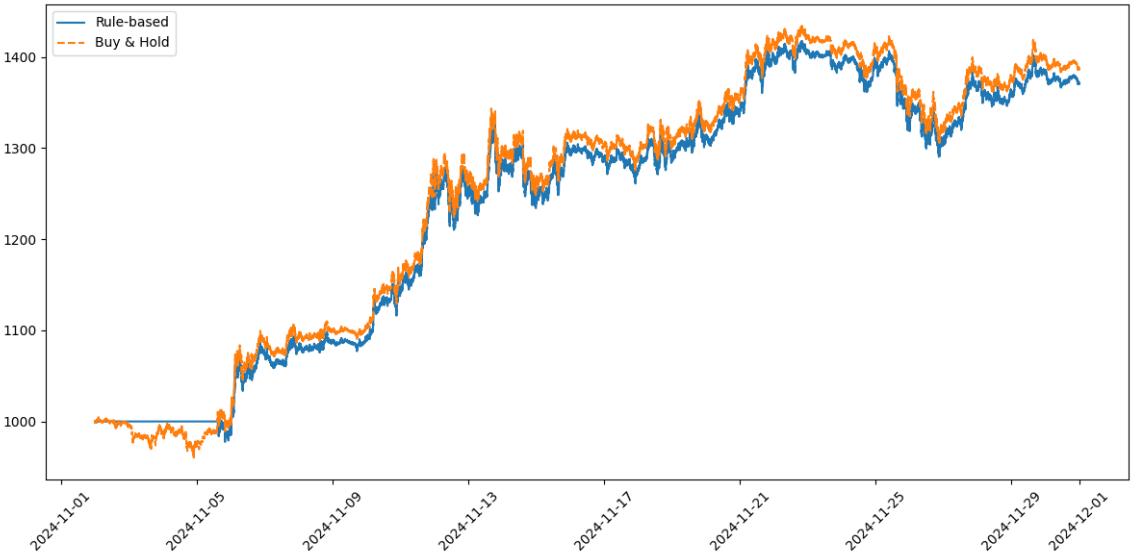
180) September



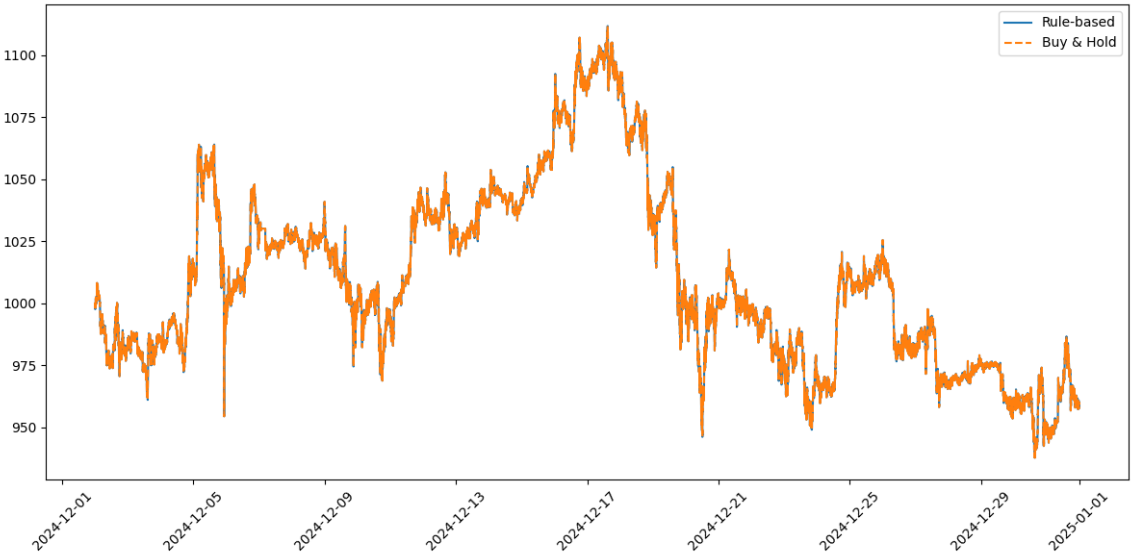
181) October



182) November

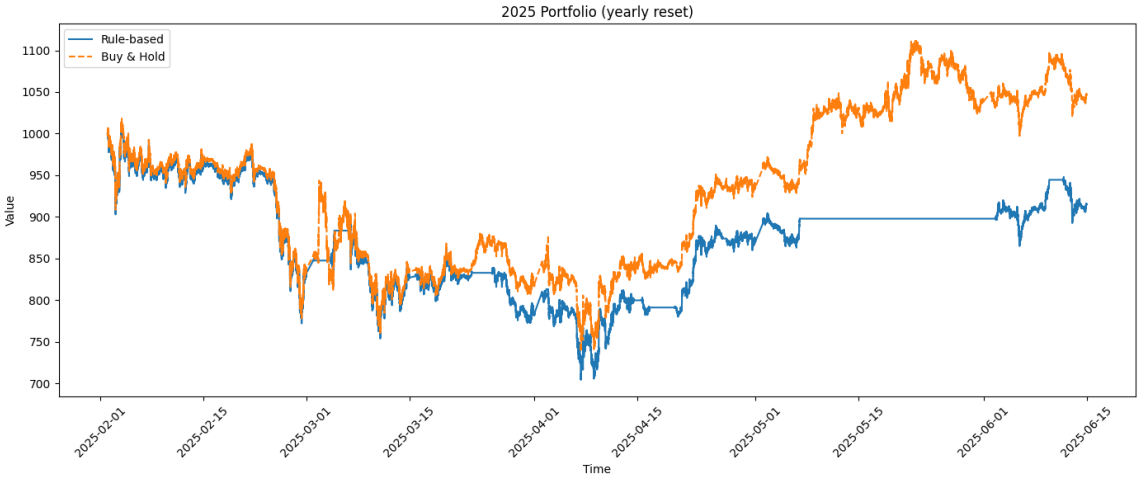


183) December

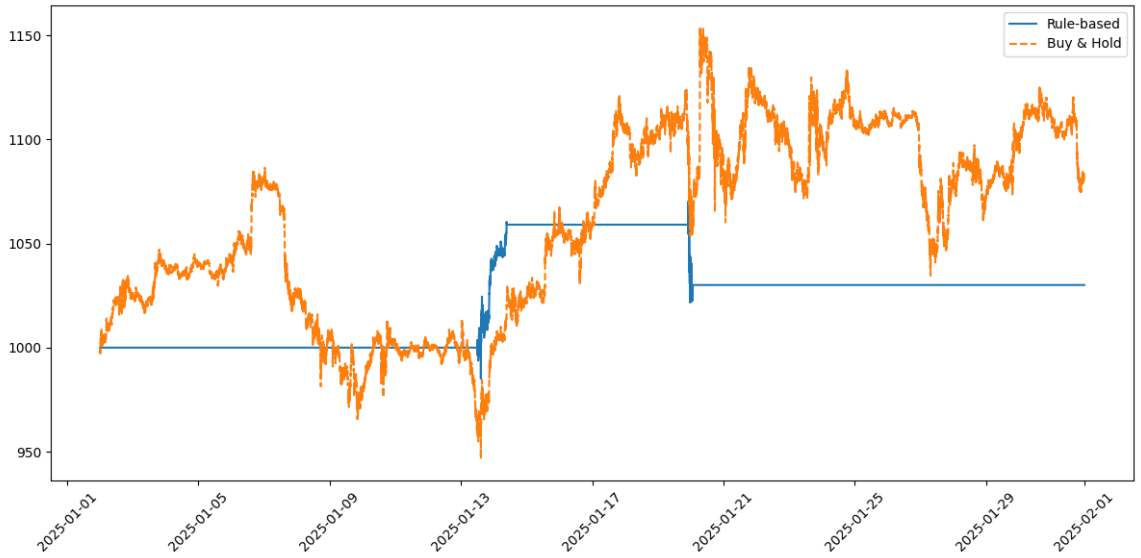


2025

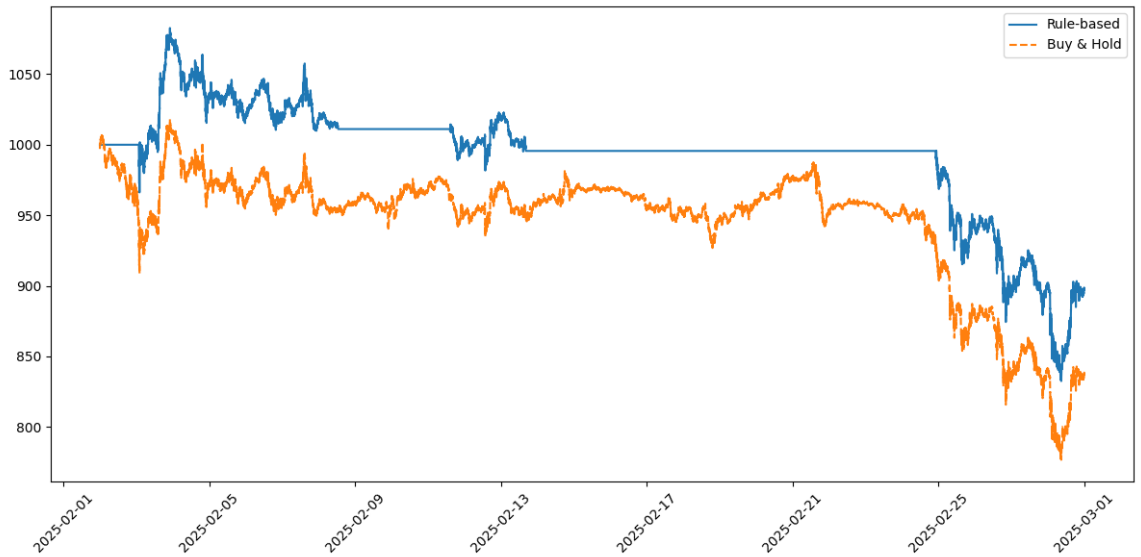
184) Full Year



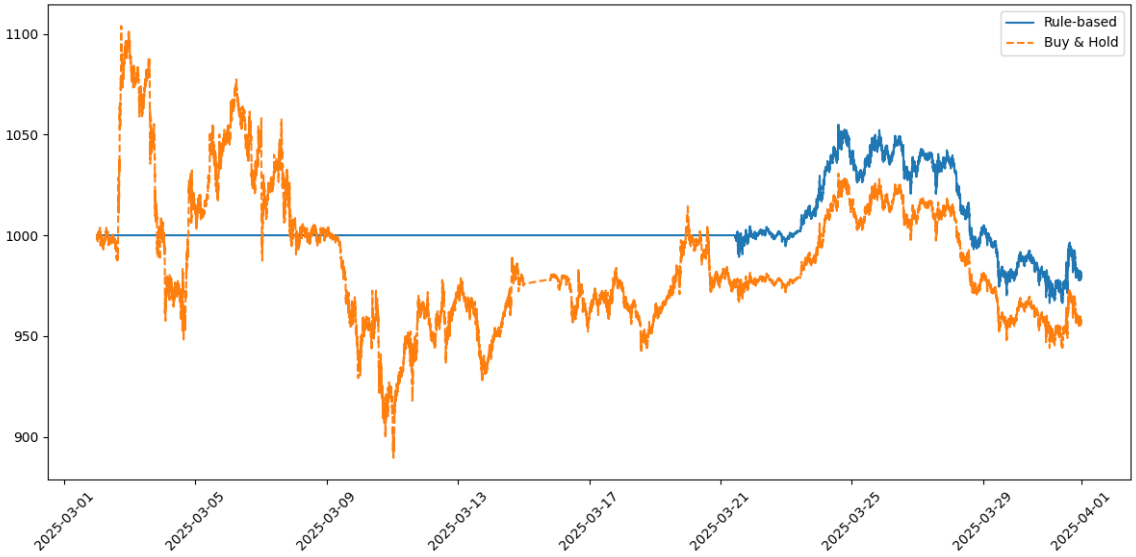
185) January



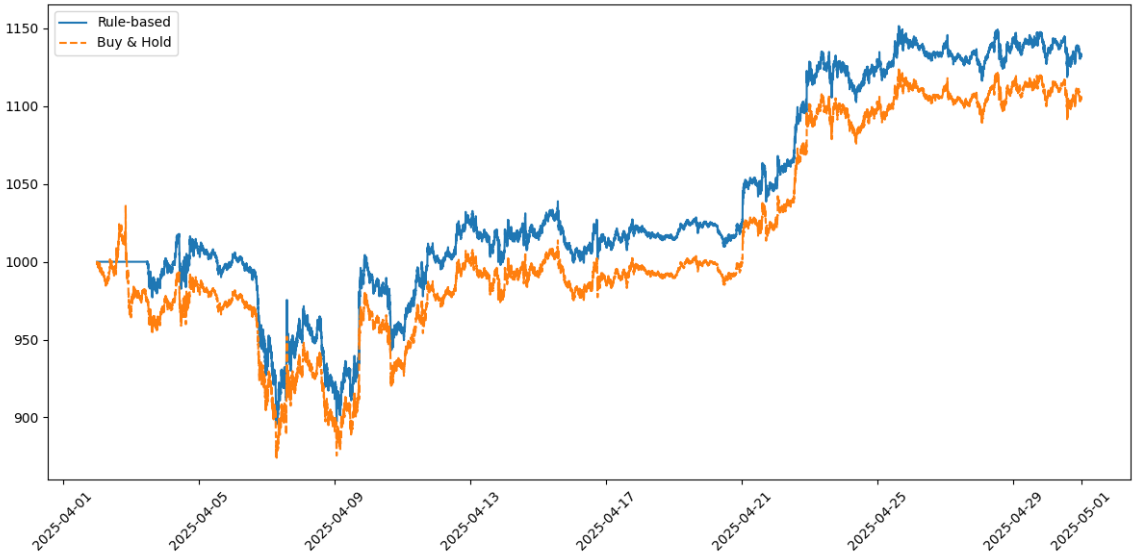
186) February



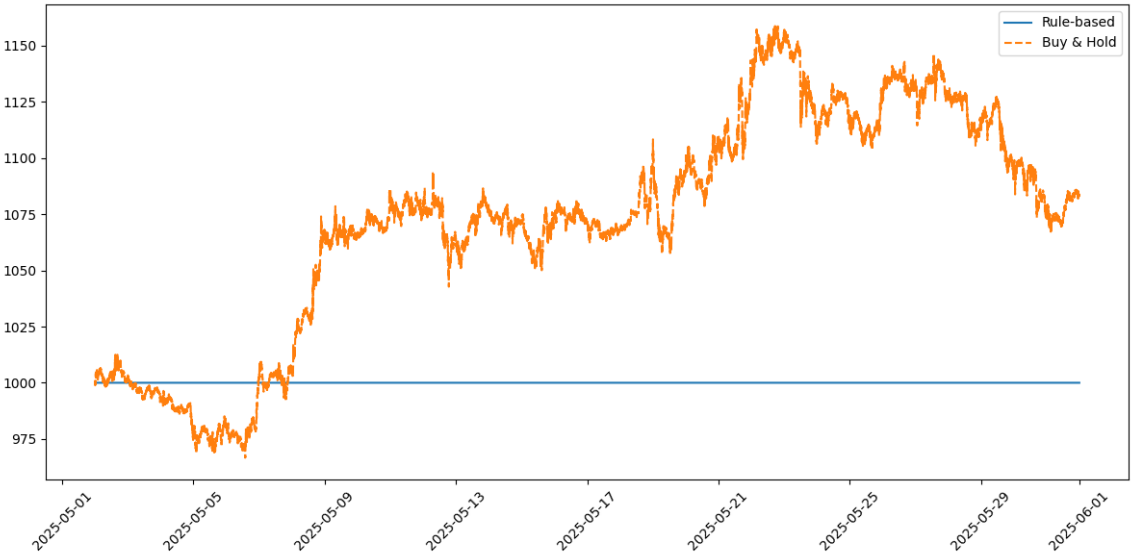
187) March



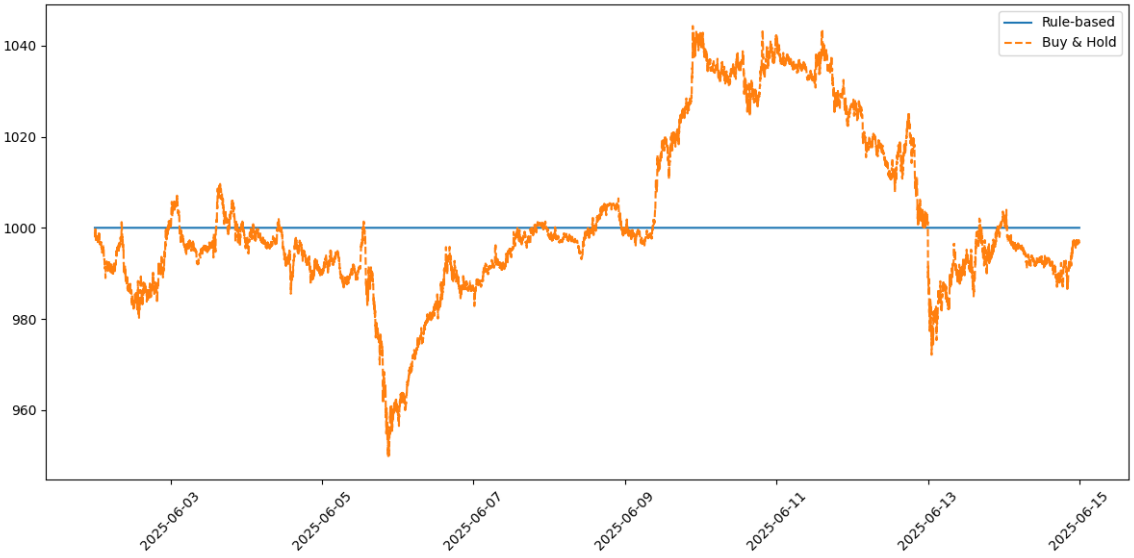
188) April



189) May



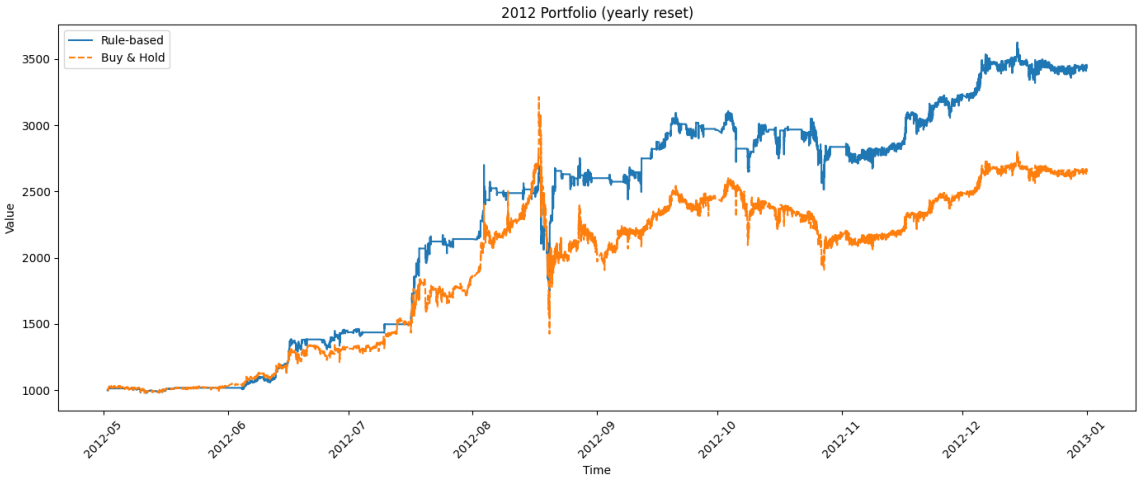
190) June



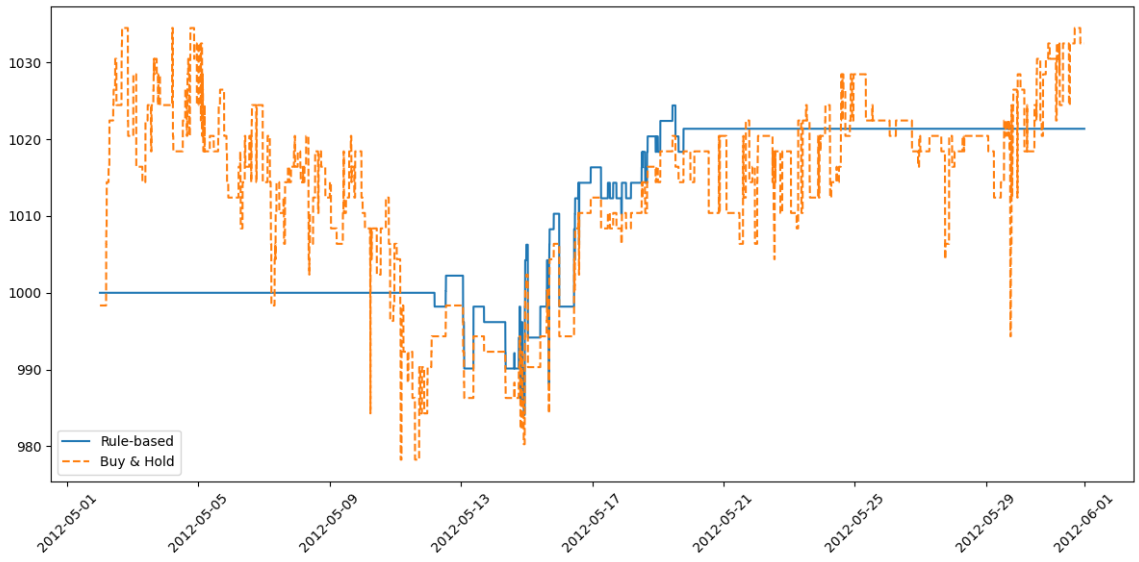
Portfolio Equity Graphs – LSTM

2012

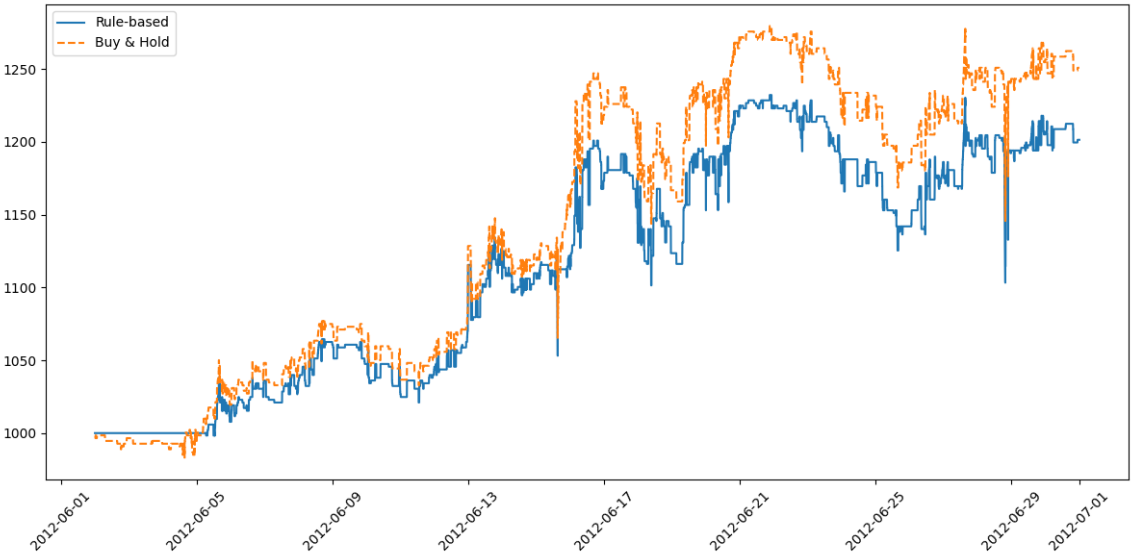
191) Full Year



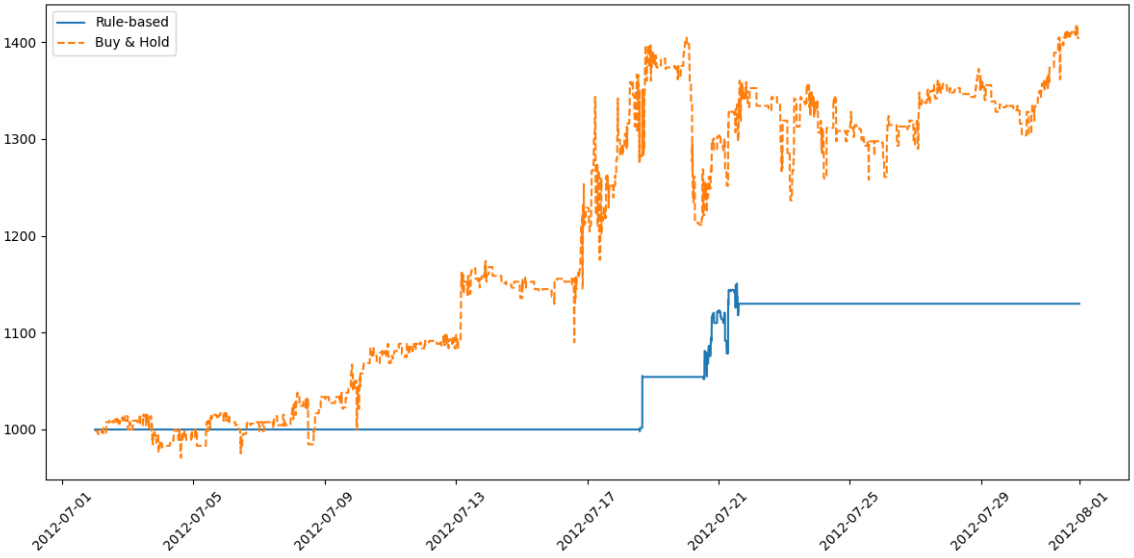
192) May



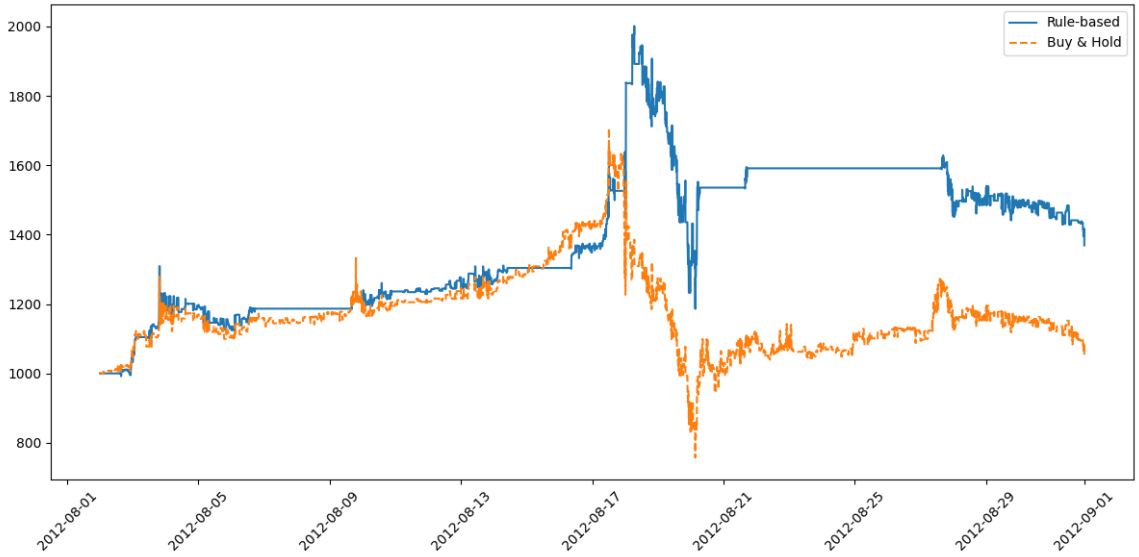
193) June



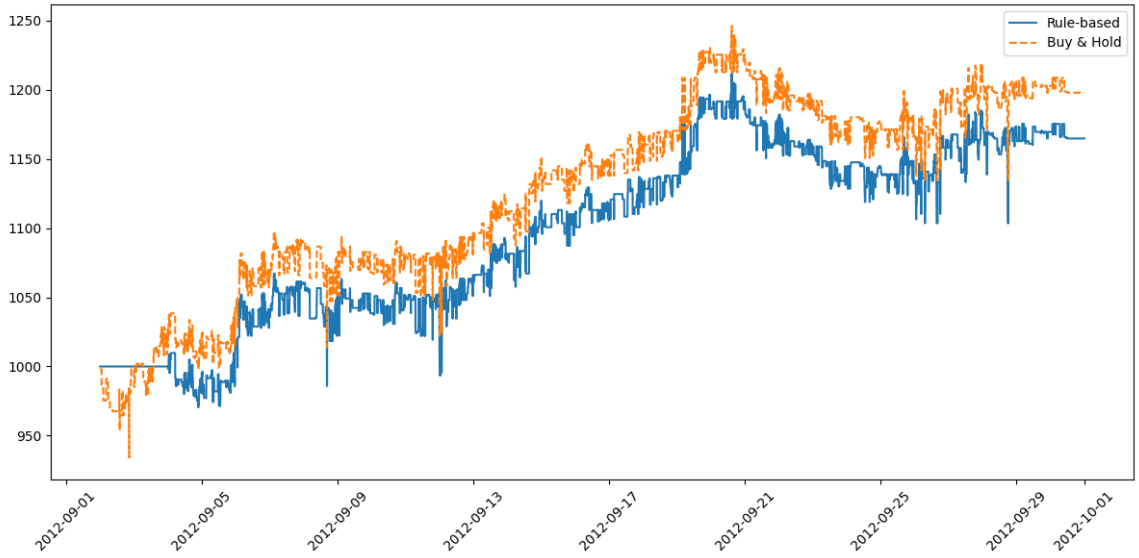
194) July



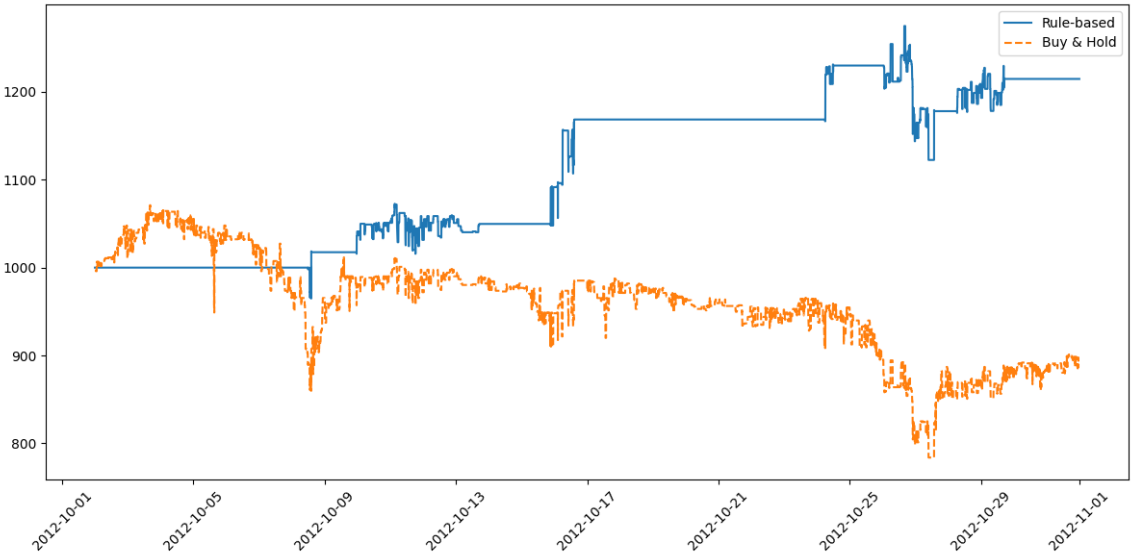
195) August



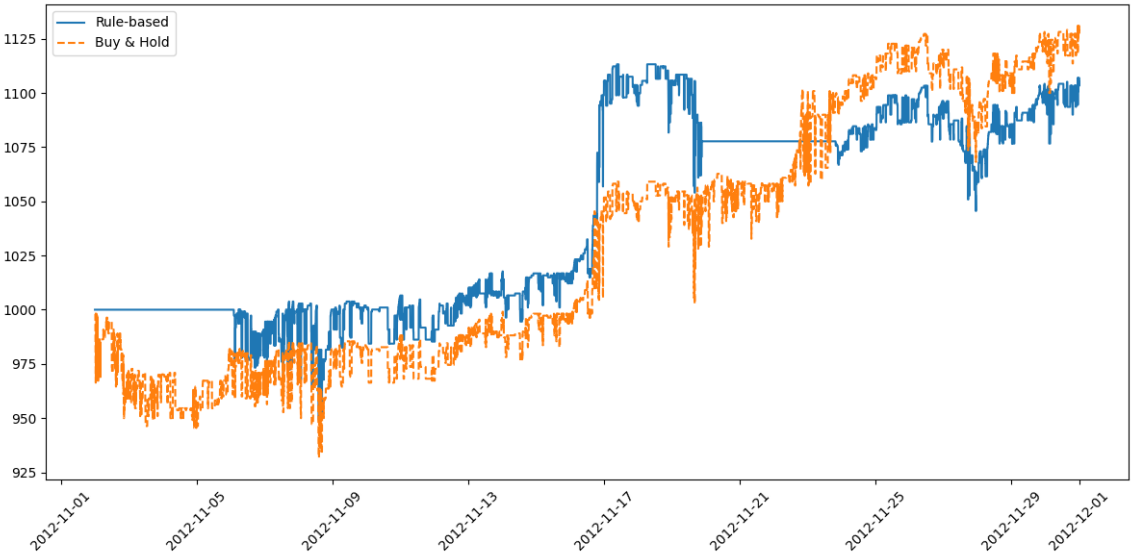
196) September



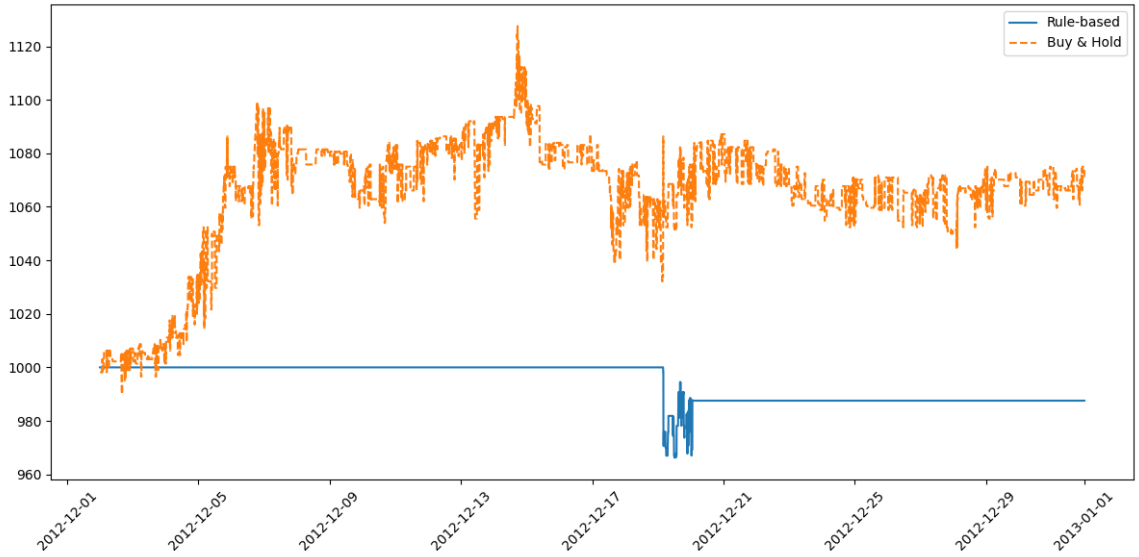
197) October



198) November

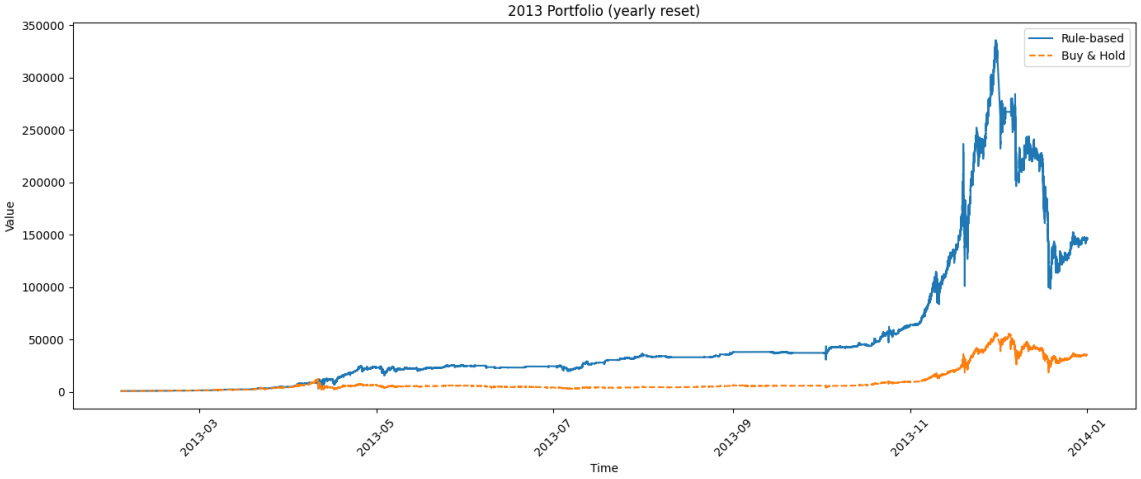


199) December

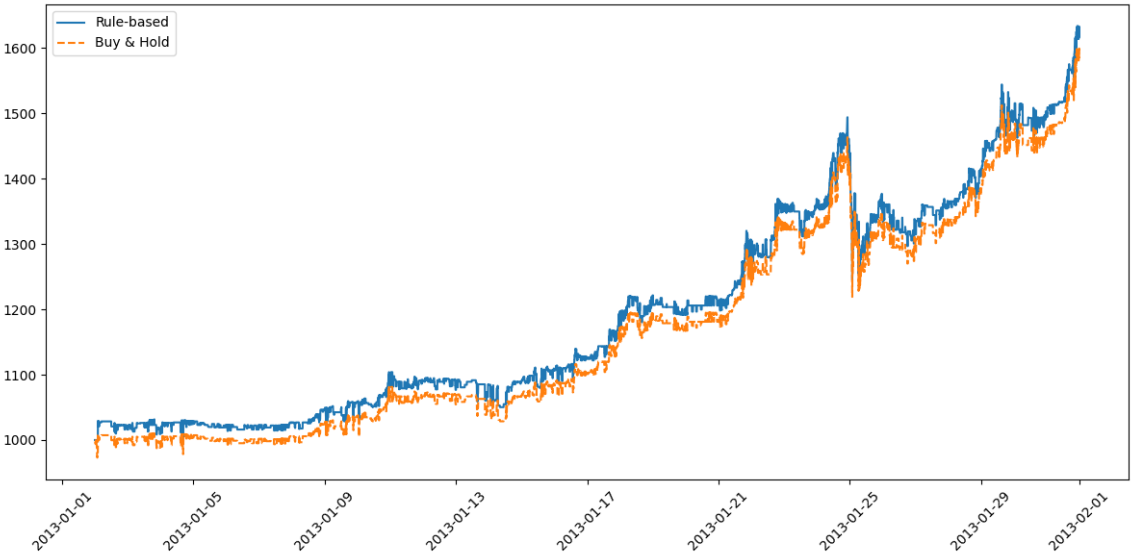


