Backtest Overfitting in the Machine Learning Era:

A Comparison of Out-of-Sample Testing Methods in a Synthetic Controlled Environment

Hamid Arian[a,*,1], Daniel Norouzi Mobarekeh[b,2] and Luis Seco[c,3]

[c] University of Toronto, 40 St George St, Toronto, Ontario Canada M5S 2E4
[a] York University, 4700 Keele St, Toronto, Ontario, Canada M3J 1P3
[b] Sharif University of Technology, Teymoori Sq, Tehran, Iran 1459973941

ARTICLE INFO

Keywords:
Quantitative Finance
Machine Learning
Cross-Validation
Probability of Backtest Overfitting

ABSTRACT

This research explores the integration of advanced statistical models and machine learning in financial analytics, representing a shift from traditional to advanced, data-driven methods. We address a critical gap in quantitative finance: the need for robust model evaluation and out-of-sample testing methodologies, particularly tailored cross-validation techniques for financial markets. We present a comprehensive framework to assess these methods, considering the unique characteristics of financial data like non-stationarity, autocorrelation, and regime shifts. Through our analysis, we unveil the marked superiority of the Combinatorial Purged (CPCV) method in mitigating overfitting risks, outperforming traditional methods like K-Fold, Purged K-Fold, and especially Walk-Forward, as evidenced by its lower Probability of Backtest Overfitting (PBO) and superior Deflated Sharpe Ratio (DSR) Test Statistic. Walk-Forward, by contrast, exhibits notable shortcomings in false discovery prevention, characterized by increased temporal variability and weaker stationarity. This contrasts starkly with CPCV's demonstrable stability and efficiency, confirming its reliability for financial strategy development. The analysis also suggests that choosing between Purged K-Fold and K-Fold necessitates caution due to their comparable performance and potential impact on the robustness of training data in out-of-sample testing. Our investigation utilizes a Synthetic Controlled Environment incorporating advanced models like the Heston Stochastic Volatility, Merton Jump Diffusion, and Drift-Burst Hypothesis, alongside regime-switching models. This approach provides a nuanced simulation of market conditions, offering new insights into evaluating cross-validation techniques. Our study underscores the necessity of specialized validation methods in financial modeling, especially in the face of growing regulatory demands and complex market dynamics. It bridges theoretical and practical finance, offering a fresh outlook on financial model validation. Highlighting the significance of advanced cross-validation techniques like CPCV, our research enhances the reliability and applicability of financial models in decision-making.

1. Introduction

1.1. Background

The financial sector has witnessed a paradigmatic shift by integrating sophisticated statistical models and machine learning techniques into its analytical framework. This pivotal transition from traditional quantitative methods to more advanced, data-driven approaches signifies a new epoch in financial analysis. This new era is marked by the capability to process and analyze extensive datasets, revealing intricate market patterns previously obscured by the limitations of conventional methods. A key catalyst for this transformation has been the significant advancements in computational technology and the rise of high-frequency trading practices. These developments have led to a fundamental change in market dynamics. As a result, the need for robust and reliable model evaluation methodologies, particularly in cross-validation techniques, has gained unprecedented importance. Such methods are integral to maintaining the integrity and effectiveness of financial models, which are essential in guiding decision-making processes across a spectrum of financial activities, from asset allocation to risk management, in both buy-side and sell-side institutions.

1.2. Motivation

The impetus for our research stems from a pivotal observation: despite substantial progress in financial modeling and an escalating reliance on machine learning algorithms, there is a glaring shortfall in effectively validating these models within the ambit of financial markets. This research gap becomes more pronounced when considering the extensive

* The authors are listed in alphabetical order.
*Corresponding author: Hamid Arian, Assistant Professor of Finance, York University, Toronto, Ontario, Canada, email: harian@yorku.ca
ORCID: 0000-0002-4624-9421
[1] Assistant Professor of Finance, York University, Toronto, Ontario, Canada
email: harian@yorku.ca
[2] BSc student of Applied Mathematics & Economics, Sharif University of Technology, Tehran, Iran
email: norouzi@risklab.ai
[3] Professor, University of Toronto, Ontario, Canada
email: luis.seco@utoronto.ca

The presentation slides and a commentary on this article are available on RiskLab's website at the University of Toronto: risklab.ca/backtesting. The architecture of the codes of this article is explained on risklab.ai/backtesting, in both Python and Julia programming languages. The reproducible results of this paper are based on authors' Python implementation on RiskLab's GitHub page: github.com/RiskLabAI.

literature on predicting market factors. Yet, there is a conspicuous lack of discussion on tailoring cross-validation algorithms to accurately assess these models (Lopez de Prado [2018, 2020]). Further complicating this landscape is the paucity of research dedicated to critically evaluating the backtesting and cross-validation algorithms. We hypothesize that the limited exploration in this domain is attributable to the inherent complexities of financial datasets, which are typically noisy, non-stationary, and characterized by intricate patterns shaped by various variables, from macroeconomic shifts to market sentiments. These unique dataset attributes often render traditional cross-validation methods insufficient or misleading (Lopez de Prado [2018]). The grave consequences of model inaccuracies in this context cannot be overstated, as they can lead to substantial financial losses and pose systemic risks. This highlights the critical need to develop and refine cross-validation methodologies for navigating financial data nuances. While hedge funds and investment firms might have practical approaches to address these challenges, there is a stark silence in the academic literature on this imperative issue. Our study seeks to bridge this gap, providing insights and methodologies vital for the rigorous evaluation of financial models, thereby catering to finance's academic and practical realms.

1.3. Literature Review

1.3.1. Evolution of Backtesting on Out-of-Sample Data

The evolution of cross-validation (CV) methodologies in quantitative finance has been marked by a significant transition from traditional data science approaches to more specialized techniques tailored for financial market data. Conventional methods like K-Fold Cross-Validation and Walk-Forward Cross-Validation, while effective in various analytical contexts, have shown limitations when applied to financial markets due to their inability to adequately account for the temporal dependencies and non-stationarity inherent in financial time series. Recognizing these shortcomings, Lopez de Prado introduced advanced CV techniques specifically designed for financial applications. Purged K-Fold Cross-Validation, as outlined by Lopez de Prado [2018], enhances the standard K-Fold method by incorporating a 'purging' mechanism, eliminating data from the training set that could inadvertently leak information about the test set. This approach is particularly critical in financial modeling to prevent lookahead biases. Further advancing the field, Lopez de Prado's Combinatorial Purged Cross-Validation (CPCV) method offers a robust solution for backtesting trading strategies. Unlike traditional CV methods, CPCV creates multiple training and testing combinations, ensuring that each data segment is used for training and validation, thus providing a more comprehensive assessment of a strategy's performance across various market scenarios. This method respects the chronological ordering of data and effectively addresses the risk of overfitting, a prevalent issue in the development of financial models.