2013

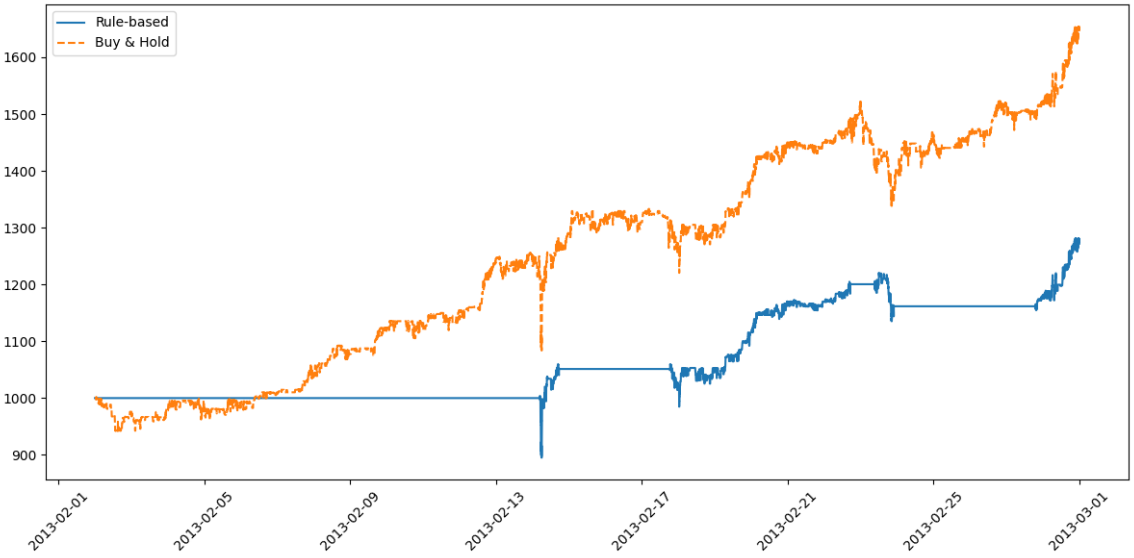
200) Full Year



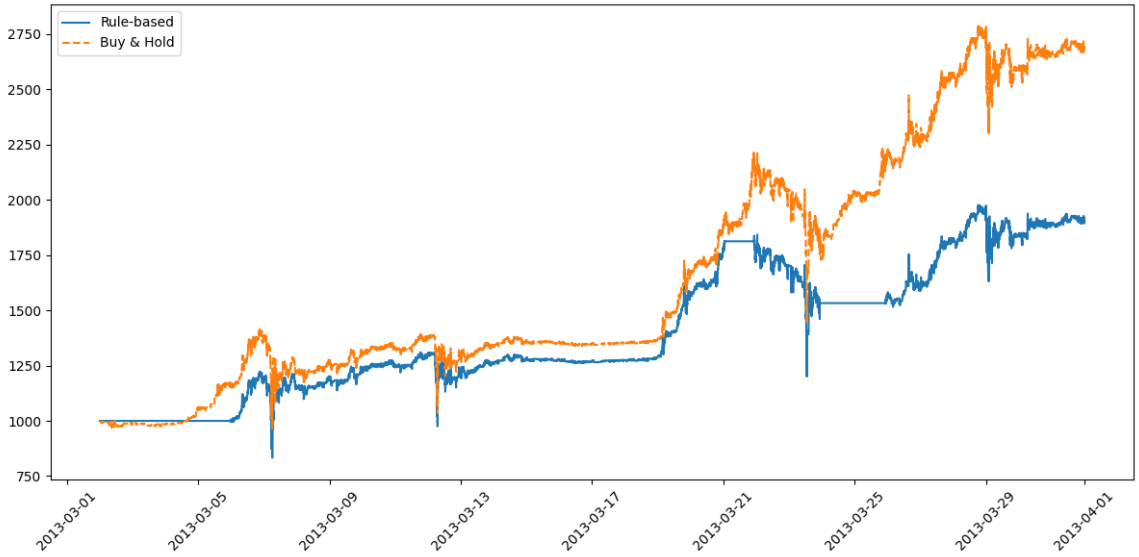
201) January



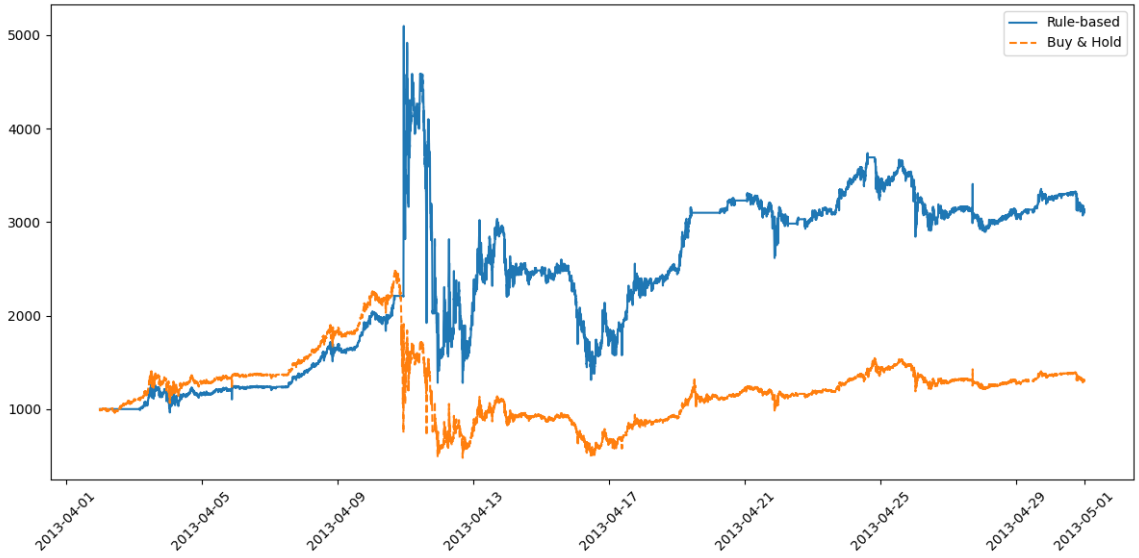
202) February



203) March

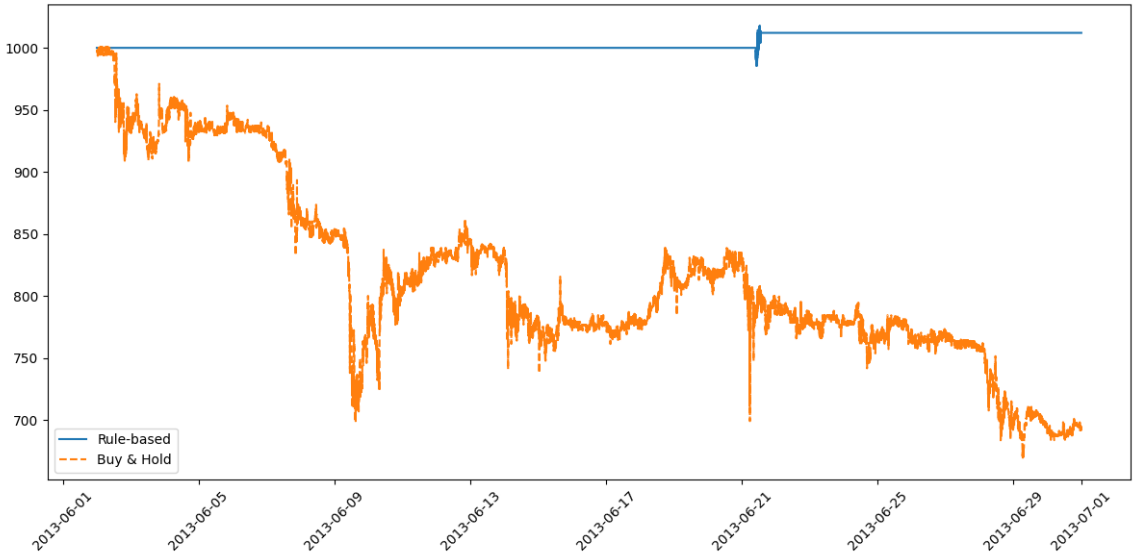


204) April

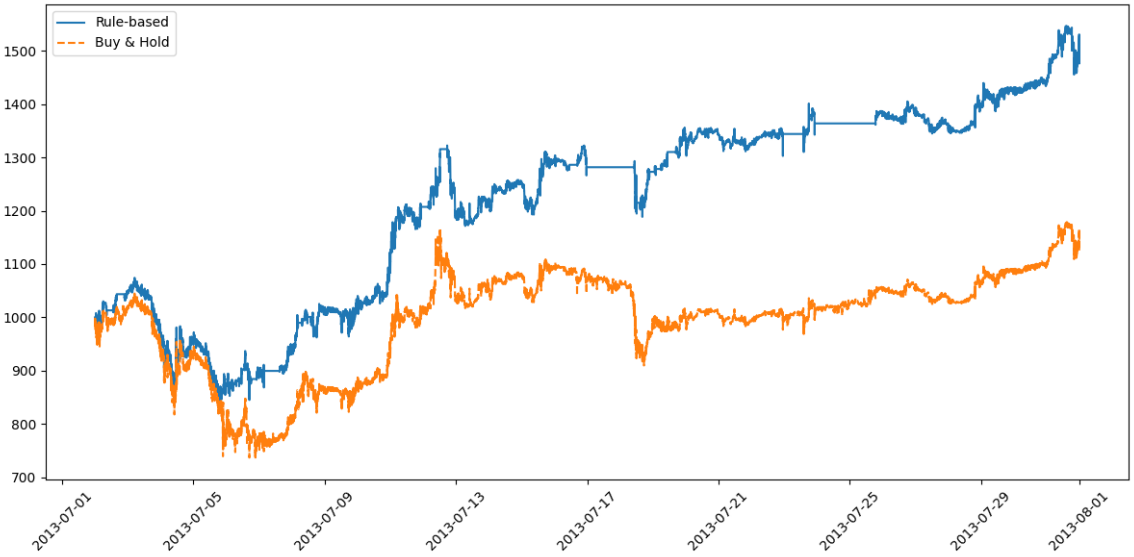


205) May

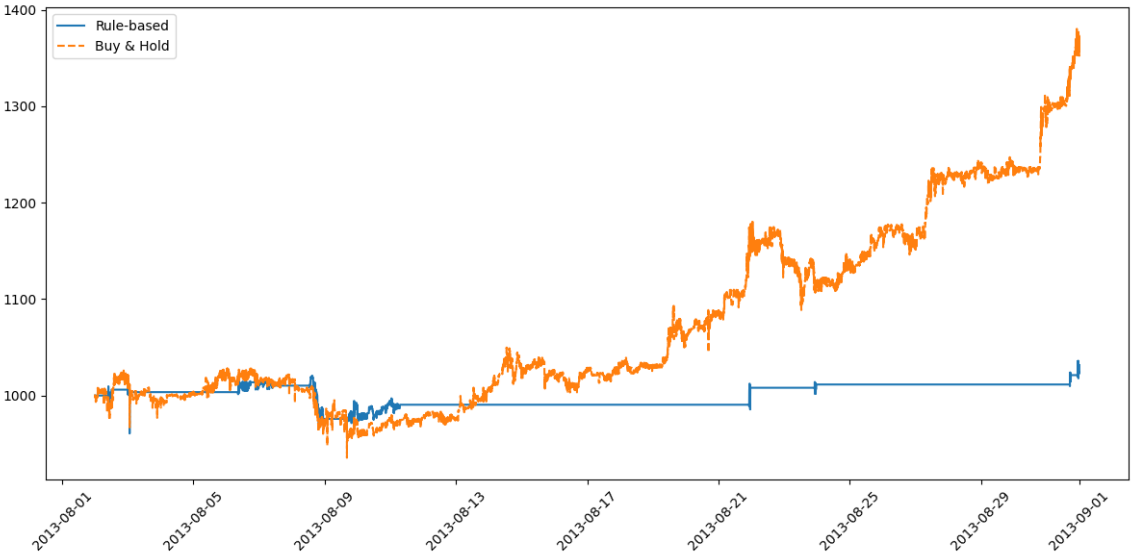
206) June



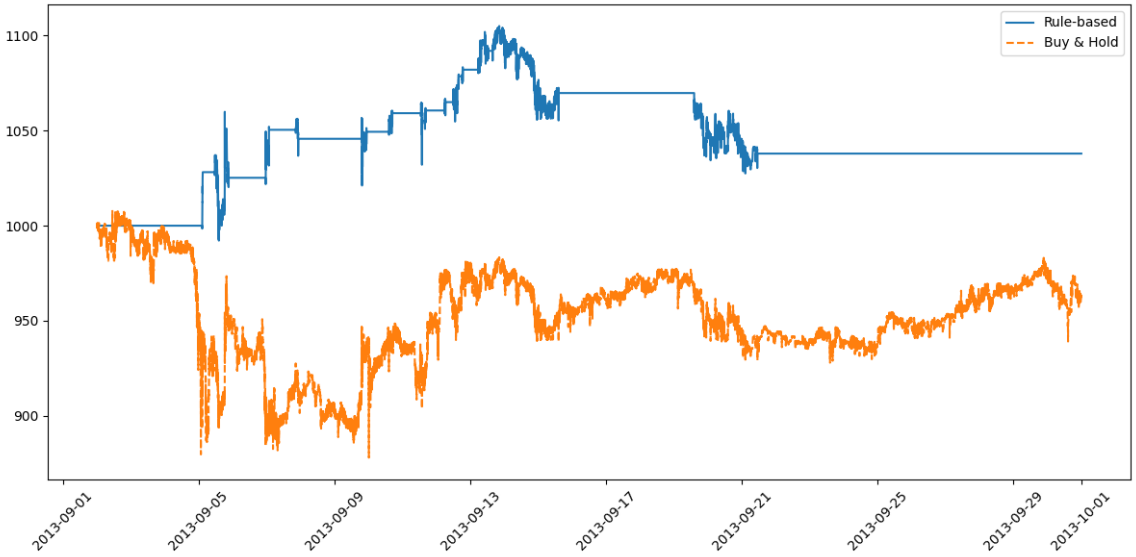
207) July



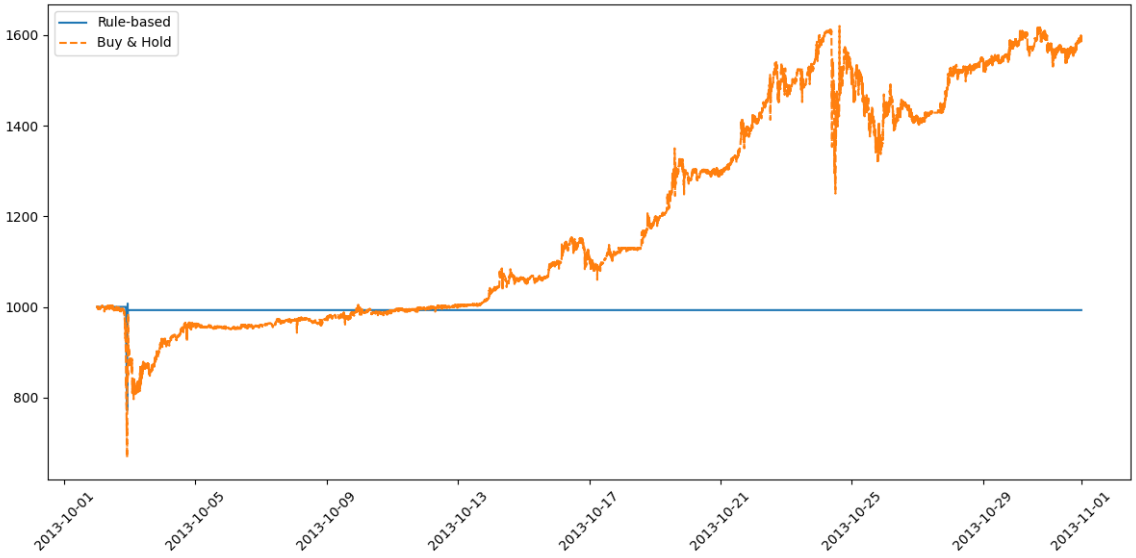
208) August



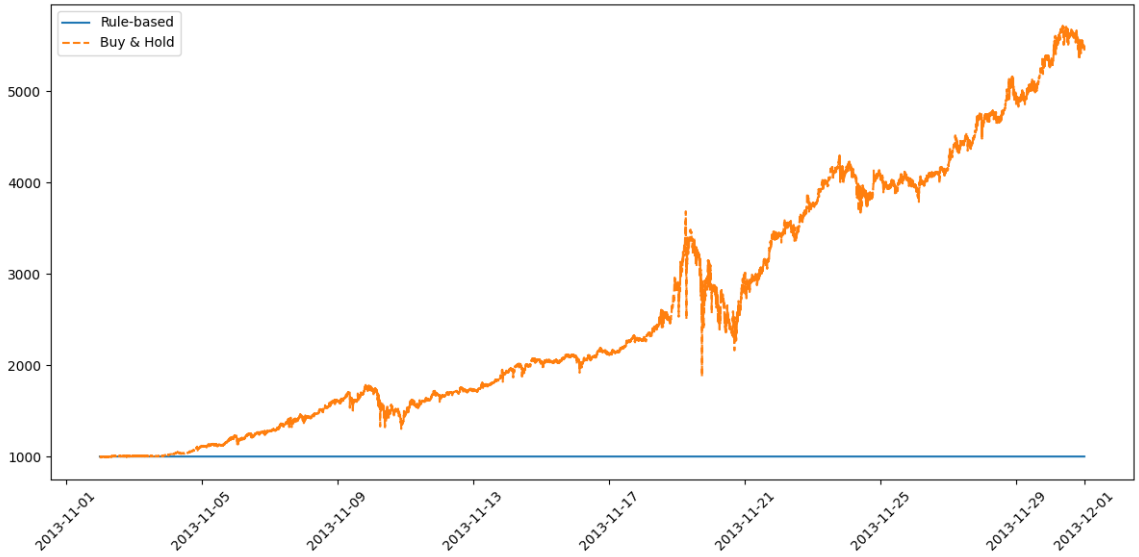
209) September



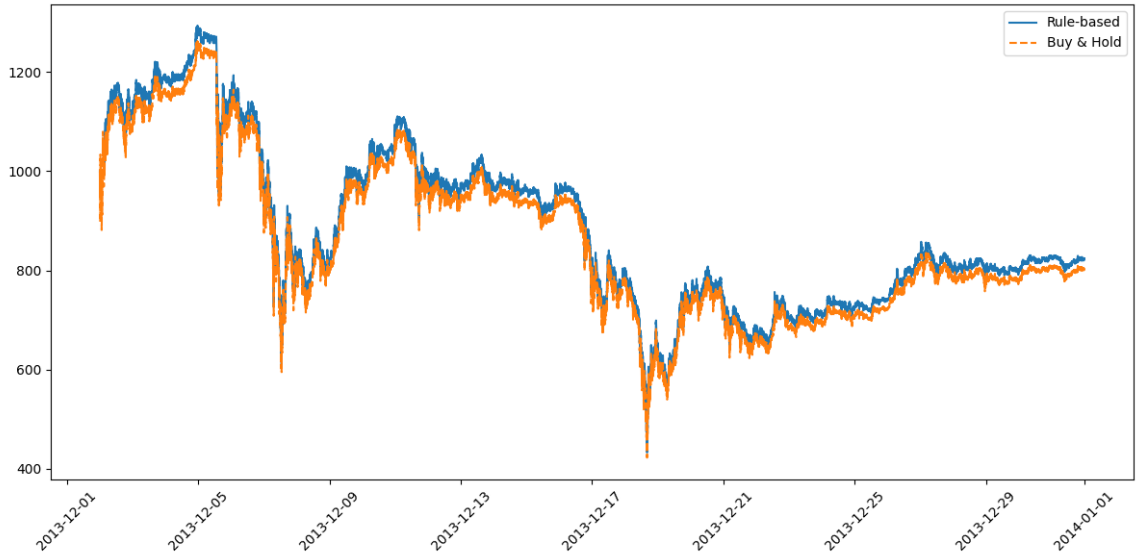
210) October



211) November

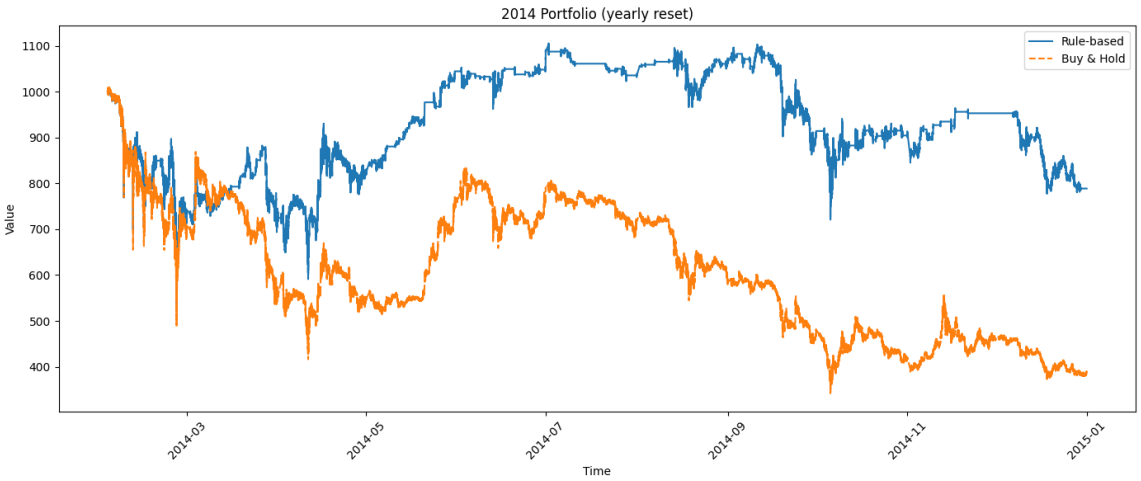


212) December

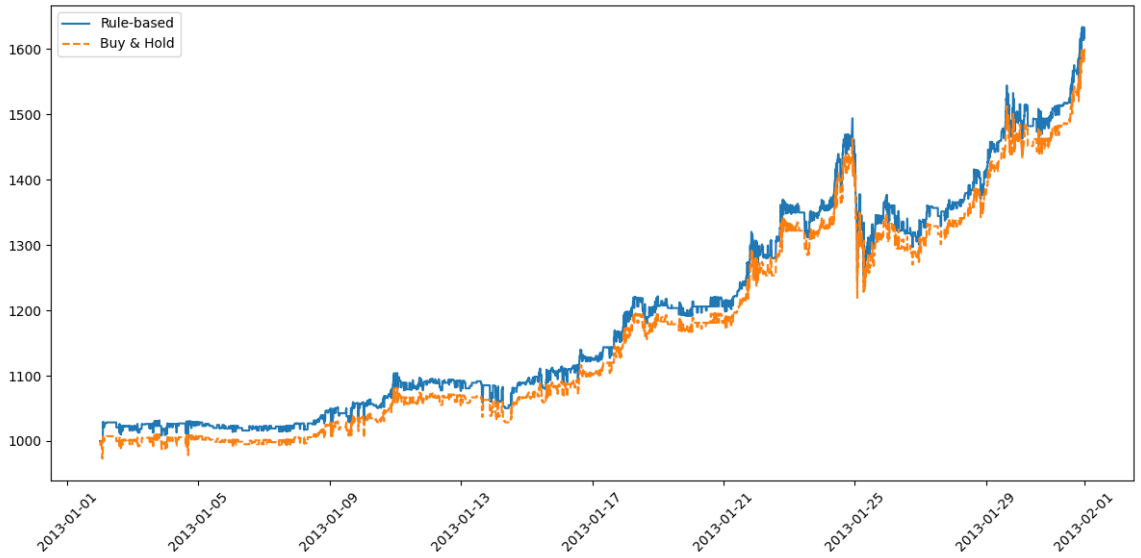


2014

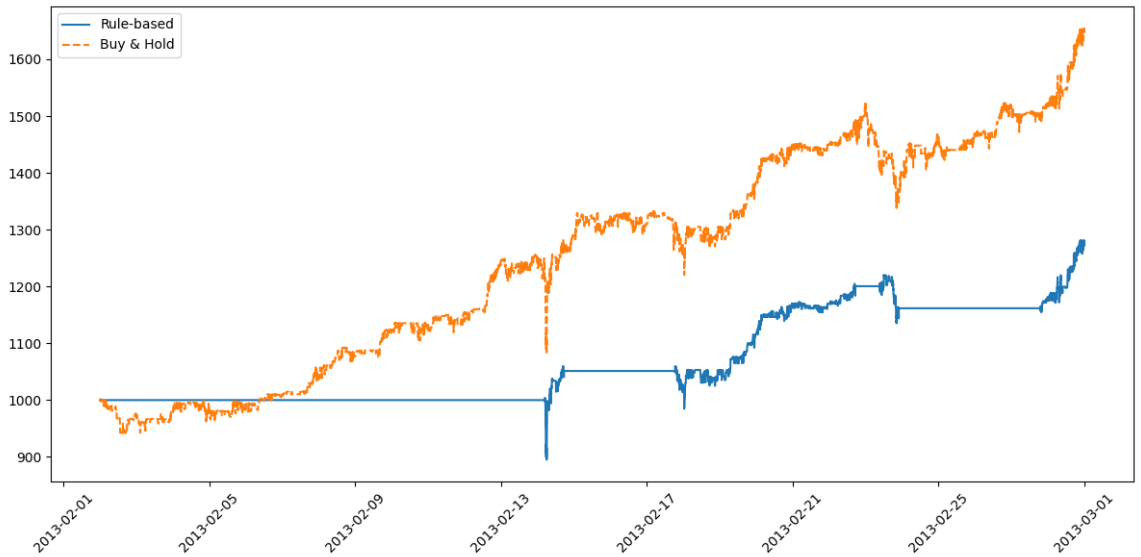
213) Full Year



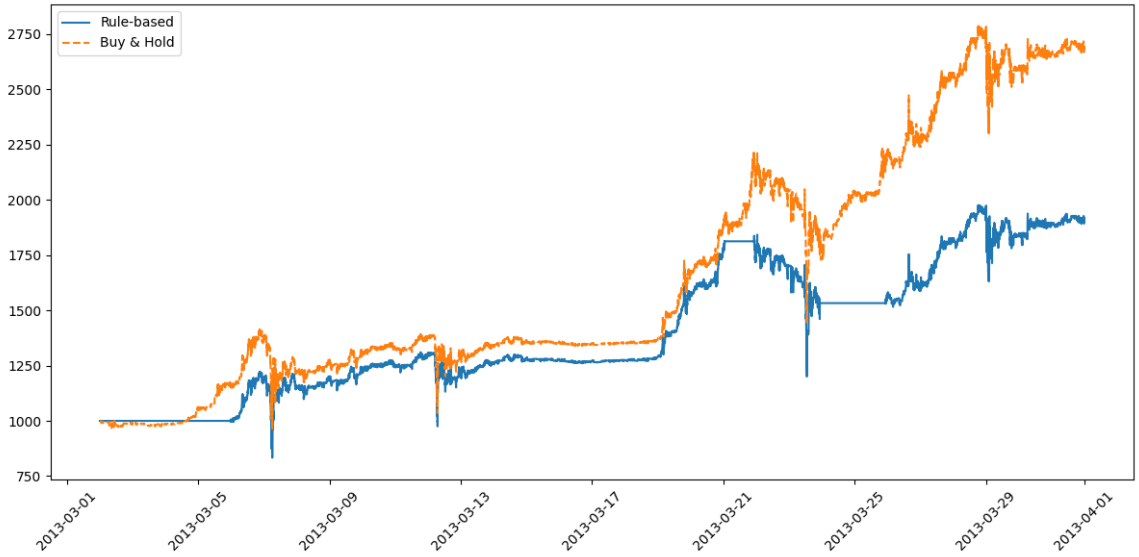
214) January



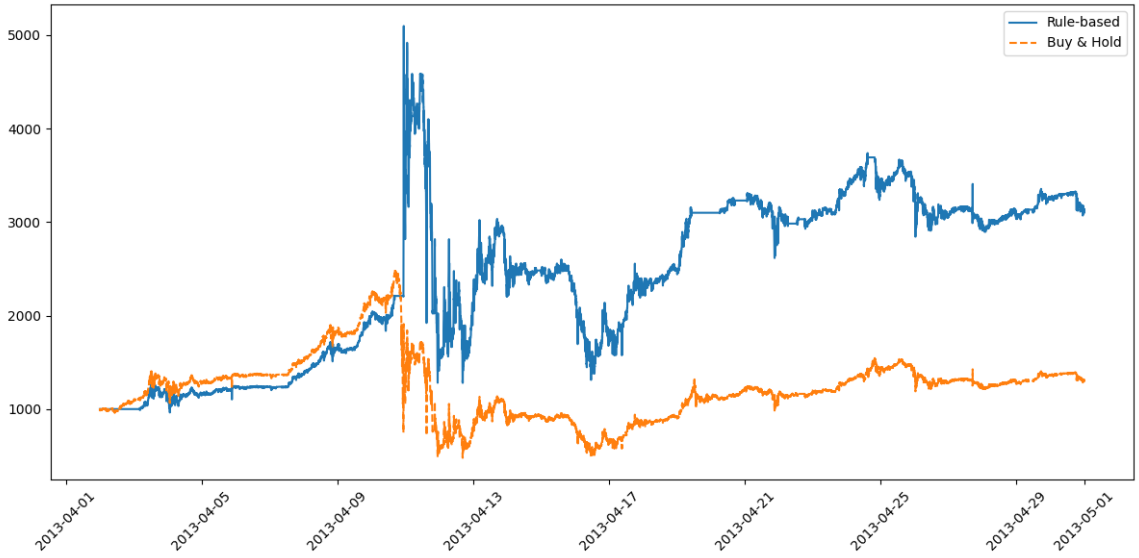
215) February



216) March

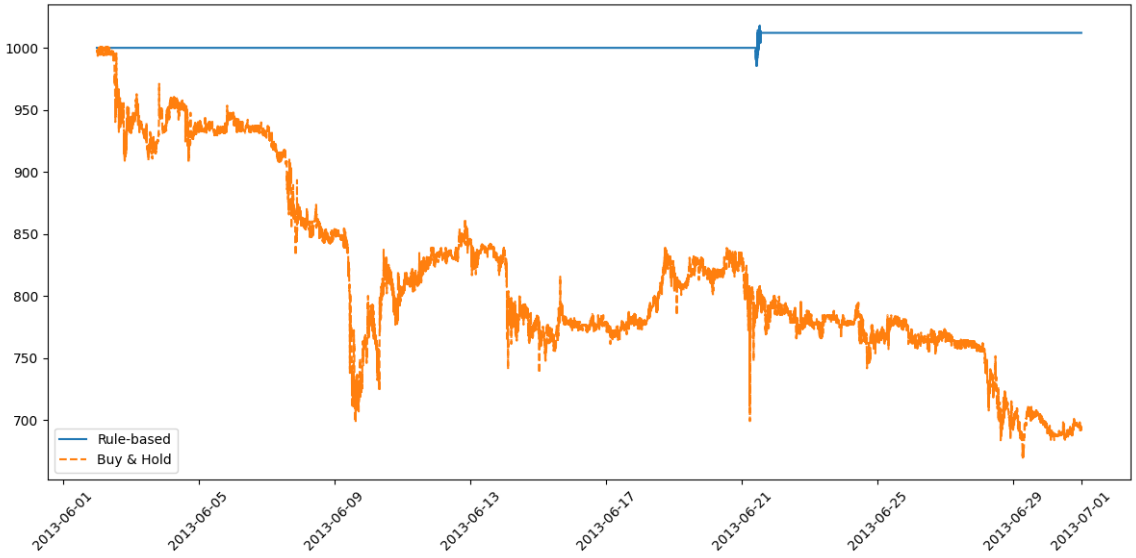


217) April

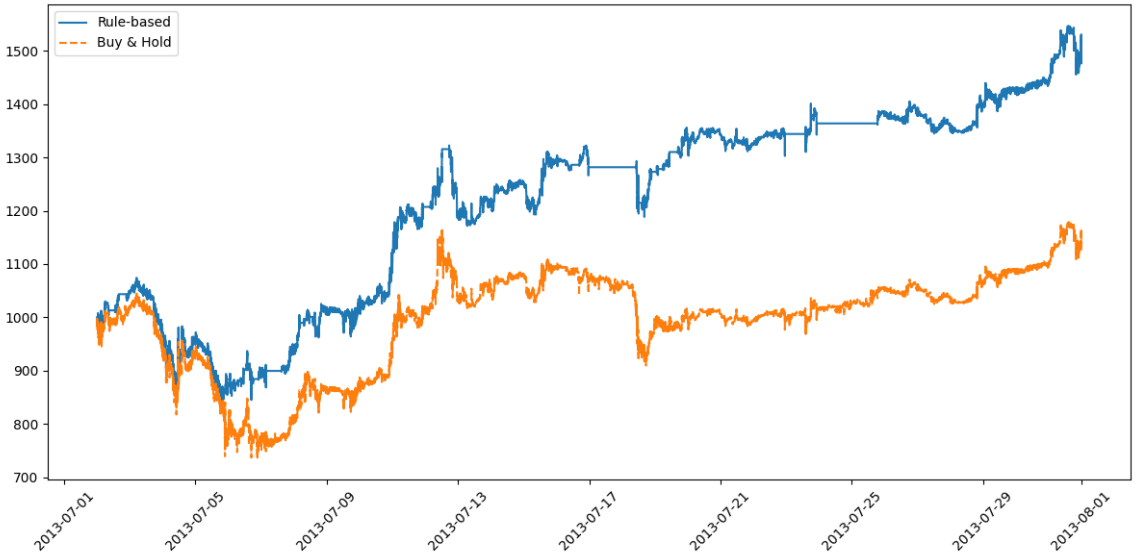


218) May

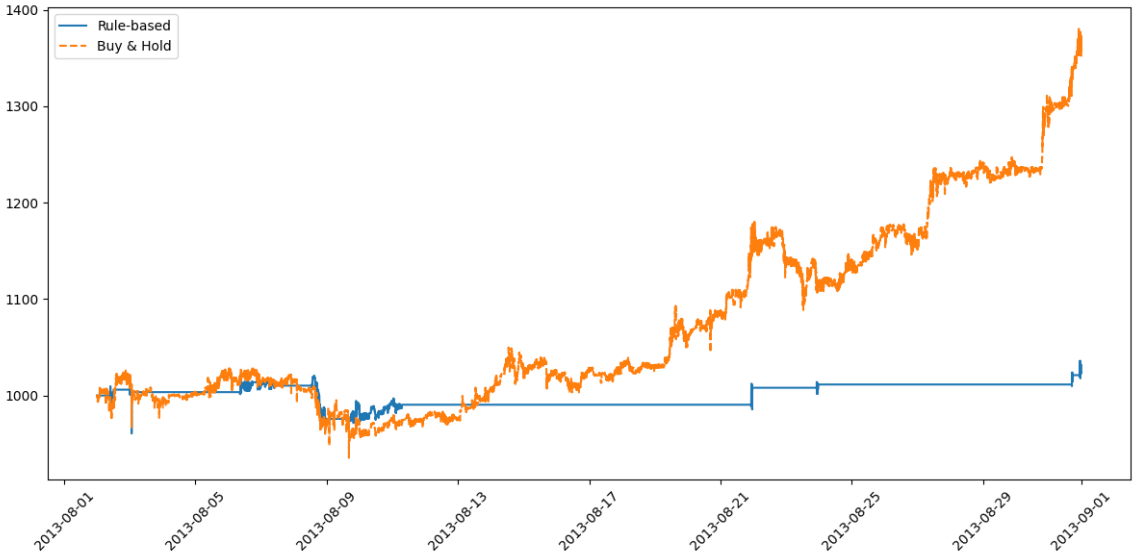
219) June



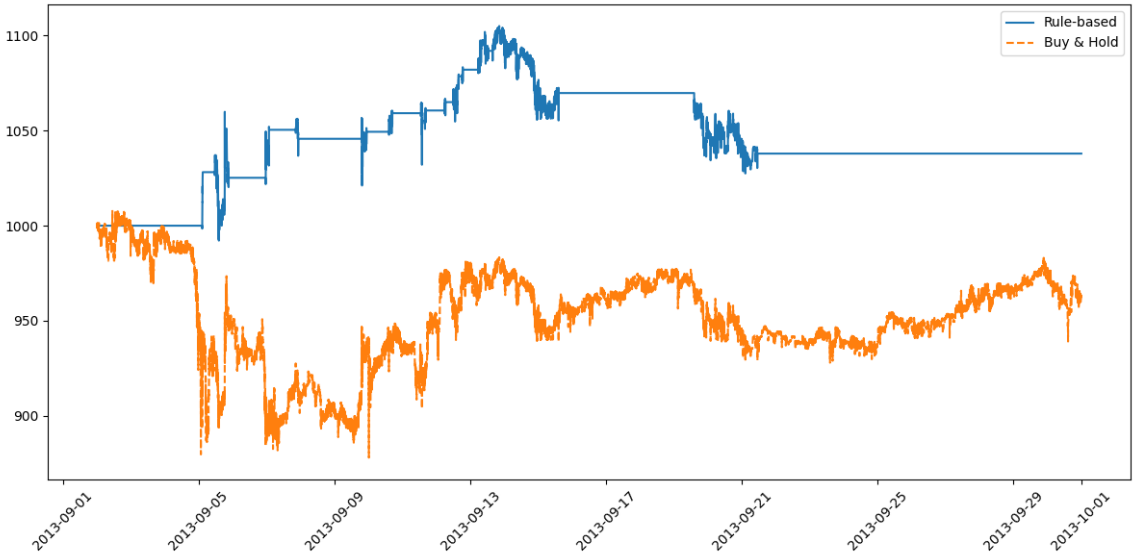
220) July



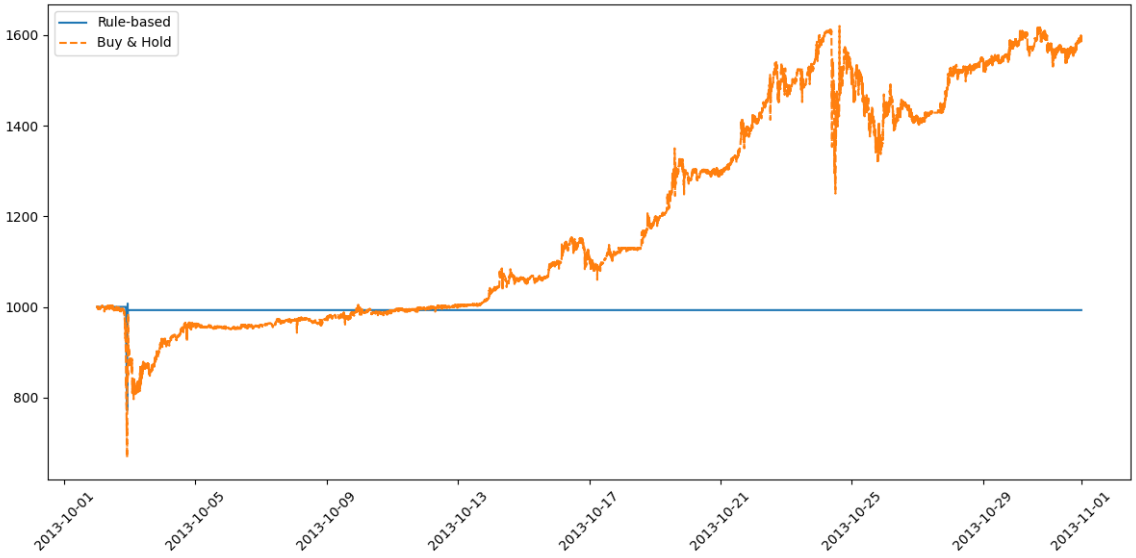
221) August



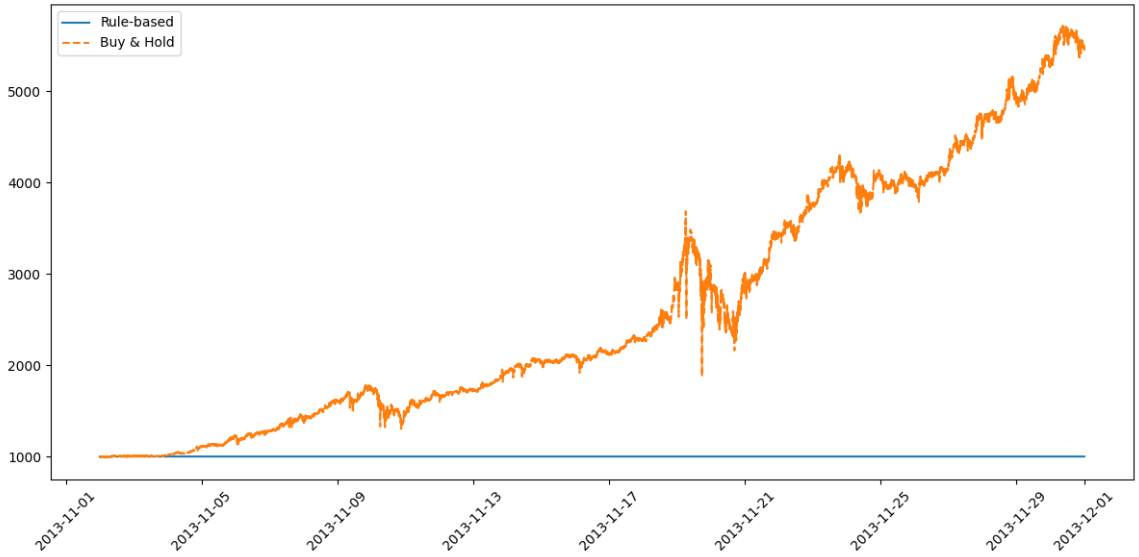
222) September



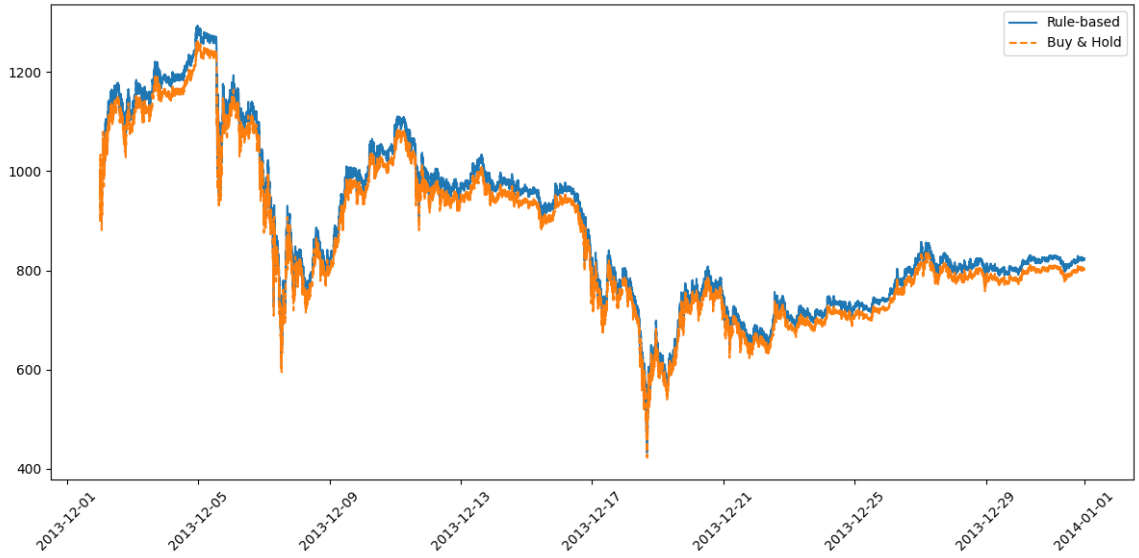
223) October



224) November

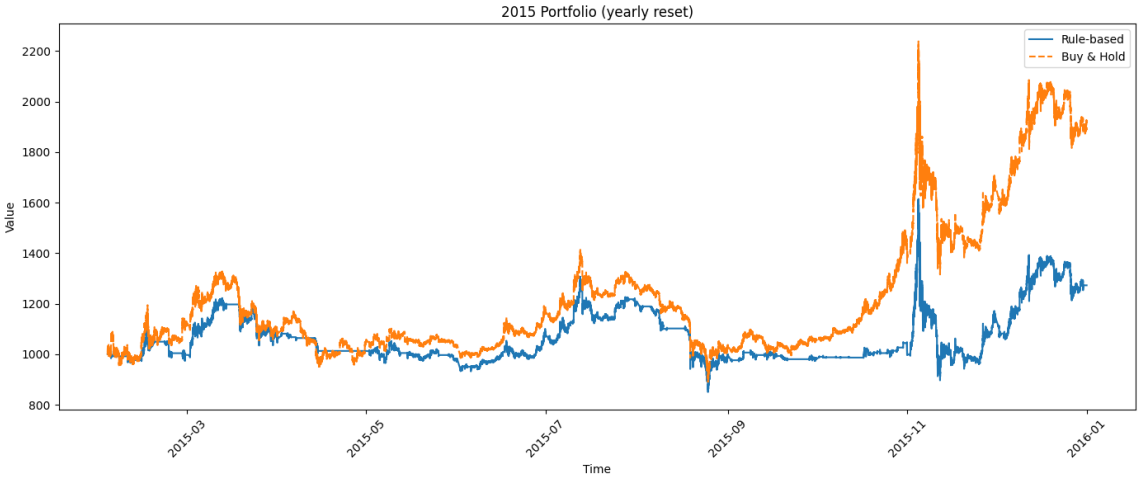


225) December

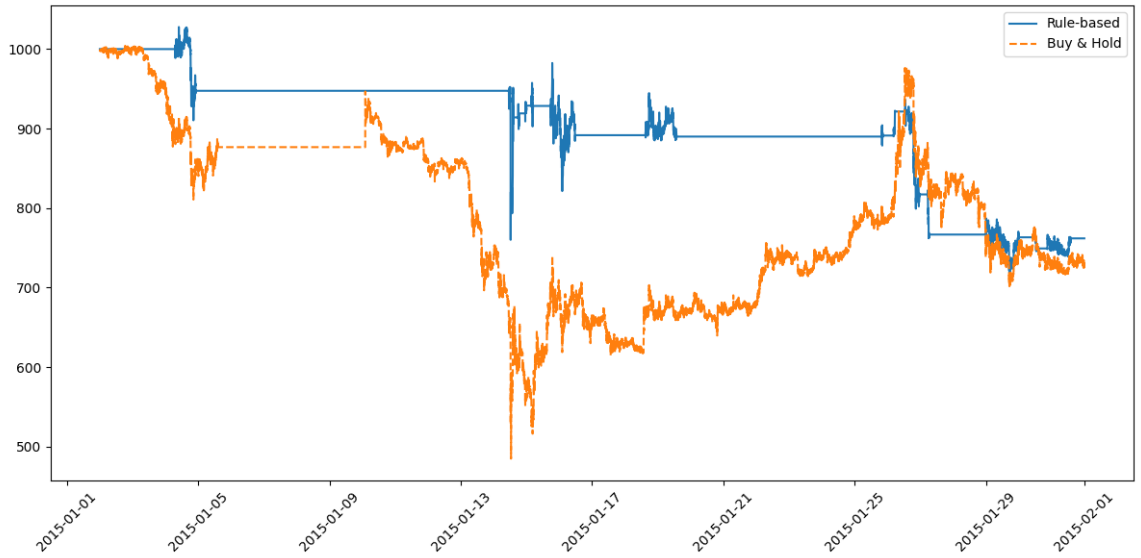


2015

226) Full year

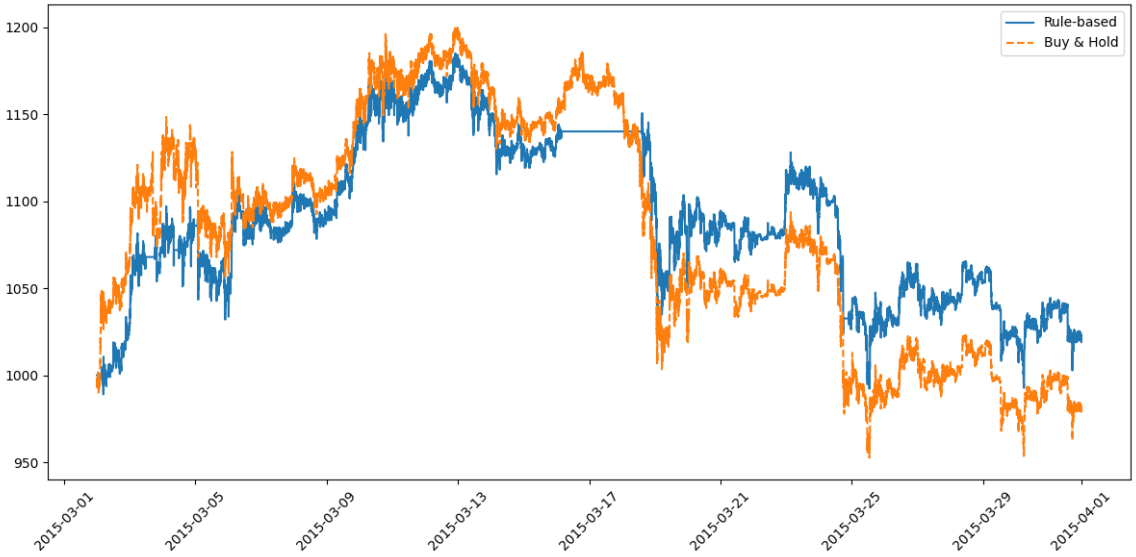


227) January

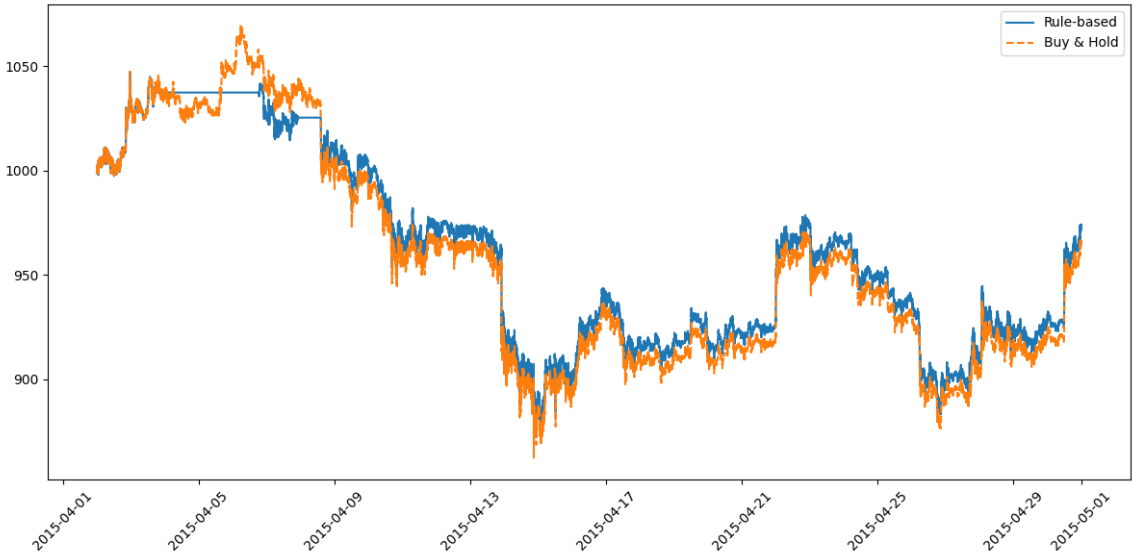


228) February

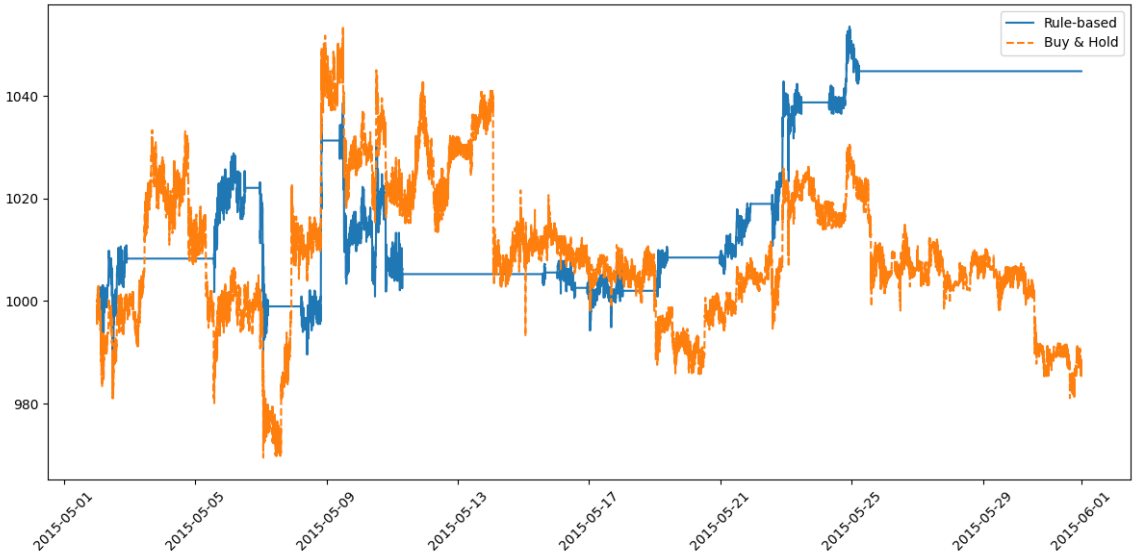
229) March



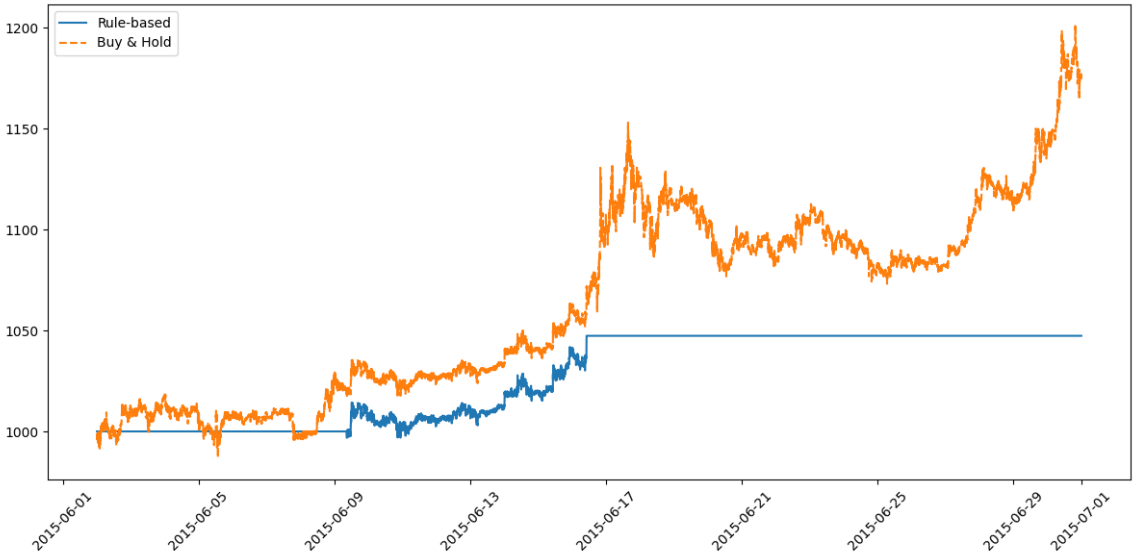
230) April



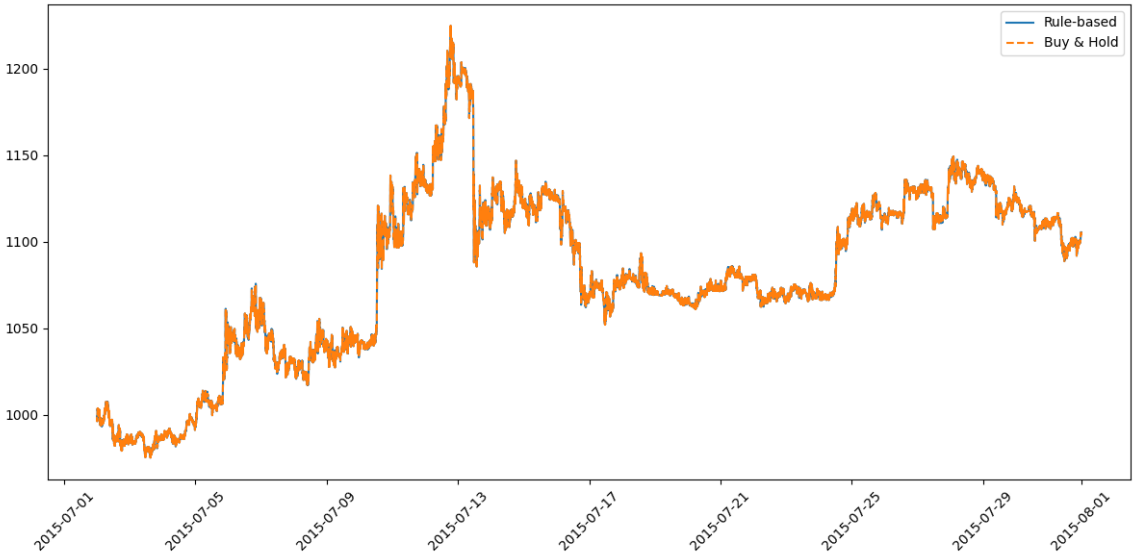
231) May



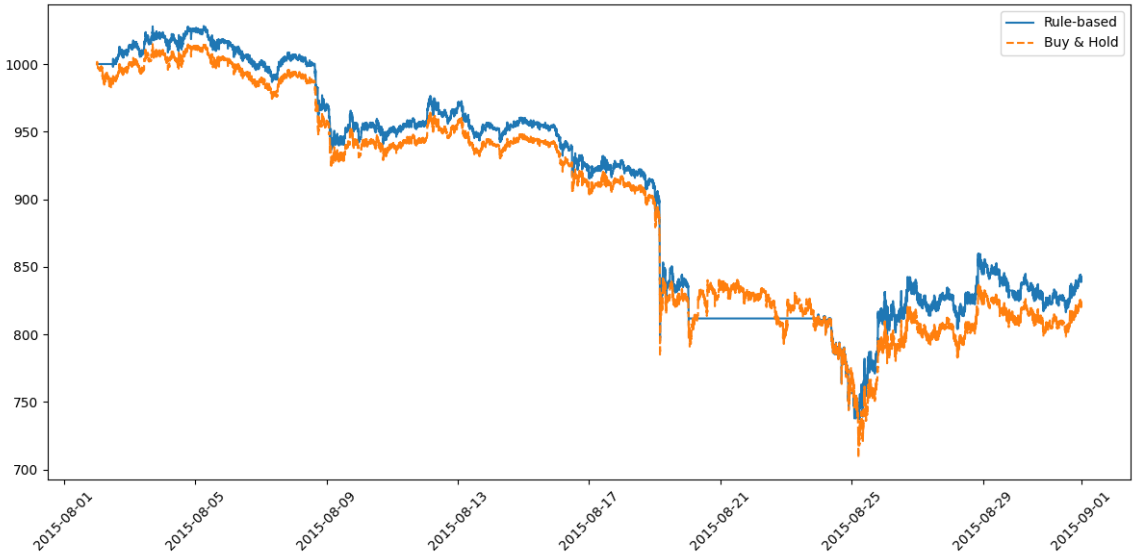
232) June



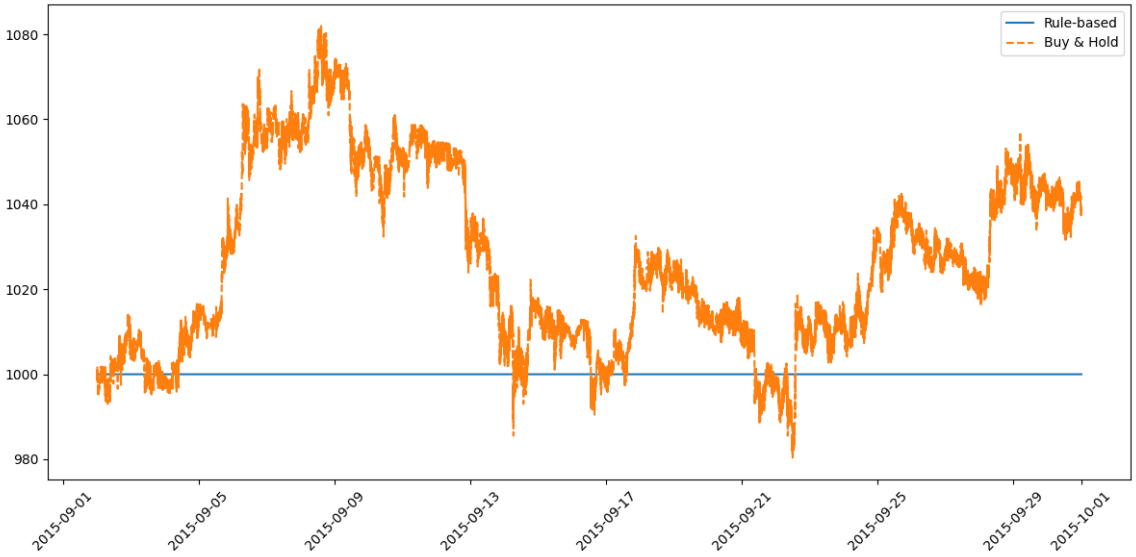
233) July



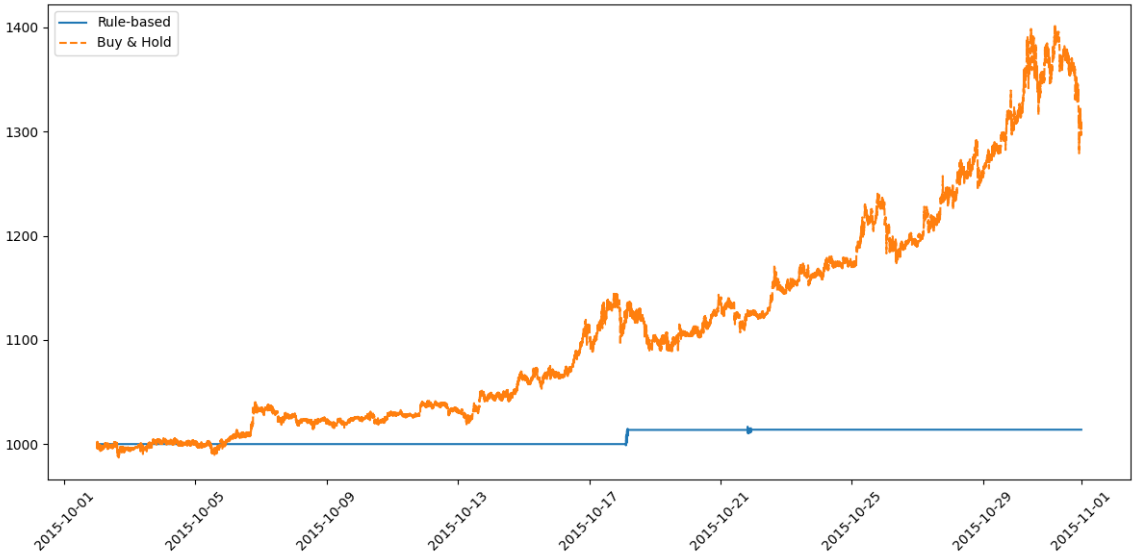
234) August



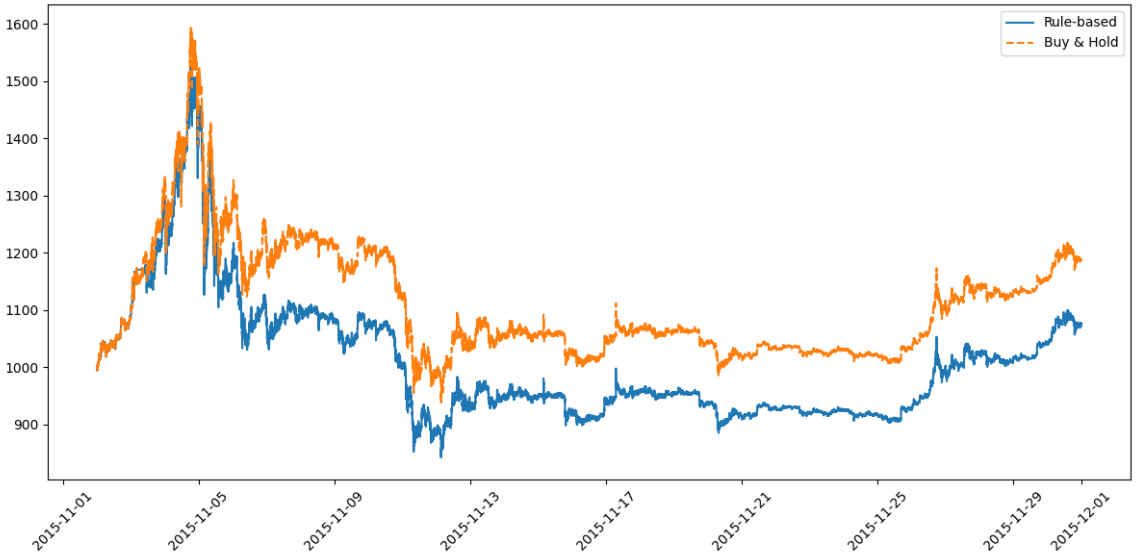
235) September



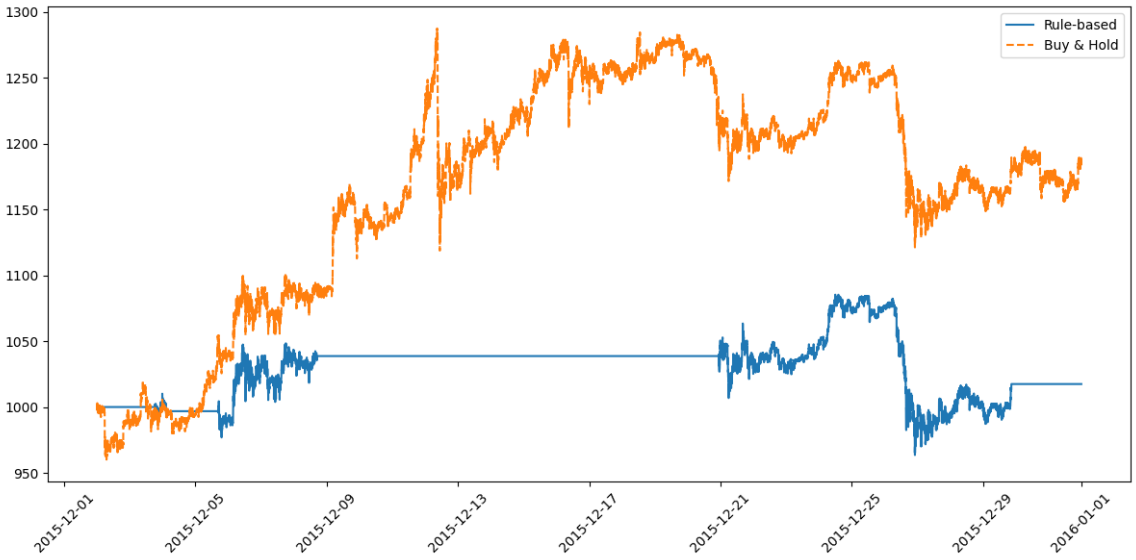
236) October



237) November

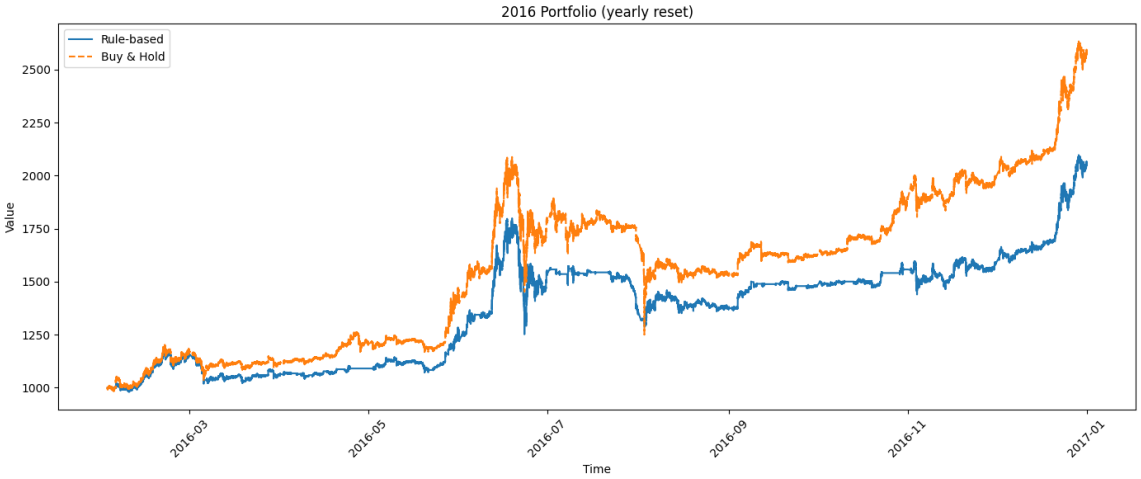


238) December

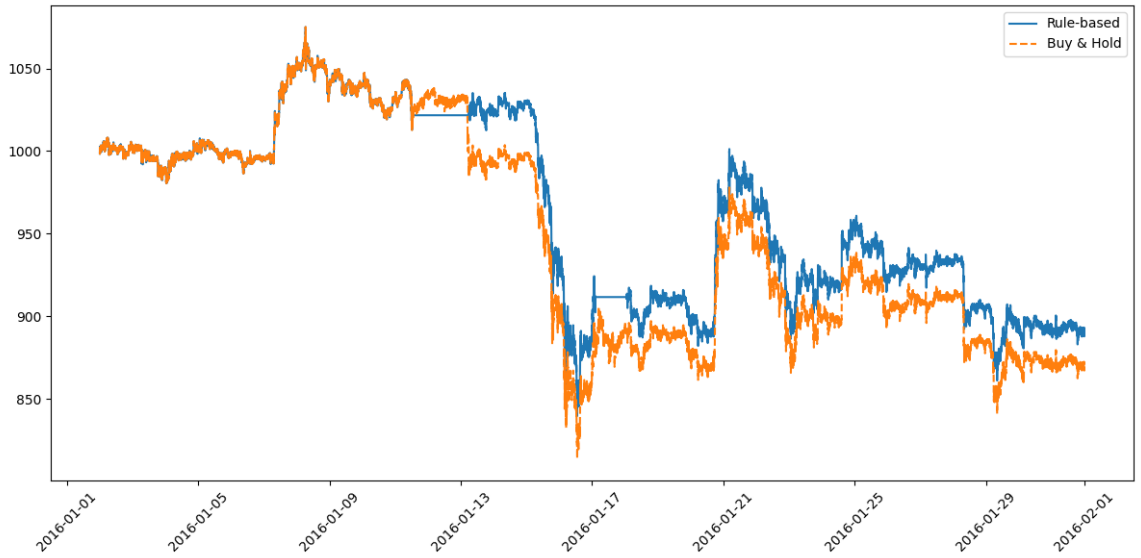


2016

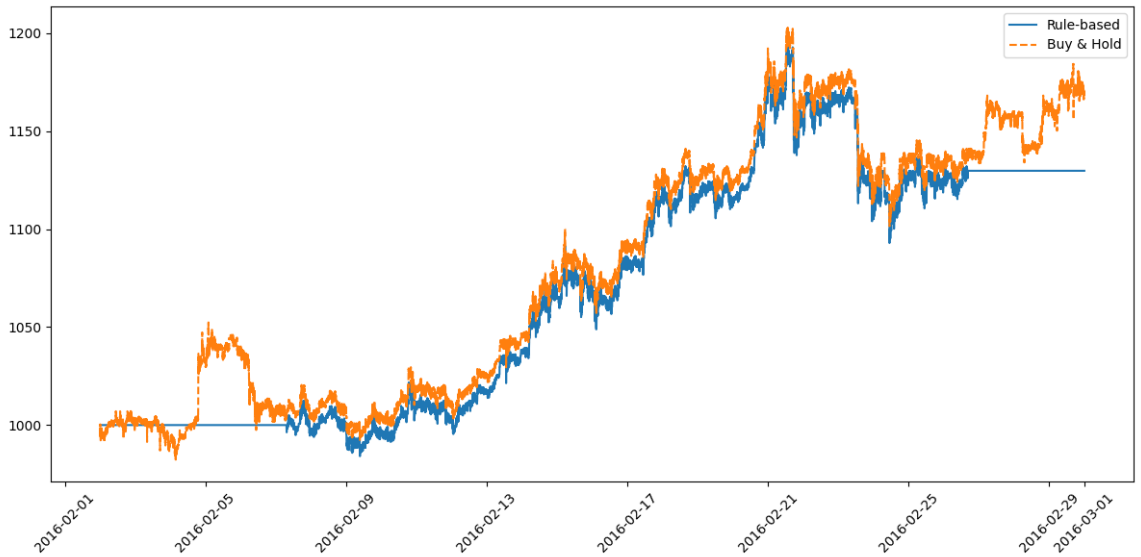
239) Full year



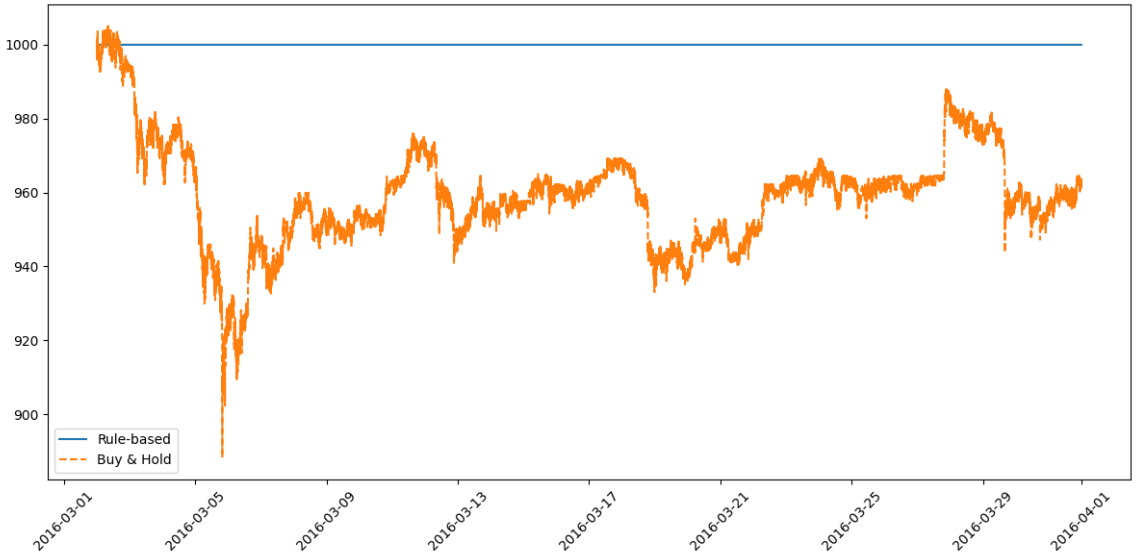
240) January



241) February

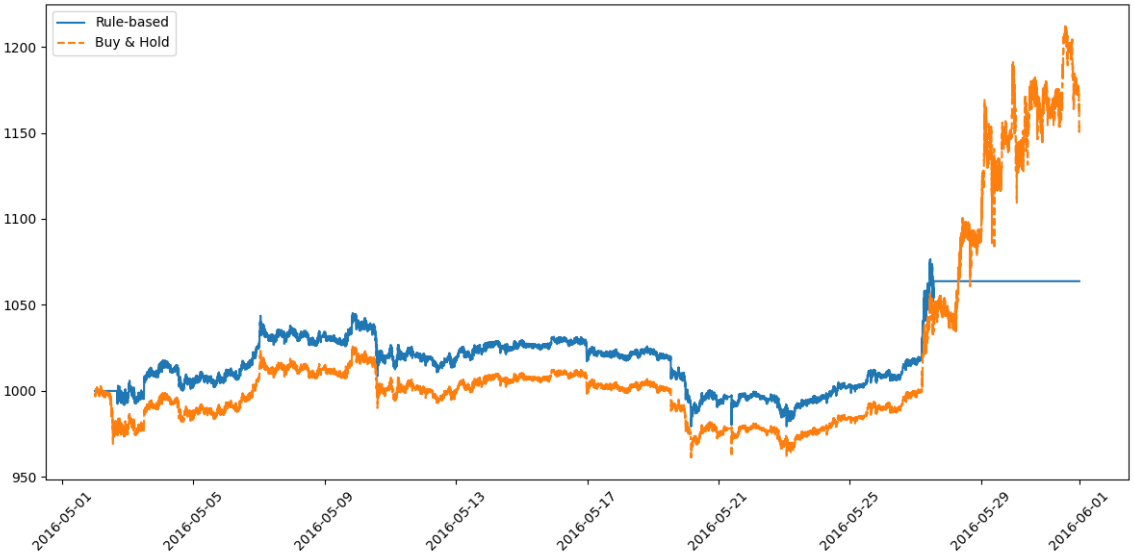


242) March

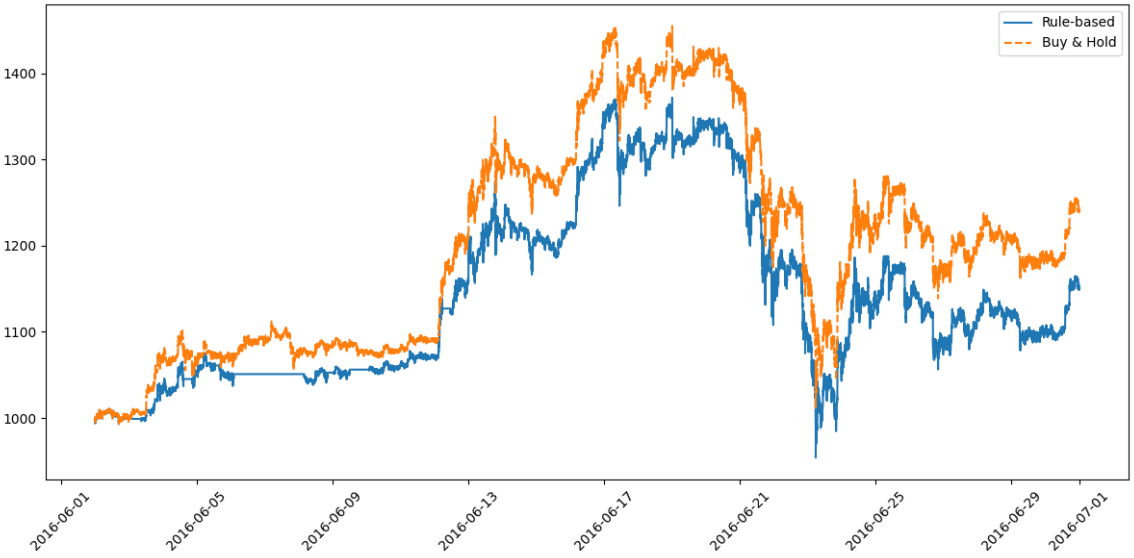