1.3.2. Rising Concerns over Backtest Overfitting and False Discoveries

The evolution of financial modeling has necessitated advanced methodologies to effectively address the challenges of overfitting and false discoveries in strategy evaluation. Pioneering contributions by Bailey et al. [2016] and Bailey and López de Prado [2014b], brought to the fore the need for rigorous evaluation of trading strategies. They introduced quantifiable metrics like the Probability of Backtest Overfitting (PBO) and the Deflated Sharpe Ratio (DSR), which provided a statistical basis to assess the reliability of backtested strategies. Despite these advancements, a significant gap exists in the literature: a comprehensive framework linking backtest overfitting assessment with the effectiveness of out-of-sample testing methodologies. Our study addresses this gap by proposing a novel framework that evaluates out-of-sample testing techniques through the prism of backtest overfitting. By integrating key concepts such as PBO and DSR into our analysis, we aim to provide a holistic evaluation of CV methods, ranging from traditional data science approaches to innovative financial models like those proposed by Lopez De Prado. This approach ensures financial models' robustness and predictive power, filling a critical void in quantitative finance.

1.3.3. Exploring Market Dynamics with Synthetic Controlled Environment

Advancements in synthetic data generation within financial analysis have seen the integration of sophisticated models that adeptly replicate complex market dynamics. Our study's Synthetic Controlled Environment embraces this complexity by merging the Heston Stochastic Volatility Model, characterized by the stochastic differential equation $dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_t^S$ (Heston [1993]), with the Merton Jump Diffusion Model (Merton [1976]), which introduces jumps in asset prices through $dS_t = \mu S_t dt + \sigma S_t dW_t + S_t dJ_t$. Following this, we explore the context of speculative market bubbles. Schatz and Sornette [2020] categorizes bubbles into Type-I and Type-II, with Type-I characterized by an efficient full price process S but inefficiencies in both pre-drawdown $\tilde{S}$ and drawdown X processes, and Type-II by an efficient drawdown process X but an overall inefficient S. This categorization provides a nuanced understanding of bubble dynamics in financial markets. Building on this foundation, our environment further incorporates the Drift Burst Hypothesis (Christensen et al. [2022]), articulating short-lived market anomalies through equations $\mu_t^{\text{db}} = a|\tau_{\text{db}} - t|^{-\alpha}$ and $\sigma_t^{\text{yb}} = b|\tau_{\text{db}} - t|^{-\beta}$, emphasizing the critical interplay between drift and volatility during such events. The addition of the Markov Chain model for regime transitions (Hamilton [1994]), characterized by its transition matrix $P = [P_{ij}]$, enables the simulation to adeptly mirror fluid market states adeptly, capturing the ephemeral nature of financial markets. This innovative amalgamation of stochastic volatility, jump-diffusion, bubble dynamics, and regime-switching, cohesively combined in our Synthetic Controlled Environment, sets a groundbreaking precedent in the domain of fi-

nancial model testing and validation, presenting a comprehensive framework for evaluating out-of-sample testing methodologies in the nuanced and intricate world of quantitative finance.

1.4. Problem Statement

Central to our study is a problem of increasing concern in financial analytics: developing a robust and reliable out-of-sample testing methodology congruent with the unique attributes of financial time series data. This issue is multifaceted. Firstly, financial time series are characterized by non-stationarity, autocorrelation, heteroskedasticity, and regime shifts, challenging the applicability of conventional out-of-sample testing methods. Secondly, the temporal dynamics of financial data, with intricate lead-lag relationships and evolutionary patterns, demand an out-of-sample testing approach that preserves the chronological sequence of data to avoid look-ahead bias and overfitting, issues frequently encountered in applying machine learning models in finance. Despite the remarkable advancements in integrating statistical models and machine learning techniques into financial analysis, a significant gap persists in accurately assessing these models, particularly under the challenges of backtest overfitting and the dynamic nature of financial markets. Our study specifically targets the inadequacy of existing cross-validation techniques, which, while robust in traditional data science contexts, fall short of fully capturing the temporal dependencies and non-stationarity of financial data. This gap is further widened by the lack of a comprehensive framework that integrates the assessment of backtest overfitting with the effectiveness of out-of-sample testing methodologies. The significance of this problem is not limited to theoretical modeling but has far-reaching implications in practical aspects like risk management, algorithmic trading, and portfolio optimization. Inaccuracies in model validation can lead to substantial financial risks and losses, accentuating the need for rigorous, tailored validation methods, especially under increasing regulatory scrutiny.

1.5. Objectives of the Study

The central objective of our study is to develop a comprehensive evaluation framework for cross-validation methods in financial modeling, particularly in the context of evolving market complexities and the challenges of backtest overfitting. By incorporating key concepts like the Probability of Backtest Overfitting (PBO) and the Deflated Sharpe Ratio (DSR), our framework holistically assesses various cross-validation approaches, ranging from traditional data science methods to more sophisticated financial models. The emphasis of this study is not inherently on the incorporation of sophisticated methodologies; rather, it centers on a critical assessment of the efficacy of these approaches when considering the distinctive attributes inherent in financial data. We aim to bridge the gap between theoretical robustness and practical reliability in financial models, enhancing their applicability in high-stakes financial decision-making, from asset allocation to risk management.

1.6. Contribution

This research significantly contributes to quantitative finance by pioneering a comprehensive framework for evaluating out-of-sample testing methodologies, particularly in financial modeling. We bridge a notable gap in the existing literature by linking the concept of backtest overfitting, as encapsulated by metrics like the Probability of Backtest Overfitting (PBO) and the Deflated Sharpe Ratio (DSR), with the efficacy of out-of-sample testing methods. Our innovative approach enhances the accuracy and reliability of cross-validation techniques, addressing the challenges posed by the temporal complexities and non-stationarity of financial time series. We leverage the advanced statistical models of the Heston Stochastic Volatility, Merton Jump Diffusion, and Markov Chain for regime transitions, combined with exploring market dynamics through speculative bubbles and the Drift Burst Hypothesis. This synthesis provides a more nuanced simulation of market conditions, offering fresh insights and methodologies that can significantly improve decision-making processes in various financial applications, from risk management to algorithmic trading. Our work advances the field by presenting a novel and holistic perspective on model validation in the ever-evolving quantitative finance domain, thus enhancing both financial practices and academic research.

1.7. Scope and Limitations

This study evaluates cross-validation methods within synthetic market environments meticulously engineered to encompass diverse market conditions. Our research uses sophisticated statistical models to dissect the intricacies of backtest overfitting in these rigorously constructed settings. A notable limitation of our approach is the reliance on synthetic data, which, while providing controlled experimental conditions, might not fully capture the complex, often unpredictable dynamics of real-world financial markets. Consequently, extrapolating our findings to actual market scenarios should be cautiously approached, especially when considering applications in live trading environments or risk management strategies. Moreover, while comprehensive, the specific choice of models and simulation parameters implies certain constraints. This necessitates further empirical validation in diverse, real-market contexts to enhance the generalizability of our results. Our study's primary aim is to enrich the domain of financial model validation, striking a crucial balance between theoretical depth and practical relevance and paving the way for subsequent research to build upon these foundational insights.

1.8. Organization of the Paper

This paper is systematically structured to explore cross-validation techniques in synthetic market environments comprehensively. The paper opens with **Introduction** 1, setting the stage by delineating the research background, objectives, and the scope of the study. Following this, the **Methodology** section 2 explores the details of the statistical models and algorithms employed, outlining the framework for synthetic data generation and analysis. The **Empirical Results** section 3 thoroughly examines our rigorous testing and analysis

findings, providing insights into the performance and robustness of various cross-validation methods. In the **Discussion** section 4, we interpret these findings, contextualizing them within the broader landscape of quantitative finance and discussing their implications. The paper culminates with the **Conclusion** section 5, where we summarize the key takeaways, acknowledge the limitations of our study, and suggest directions for future research.