243) April

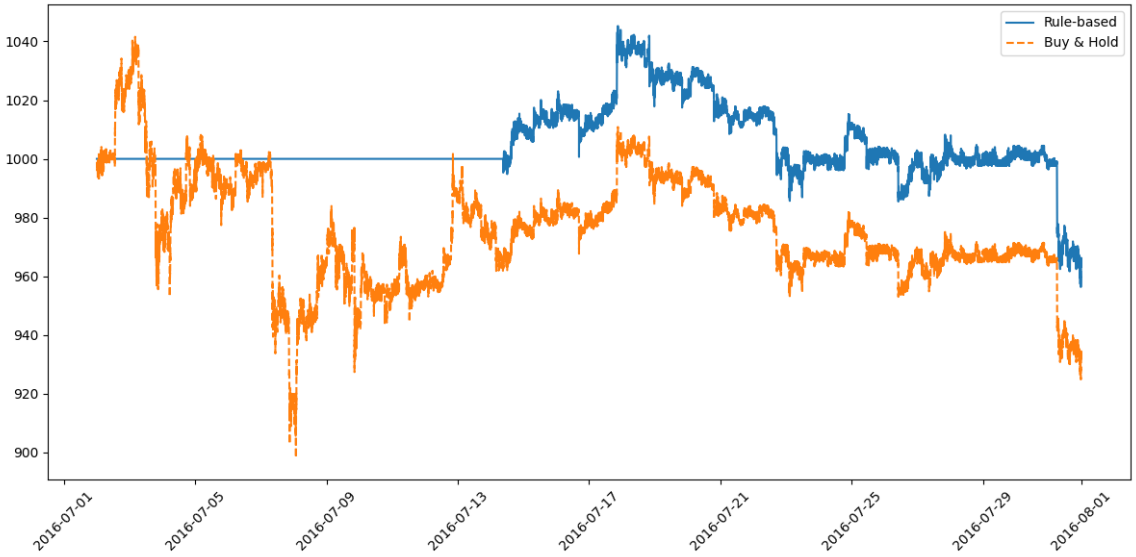
244) May



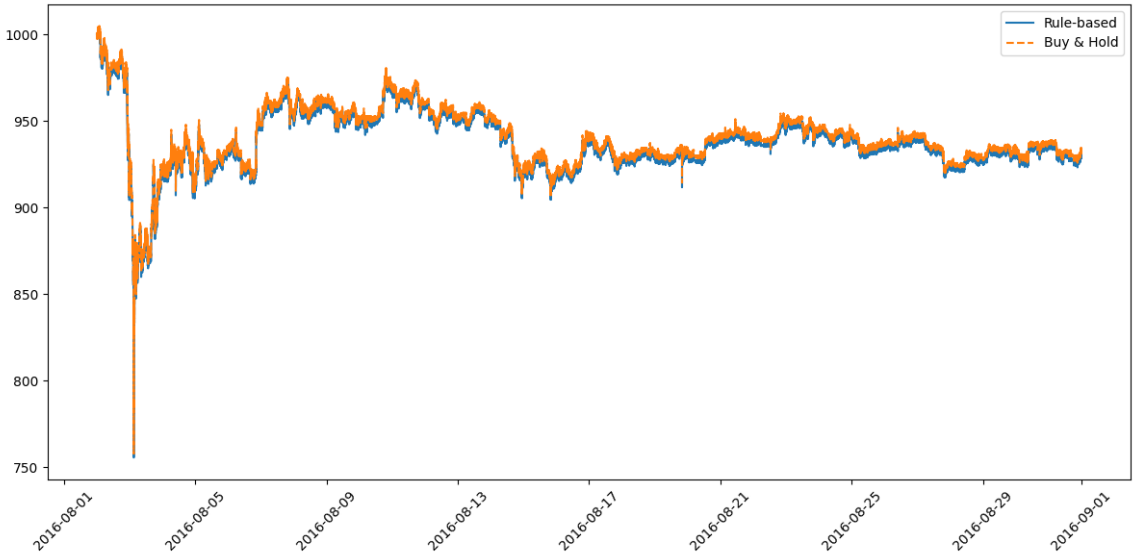
245) June



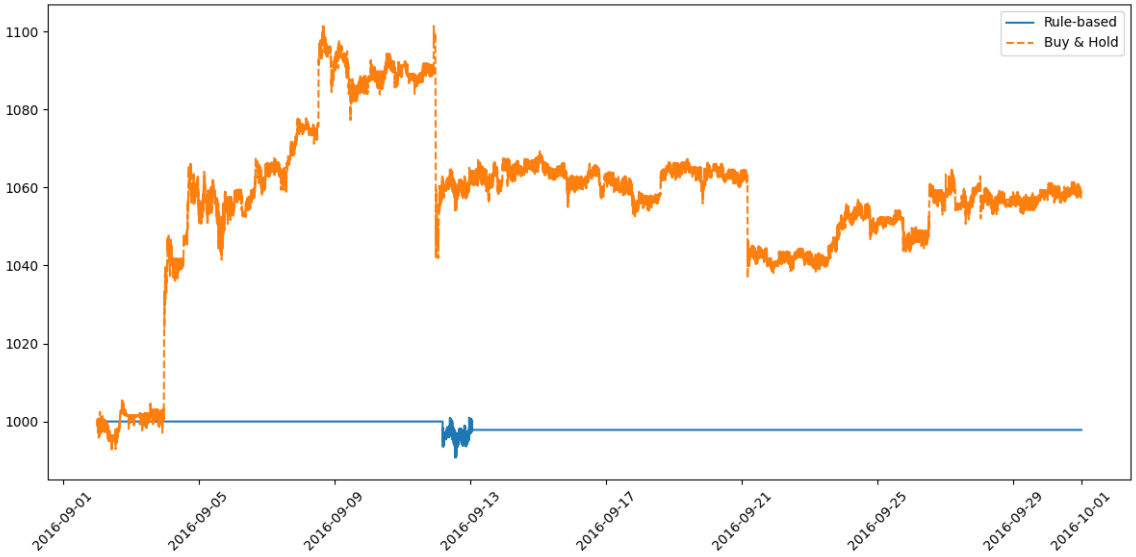
246) July



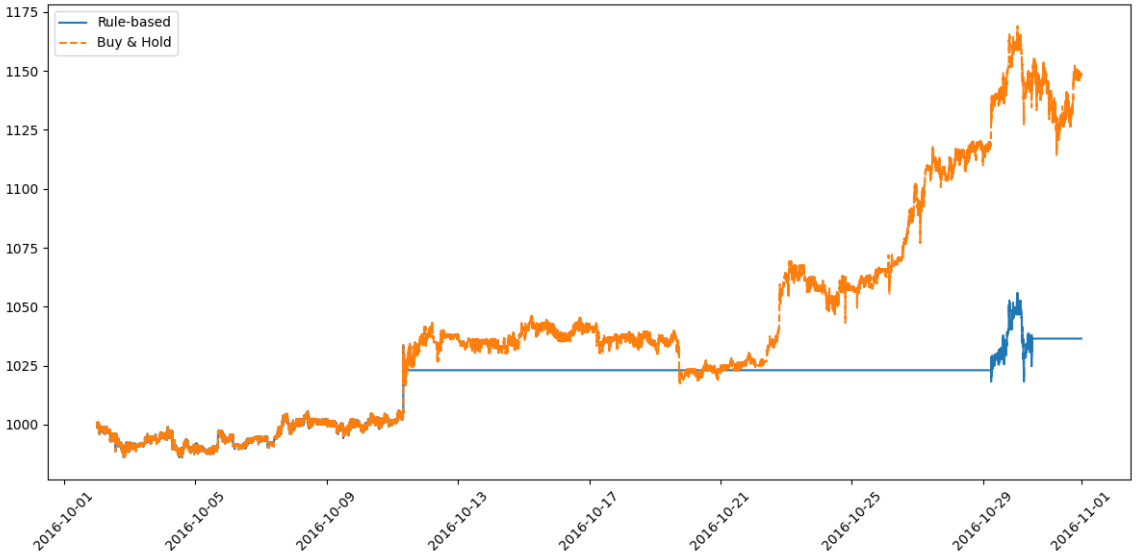
247) August



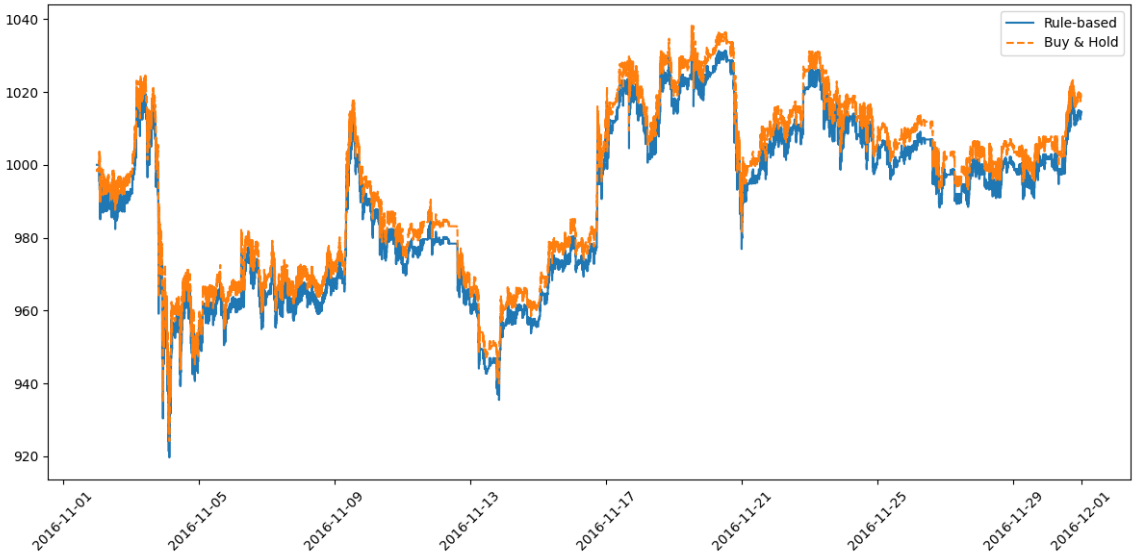
248) September



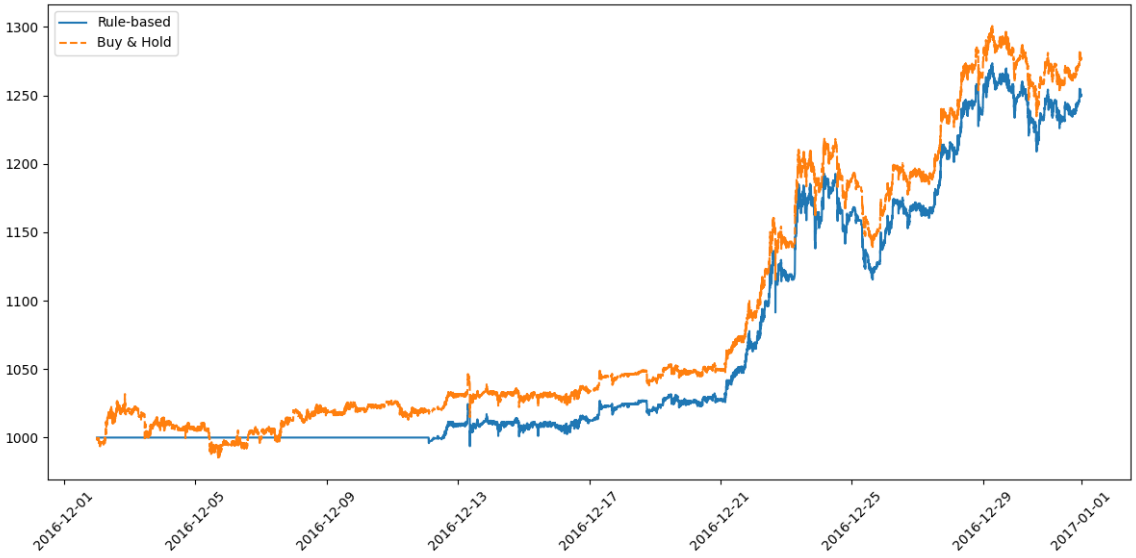
249) October



250) November

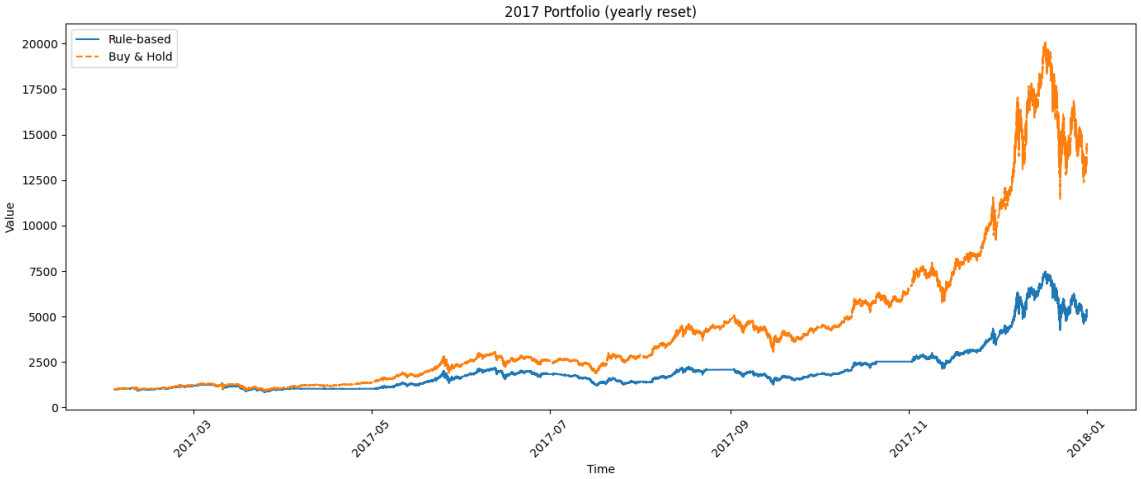


251) December



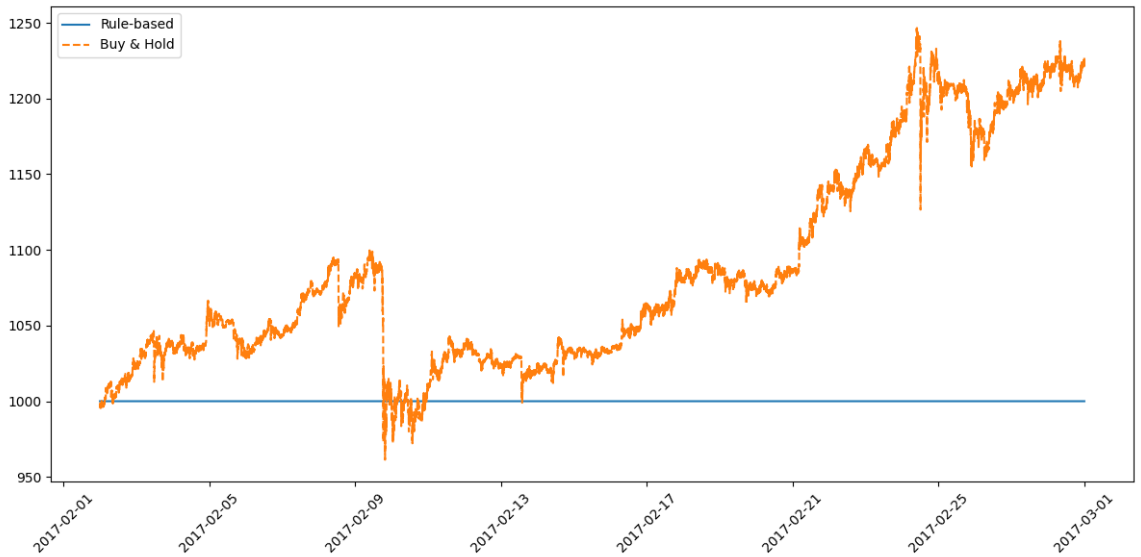
2017

252) Full Year

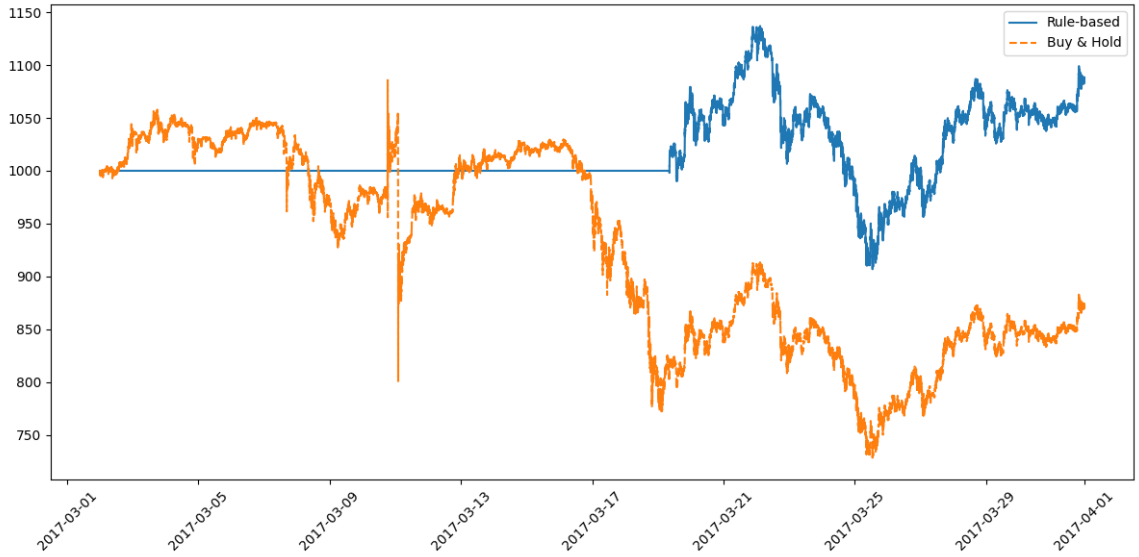


253) January

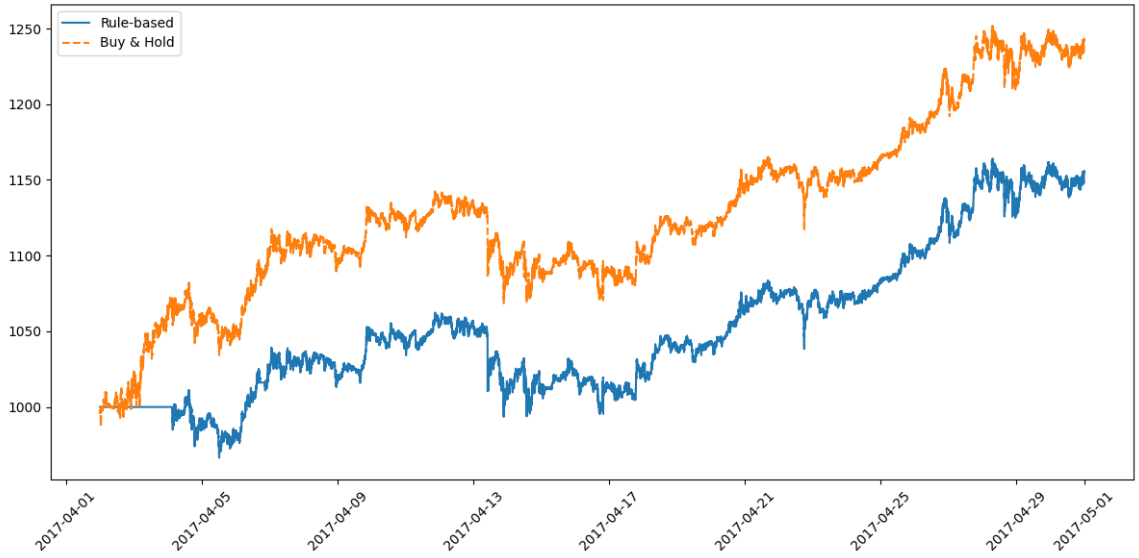
254) February



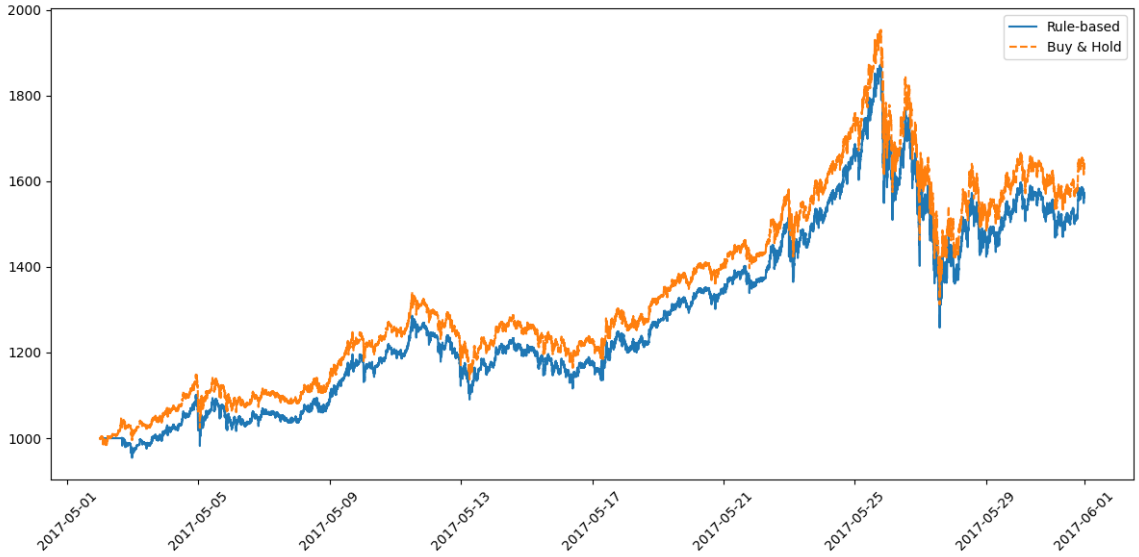
255) March



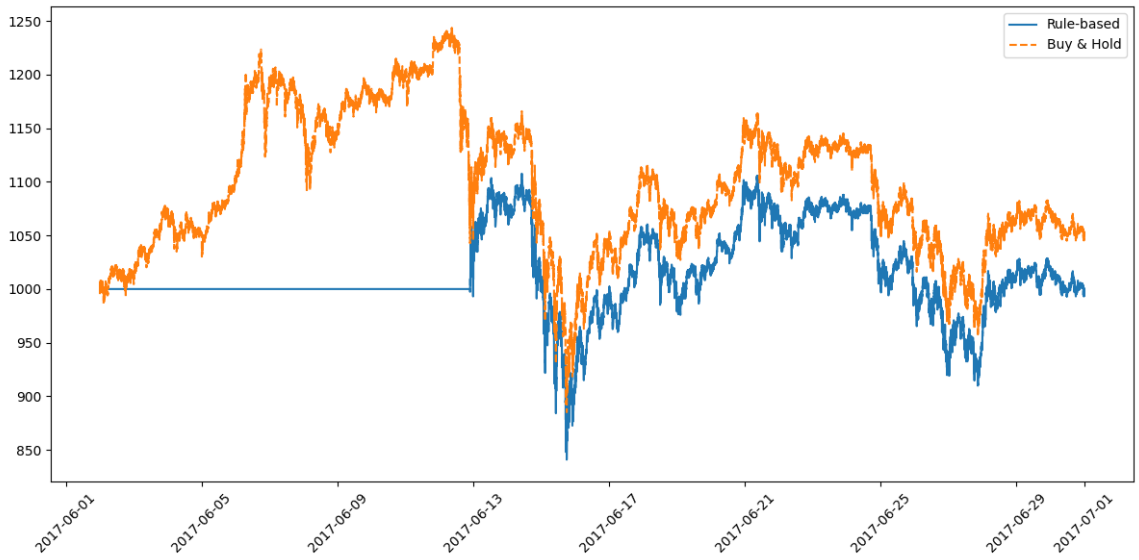
256) April



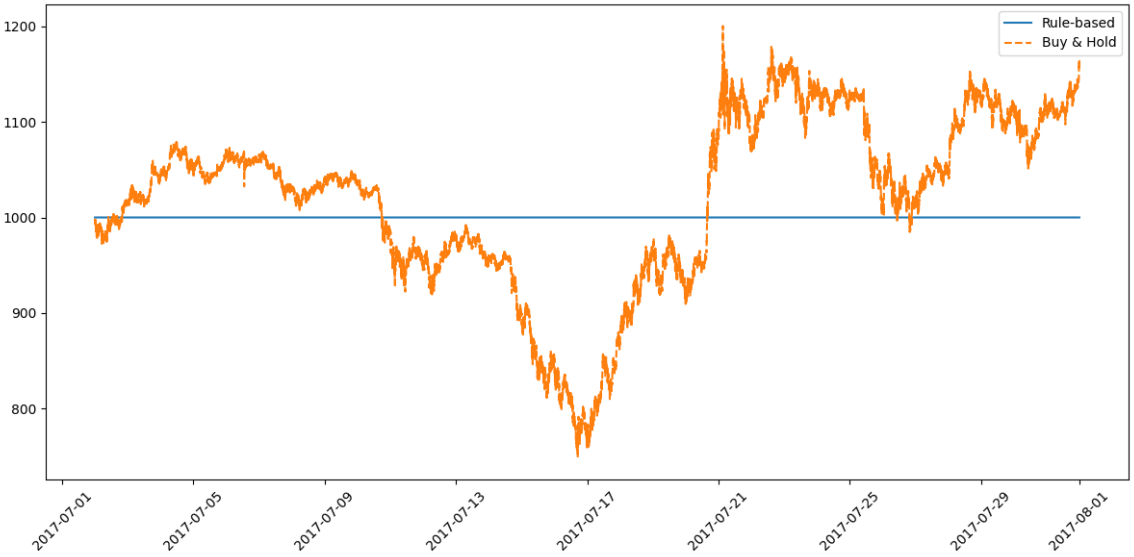
257) May



258) June

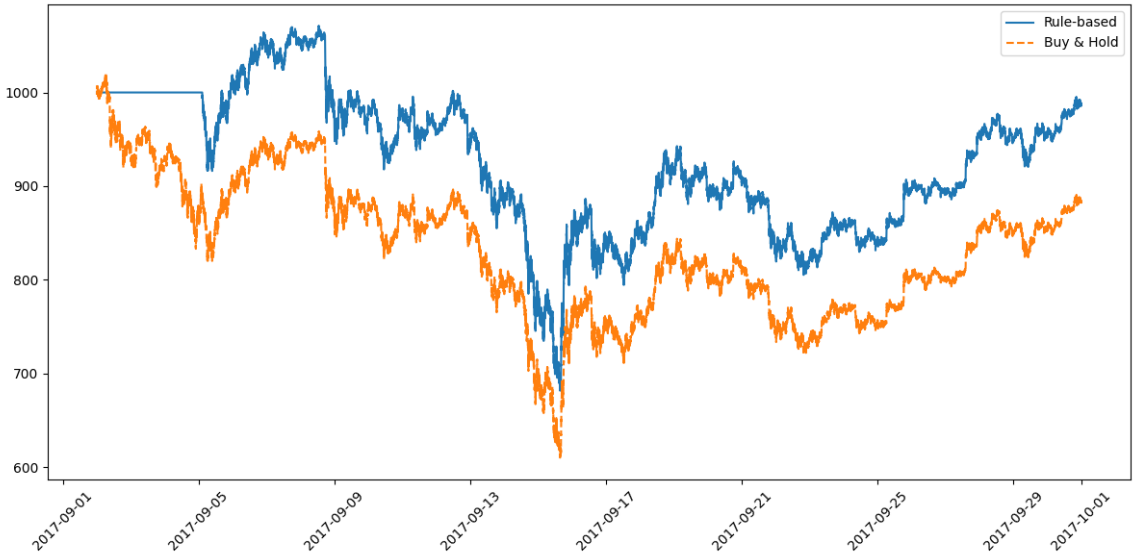


259) July

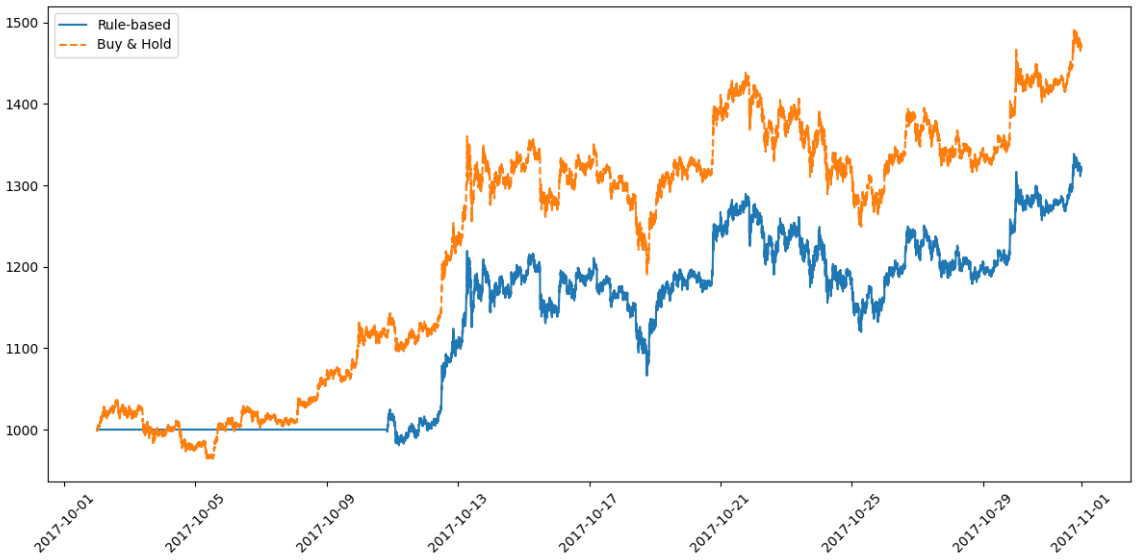


260) August

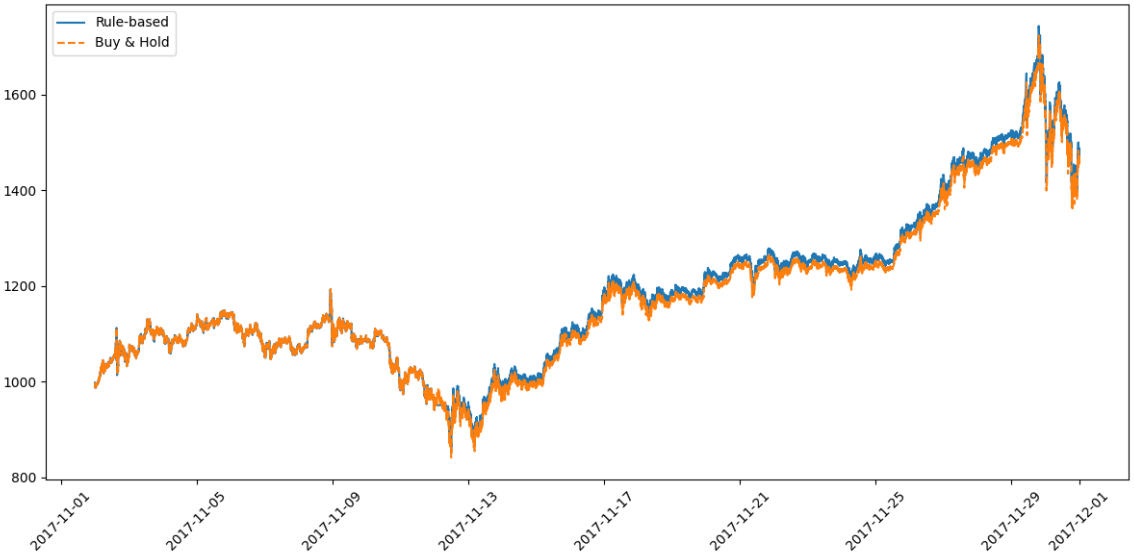
261) September



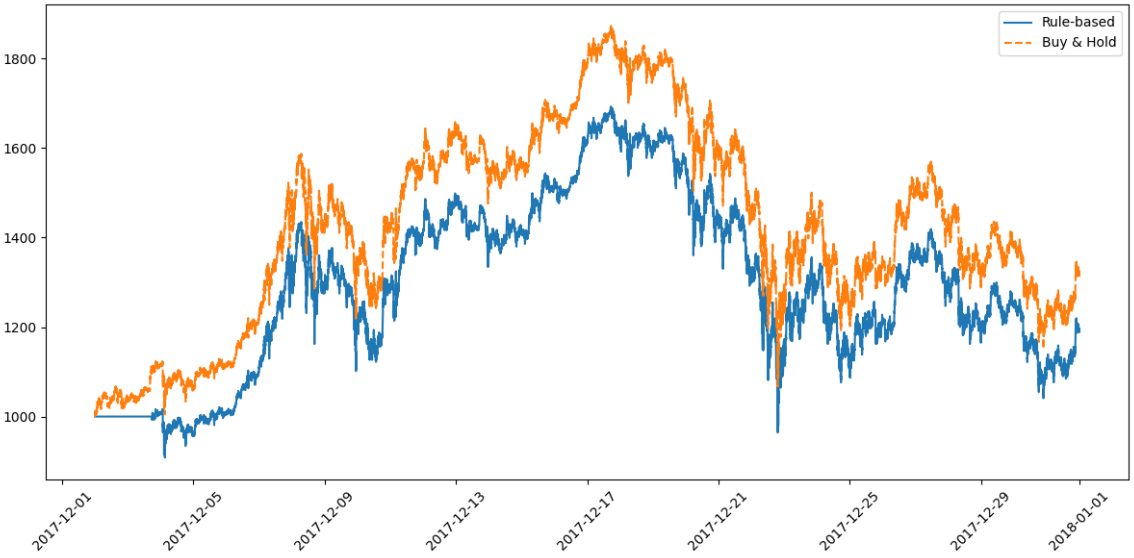
262) October



263) November

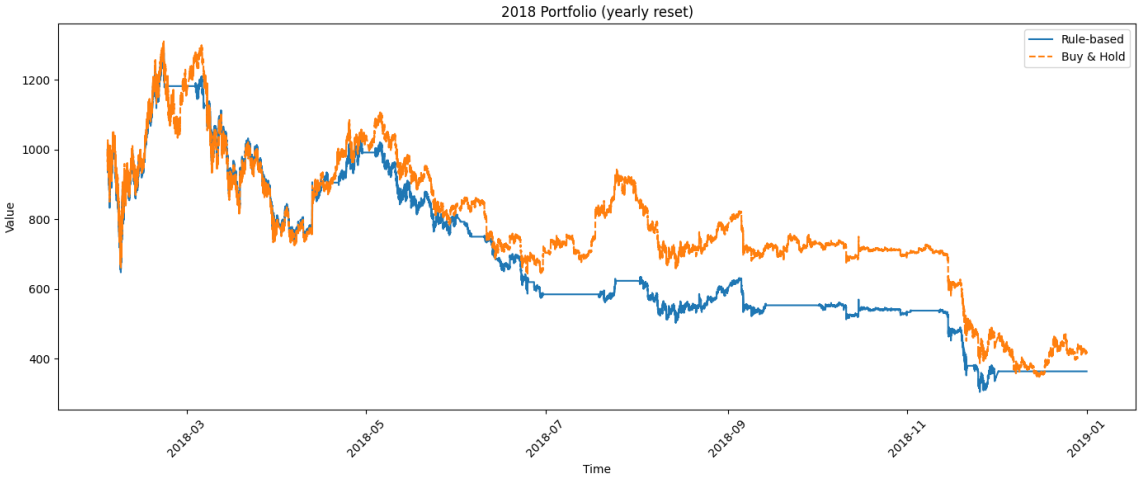


264) December

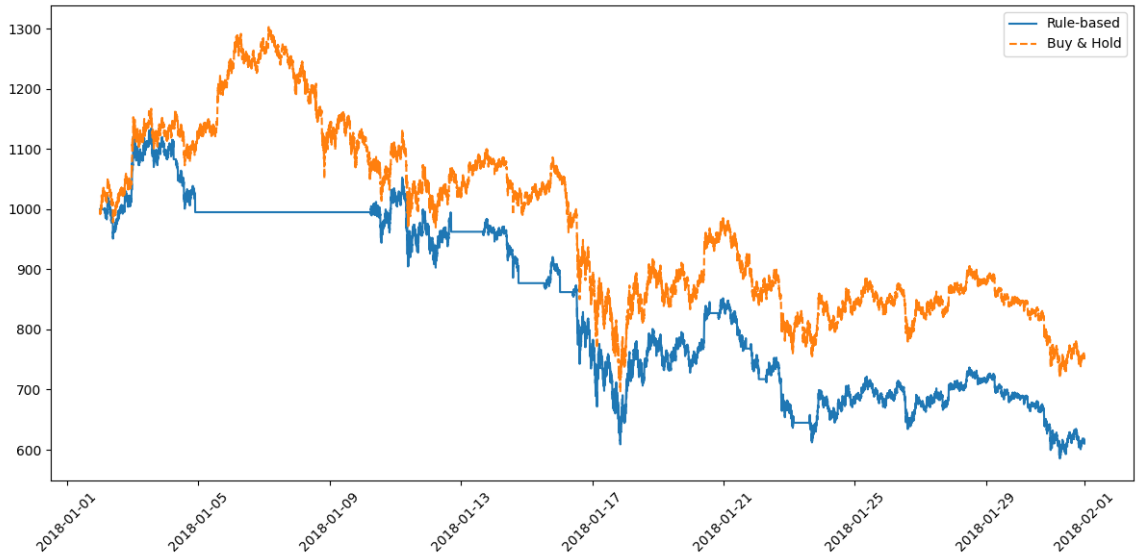


2018

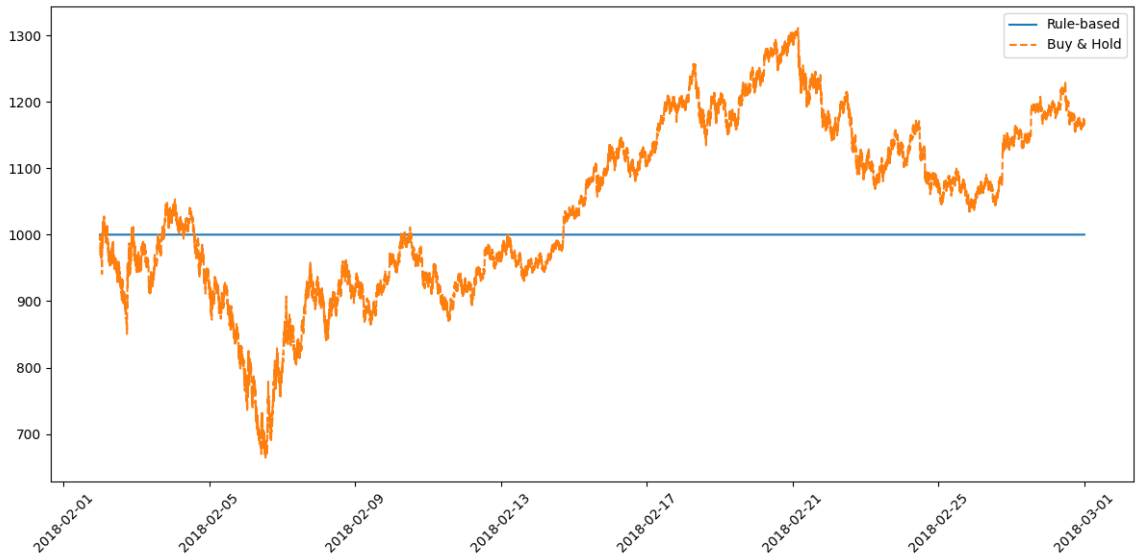
265) Full Year



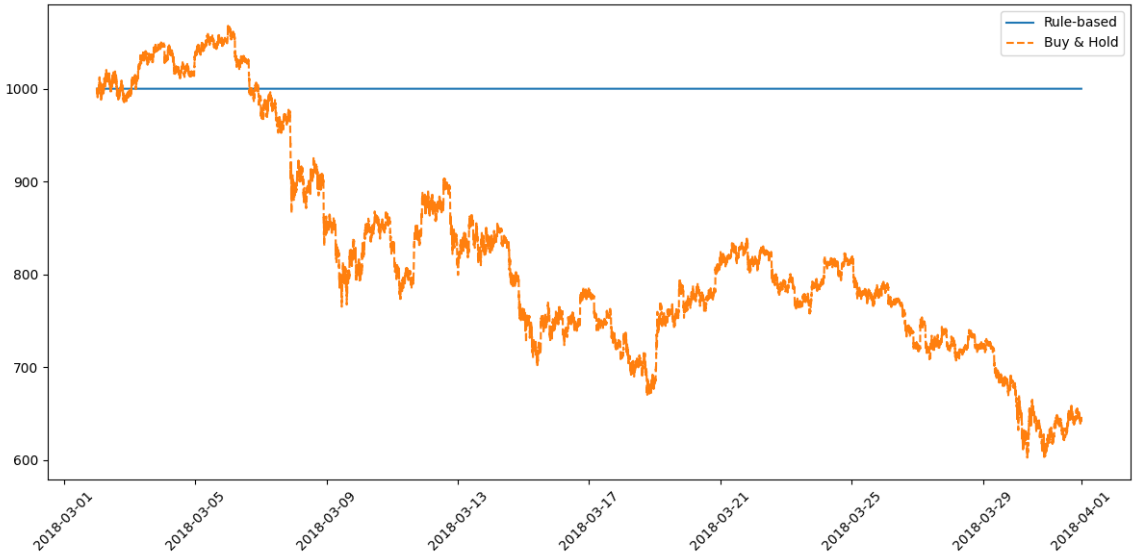
266) January



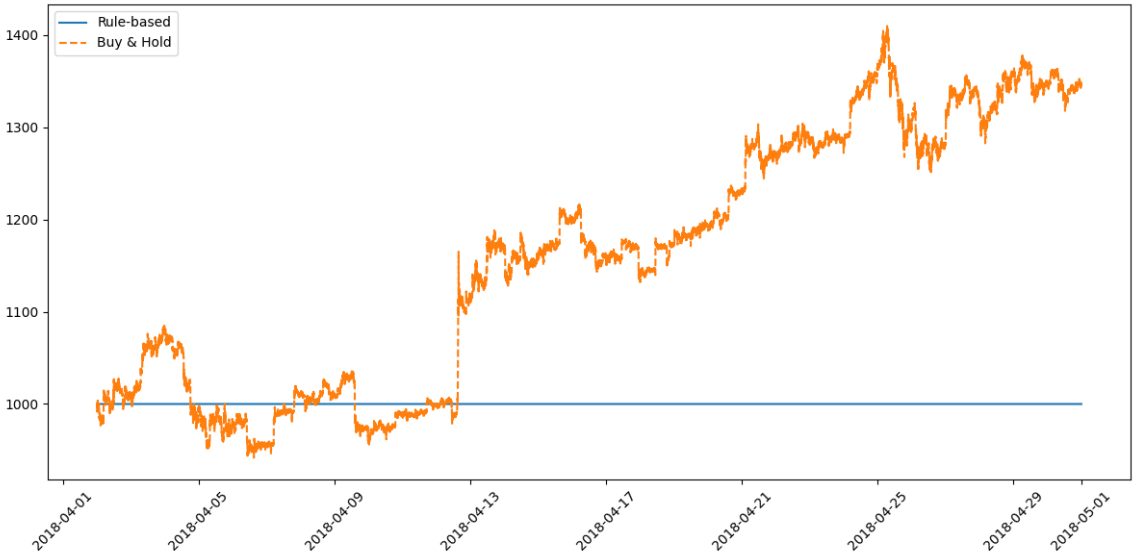
267) February



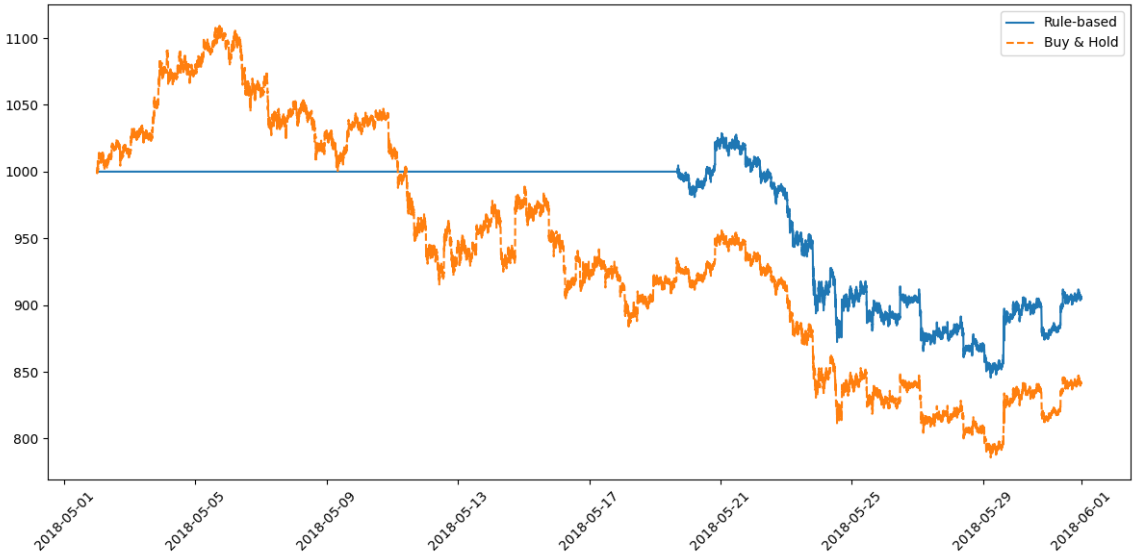
268) March



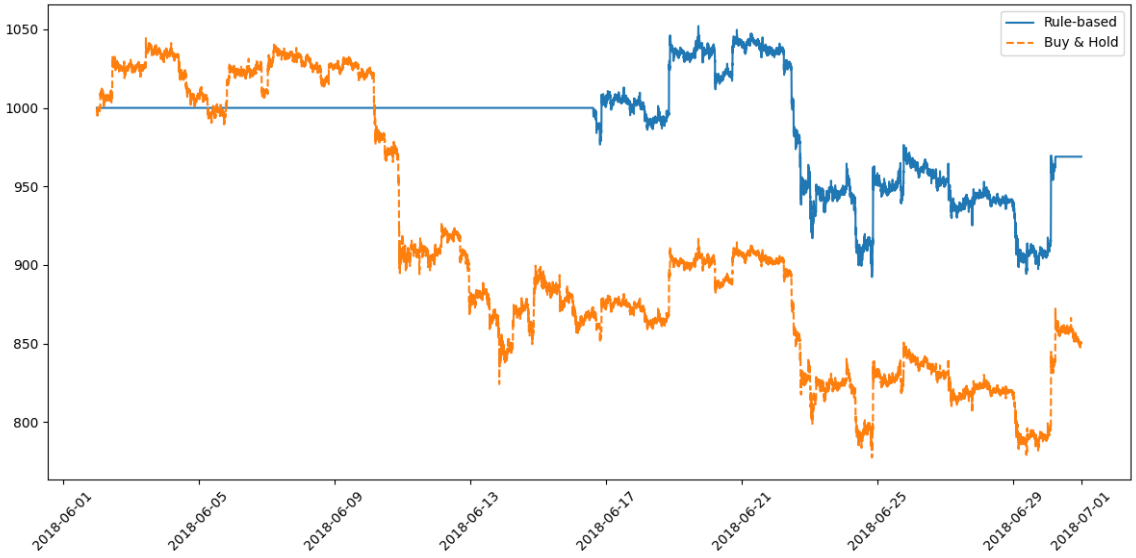
269) April



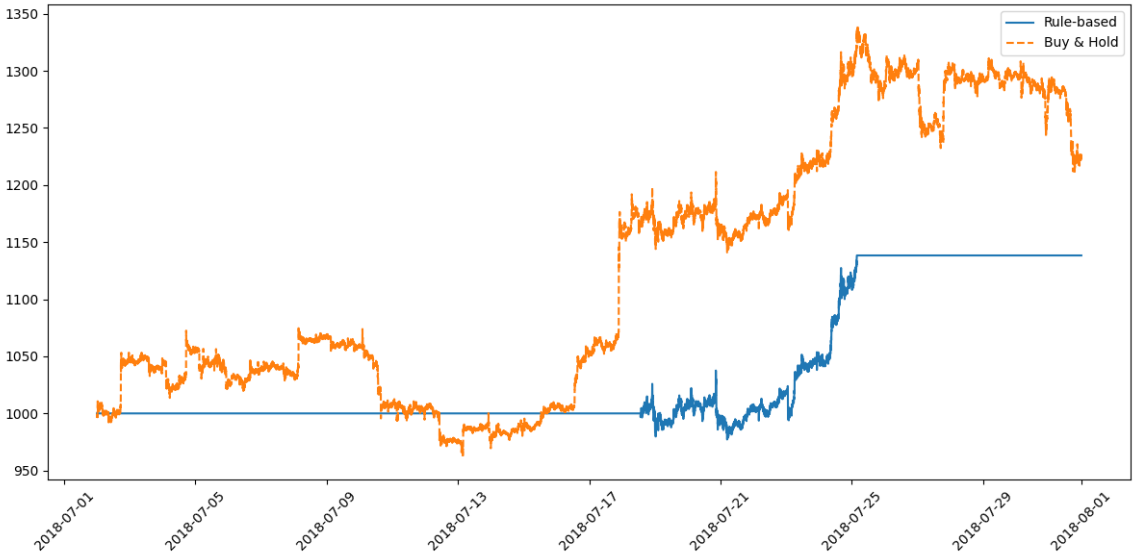
270) May



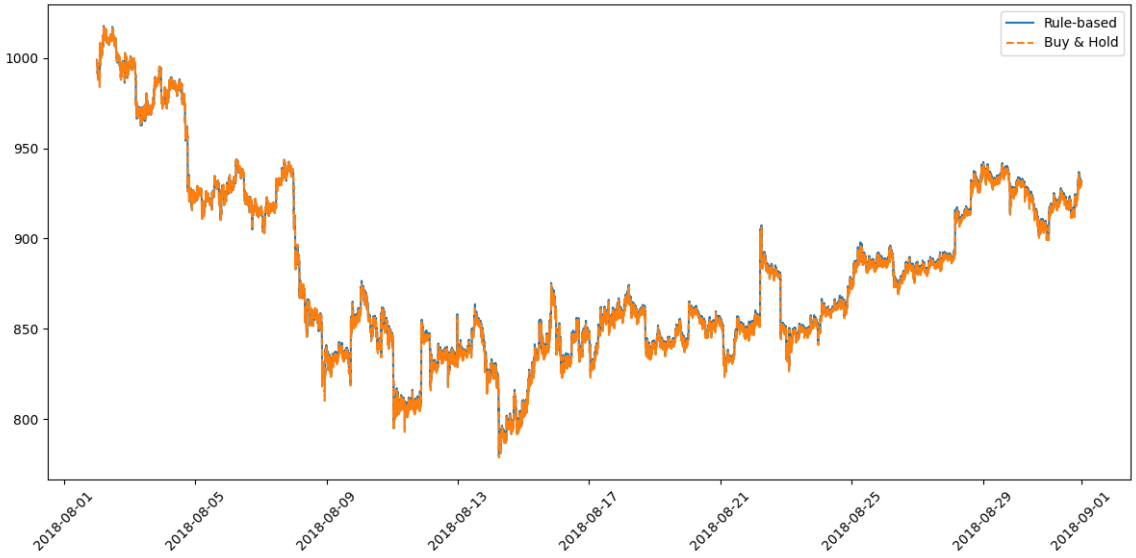
271) June



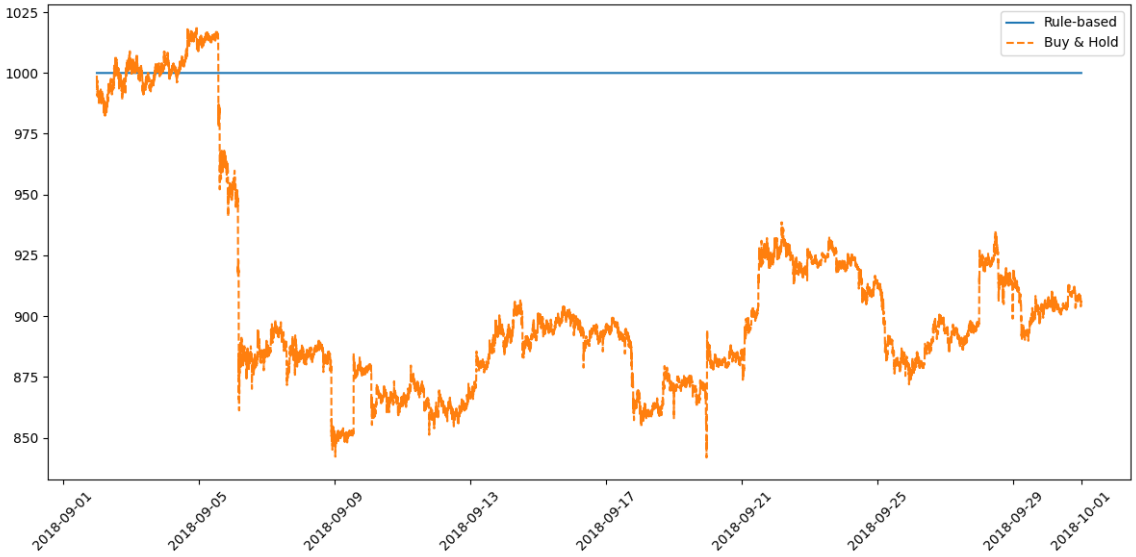
272) July



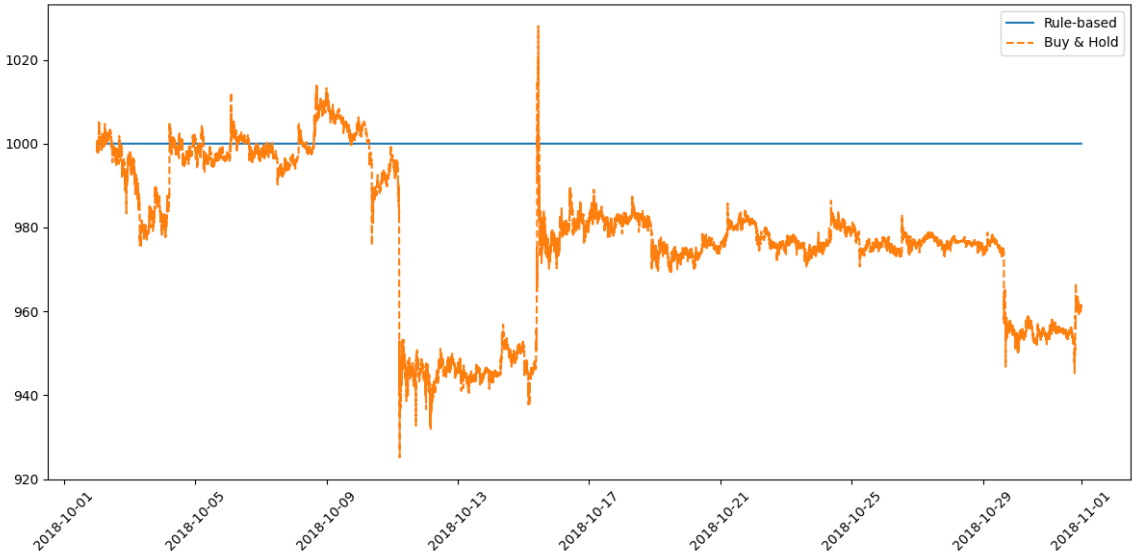
273) August



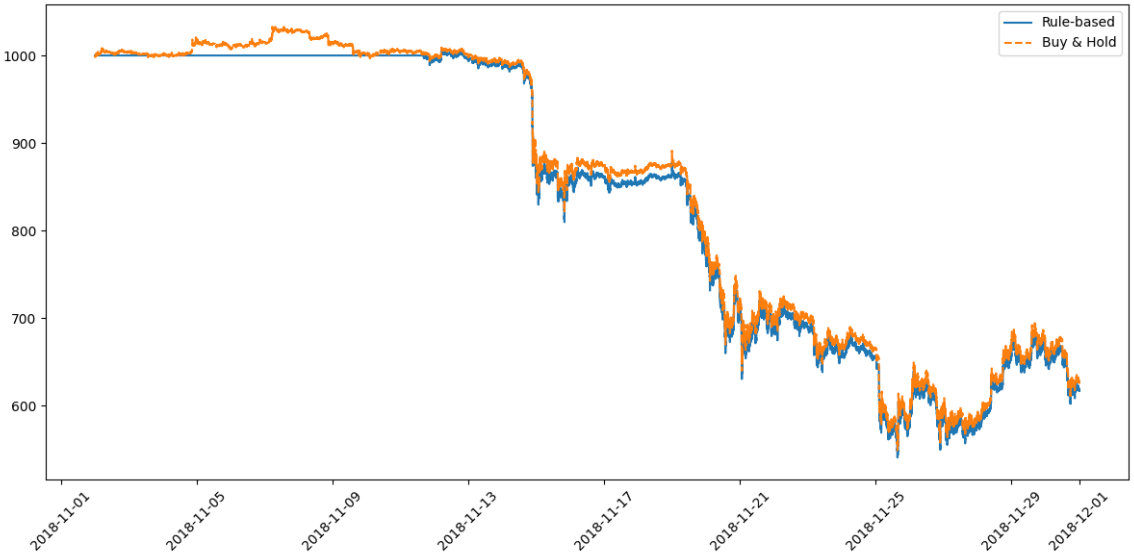
274) September



275) October



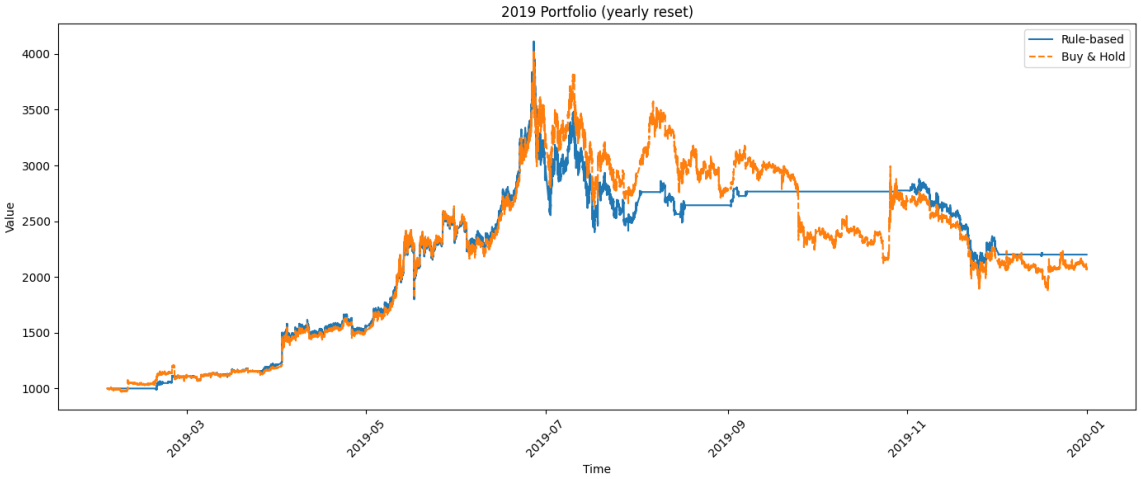
276) November



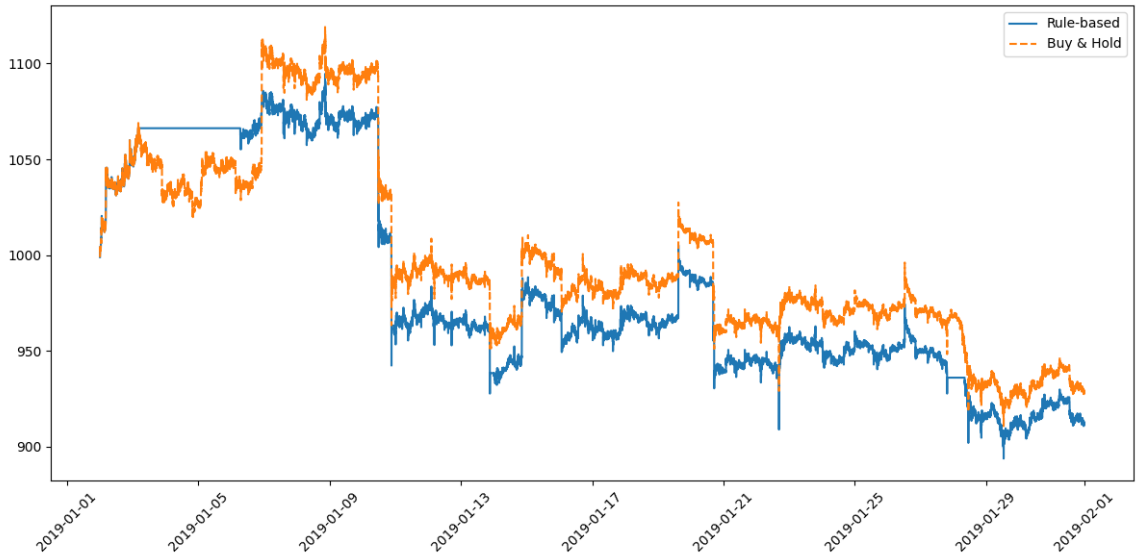
277) December

2019

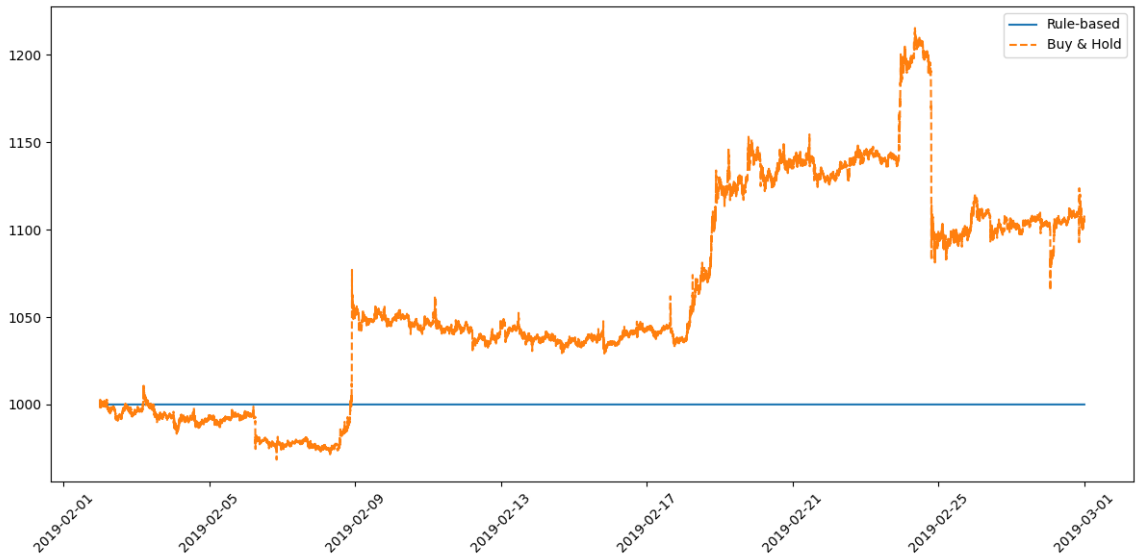
278) Full Year



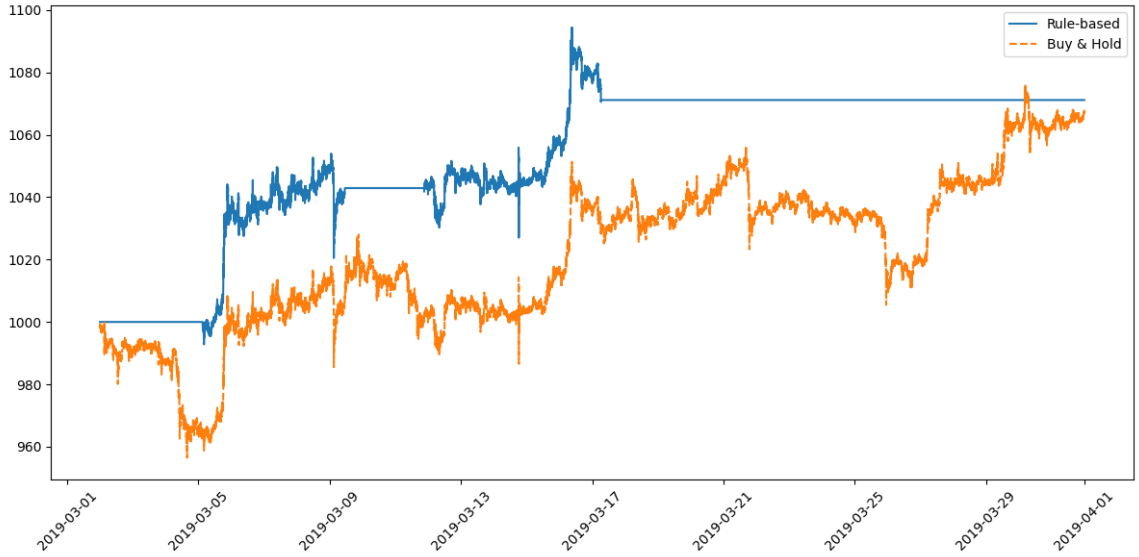
279) January



280) February

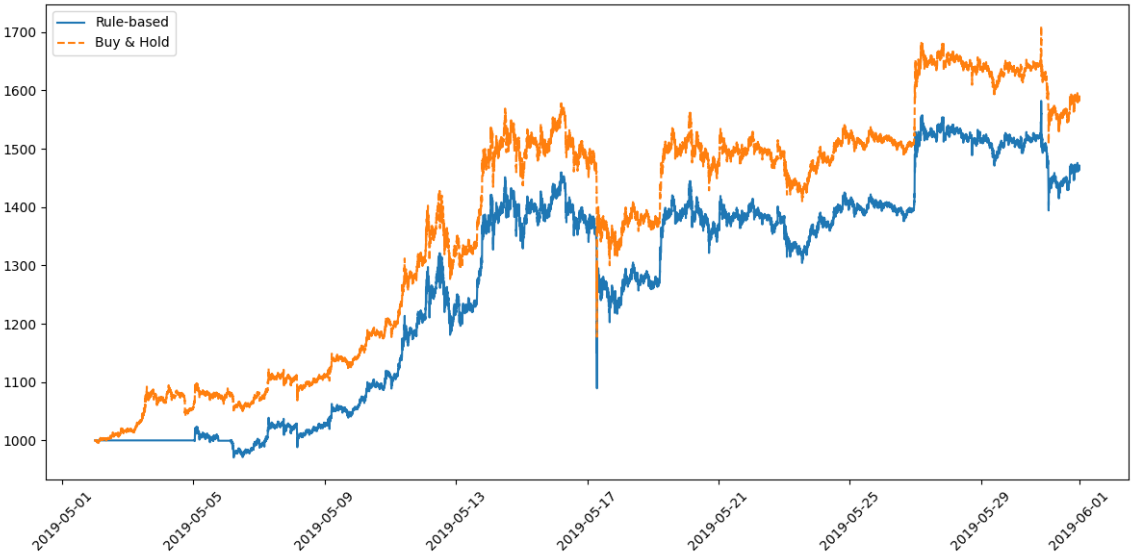


281) March



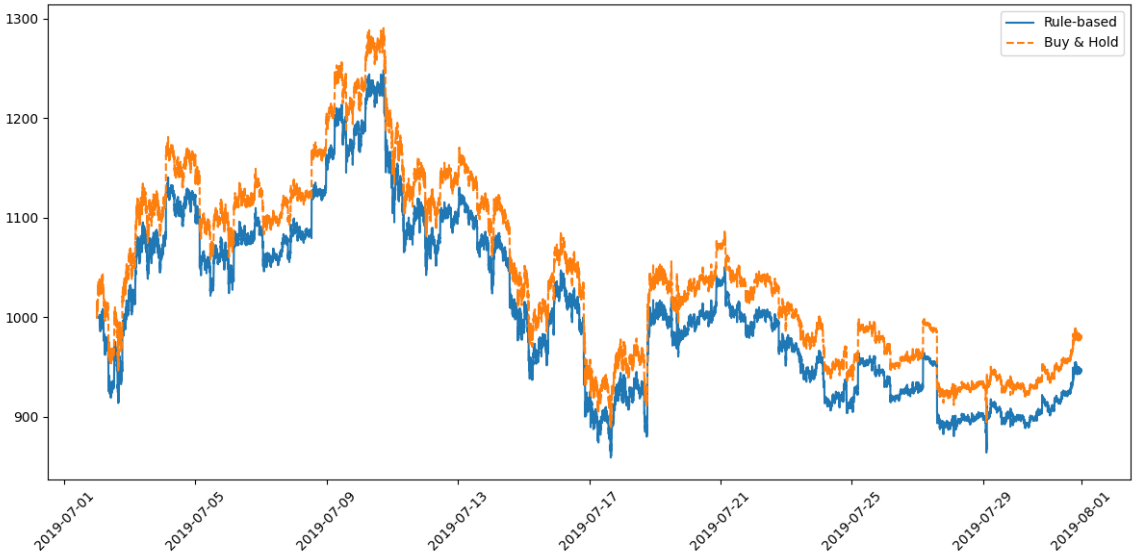
282) April

283) May



284) June

285) July

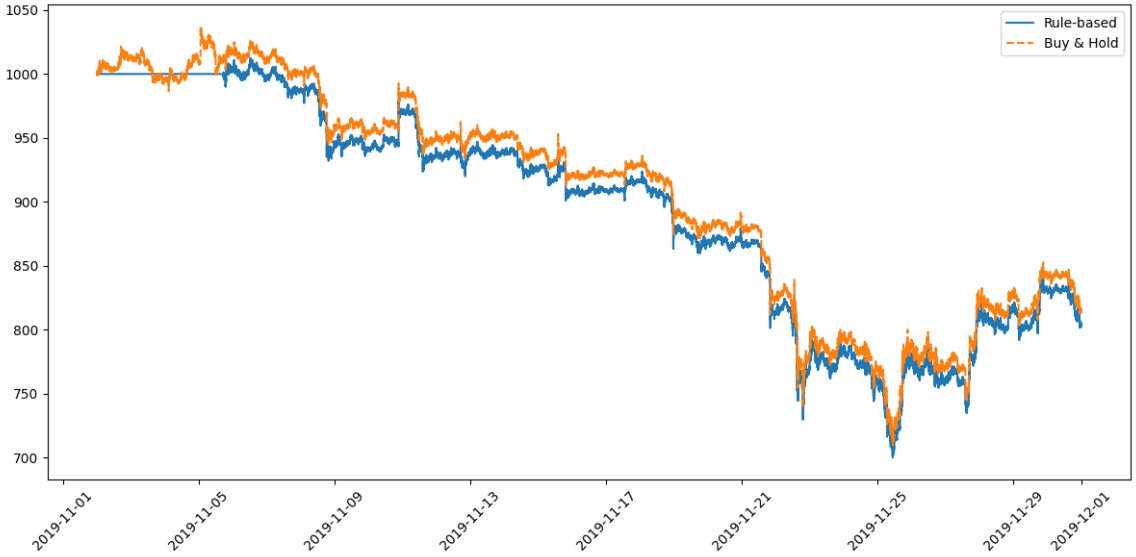


286) August

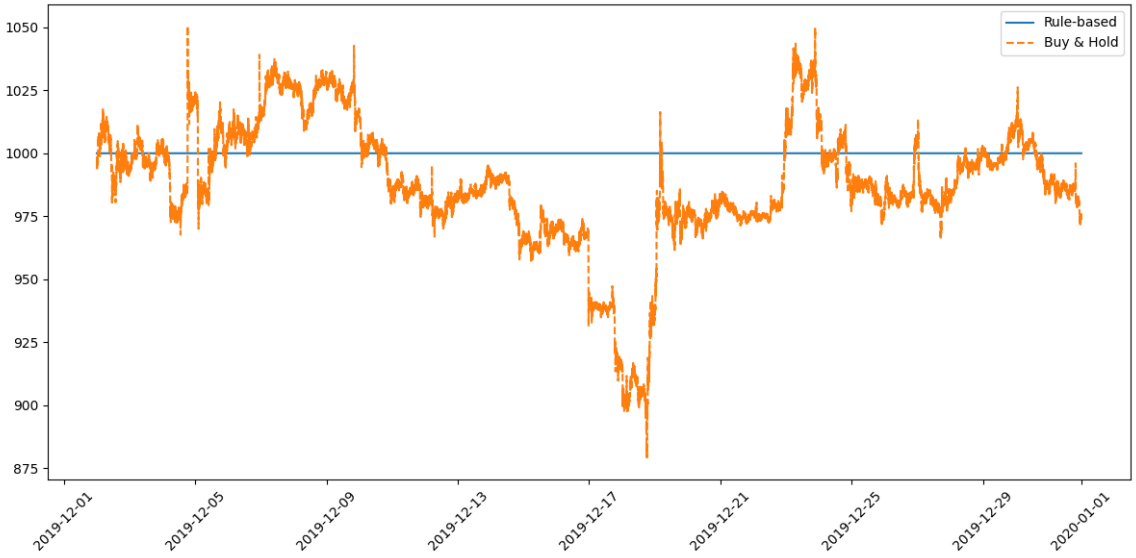
287) September

288) October

289) November

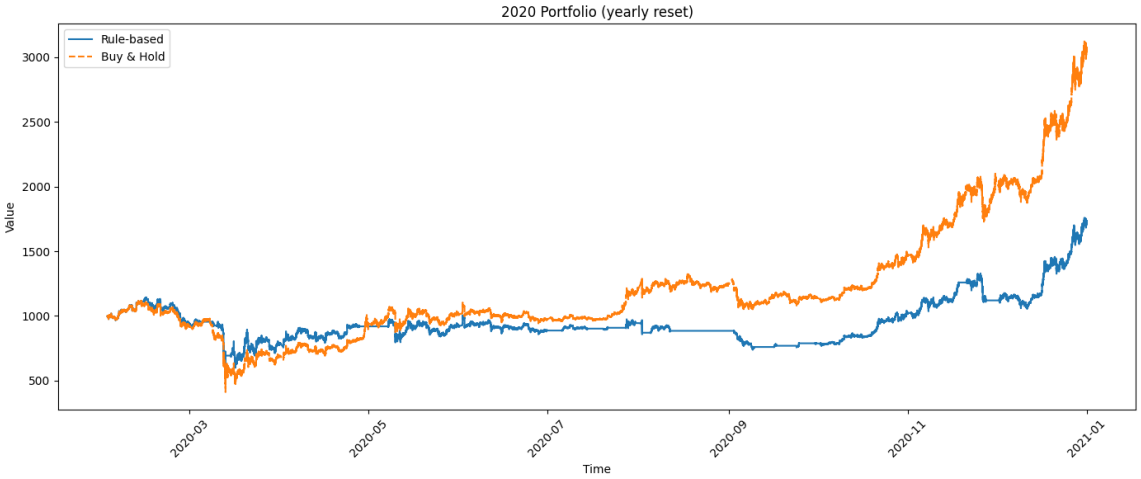


290) December

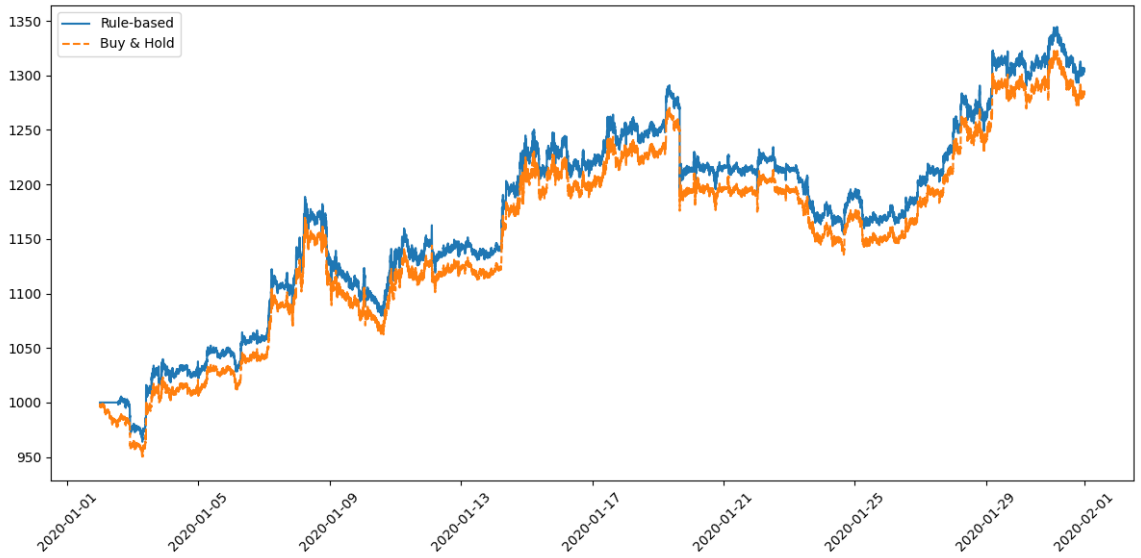