2. Methodology

The methodology section forms the backbone of our research, presenting a comprehensive and systematic approach to exploring and analyzing financial market dynamics through machine learning and statistical methods. This section outlines the construction and utilization of a Synthetic Controlled Environment, which integrates complex market models such as the Heston Stochastic Volatility and Merton Jump Diffusion models and incorporates regime-switching dynamics through Markov chains. Additionally, it addresses the drift burst hypothesis to model market anomalies like speculative bubbles and flash crashes. The methodology elaborates on developing and evaluating a prototypical financial machine-learning strategy, encompassing event-based sampling, trade directionality, bet sizing, and feature selection. Crucially, the methodology also delves into assessing backtest overfitting through advanced statistical techniques, ensuring the validity and robustness of the proposed trading strategies. The methodologies are meticulously designed to capture the intricate nuances of financial markets, thereby enabling a thorough and accurate analysis of trading strategies within a controlled yet realistic market simulation.

2.1. Synthetic Controlled Environment

In financial analysis, constructing a Synthetic Controlled Environment is essential for thoroughly examining market dynamics and validating theoretical models. This segment delineates an integrated simulation architecture synthesizing the Heston model's stochastic volatility, Merton's jump-diffusion framework, and the Markov chains' regime-switching nuance. It also contemplates the drift burst hypothesis to capture transient market anomalies. These components construct a nuanced and comprehensive emulation of the financial market's complexity, serving as a critical substrate for the exploration and scrutiny of econometric theories.

2.1.1. Random Walk: The Heston Stochastic Volatility Model

In modeling the stochastic behavior of the market price, we employ the foundational Heston model, as articulated by Heston [1993]. This model provides a framework that captures the intrinsic volatility dynamics of a financial asset.

At the heart of the Heston model lies the premise that the asset price, S_t, evolves according to the following stochastic differential equation:

$$dS_t = \mu S_t \mathrm{d}t + \sqrt{v_t} S_t \mathrm{d}W_t^S, \tag{2.1}$$

where the instantaneous variance, v_t, adheres to the Feller square-root or Cox-Ingersoll-Ross (CIR) process:

$$dv_t = \kappa(\theta - v_t)\mathrm{d}t + \xi\sqrt{v_t}\mathrm{d}W_t^v, \tag{2.2}$$

with W_t^S and W_t^v representing Wiener processes, exhibiting a correlation of ρ.

The model described in Eqn. (2.1) and Eqn. (2.2) uses four main parameters. θ is the long-term average variance, showing the expected variance that v_t will approach as t increases. ρ describes the correlation between the two Wiener processes in the model. κ shows how quickly v_t returns to its long-term average, θ. And ξ is known as the 'volatility of volatility', indicating how much v_t can vary.

A salient feature of this model is the Feller condition, expressed as $2\kappa\theta > \xi^2$. Ensuring this inequality guarantees the strict positivity of the process, ensuring no negative values for variance.

2.1.2. Jumps: The Merton Jump Diffusion Model

The Merton Jump Diffusion model by Merton [1976] enhances the geometric Brownian motion proposed by the Black-Scholes model by integrating a discrete jump component to capture abrupt stock price movements. The stock price dynamics are given by:

$$dS_t = \mu S_t dt + \sigma S_t dW_t + S_t dJ_t, \tag{2.3}$$

In Eqn. (2.3), $\mu S_t dt$ is the drift term that captures the expected return, $\sigma S_t dW_t$ embodies the continuous random fluctuations with σ being the stock's volatility, and dW_t the standard Brownian motion increment, and $S_t dJ_t$ accounts for instantaneous jumps in the stock price.

The jump process J_t in Eqn. (2.3) is defined as:

$$J_t = \sum_{i=1}^{N(t)} Y_i, \tag{2.4}$$

where $N(t)$ is a Poisson process with intensity λ, and Y_i represents logarithmic jump sizes, normally distributed with mean m and standard deviation s.

To simulate paths of S_t, one evolves the stock price using the drift and diffusion terms, determines jumps based on $N(t)$, and adjusts the stock price according to the magnitude from the Y_i distribution. By merging continuous price movements with jumps, this model potentially offers a more accurate representation of real-world stock price behaviors than mere geometric Brownian motion.

2.1.3. Speculative Bubbles & Flash Crashes: The Drift Burst Hypothesis

In the study Christensen et al. [2022], the authors introduce the drift burst hypothesis to elucidate the short-lived flash crashes evident in high-frequency tick data. This methodology zeroes in on the complex dance between drift and volatility. They theorize that a sudden uptick in drift is only viable if there's a simultaneous surge in volatility. To articulate this, they introduce the "volatility burst" concept, denoting a rapid escalation in market volatility.

The drift's sudden increase is concisely encapsulated in the equation:

$$\mu_t^{\text{db}} = a|\tau_{\text{db}} - t|^{-\alpha}. \tag{2.5}$$

In Eqn. (2.5), μ_t^{db} describes the drift at a given time t according to its distance relative to the bursting time τ_{db}. The factor a sets the scale of the drift, while $\frac{1}{2} < \alpha < 1$ measures how intense this drift spike is.

Similarly, the abrupt rise in volatility, or the "volatility burst", is represented as:

$$\sigma_t^{\text{vb}} = b|\tau_{\text{db}} - t|^{-\beta}. \tag{2.6}$$

In Eqn. (2.6), σ_t^{vb} indicates the volatility at time t. The parameter b quantifies the size of this volatility surge, and $0 < \beta < \frac{1}{2}$, gauges its sharpness.

2.1.4. Regime Transitions: Markov Chain

A regime-switching time series model is applied to simulate market dynamics, following Hamilton [1994] as mentioned by Lopez de Prado [2020]. The market is segmented into discrete regimes, each with unique characteristics. The market's transition between these regimes at any given time t is determined by a Markov chain, where the transition probability $p_{t,n}$ depends solely on the state immediately prior. This approach captures the fluid nature of financial markets, which fluctuate between different states, reflecting shifts in volatility and trends. By employing a Markov chain, these transitions are modeled with mathematical precision while maintaining economic plausibility, recognizing that financial markets tend to exhibit a memory of only the most recent events.

A Markov chain is a mathematical system that transitions from one state to another in a state space. It is defined by its set of states and the transition probabilities between these states. The fundamental property of a Markov chain is that the probability of moving to the next state depends only on the present state and not on the sequence of events that preceded it.

Given a finite number of states $S = \{s_1, s_2, \ldots, s_n\}$, the probability of transitioning from state s_i to state s_j in one step is denoted by P_{ij}:

$$P_{ij} = P(X_{n+1} = s_j | X_n = s_i), \tag{2.7}$$

where X_n represents the state at time n, and P_{ij} is the entry in the i-th row and j-th column of the transition matrix P. The matrix $P = [P_{ij}]$ is called the transition matrix of the Markov chain. Each entry P_{ij} represents the one-step transition probability from state s_i to state s_j as in Eqn. (2.7):

$$P = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ P_{21} & P_{22} & \cdots & P_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{bmatrix}. \tag{2.8}$$

2.1.5. Market Synthesis: Discrete Simulation

In our study, we employ a discrete simulation approach to model market dynamics, which can be effectively represented by the Euler-Maruyama method for stochastic differential equations. This method provides a numerical approximation of the continuous market processes in a discrete framework. By applying Ito's Lemma, the approximation is given by:

$$\Delta S_t \approx \left(\mu - \frac{1}{2}v_t - \lambda\left(m + \frac{v^2}{2}\right)\right) S_t \Delta t + \sqrt{v_t} S_t Z \sqrt{\Delta t} + Y \Delta N(t). \tag{2.9}$$

In Eqn. (2.9), ΔS_t is the change in asset price, μ represents the drift rate, and $\sqrt{v_t}$ is the volatility factor scaled by the standard normal random variable Z. Y is a normally distributed jump size with mean m and variance v^2, and $\Delta N(t)$ denotes the jump process increments characterized by a Poisson distribution with intensity $\lambda \Delta t$.