2020

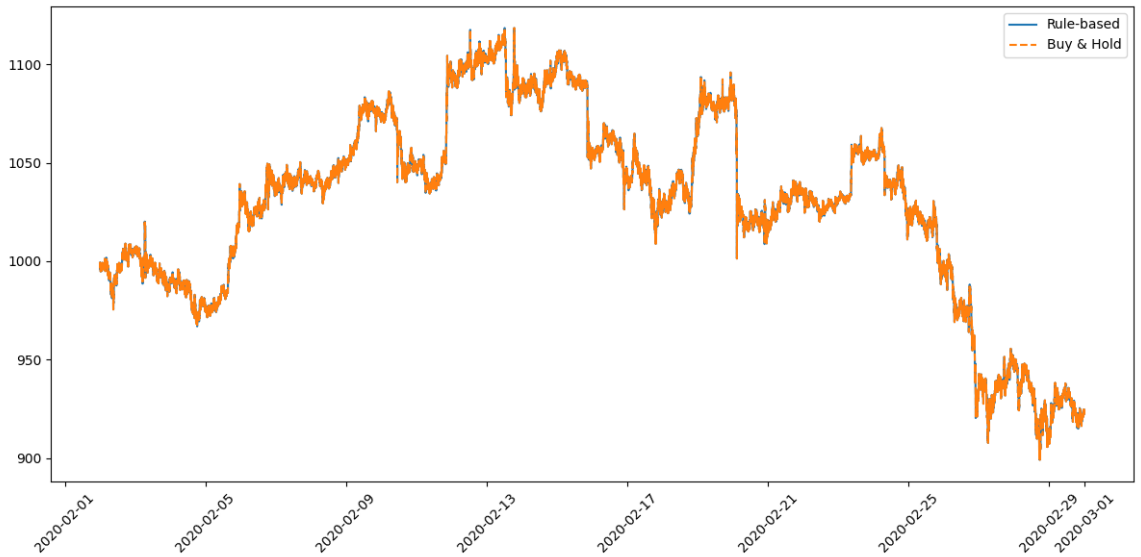
291) Full Year



292) January

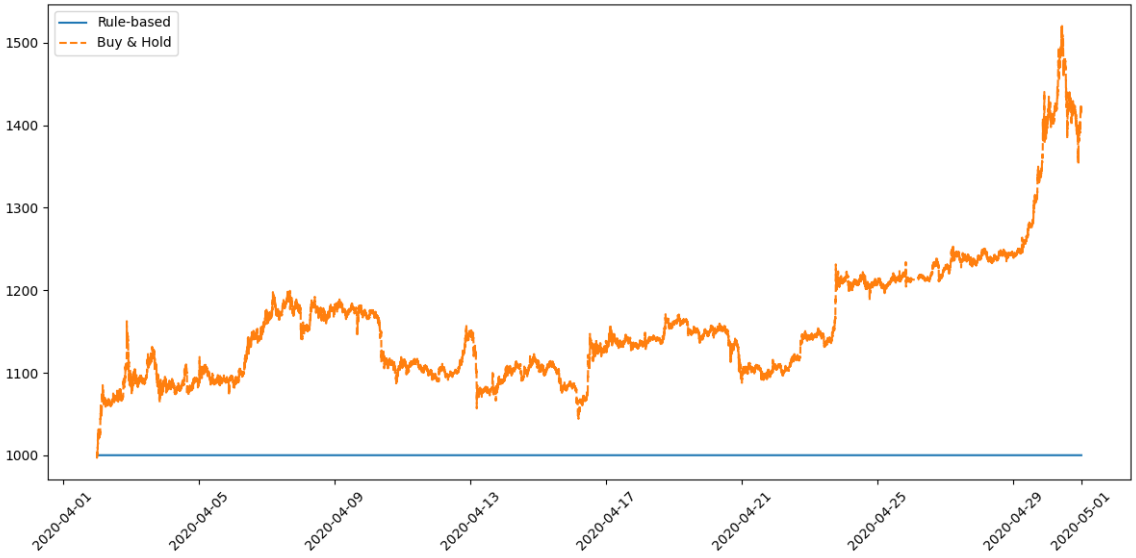


293) February

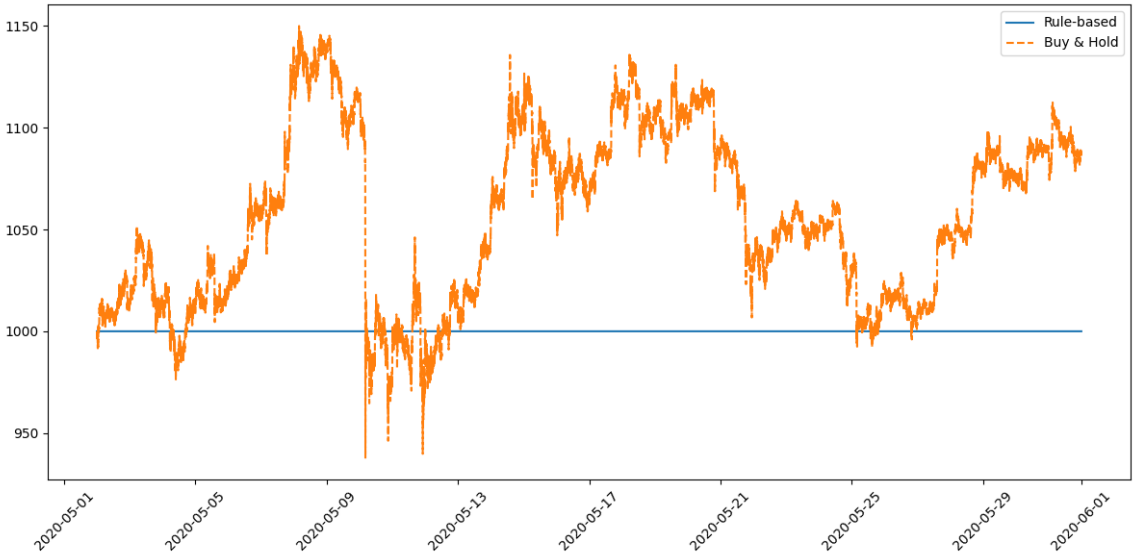


294) March

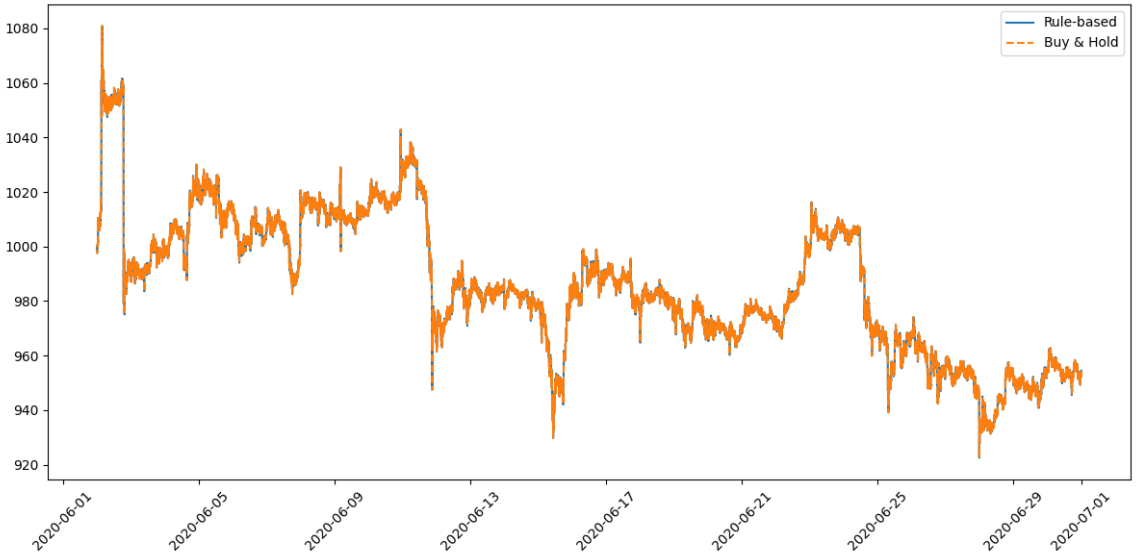
295) April



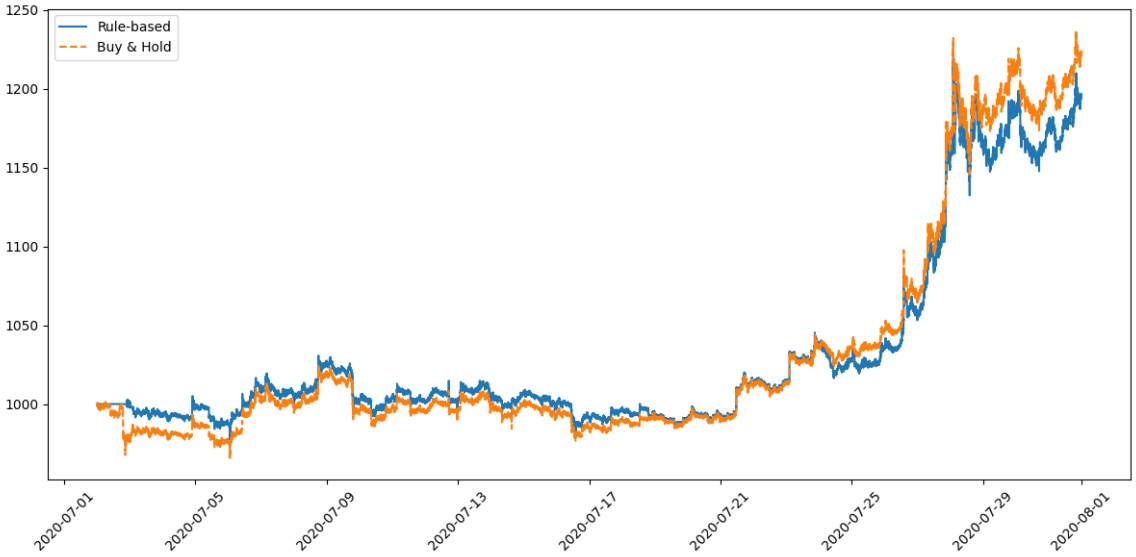
296) May



297) June

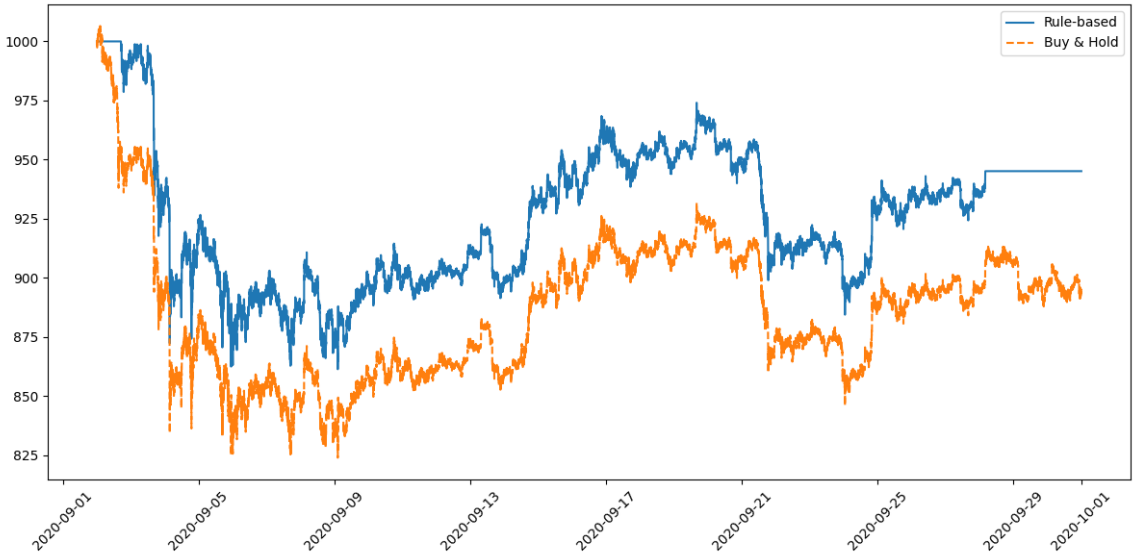


298) July



299) August

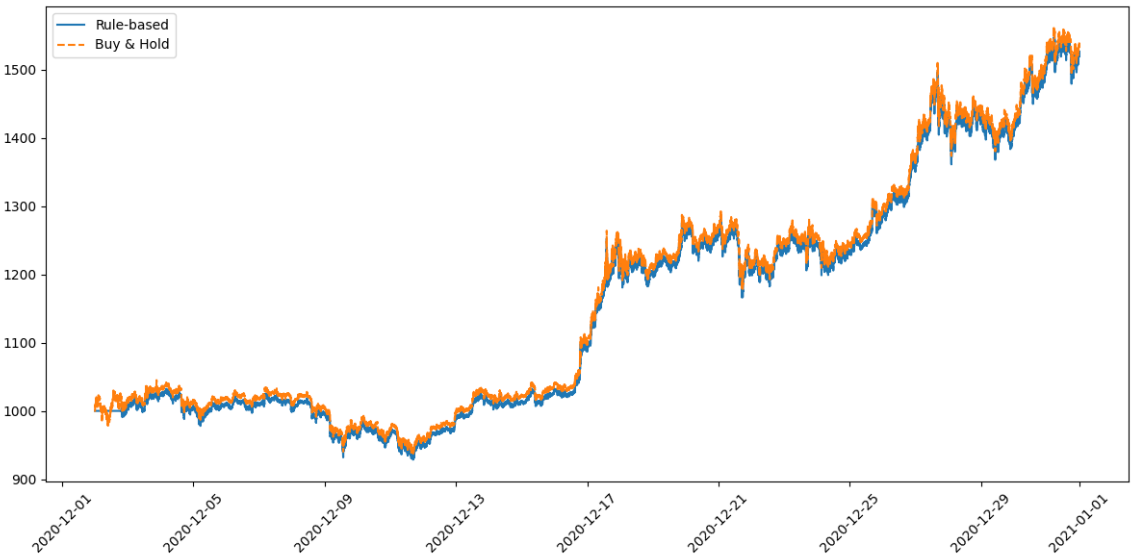
300) September



301) October

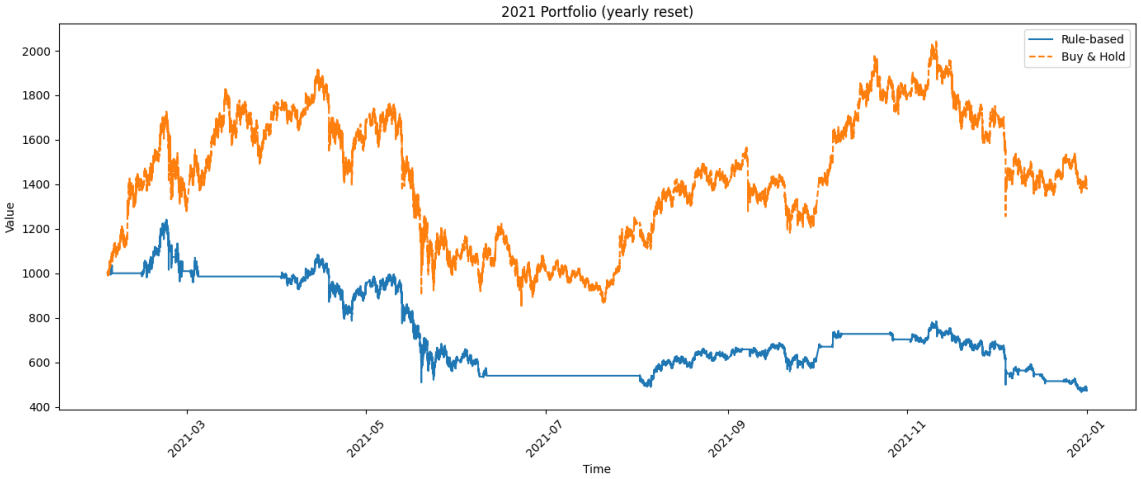
302) November

303) December



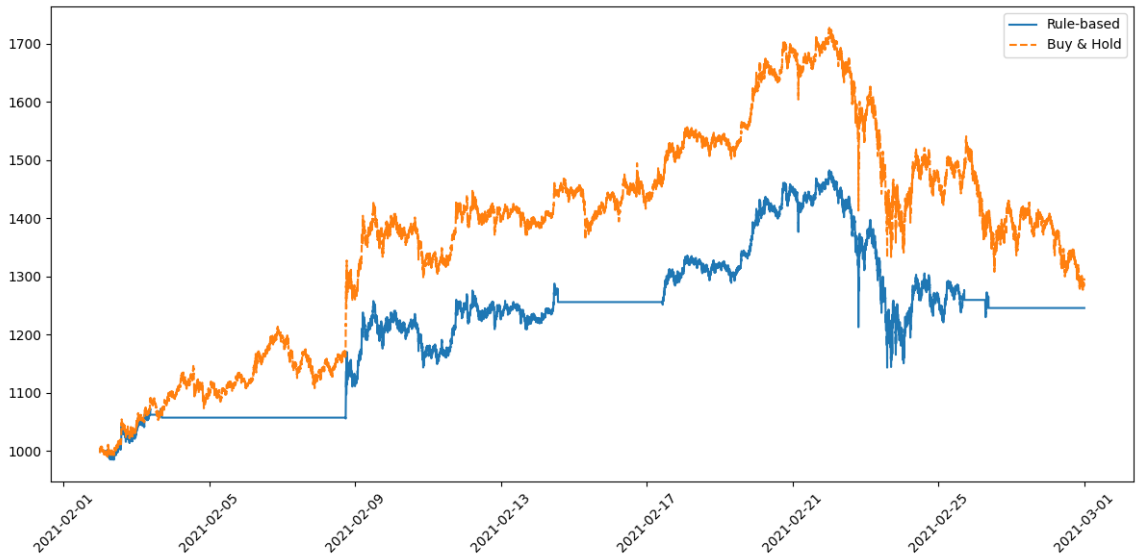
2021

304) FullYear

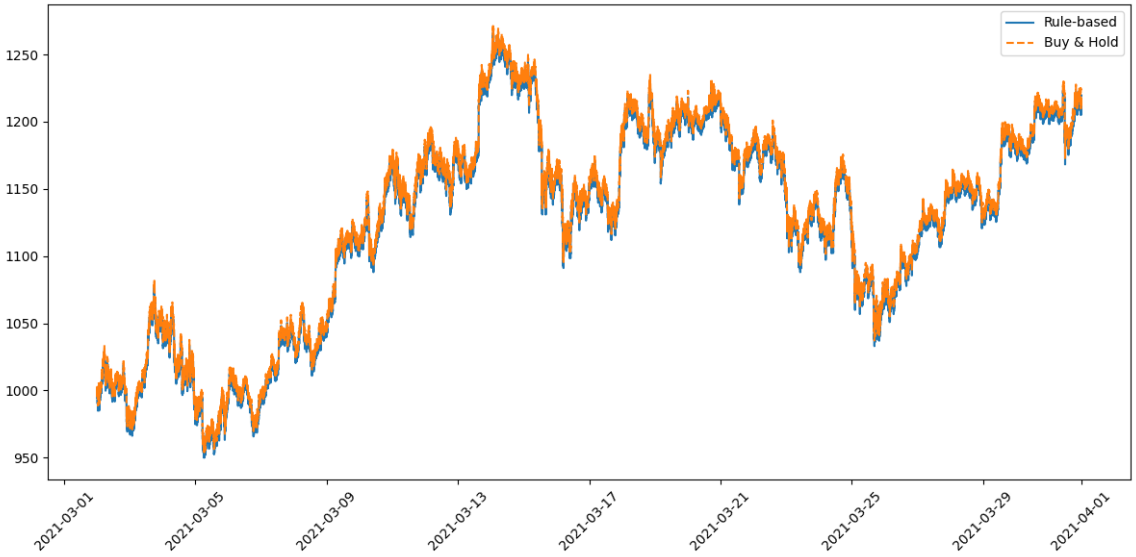


305) January

306) February

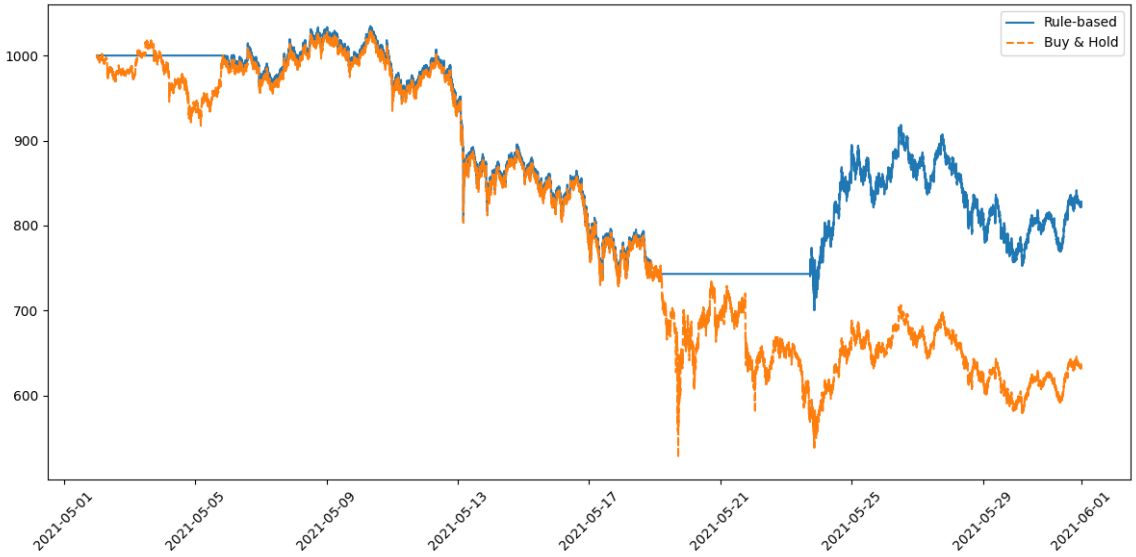


307) March

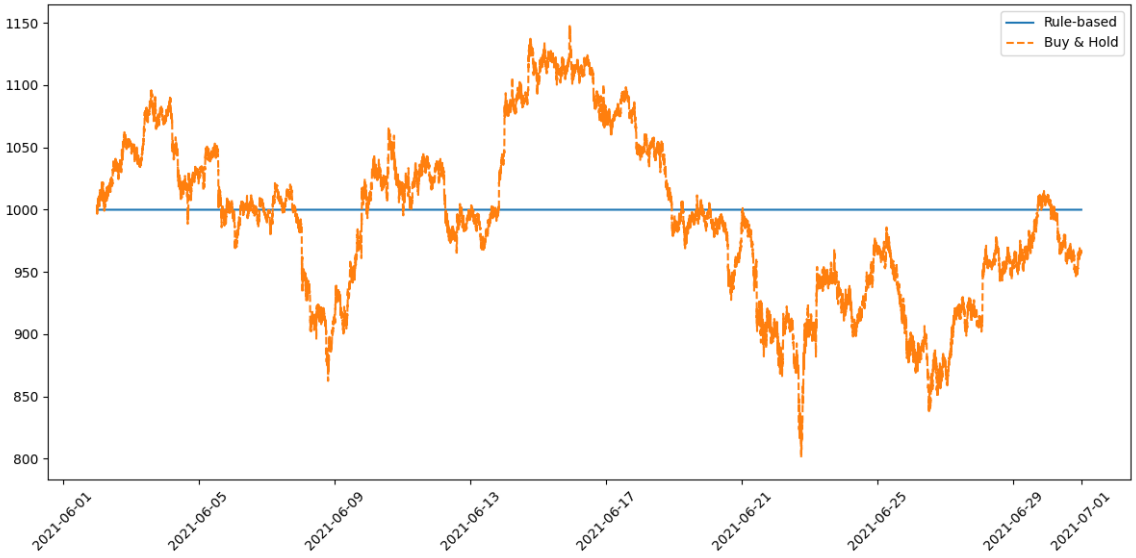


308) April

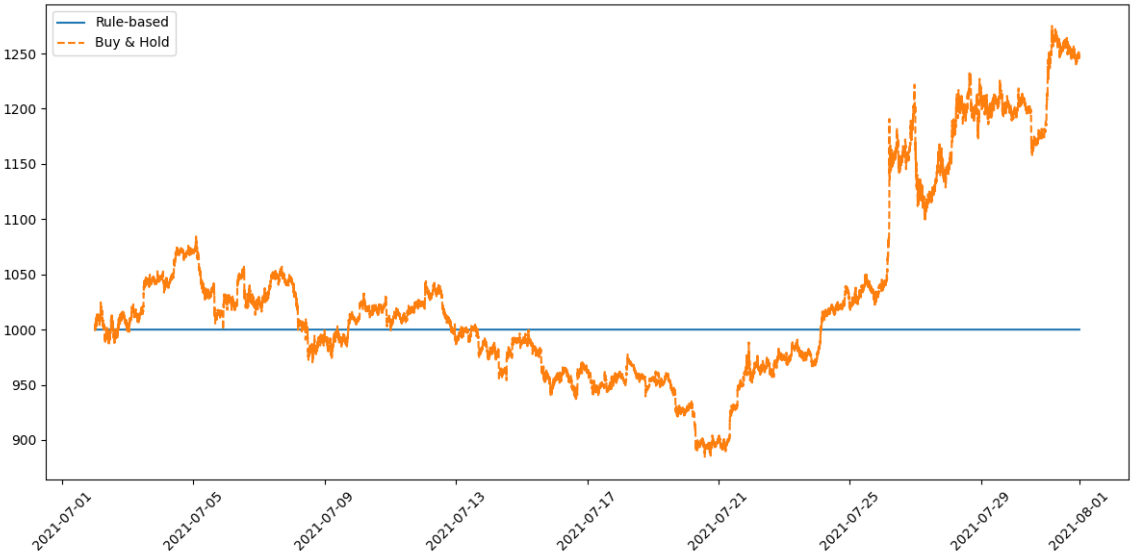
309) May



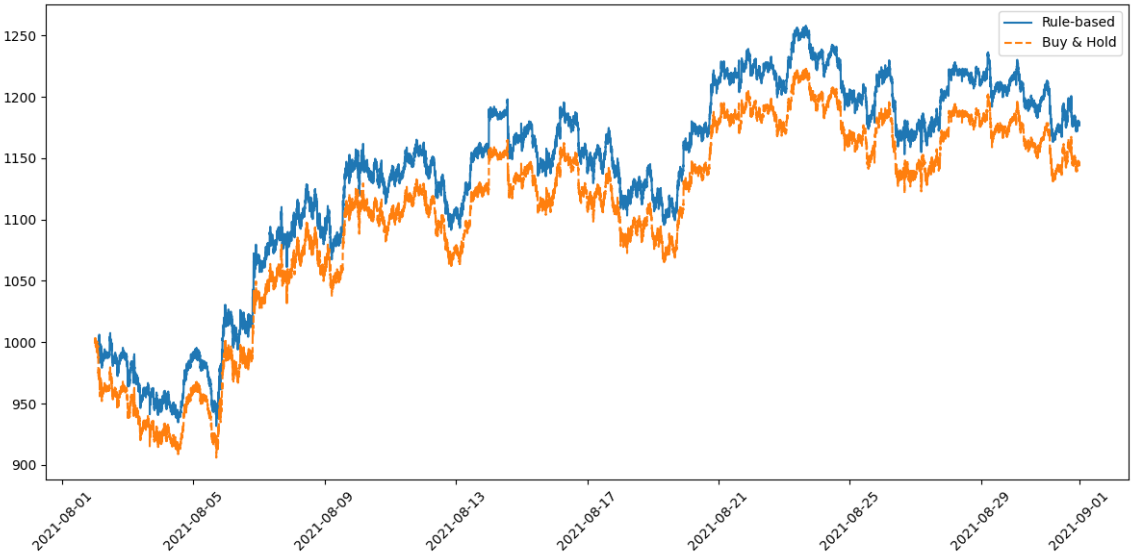
310) June



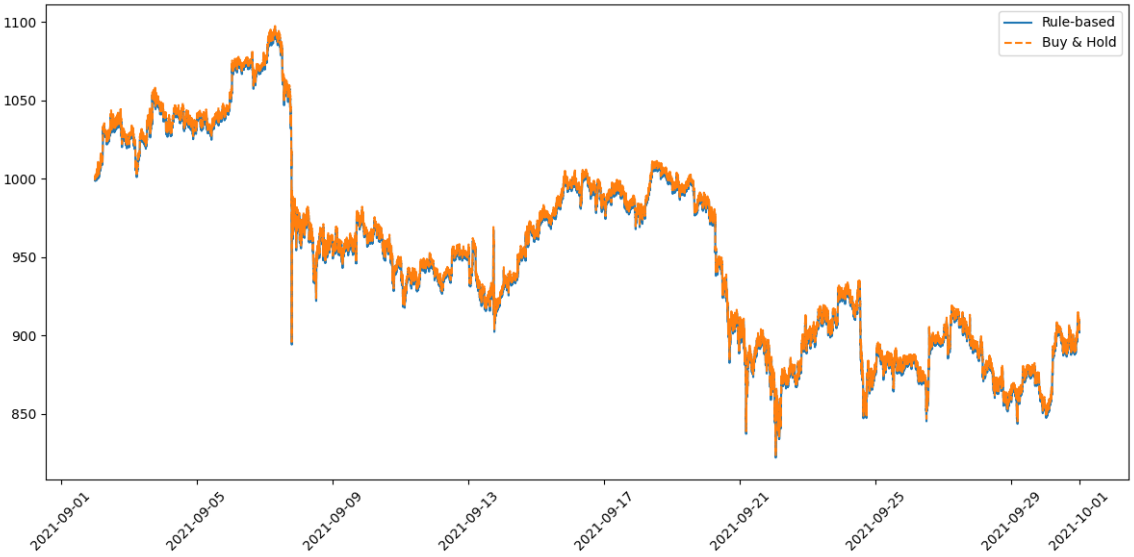
311) July



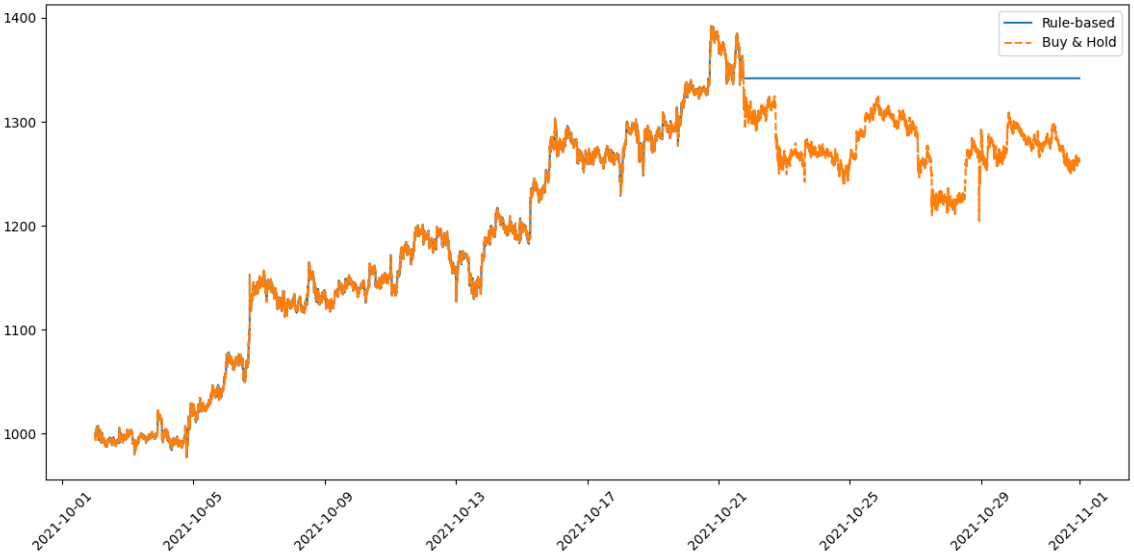
312) August



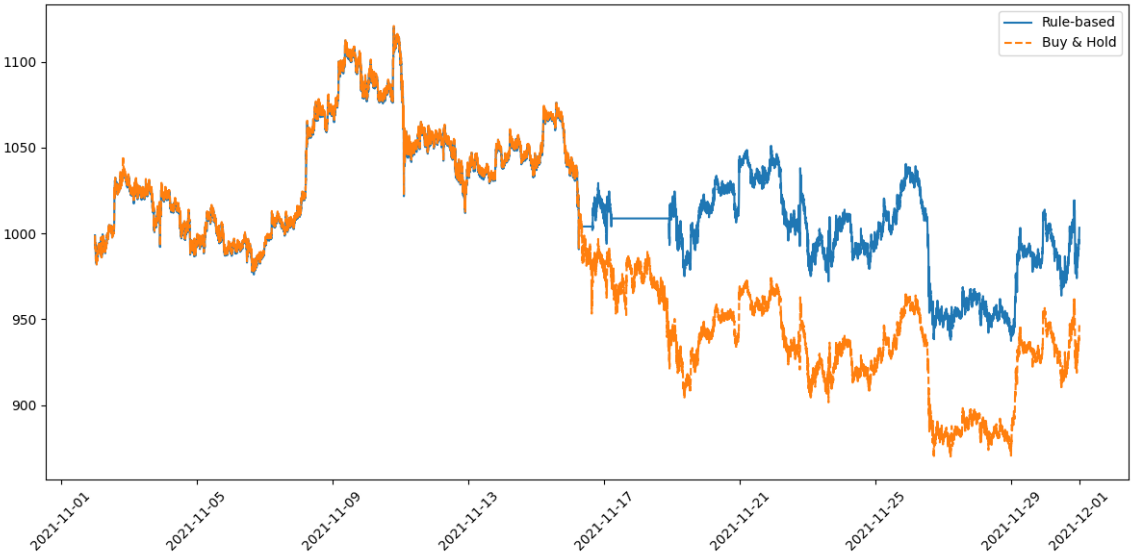
313) September



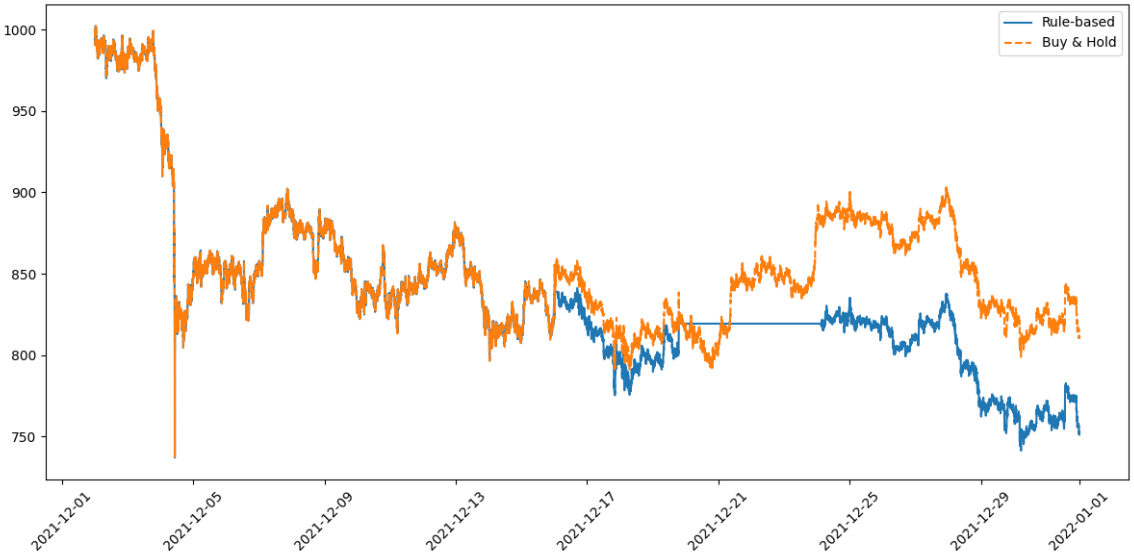
314) October



315) November

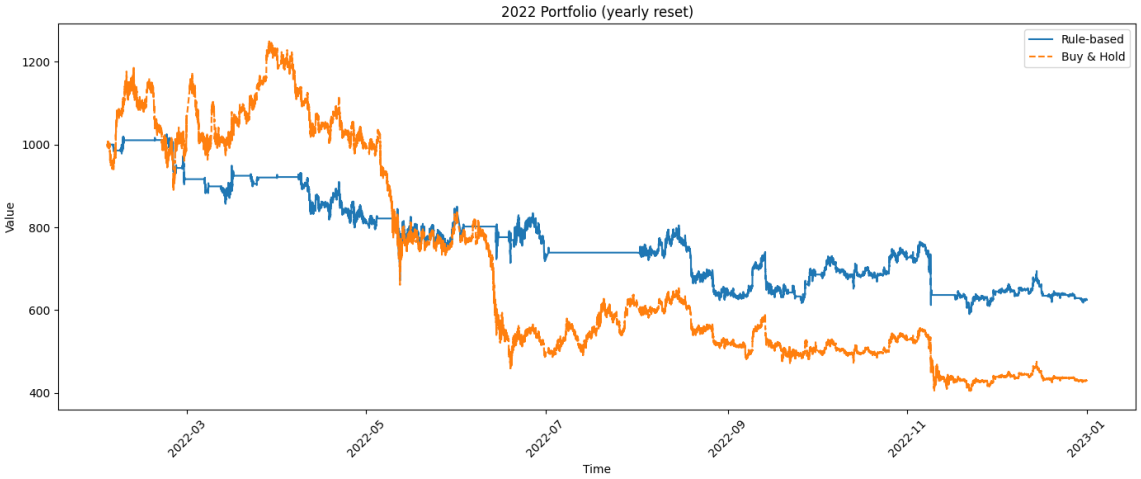


316) December

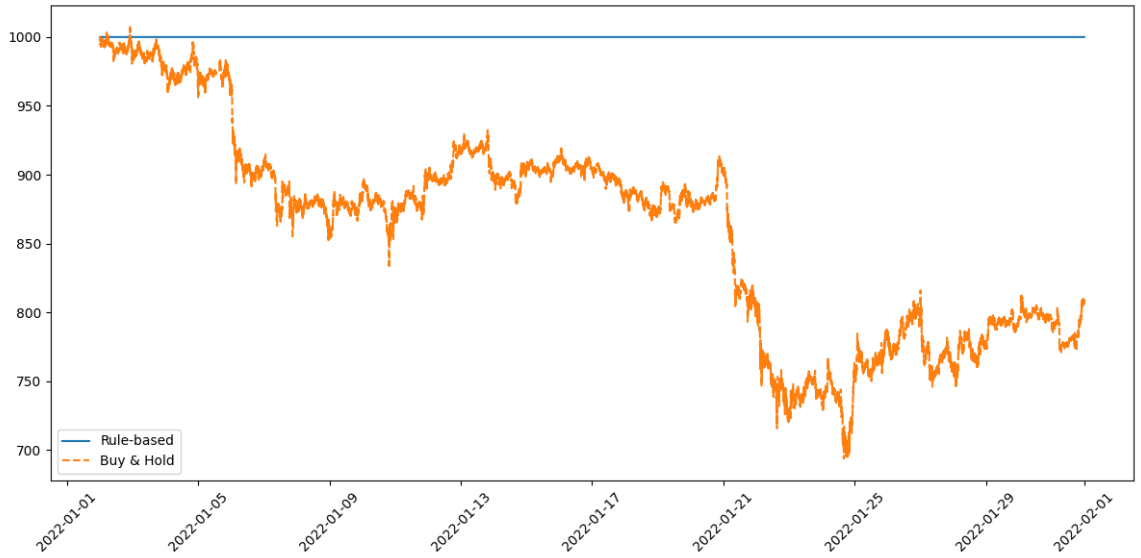


2022

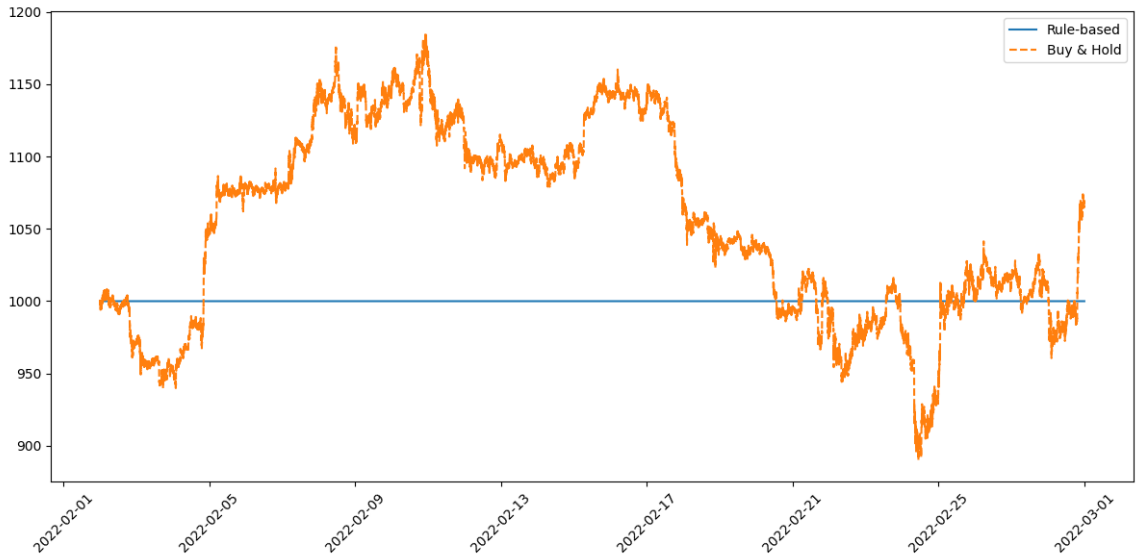
317) Full Year



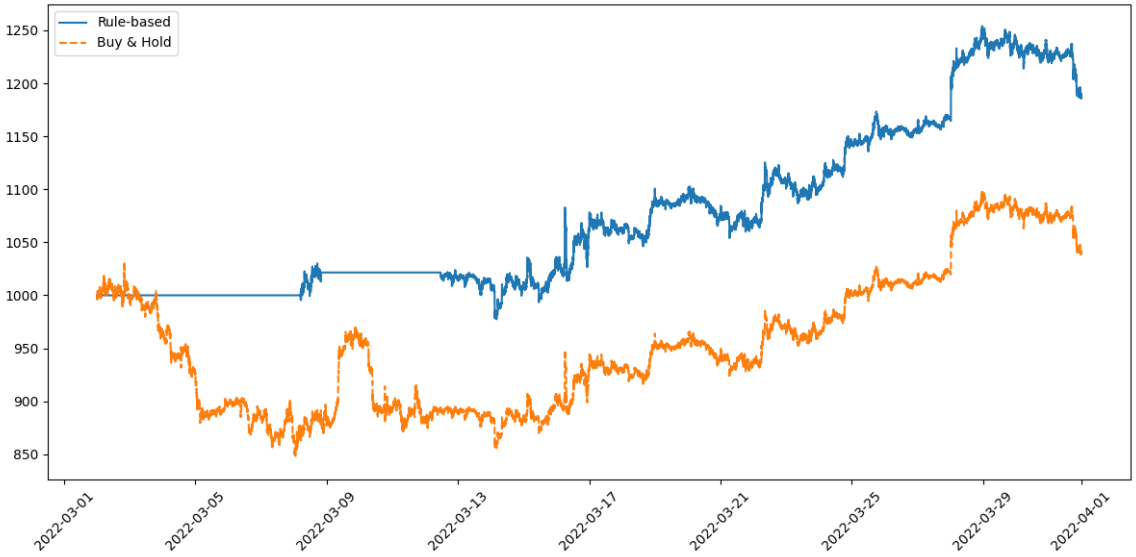
318) January



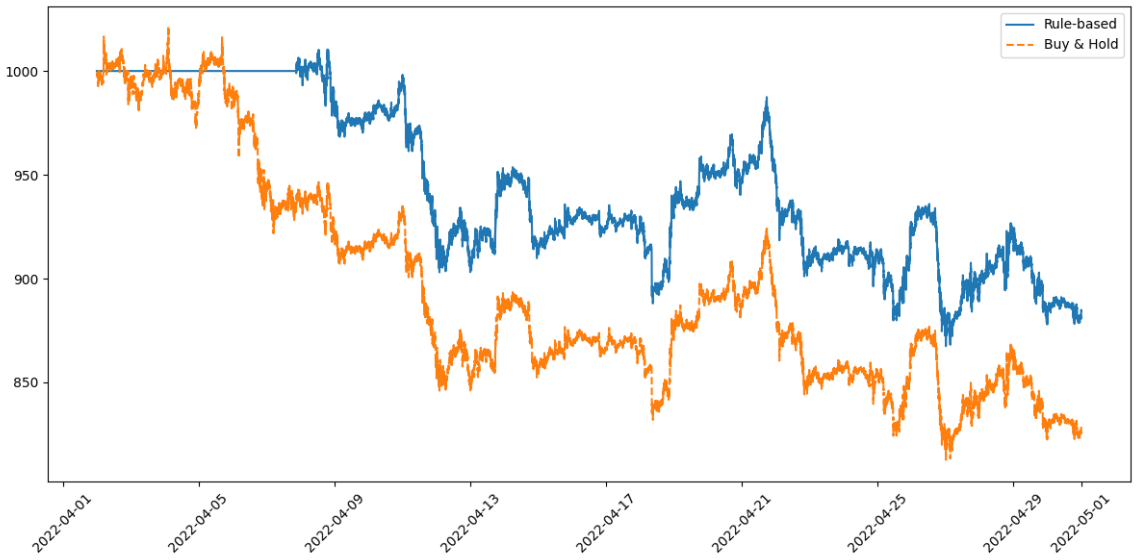
319) February



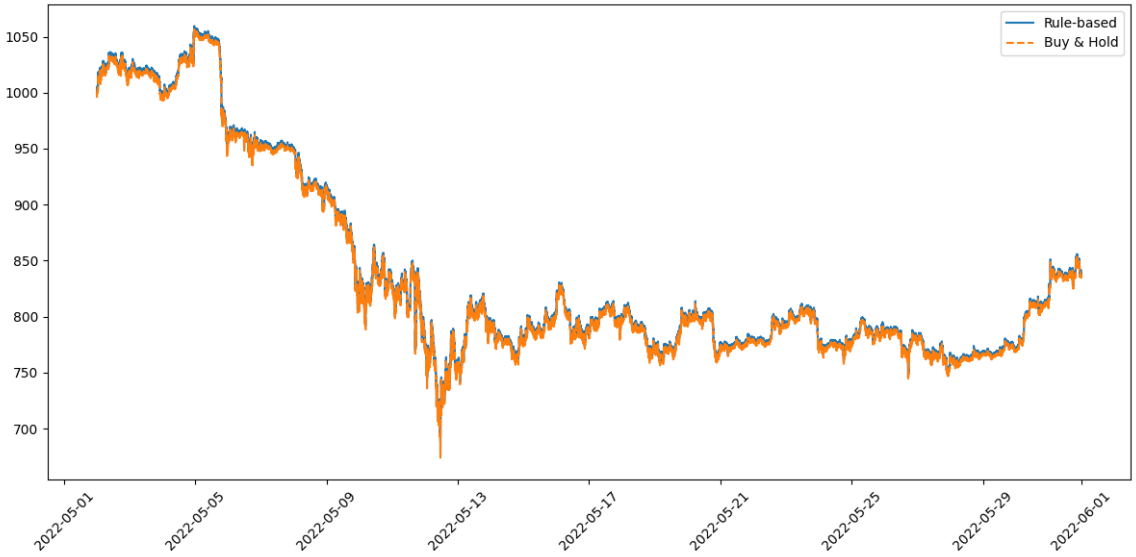
320) March



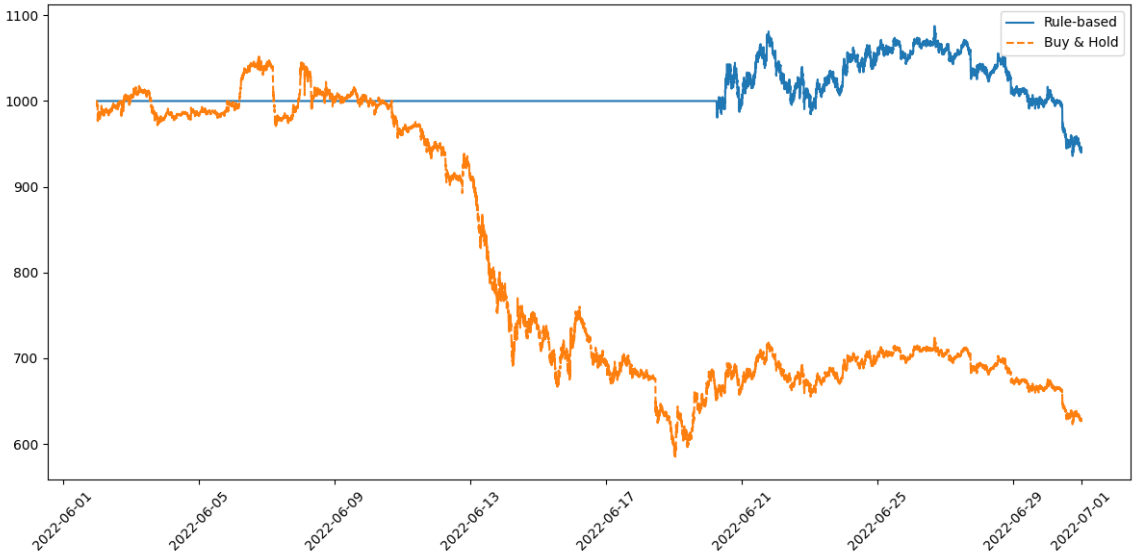
321) April



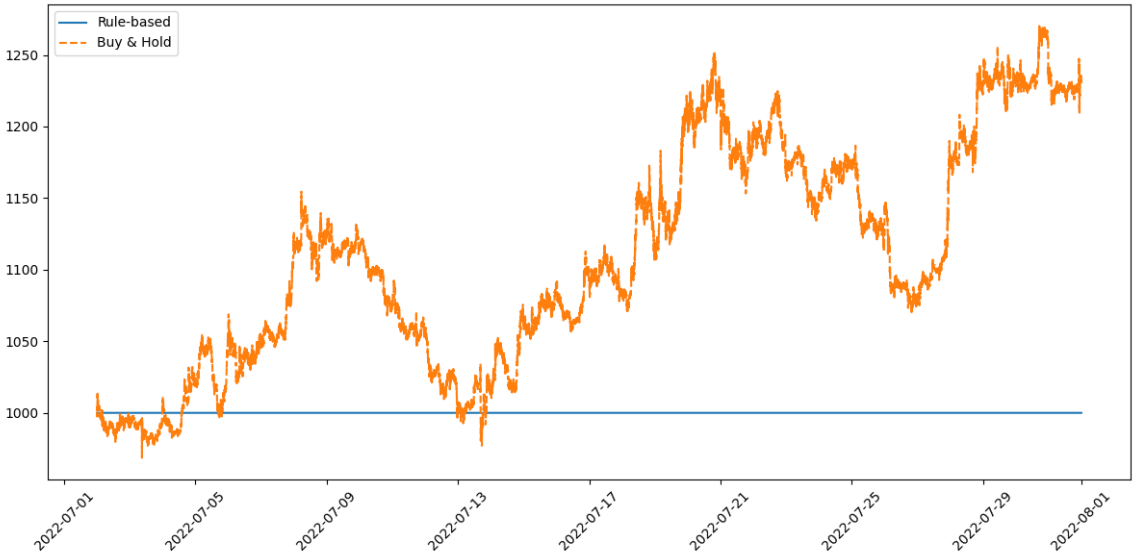
322) May



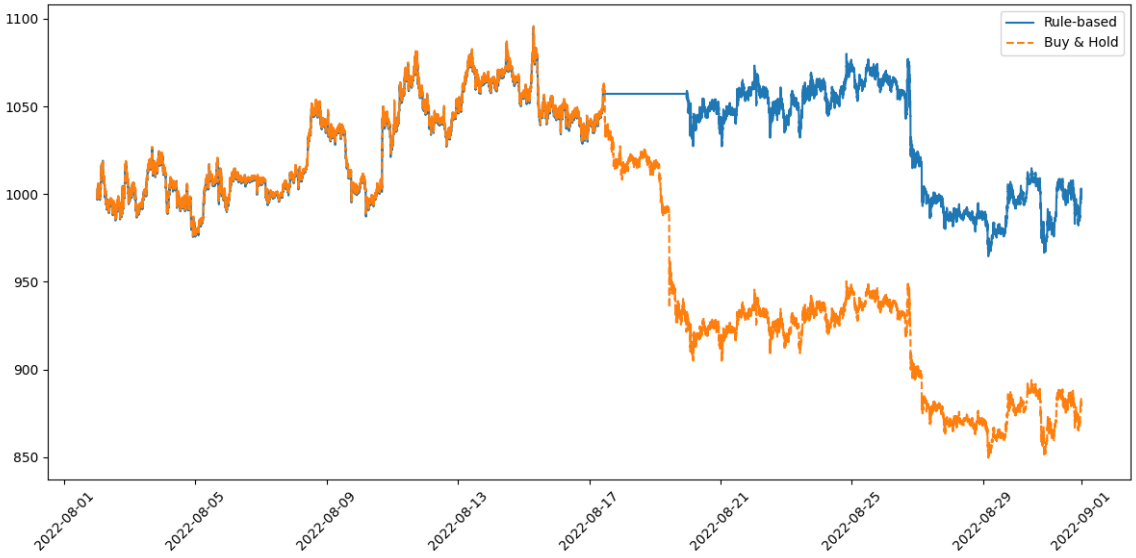
323) June



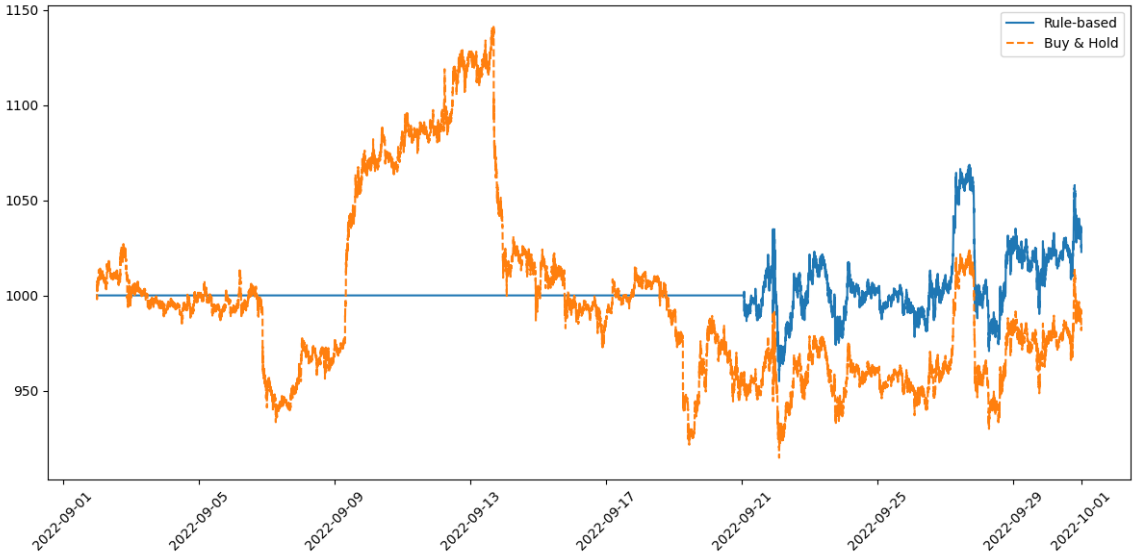
324) July



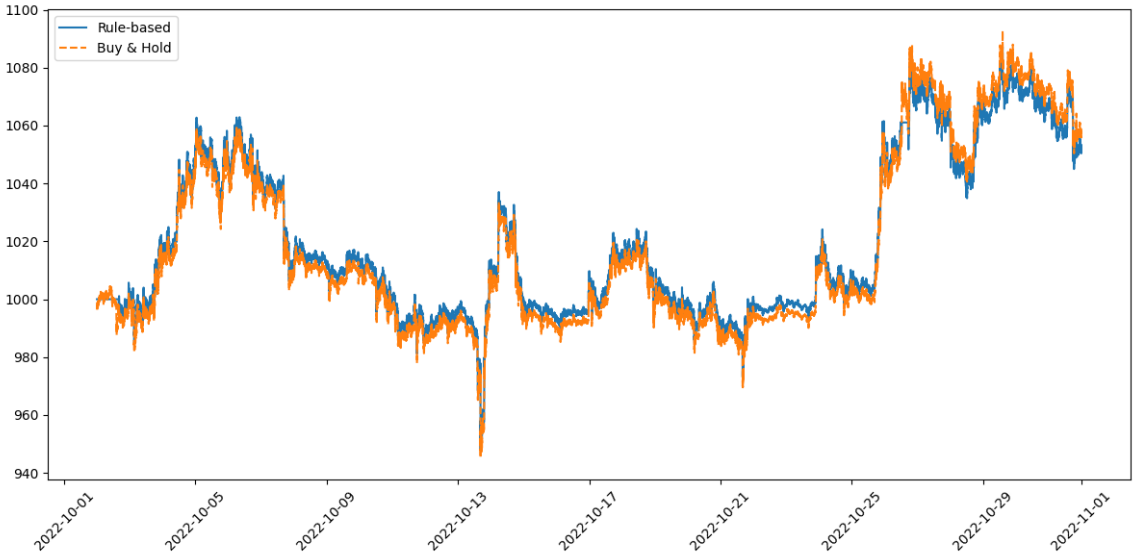
325) August



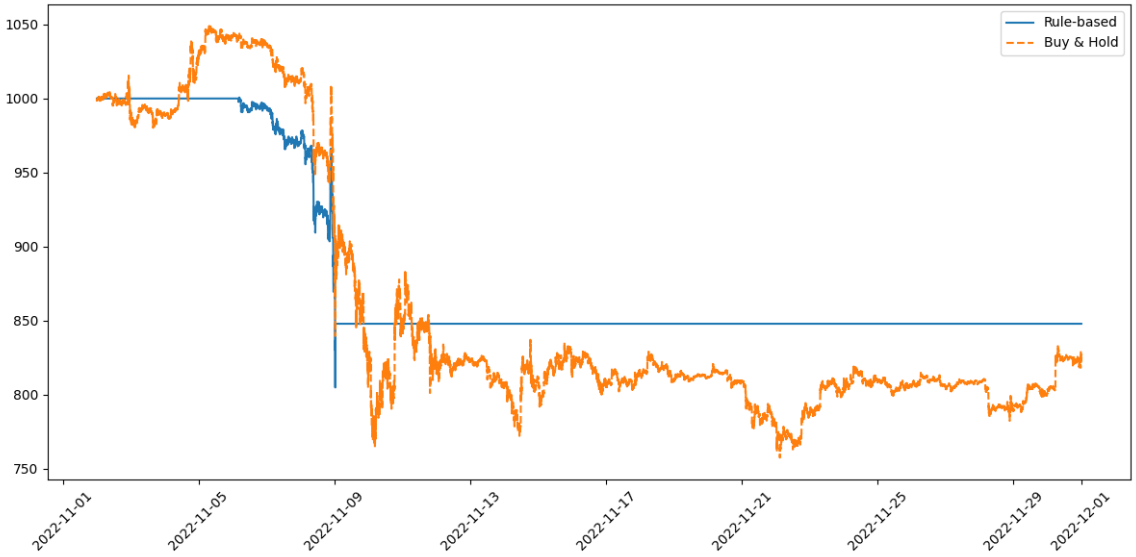
326) September



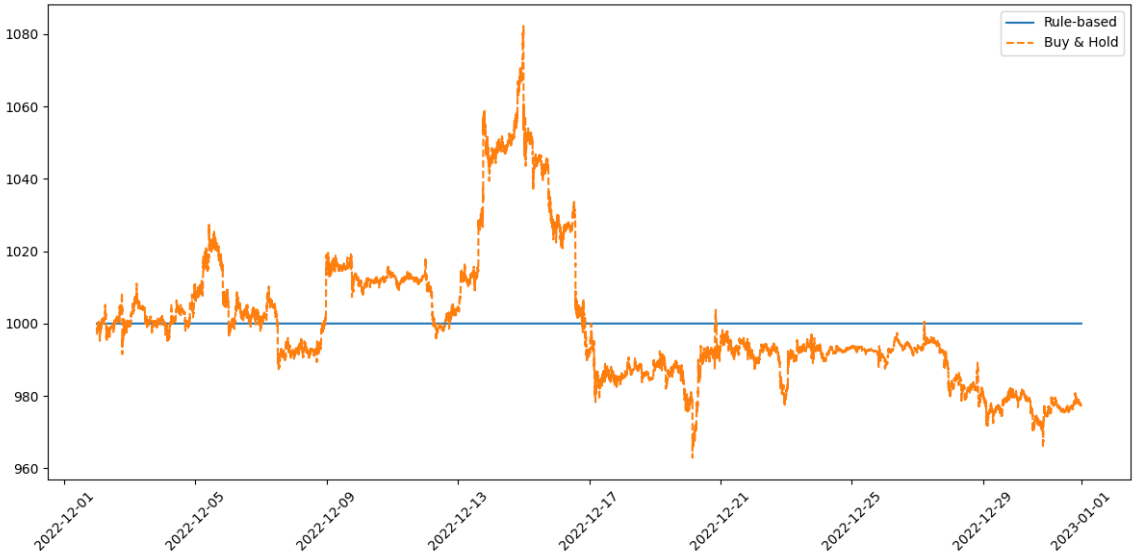
327) October



328) November

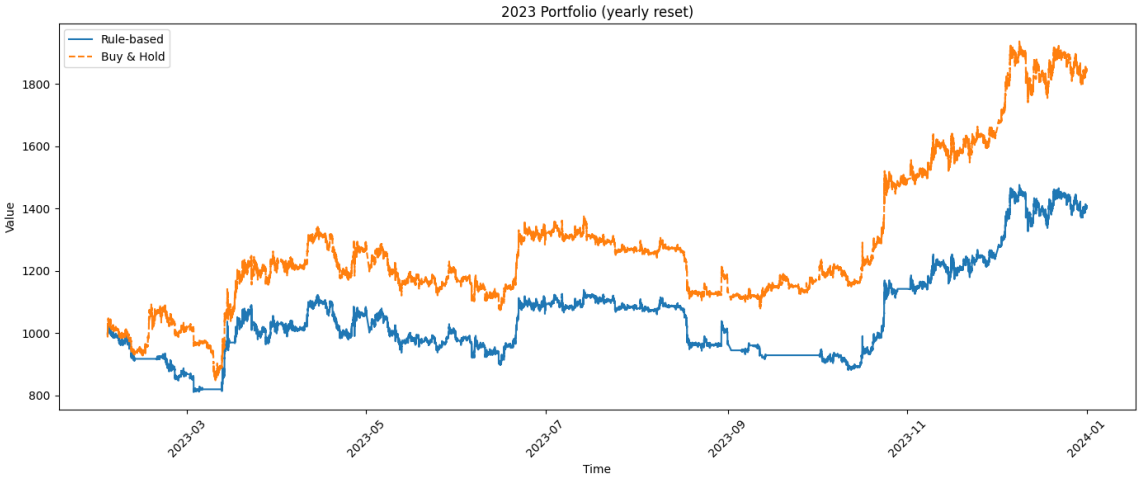


329) December

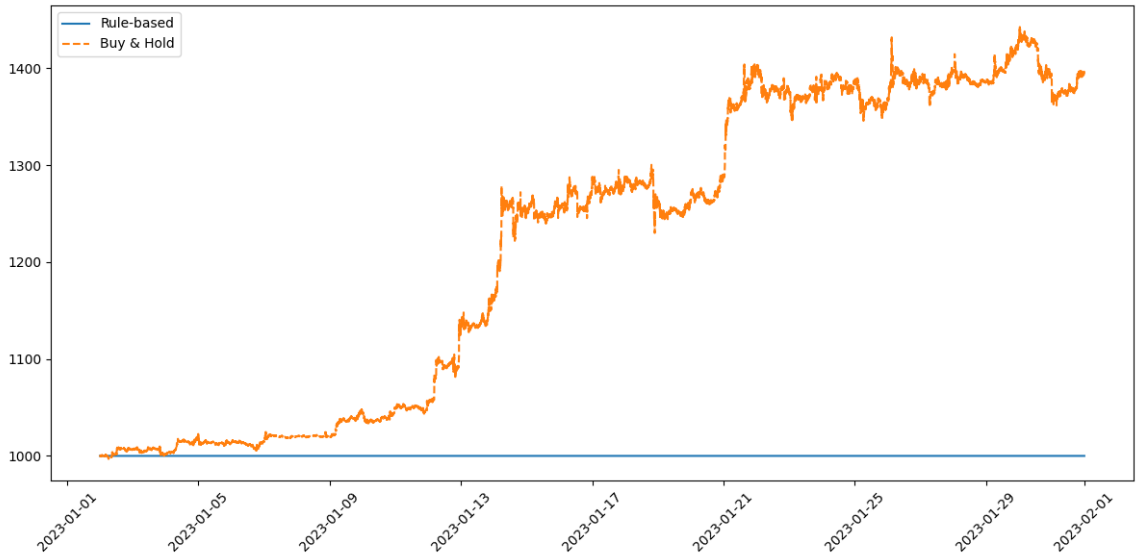


2023

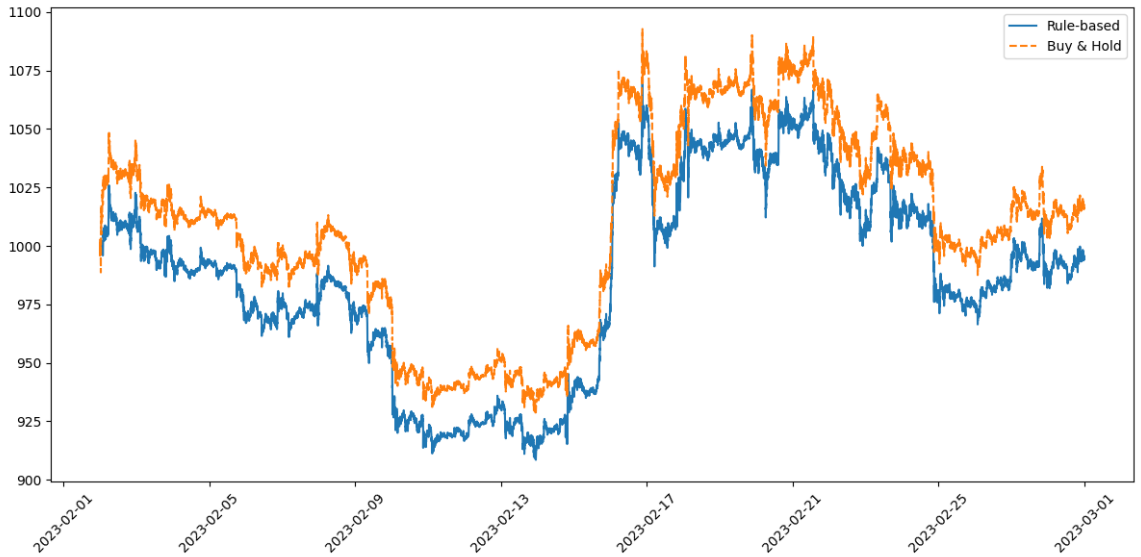
330) Full Year



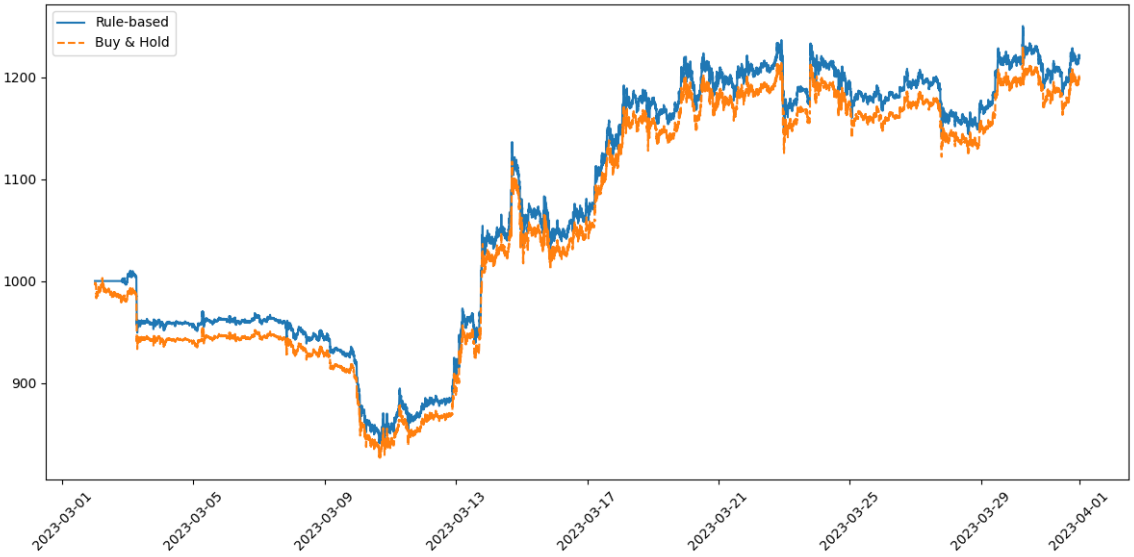
331) January



332) February

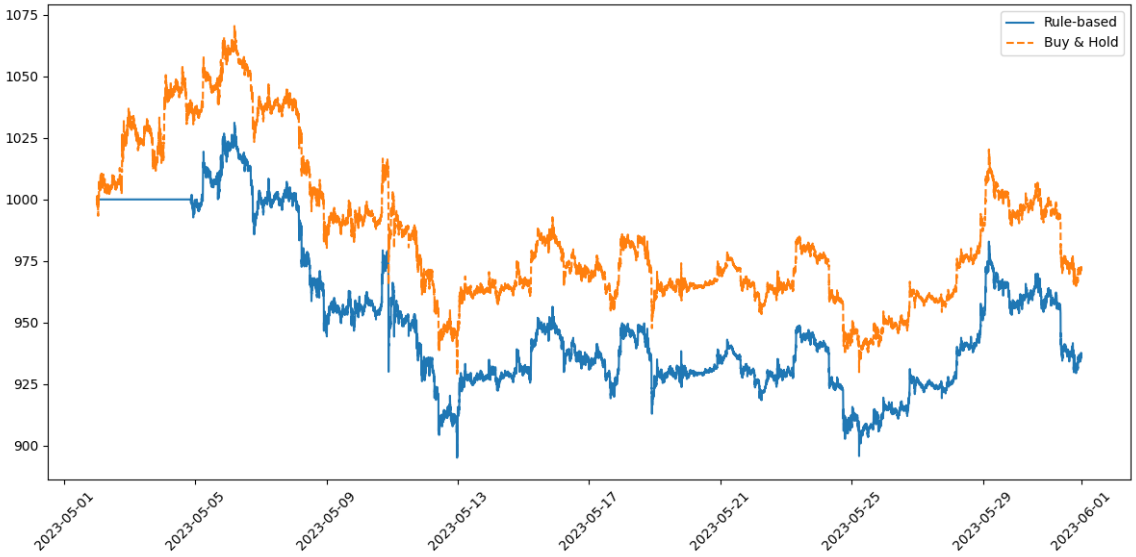


333) March

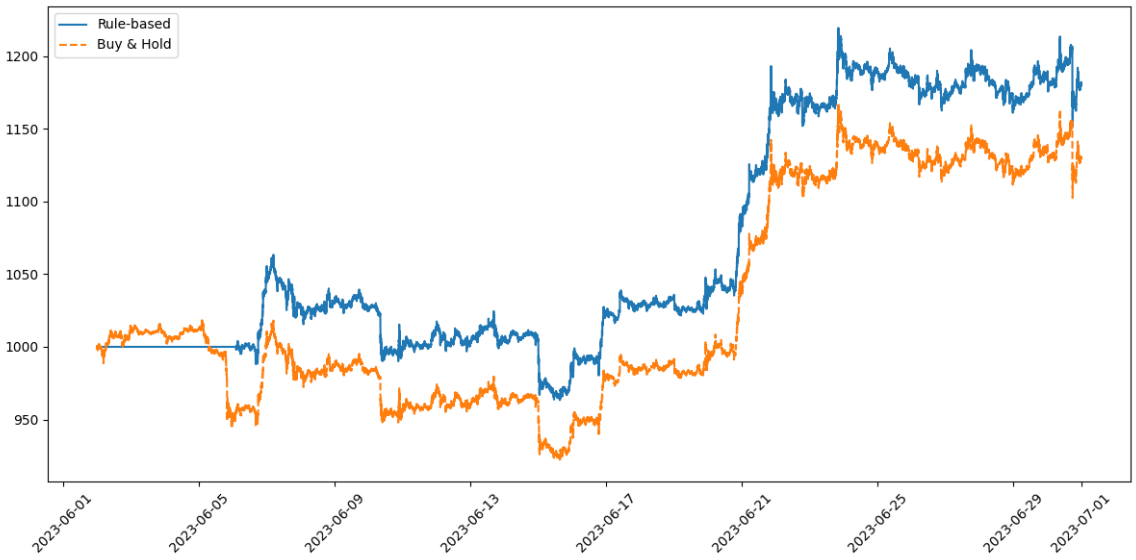


334) April

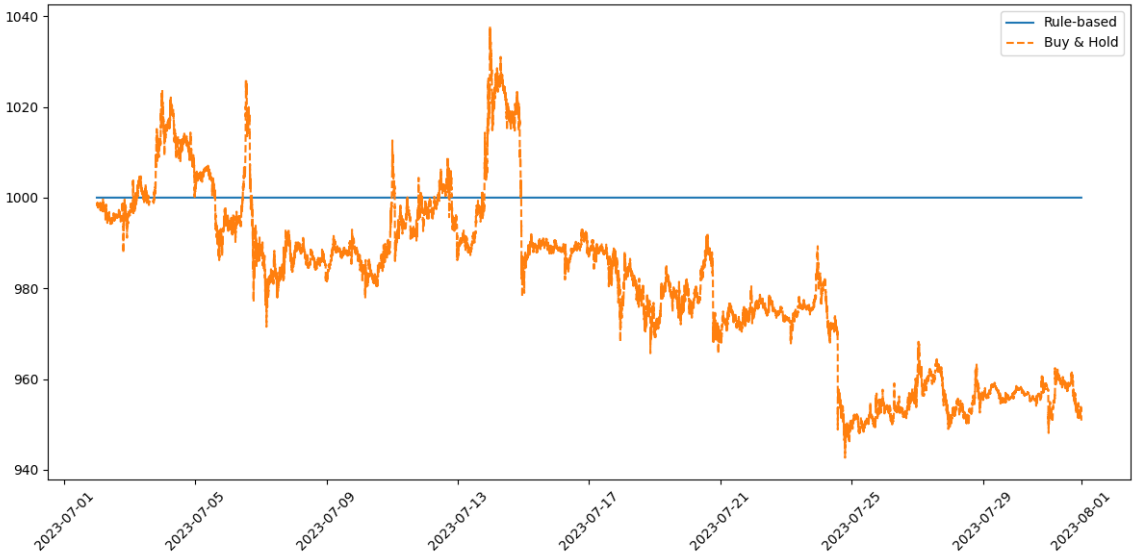
335) May



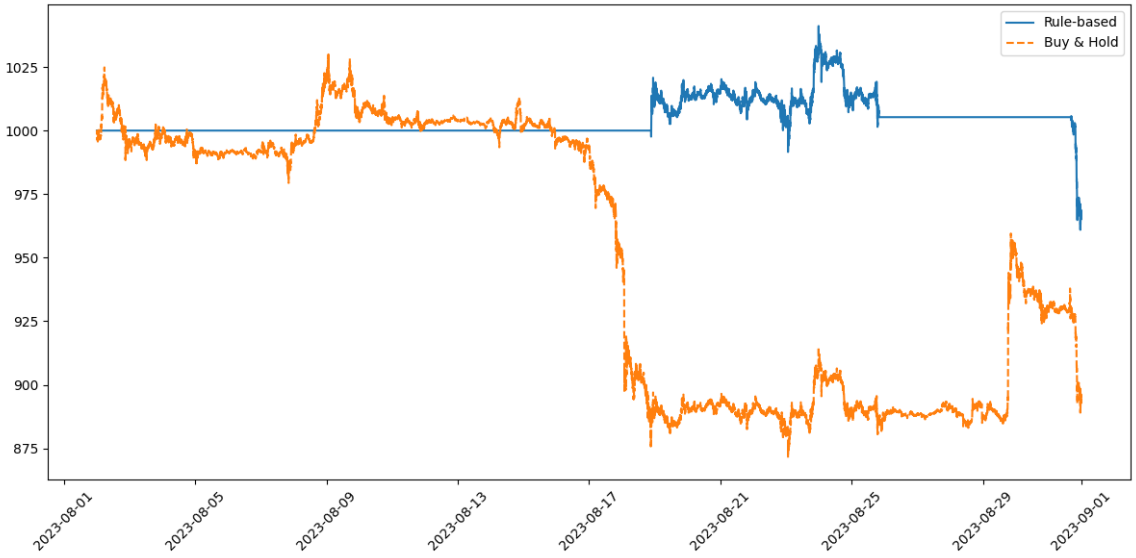
336) June