The variation in instantaneous variance v_t is captured by Eqn. (2.10):

$$\Delta v_t = \kappa(\theta - v_t)\Delta t + \xi \sqrt{v_t}(\rho_\epsilon \epsilon_t^P + \sqrt{1 - \rho_\epsilon^2}\epsilon_t^v)\sqrt{\Delta t}, \tag{2.10}$$

where κ is the rate at which v_t reverts to its long-term mean θ, and ξ measures the volatility of the variance. The correlated standard normal white noises ϵ_t^v and ϵ_t^P introduce randomness with a correlation coefficient ρ_ϵ. The factor $\sqrt{\Delta t}$ is introduced to scale the model appropriately in the discrete-time setting, reflecting the properties of Brownian motion increments.

Incorporating the Markov chain regime transition model into our discrete simulation, the constants μ, θ, ξ, ρ_ϵ, λ, m, and v^2 are adjusted for each regime. The adjustment is dictated by the state transitions determined by the Markov chain, where each state encapsulates a distinct market regime with its own parameter set. As the market transitions between regimes, these parameters change accordingly, aligning the simulation with the underlying stochastic process that reflects the dynamic financial market environment.

2.2. Prototypical Financial Machine Learning Strategy

Developing a coherent machine-learning strategy in quantitative finance necessitates a meticulous fusion of statistical techniques and market knowledge. Our proposed methodology rigorously combines event-based triggers, trend-following mechanisms, and risk assessment tools to formulate a prototypical financial machine-learning strategy. It commences with precisely identifying market events through CUSUM filtering and progresses to ascertain trade directionality via momentum analysis. The core of the strategy harnesses meta-labeling to assess trade viability and employs an averaging approach to bet sizing sensitive to market conditions and position overlap. Integrating fractionally differentiated features alongside traditional technical indicators forms a robust feature set, ensuring the preservation of temporal dependencies and adherence to stationarity—a prerequisite for

the successful application of predictive modeling in financial contexts.

2.2.1. Sampling: CUSUM Filtering

Portfolio management often relies on event-based triggers for investment decisions. These events may include structural breaks, signals, or microstructural changes, often prompted by macroeconomic news, volatility shifts, or significant price deviations. In this context, it is crucial to identify such events accurately, leveraging machine learning (ML) to ascertain the potential for reliable predictive models. The redefinition of significant events or the enhancement of feature sets is a continual process refined upon discovering non-predictive behaviors.

We employ the Cumulative Sum (CUSUM) filter as an event-based sampling technique for methodological rigor, as mentioned by Lopez de Prado [2018]. This method detects deviations in the mean of a quantity, denoting an event when a threshold is crossed. Given independent and identically distributed (IID) observations from a locally stationary process $\{y_t\}_{t=1,\dots,T}$, we define the CUSUM as:

$$S_t = \max\left\{0, S_{t-1} + y_t - \mathbb{E}_{t-1}\left(y_t\right)\right\}, \tag{2.11}$$

with the initial condition $S_0 = 0$. A signal for action is suggested at the smallest time t where $S_t \geq h$, with h being the predefined threshold or filter size. It's notable that S_t is reset to zero if $y_t \leq \mathbb{E}_{t-1}\left(y_t\right) - S_{t-1}$, which intentionally ignores negative shifts.

To encompass both positive and negative shifts, we extend this to a symmetric CUSUM filter:

$$\begin{aligned} S_t^+ &= \max\left\{0, S_{t-1}^+ + y_t - \mathbb{E}_{t-1}\left(y_t\right)\right\}, \quad S_0^+ = 0, \\ S_t^- &= \min\left\{0, S_{t-1}^- + y_t - \mathbb{E}_{t-1}\left(y_t\right)\right\}, \quad S_0^- = 0, \\ S_t &= \max\left\{S_t^+, -S_t^-\right\}. \end{aligned} \tag{2.12}$$

Adopting Lam and Yam [1997]'s strategy, we generate alternating buy-sell signals upon observing a return h relative to a prior peak or trough, akin to the filter trading strategy by Fama and Blume [1966]. Our application of the CUSUM filter using Eqn. (2.12), however, is distinct; we only sample at bar t if $S_t \geq h$, subsequently resetting S_t assuming $\mathbb{E}_{t-1}\left(y_t\right) = y_{t-1}$. We define y_t as the natural logarithm of the asset's price to capture proportional price movements. The threshold h is not static; instead, it dynamically adjusts directly to the daily volatility, ensuring sensitivity to market conditions.

2.2.2. Side Determination: Momentum Strategy

We employ a momentum strategy based on moving averages to determine the direction of trades signaled by the event-based CUSUM filter sampling. Specifically, we calculate two moving averages of the prices, a short-term moving average $\mathrm{MA}_{\mathrm{short}}(y_t)$ and a long-term moving average $\mathrm{MA}_{\mathrm{long}}(y_t)$, to identify the prevailing trend. The short-term moving average is responsive to recent price changes, while the long-term moving average captures the underlying trend. These moving averages are formulated as follows:

$$\begin{aligned} \mathrm{MA}_{\mathrm{short}}(y_t) &= \frac{1}{N_{fast}} \sum_{i=0}^{N_{fast}-1} y_{t-i}, \\ \mathrm{MA}_{\mathrm{long}}(y_t) &= \frac{1}{N_{slow}} \sum_{i=0}^{N_{slow}-1} y_{t-i}, \end{aligned} \tag{2.13}$$

where N_{fast} and N_{slow} represent the number of periods for the fast (short-term) and slow (long-term) moving averages, respectively.

A position is taken based on the relative positioning of these moving averages post a CUSUM event. A trade is initiated based on these conditions:

1. **Long Position:** Triggered when $\mathrm{MA}_{\mathrm{short}}(y_t)$ surpasses $\mathrm{MA}_{\mathrm{long}}(y_t)$, signaling upward market momentum.
2. **Short Position:** Initiated when $\mathrm{MA}_{\mathrm{short}}(y_t)$ falls below $\mathrm{MA}_{\mathrm{long}}(y_t)$, indicating downward market momentum.

The strategy, thus, aligns the position with the current market trend, as indicated by the momentum in prices.

2.2.3. Size Determination: Meta-Labeling via Triple-Barrier Method

In our trading framework, once the side of a position is determined through the momentum strategy, it undergoes a rigorous evaluation via the triple-barrier method to ascertain its potential profitability. This evaluation forms the basis for position sizing, leveraging a meta-labeling approach introduced by Lopez de Prado [2018].