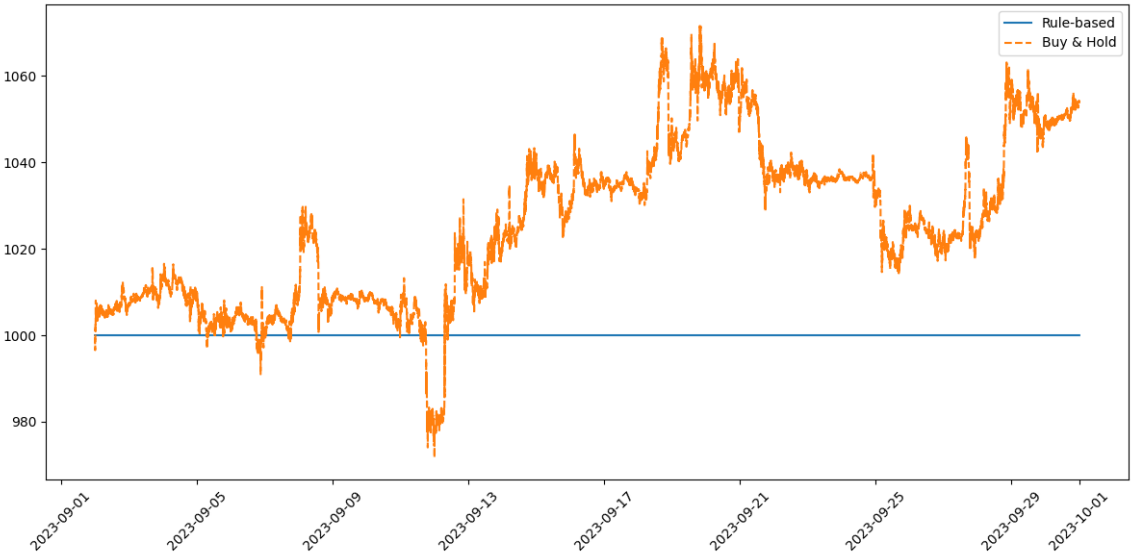
337) July



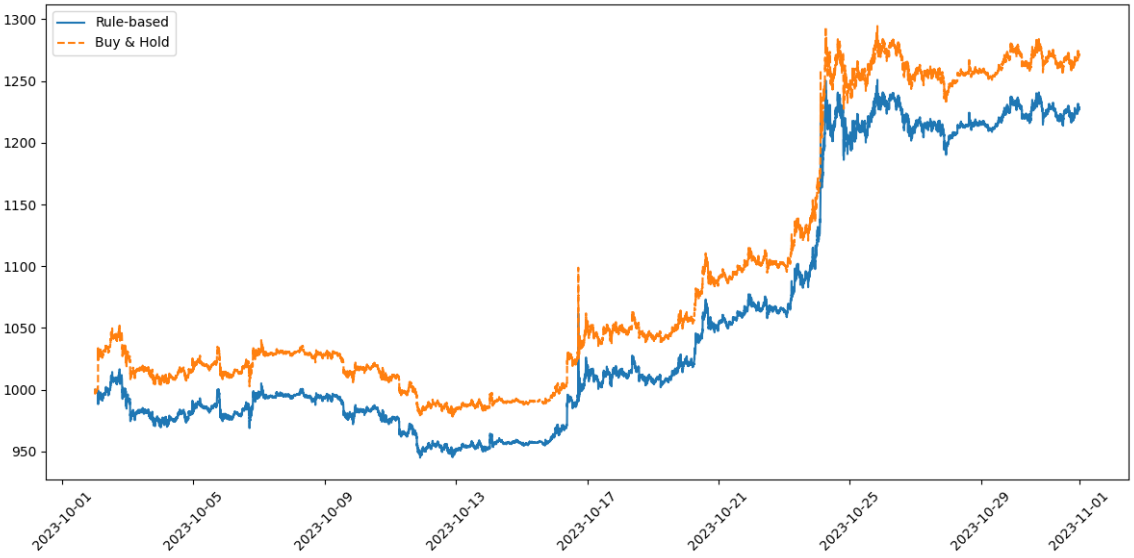
338) August



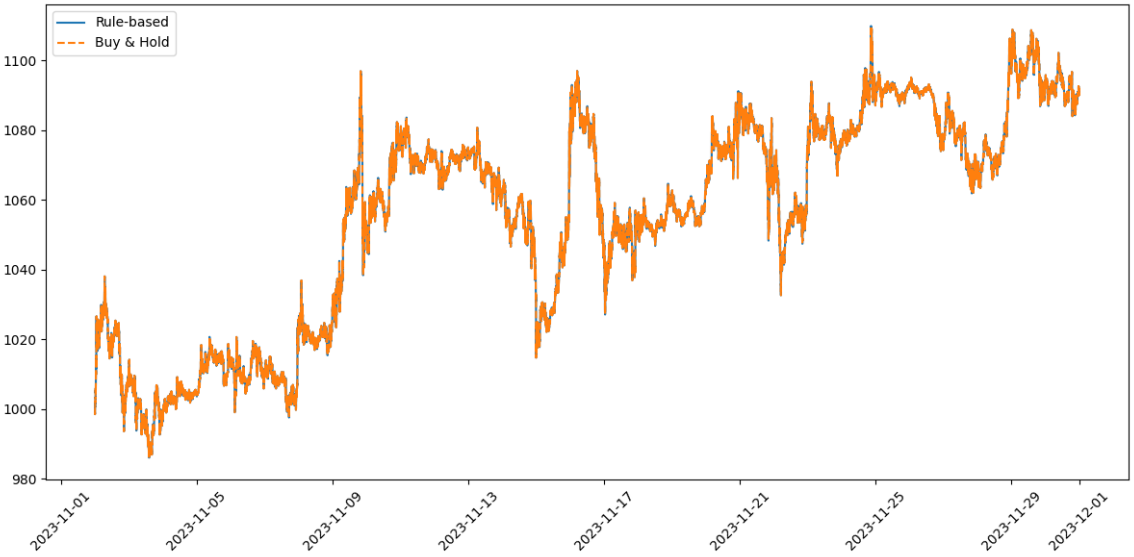
339) September



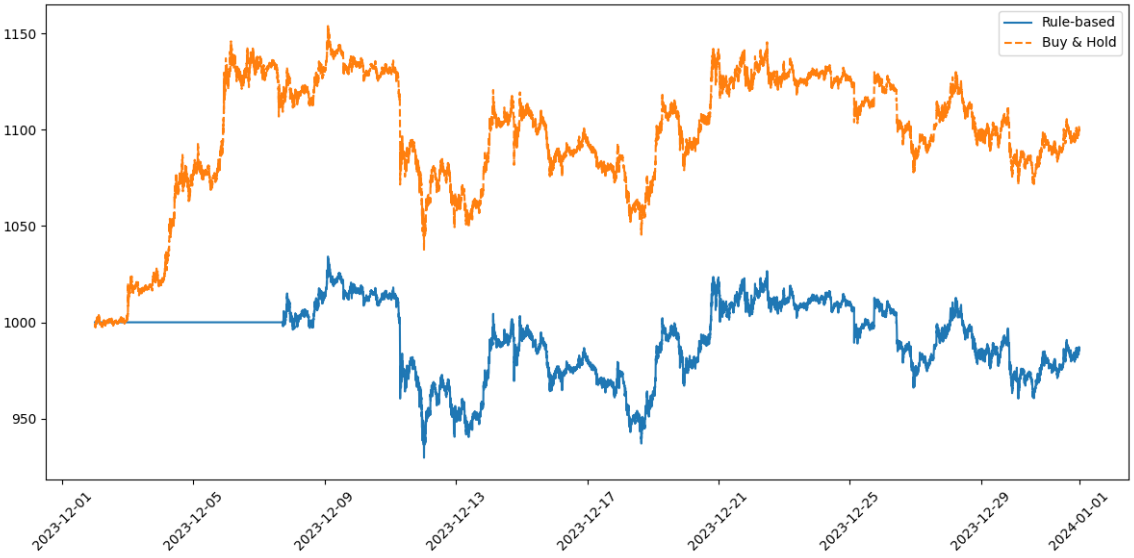
340) October



341) November

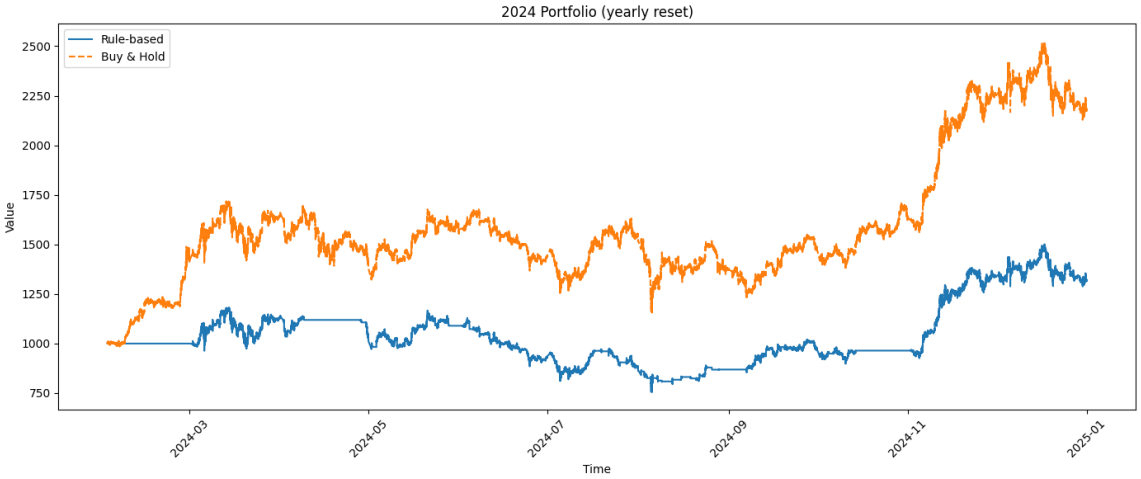


342) December



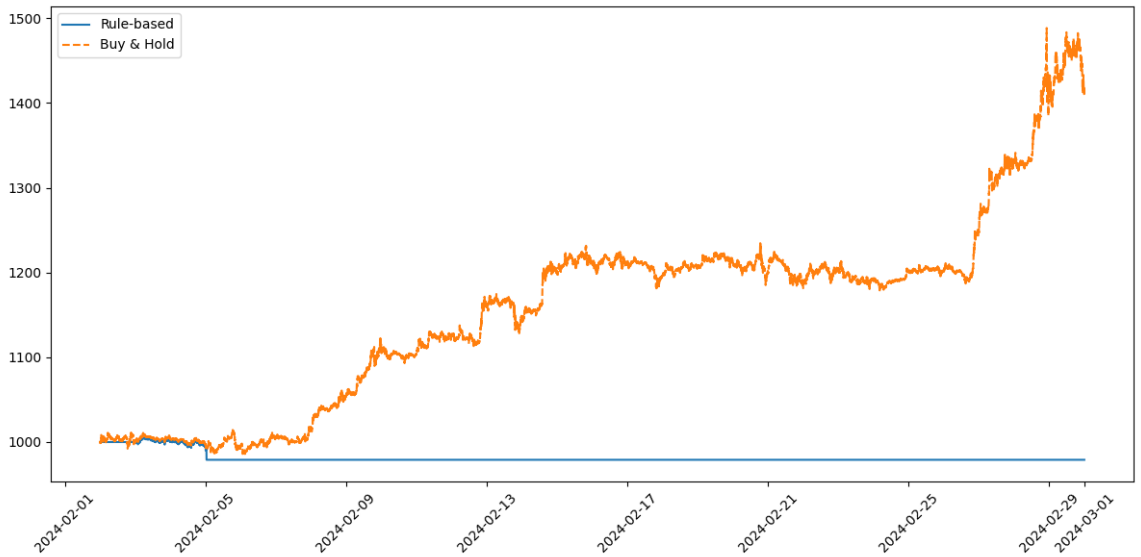
2024

343) Full Year



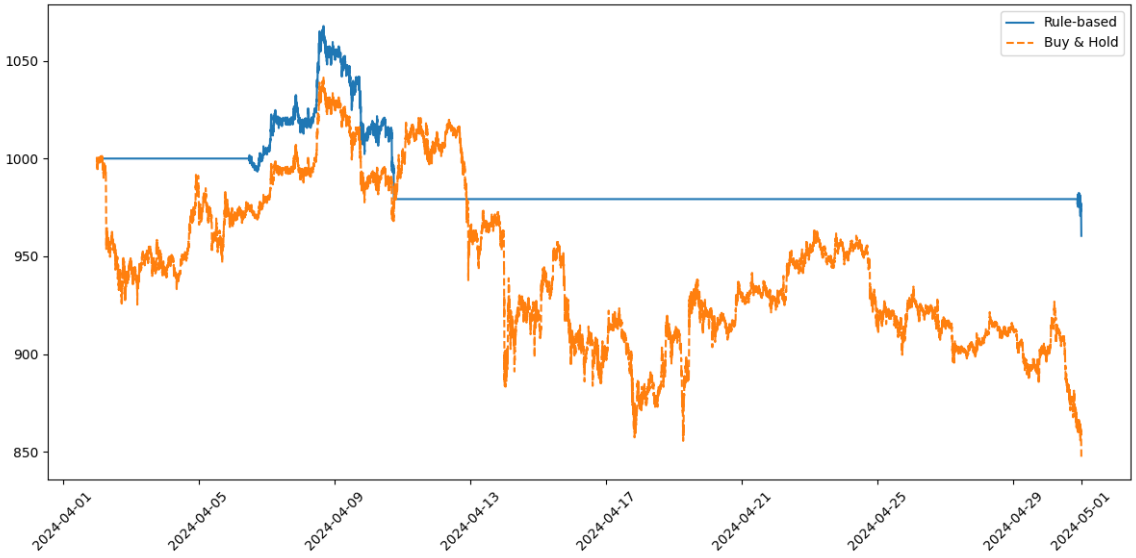
344) January

345) February



346) March

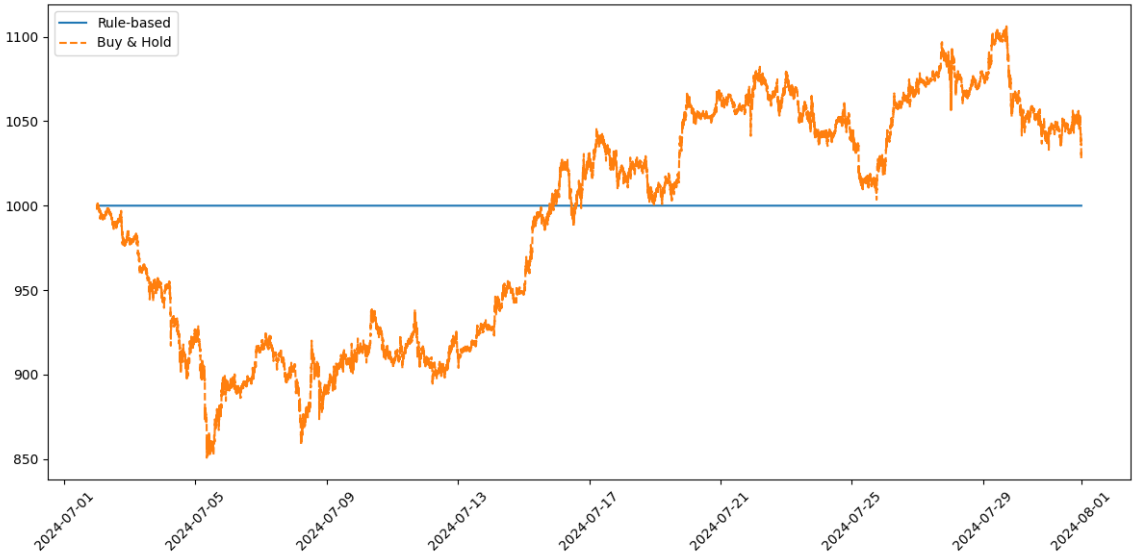
347) April



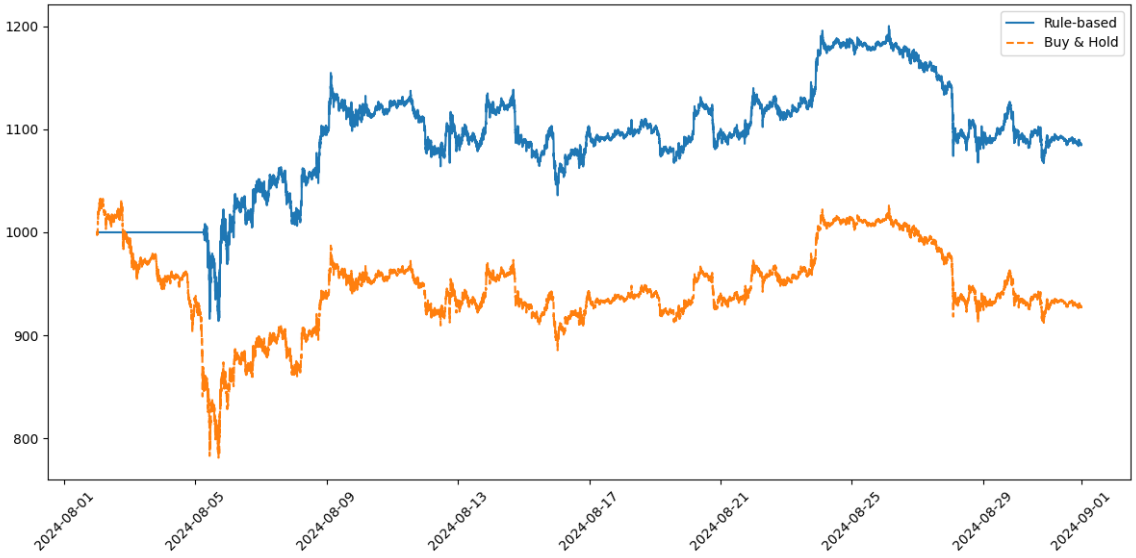
348) May

349) June

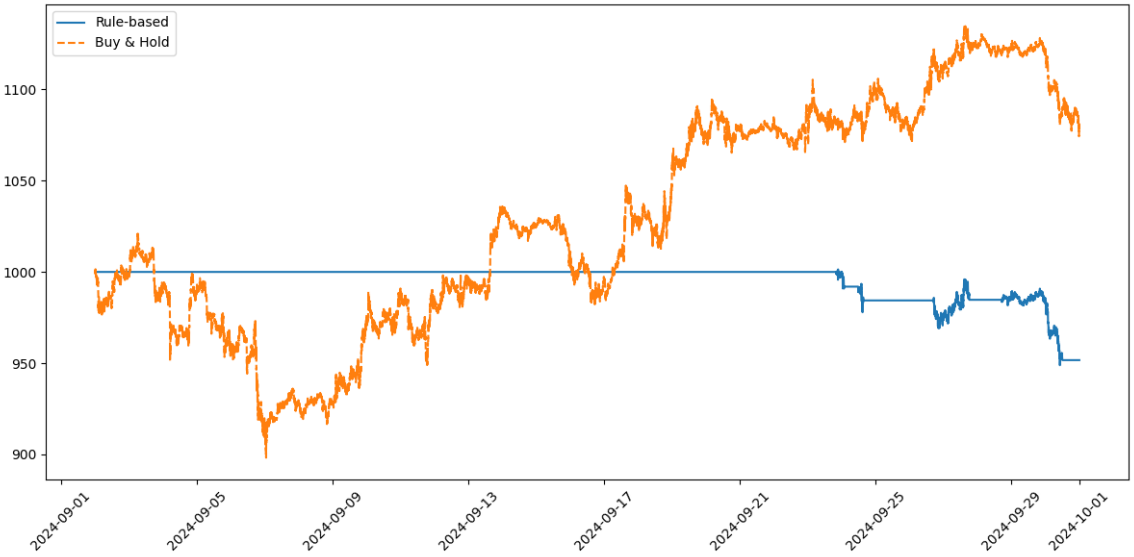
350) July



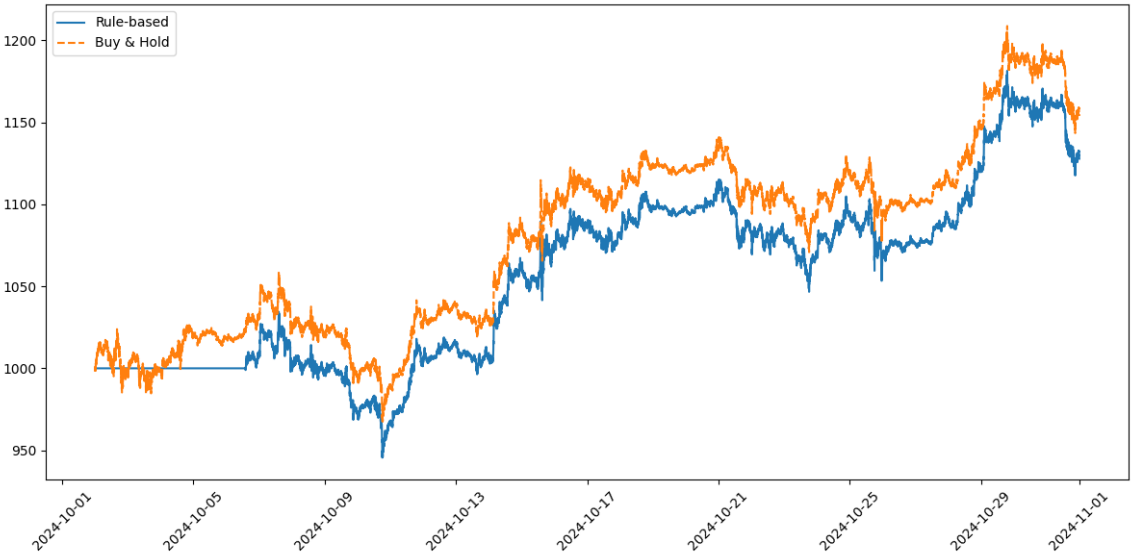
351) August



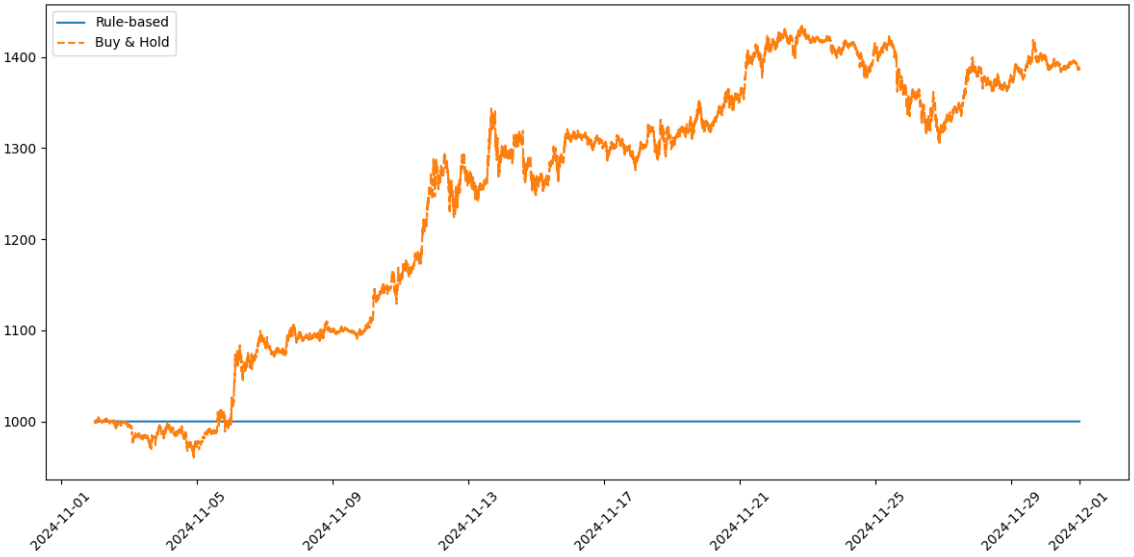
352) September



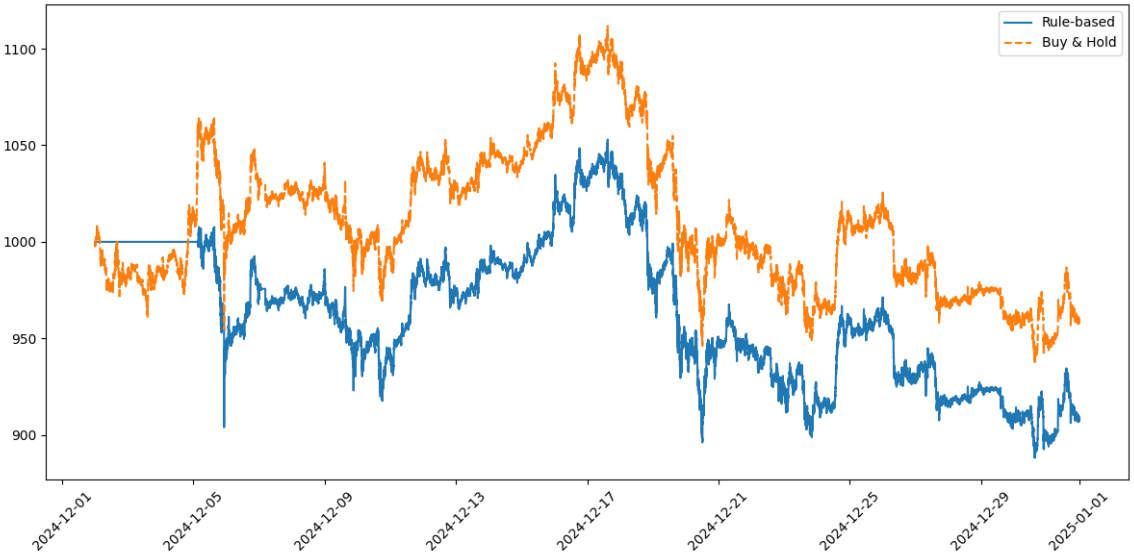
353) October



354) November

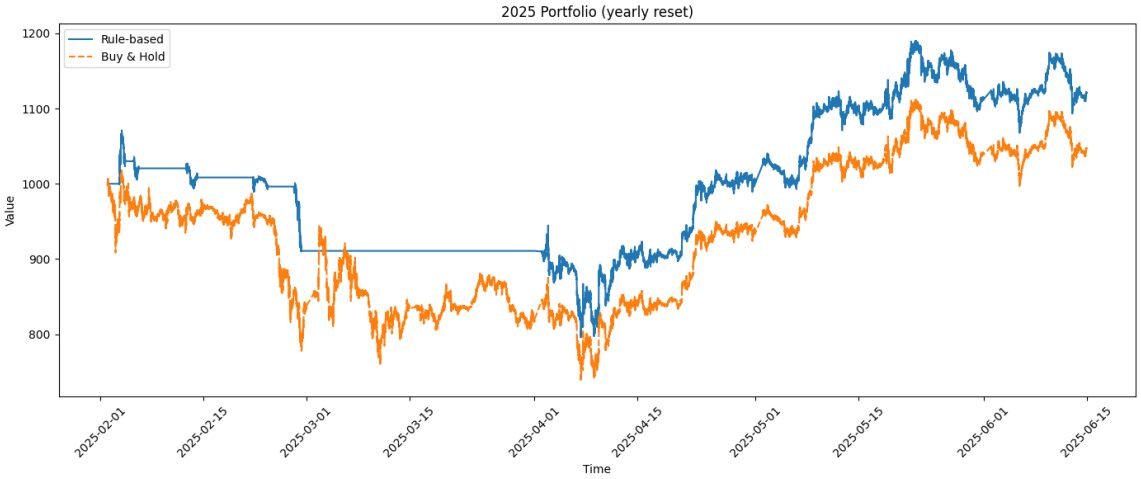


355) December

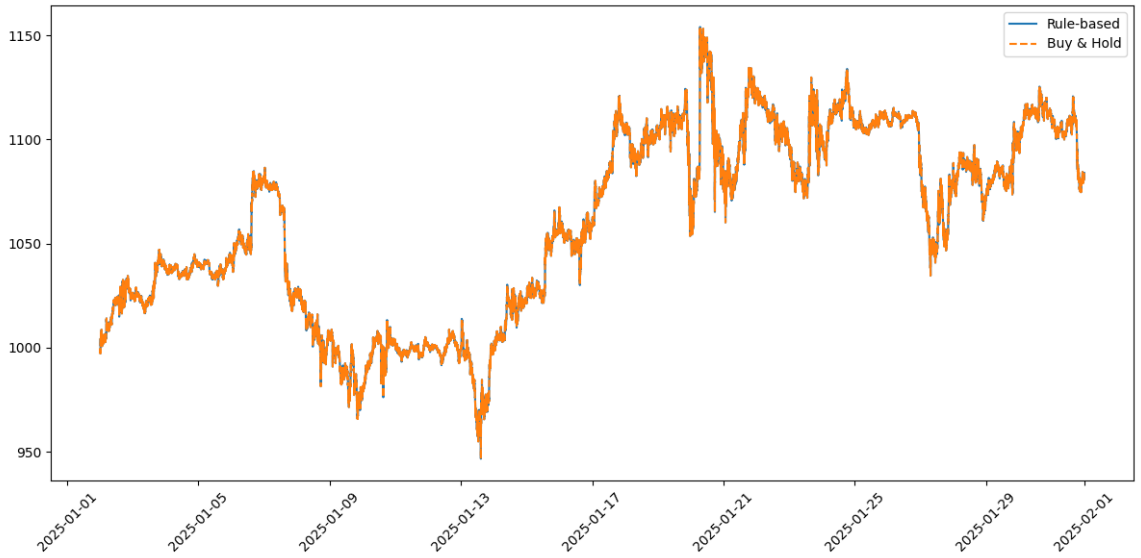


2025

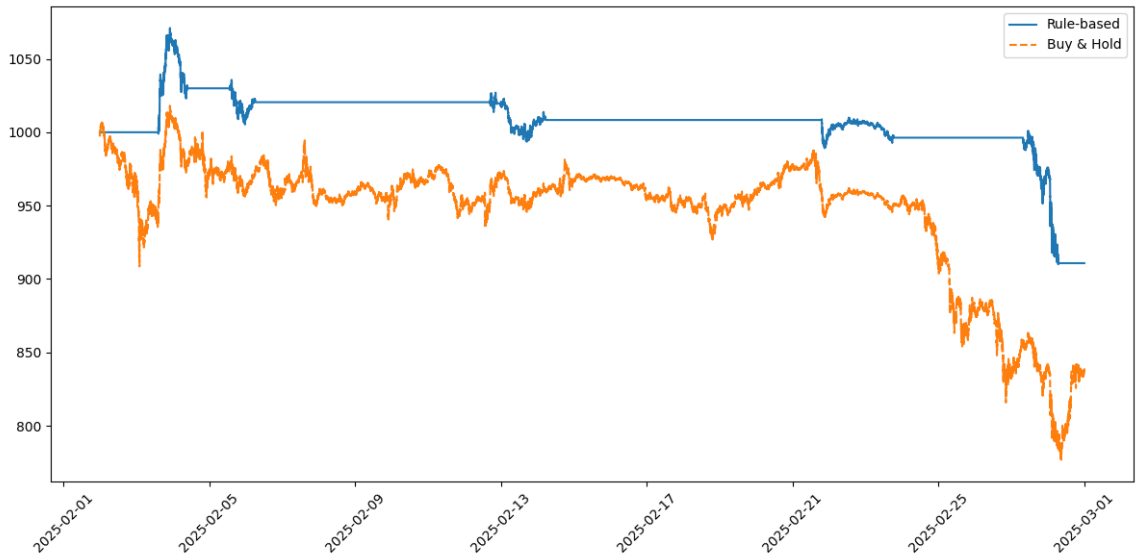
356) Full Year



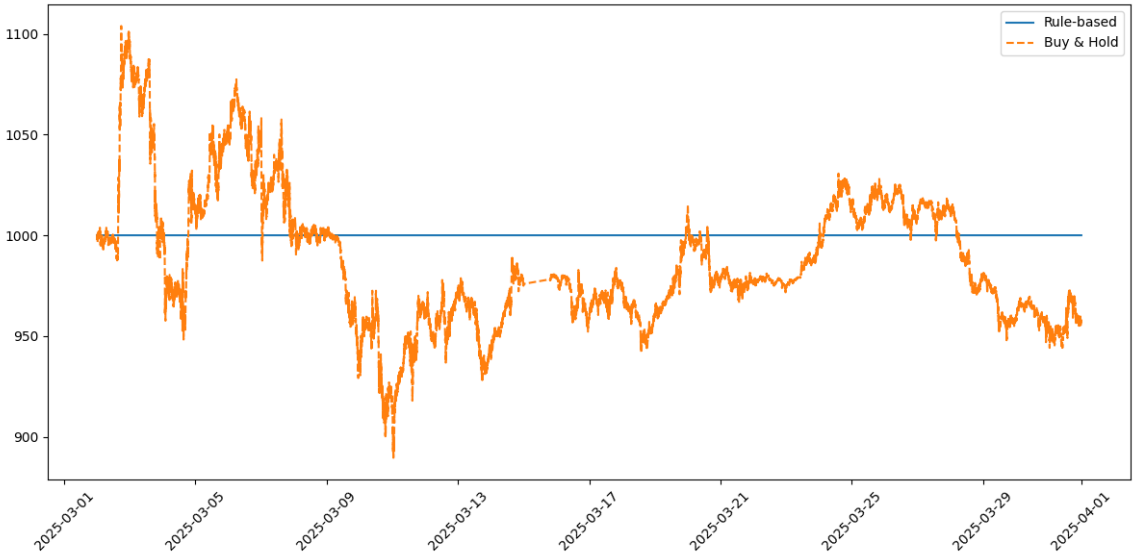
357) January



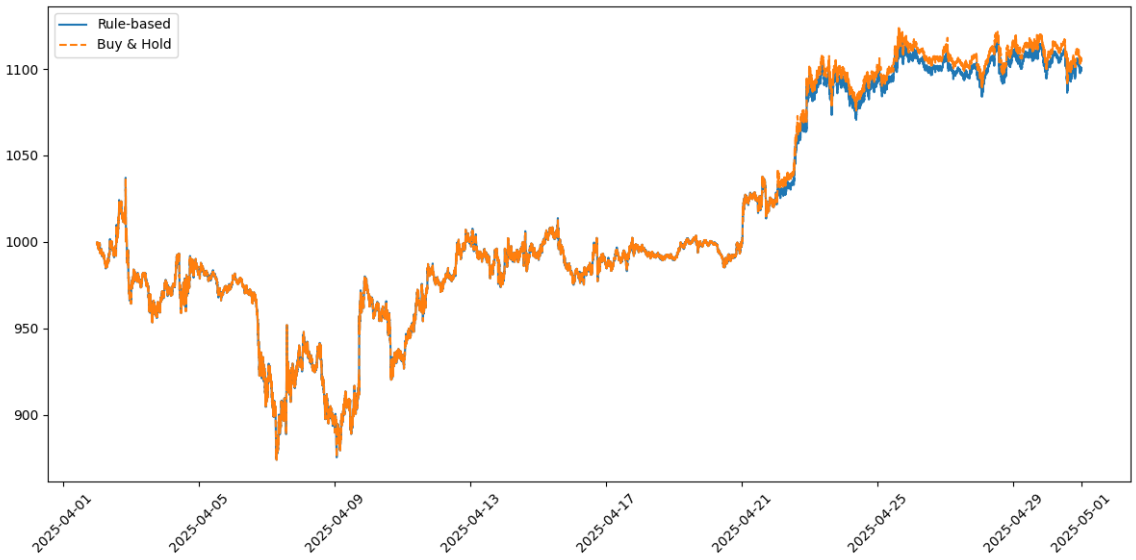
358) February



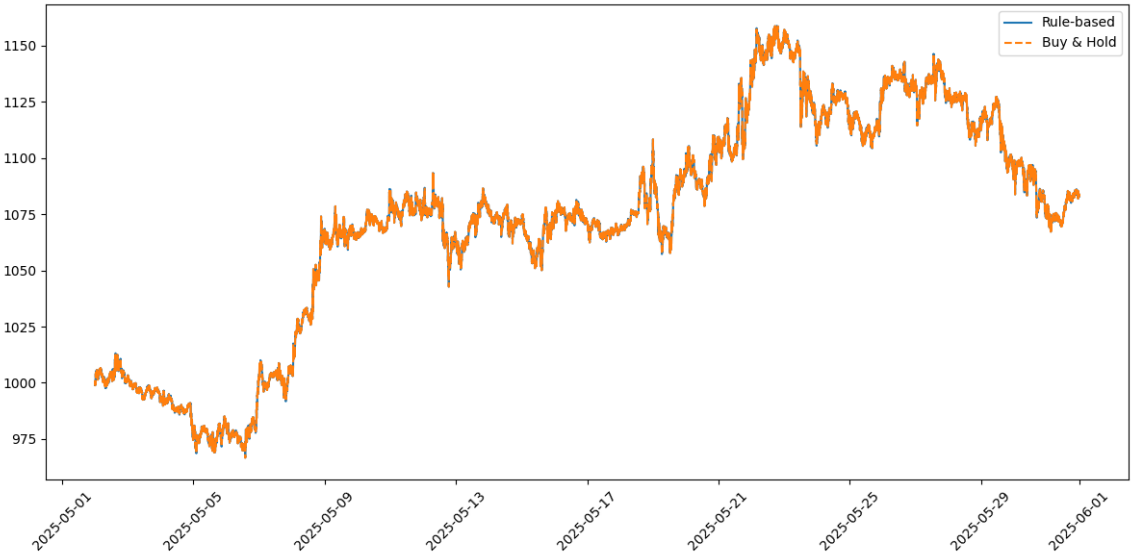
359) March



360) April



361) May



362) June