Upon identification of a trade's direction, the triple-barrier method applies three distinct barriers to determine the outcome of the position. The horizontal barriers are set according to a dynamic volatility-adjusted threshold for profit-taking and stop-loss, while the vertical barrier is defined by a predetermined expiration time, denoted as h. The label assignment is as follows: hitting the upper barrier signifies a successful trade, hence labeled 1; conversely, touching the lower barrier first indicates a loss, labeled -1. If the vertical time barrier expires first, the label is determined by the sign of the return, reflecting the result of the trade within the period $[t_{i,0}, t_{i,0} + h]$.

The role of meta-labeling in this context is to scrutinize further the trades indicated by the primary momentum model. It confirms or refutes the suggested positions, effectively filtering out false positives and allowing for a calculated decision on the actual size of the investment. The meta-labeling process directly informs the appropriate risk allocation for each position by assigning a confidence level to each potential trade. This methodological step enhances the precision of our strategy and ensures that position sizing is aligned with the evaluated profitability of the trade, as indicated by the outcome of the triple-barrier assessment.

2.2.4. Sample Weights: Label Uniqueness

The validity of the Independent and Identically Distributed (IID) assumption is a common shortfall in financial machine

learning, as the overlapping intervals in the data often violate it. Specifically, labels y_i and y_j may not be IID if there is a shared influence from a common return $r_{t_{j,0},\min\{t_{i,1},t_{j,1}\}}$, where $t_{i,1} > t_{j,0}$ for consecutive labels $i < j$. To address the non-IID nature of financial datasets without compromising the model granularity, we utilize sample weights as introduced by Lopez de Prado [2018]. This method recognizes the interconnectedness of data points and adjusts their influence on the model accordingly. By weighing samples based on their unique information and return impact, we enhance model robustness, enabling more accurate analysis of financial time series.

We define concurrent labels at time t as those that are both influenced by at least one shared return

$$r_{t-1,t} = \frac{p_t}{p_{t-1}} - 1. \tag{2.14}$$

The concurrency of labels y_i and y_j does not necessitate a complete overlap in period; rather, it is sufficient that there is a partial temporal intersection involving the return at time t.

To quantify the extent of overlap, we construct a binary indicator array $\{1_{t,i}\}_{i=1,\ldots,I}$ for each time t, where $1_{t,i}$ is set to 1 if the interval $[t_{i,0}, t_{i,1}]$ overlaps with $[t-1, t]$, and 0 otherwise. We then calculate the concurrency count at time t, given by

$$c_t = \sum_{i=1}^{I} 1_{t,i}. \tag{2.15}$$

The uniqueness of a label is inversely proportional to the number of labels concurrent with it (Eqn. (2.15)). Consequently, we assign sample weights by inversely scaling them with the concurrency count while considering the magnitude of returns over the label's lifespan. For label i, the preliminary weight $\tilde{w}_i$ is computed as the norm of the sum of proportionally attributed returns:

$$\tilde{w}_i = \left\| \sum_{t=t_{i,0}}^{t_{i,1}} \frac{r_{t-1,t}}{c_t} \right\|. \tag{2.16}$$

To facilitate a consistent scale for optimization algorithms that default to an assumption of unit sample weights, we normalize these preliminary weights calculated in Eqn. (2.16) to sum to the total number of labels I:

$$w_i = \frac{\tilde{w}_i}{\sum_{j=1}^{I} \tilde{w}_j}. \tag{2.17}$$

Eqn. (2.17) ensures that $\sum_{i=1}^{I} w_i = I$. Through this weighting scheme, we emphasize observations with greater absolute log returns that are less common, thereby enhancing the model's capacity to learn from unique and significant market events.

2.2.5. *Financial Features: Fractional Differentiation & Technical Analysis*

In pursuing a robust financial machine-learning model, our methodology encompasses diverse features that balance memory preservation with the necessity for stationarity. Fractional differentiation of log prices is employed to maintain as much informative historical price behavior as possible while ensuring the data adheres to the stationary requirement of predictive models (Lopez de Prado [2018]). Additionally, we incorporate exponentially weighted moving averages (EWMA) of volatility, capturing recent market volatility trends and a suite of technical analysis indicators that provide insights into market sentiment and dynamics. Technical analysis features are extracted from historical price and volume data and are widely used to capture market sentiment and trends, which are indicative of future price movements and provide structured information from the otherwise noisy market data, aiding the machine learning model to discern patterns associated with profitable trading opportunities. The features used for this problem are as follows:

1. **FracDiff:** The fractionally differentiated log price. Financial time series are characterized by a low signal-to-noise ratio and memory, challenging traditional stationarity transformations like integer differentiation, which remove this memory and potentially valuable predictive signals (Lopez De Prado [2015]). To address this, fractional differentiation is employed to preserve memory while ensuring stationarity.
   Consider a time series $\{X_t\}$ and the backshift operator B such that $B^k X_t = X_{t-k}$ for any non-negative integer k. The binomial theorem applied to an integer power can be extended to real powers using the binomial series and applied to the backshift operator:

$$(1-B)^d = \sum_{k=0}^{\infty} \binom{d}{k} (-B)^k = \sum_{k=0}^{\infty} \frac{\prod_{i=0}^{k-1}(d-i)}{k!} (-B)^k. \tag{2.18}$$

   The expansion in Eqn. (2.18) yields weights ω_k, which are applied to past values of the series to compute the fractionally differentiated series $\tilde{X}_t$:

$$\tilde{X}_t = \sum_{k=0}^{\infty} \omega_k X_{t-k}, \quad \text{with } \omega_k = (-1)^k \prod_{i=0}^{k-1} \frac{d-i}{k!}. \tag{2.19}$$

   An approach to fractional differentiation employs a fixed-width window by truncating the infinite series based on a threshold criterion for the weights. The fixed-width window approach can be formalized as follows: find the smallest l^* such that the modulus of the weights $\|\omega_{l^*}\|$ is not less than the threshold τ, and $\|\omega_{l^*+1}\|$ falls below τ. The adjusted weights $\tilde{\omega}_k$ are

then defined by:

$$\tilde{\omega}_k = \begin{cases} \omega_k & \text{if } k \leq l^*, \\ 0 & \text{if } k > l^*. \end{cases} \tag{2.20}$$

Applying these truncated weights, the fractionally differentiated series $\tilde{X}_t$ is obtained through a finite sum:

$$\tilde{X}_t = \sum_{k=0}^{l^*} \tilde{\omega}_k X_{t-k}, \quad \text{for } t = T - l^* + 1, \ldots, T. \tag{2.21}$$

The resultant series in Eqn. (2.21) is a driftless mixture of the original level and noise components, providing a stationary series despite its non-Gaussian distribution that exhibits memory-induced skewness and kurtosis.

For a given time series $\{X_t\}_{t=1,\ldots,T}$, the fixed-width window fractional differentiation (FFD) approach is utilized to determine the order of differentiation d^* that achieves stationarity in the series $\{\tilde{X}_t\}_{t=l^*,\ldots,T}$ using ADF tests. The value of d^* indicates the memory that must be eliminated to attain stationarity.

2. **Volatility:** Volatility is a fundamental feature that captures the magnitude of price movements and is critical for modeling risk and return in financial markets. The exponentially weighted moving average (EWMA) of volatility gives more weight to recent observations, making it a responsive measure of current market conditions. The EWMA volatility for a given day t is calculated as follows:

$$\sigma_t^{EWMA} = \sqrt{\lambda \sigma_{t-1}^2 + (1-\lambda) r_t^2}, \tag{2.22}$$

where r_t is the log return at time t, and λ is the decay factor that determines the weighting of past observations.
3. **Z-Score:** The Z-Score standardizes the log prices by their deviation from a rolling mean relative to the rolling standard deviation, highlighting price anomalies.
4. **Log MACD Histogram:** The difference between the logarithmically transformed MACD line and its corresponding signal line indicates momentum shifts.
5. **ADX:** The Average Directional Index measures the strength of a trend over a given period, with higher values indicating stronger trends.
6. **RSI:** The Relative Strength Index identifies conditions where the asset is potentially overbought or oversold, often signaling possible reversals.
7. **CCI:** The Commodity Channel Index detects cyclical trends in asset prices, often used to spot impending market reversals.
8. **Stochastic:** The Stochastic Oscillator compares the closing price to its price range over a specified period, indicating momentum.
9. **ROC:** The Rate of Change measures the velocity of price changes, with positive values indicating upward momentum and negative values indicating downward momentum.
10. **ATR:** The Average True Range quantifies market volatility by averaging true ranges over a period, reflecting the degree of price volatility.
11. **Log DPO:** The logarithm of the Detrended Price Oscillator compares rolling means at different periods to identify cyclical patterns in the price data.
12. **MACD Position:** Indicates the position of the MACD Histogram relative to its signal line, with values above zero suggesting a bullish crossover and below zero a bearish crossover.
13. **ADX Strength:** Reflects the trend's strength as measured by the ADX, categorizing trends as strong if above a threshold value and weak if below.
14. **RSI Signal:** Categorizes the RSI reading as signaling overbought conditions above a high threshold or oversold conditions below a low threshold.
15. **CCI Signal:** Provides a signal based on the CCI reading, indicating overbought or oversold conditions when crossing predefined threshold levels.
16. **Stochastic Signal:** Generates a signal from the Stochastic Oscillator, identifying overbought or oversold conditions based on threshold levels.
17. **ROC Momentum:** Categorizes the momentum based on the ROC, with positive values indicating an upward momentum and negative values a downward momentum.
18. **Kumo Breakout:** Identifies price breakouts from the Ichimoku Cloud, suggesting a bullish breakout when the price is above the cloud and bearish when below.
19. **TK Position:** Indicates the position of the Tenkan-sen relative to the Kijun-sen in the Ichimoku Indicator, with values above one suggesting a bullish crossover and below one a bearish crossover.
20. **Price Kumo Position:** Categorizes the price position relative to the Ichimoku Cloud, suggesting bullish sentiment when above the cloud and bearish when below.
21. **Cloud Thickness:** Measures the thickness of the Ichimoku Cloud by taking the logarithm of the ratio between the cloud spans, indicating market volatility and support/resistance strength.
22. **Momentum Confirmation:** Confirms the momentum indicated by the Ichimoku Indicator, with the Tenkan-sen above the cloud suggesting bullish momentum and below suggesting bearish momentum.

2.2.6. Bet Sizing: Averaging Active Bets

Proper bet sizing is crucial in implementing a successful investment strategy informed by machine learning predictions. We denote by $p[x]$ the probability of a label x occurring, where $x \in \{-1, 1\}$. To determine the appropriateness of a bet, we test the null hypothesis:

Null Hypothesis 1. $H_0 : p[x = 1] = \frac{1}{2}$.

Calculating the test statistic:

$$z = \frac{p[x=1] - \frac{1}{2}}{\sqrt{p[x=1](1-p[x=1])}} \sim Z, \tag{2.23}$$

where $z \in (-\infty, +\infty)$ and Z represents the standard normal distribution. The bet size is then derived as

$$m = 2Z[z] - 1, \tag{2.24}$$

with $m \in [-1, 1]$ and $Z[\cdot]$ being the cumulative distribution function (CDF) of Z for Eqn. (2.23). This formulation accounts for predictions originating from both meta-labeling and standard labeling estimators.

The process of bet sizing involves determining the size of individual bets based on the probability of outcomes and managing the aggregation of multiple bets that may be active concurrently. To manage multiple concurrent bets, we define a binary indicator $\{1_{t,i}\}$ for each bet i at time t. This indicator takes the value of 1 if bet i is active within the interval $(t-1, t]$, and 0 otherwise. The aggregate bet size at time t is then the average of all active bet sizes as shown in Eqn. (2.25):

$$m_t = \frac{\sum_{i=1}^{I} m_i 1_{t,i}}{\sum_{i=1}^{I} 1_{t,i}}, \tag{2.25}$$

where m_i is the individual bet size.

2.3. Strategy Trials

This section presents our strategy trials, which are integral to our financial machine-learning research. We employ a comprehensive methodology, examining machine learning models like k-Nearest Neighbors, Decision Trees, and XGBoost, each with unique parameter settings. Our approach deliberately tests these models under conditions conducive to overfitting to assess their robustness and adaptability. We also introduce the Momentum Cross-Over Strategy, utilizing various moving average window lengths to align trades with market trends. This combination of diverse models and adaptive strategies, processed through a systematic pipeline that includes event-based sampling, meta-labeling, and iterative optimization, is designed to rigorously evaluate the efficacy of trading strategies in complex market scenarios. The trials aim to balance the exploration of machine learning potentials in finance with the pragmatic challenges of real-world market conditions.

2.3.1. Machine Learning Models: An Overview

In our strategic analysis, we leverage various machine learning models, each with a distinct set of parameters. This approach is designed to rigorously test the models under varying conditions, potentially increasing the risk of overfitting. This methodological choice serves a dual purpose: firstly, to rigorously challenge the robustness of the models under extreme parameter conditions, and secondly, to examine the models' performance in scenarios prone to overfitting. This deliberate stress testing provides valuable insights into the resilience and adaptability of the algorithms in complex financial environments. The following models and their respective parameter sets are integral to this analysis:

1. **K-Nearest Neighbors (k-NN):** The k-NN model is predicated on feature similarity and is highly sensitive to the number of neighbors chosen. By experimenting with small numbers of neighbors, we expose the model to potential overfitting, where it might rely too heavily on immediate, possibly noisy data points. The model used in our study is a custom pipeline integrating standard scaling with the KNeighborsClassifier.
2. **Decision Tree:** Decision Trees, while interpretable, can easily overfit the training data, especially without constraints on tree depth. Our configuration tests the model in its most unconstrained form, providing insights into its behavior without regularizing parameters. Our implementation uses a Decision Tree Classifier with a predefined random state for reproducibility. The parameters include the maximum depth of the tree, the minimum number of samples required to split an internal node, and the minimum number of samples required to be at a leaf node.
3. **XGBoost:** XGBoost is an advanced implementation of gradient boosting algorithms known for its efficiency, flexibility, and portability. However, with excessively high values for parameters such as the number of estimators and learning rates, there is a risk of overfitting, where the model becomes overly tailored to the training data. It excels in handling sparse data and scales effectively across multiple cores. In our setup, the XGBoost Classifier is employed with specific parameters like the number of trees, maximum depth of trees, learning rate, and subsampling ratio of the training instances.

Each model is exhaustively assessed across its parameter space to evaluate its efficacy and robustness in various market scenarios. This extensive parameterization is a deliberate strategy to test the models' susceptibility to overfitting, a critical consideration in financial machine-learning applications.

2.3.2. Momentum Cross-Over Strategy: An Overview

The Momentum Cross-Over Strategy is a key element of our strategy trials, aiming to align trade directions with market trends detected through moving averages. This strategy's adaptability lies in its various combinations of window lengths for the moving averages, allowing it to capture market momentum over different time frames. By experimenting with multiple window length pairs, the strategy adjusts to various market conditions and introduces flexibility that increases the likelihood of overfitting. This approach ensures a thorough examination of market trends, aiming to optimize trade positions in line with the prevailing market direction.

2.3.3. Trials on Synthesized Data: The Pipeline

Our strategy trials employ a streamlined pipeline to assess the potential for overfitting in various trading strategies. The pipeline integrates event-based sampling, momentum strategy, machine learning models, and meta-labeling to simulate diverse market conditions and test strategy efficacy. The key steps of this pipeline are:

1. **CUSUM Sampling:** The process begins with the CUSUM

filter, identifying significant market shifts based on deviations in log prices. This method generates signals for potential trading opportunities.

2. **Momentum Cross-Over Strategy:** Following CUSUM signals, the Momentum Cross-Over Strategy is applied. This step involves choosing window sizes for calculating moving averages and determining the trade direction based on their relative positions.
3. **Machine Learning Model Selection:** A machine learning model, such as k-NN, Decision Tree, or XGBoost, is selected with specific parameters. This stage tests model responses to trading signals, emphasizing the analysis of overfitting risks under varying parameter settings.
4. **Meta-Labeling and Sample Weights:** Trade signals are processed through meta-labeling using the Triple-Barrier Method while concurrently assigning sample weights to tackle the non-IID nature of financial data, thus enhancing the model's learning efficacy.
5. **Model Fitting and Testing:** The chosen model is fitted to the data, now with meta-labels and weights, to evaluate its predictive accuracy under synthesized conditions.

This pipeline approach critically examines the interplay between different components of trading strategies, focusing on the risk of overfitting. By simulating complex market scenarios, we aim to validate the robustness and adaptability of these strategies for real-world application.

2.4. Backtesting on Out-of-Sample Data: Cross-Validation

In quantitative finance, the rigor of a trading strategy is often validated through backtesting on out-of-sample data. This process involves assessing the strategy's performance using data not employed during the model's training phase, providing insights into its real-world applicability. Cross-validation (CV) techniques are pivotal, offering structured methods to evaluate the strategy's effectiveness and robustness under various market conditions. The methodologies for backtesting range from conventional approaches like K-Fold Cross-Validation, which divides the data into multiple segments for iterative testing, to more specialized methods like Walk-Forward Cross-Validation and Combinatorial Purged Cross-Validation. Each method has distinct characteristics in handling the data, particularly addressing the challenges posed by the temporal dependencies and non-stationarity in financial time series. Understanding these methods' nuances in constructing backtest pathways is crucial for accurate model validation and developing robust trading strategies.

2.4.1. Conventional Approach: K-Fold Cross-Validation

K-fold cross-validation is a widely recognized statistical method for validating the performance of predictive models, particularly in machine learning contexts. It involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (the training set), and validating the analysis on the other subset (the test set). To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

In financial modeling, especially for backtesting trading strategies, applying K-Fold cross-validation presents unique challenges. Financial data are typically time-series data characterized by temporal dependencies and non-stationarity. These features of financial data violate the fundamental assumption of traditional K-Fold cross-validation, which assumes that the observations are independent and identically distributed (i.i.d).

The process of K-Fold cross-validation in financial backtesting involves the following steps:

1. The entire dataset is divided into k consecutive folds or segments.
2. For each iteration, a different fold is treated as the test set (or validation set), and the remaining $k - 1$ folds are combined to form the training set.
3. The model is trained on the training set and validated on the test set.
4. The performance metric (e.g., Sharpe ratio, annualized return, drawdown) is recorded for each iteration.
5. After iterating through all folds, the performance metrics are aggregated to provide an overall performance estimate.

However, the temporal order of financial data necessitates careful handling. Shuffling or random data partitioning, as commonly done in other domains, can lead to significant biases and erroneous conclusions. For instance, using future data in constructing the training set, even inadvertently, introduces lookahead bias, severely compromising the model's validity.

Moreover, financial markets are influenced by macroeconomic factors and market regimes, leading to structural breaks. These factors can result in model performance that varies significantly across different periods, making it difficult to generalize the results obtained from a conventional K-Fold cross-validation approach.

Despite these limitations, K-Fold cross-validation is often used in preliminary model assessments, given its simplicity and widespread understanding in the statistical community. However, researchers in quantitative finance must supplement or replace this method with more appropriate techniques, such as Combinatorial Purged Cross-Validation, that account for the peculiarities of financial time series data.

It is crucial to interpret the results of K-Fold cross-validation in the context of financial markets with caution, understanding that its assumptions may not fully align with the underlying data characteristics.

2.4.2. Time-Consistent Validation: Walk-Forward Cross-Validation

Walk-forward cross-validation (WFCV) is a method specifically tailored for time series data, addressing the unique

challenges posed by financial market data's temporal dependencies and non-stationarity. Unlike conventional K-Fold cross-validation, which can inadvertently introduce lookahead bias by shuffling data, WFCV respects the chronological order of observations, ensuring a more realistic and robust validation of trading strategies.

The WFCV process involves the following steps:

1. The dataset is divided into an initial training period and a subsequent testing period. The size of these periods can be fixed or expanded.
2. The model is trained on the initial training set and then tested on the subsequent testing period.
3. After the first validation, the training and testing windows are rolled forward. This means expanding or shifting the training period and testing on the new subsequent period.
4. This process is repeated until the entire dataset is traversed, with each iteration using a new testing period immediately following the training period.
5. Performance metrics are recorded for each testing period and aggregated to evaluate the strategy's overall effectiveness.

WFCV's primary advantage lies in its alignment with the practical scenarios encountered in live trading. Training and testing on consecutive data segments closely mimic the real-world situation where a model is trained on past data and deployed on future, unseen data. This sequential approach helps understand how a strategy adapts to evolving market conditions and objectively assesses its predictive power and robustness over time.

However, WFCV has its limitations. The repetitive retraining process can be computationally intensive, especially for large datasets and complex models. Additionally, the choice of the size of the training and testing windows can significantly impact the results, requiring careful consideration and sensitivity analysis.

WFCV is particularly pertinent in financial machine learning due to its ability to mitigate overfitting and model decay risks — common challenges in quantitative finance. It ensures that models are continuously updated and validated against the most recent data, reflecting the dynamic nature of financial markets.

Despite its advantages, WFCV should be employed as part of a comprehensive strategy validation framework, alongside other methods like combinatorial purged cross-validation, to fully account for the complexities of financial time series to ensure robust model validation.

2.4.3. Leakage-Resistant Validation: Purged K-Fold

Purged K-Fold Cross-Validation is an advanced validation technique developed by Lopez de Prado [2018] to address the issue of information leakage in financial time series, a common pitfall in traditional cross-validation methods. This method is particularly suited for validating financial models where the integrity of the temporal order of data is crucial for preventing look-ahead biases and ensuring realistic performance estimation.

The Purged K-Fold process involves several key modifications to the standard K-fold cross-validation:

1. The dataset is partitioned into k folds, ensuring that each fold is a contiguous segment of time to maintain the temporal order of observations.
2. Each fold is used once as the validation set, while the remaining folds form the training set. However, unlike standard K-Fold cross-validation, a "purging" process is implemented.
3. The purging process involves removing observations from the training set that occur after the start of the validation period. This is done to eliminate the risk of information leakage from the future (validation period) into the past (training period).
4. Additionally, an "embargo" period is applied after each training fold ends and before the next validation fold starts. This embargo period serves as a buffer zone to further mitigate the risk of leakage due to temporal dependencies that purging might not fully address.
5. The model is trained on the purged and embargoed training data and then validated on the untouched validation fold.
6. Performance metrics are recorded for each fold and aggregated to provide an overall assessment.

This methodology is particularly effective in financial machine learning, where models often capture temporal relationships, and even subtle information leakage can lead to over-optimistic performance estimates. Purged K-Fold Cross-Validation ensures a more robust and realistic evaluation of the model's predictive power by incorporating the purging and embargo mechanisms.

Purged K-Fold is especially relevant for strategies that rely on features extracted from historical data, as it ensures that the model is not inadvertently trained on future data. This method is essential for preventing the common pitfalls of overfitting and selection bias in financial modeling.

While Purged K-Fold Cross-Validation offers significant advantages in maintaining data integrity, it requires careful consideration of the lengths of the purge and embargo periods, which should be tailored to the specific temporal dependencies in the analyzed financial data.

2.4.4. Multi-Scenario, Leakage-Free Validation: Combinatorial Purged Cross-Validation

Combinatorial Purged Cross-Validation (CPCV) is introduced by Lopez de Prado [2018] as an innovative approach to address the limitations of single-path testing inherent in conventional Walk-Forward and Cross-Validation methods. This method is specifically designed for the complex environment of financial machine learning, where temporal dependencies and non-stationarity are prevalent. CPCV generates multiple backtesting paths and integrates a purging mechanism to eliminate the risk of information leakage from training observations.

The CPCV method is implemented as follows:

1. The dataset, consisting of T observations, is partitioned into N non-overlapping groups. These groups maintain the chronological order of data, where the first $N-1$ groups each have a size of $\lfloor T/N \rfloor$, and the N-th group contains the remaining observations.
2. For a selected size k of the testing set, CPCV calculates the number of possible training/testing splits as $\binom{N}{N-k}$. Each combination involves k groups for testing, and the total number of groups tested is $\binom{N}{N-k} \times k$, ensuring a uniform distribution across all N groups.
3. From the combinatorial splits, each group is uniformly included in the testing sets. This process results in a comprehensive series of backtest paths, given by the combinatorial number $\binom{N}{k}$.
4. Paths are generated by training classifiers on a portion of the data, specifically $1 - \frac{k}{N}$, for each combination. The algorithm ensures that the portion of data in the training set is balanced against the number of paths and size of the testing sets.
5. The CPCV backtesting algorithm involves purging and embargoing as introduced before. Each path results from combining forecasts from different groups and split combinations, ensuring a comprehensive evaluation of the classifier's performance.
6. After processing all paths, the performance metrics from each path are aggregated to assess the overall effectiveness of the model, providing insights into its robustness and consistency across various market conditions.

CPCV's unique combinatorial approach allows for a thorough evaluation of the model under diverse scenarios, addressing the critical overfitting issue. It provides a more nuanced and accurate assessment of a model's predictive capabilities in the dynamic field of financial markets.

While CPCV offers an extensive validation framework, its combinatorial nature can be computationally demanding. Therefore, it's essential to consider computational resources and execution time, particularly for large financial datasets.

2.4.5. Scenario Creation: Constructing Backtest Pathways

The creation of backtest pathways varies significantly among different cross-validation methods. Traditional Cross-Validation (CV), Walk-Forward (WF) Validation, and Combinatorial Purged Cross-Validation (CPCV) each have distinct methodologies for generating these paths. Understanding these differences is crucial for selecting the appropriate validation method in financial modeling.

1. **Traditional Cross-Validation (CV)**:
   (a) In traditional CV, the dataset is divided into k folds. Each fold is a validation set once, while the remaining folds constitute the training set.
   (b) The backtest path in CV is linear and sequential. Each fold's validation results contribute to a single aggregated performance metric.
   (c) This method does not account for the temporal order of data, which can lead to unrealistic backtest paths in financial time series due to potential information leakage and autocorrelation.
2. **Walk-Forward (WF) Validation**:
   (a) WF Validation involves an expanding and rolling window approach. The dataset is sequentially divided into a training set followed by a validation set.
   (b) The unique aspect of WF is its chronological alignment. The window rolls forward, ensuring the validation set always follows the training set in time.
   (c) WF creates a single backtest path that closely mimics real-world trading scenarios. However, it tests the strategy only once, providing limited insight into its robustness under different market conditions.
3. **Combinatorial Purged Cross-Validation (CPCV)**:
   (a) CPCV enhances backtest pathways by introducing a combinatorial approach. The dataset is divided into N groups, from which k groups are selected in various combinations for training and testing.
   (b) This method generates multiple backtest paths, each representing a different combination of training and validation sets. It addresses the issue of single-path dependency seen in WF and traditional CV.
   (c) CPCV also incorporates purging and embargoing to prevent information leakage, making each path more realistic and reducing the risk of overfitting.
   (d) The key advantage of CPCV is its ability to provide a comprehensive view of the strategy's performance across a range of scenarios, unlike the single scenario tested in WF and traditional CV.

Each CV method's approach to constructing backtest pathways has implications for its utility in financial modeling. Traditional CV's disregard for temporal order limits its applicability for financial time series. WF's single-path approach offers a realistic scenario but lacks robustness testing. CPCV, with its multiple, purged combinatorial paths, offers a comprehensive evaluation of a strategy's performance, making it particularly suitable for complex financial markets where multiple scenarios are critical for understanding a strategy's effectiveness.

2.5. Assessment of Backtest Overfitting

In the quest to develop robust trading strategies within quantitative finance, the assessment of backtest overfitting emerges as a crucial facet. This section delves into the methodologies deployed to evaluate and mitigate the risk of overfitting, a common pitfall where strategies appear effective in retrospective analyses but falter in prospective applications. Two pivotal concepts, the Probability of Backtest Overfitting (PBO) and the Deflated Sharpe Ratio (DSR), are harnessed to scrutinize the reliability of backtested strategies.

PBO is gauged through Combinatorially Symmetric Cross-Validation (CSCV), a technique that rigorously tests strategy performance across diverse market scenarios. Concurrently, DSR offers a refined perspective on strategy efficacy by adjusting the Probabilistic Sharpe Ratio (PSR) for multiple trials, thus enhancing the authenticity of our backtesting results. Together, these methodologies furnish a comprehensive framework for evaluating the integrity of trading strategies, ensuring that they are not merely artifacts of historical data but are genuinely predictive and robust against future market conditions.

2.5.1. *Probability of Backtest Overfitting: Combinatorially Symmetric Cross-Validation*

Backtest trials are pivotal in the realm of quantitative finance, particularly in the development of trading strategies. Utilizing the methodology outlined in previous sections, we perform multiple backtest trials, ideally selecting the optimal strategy based on its performance in these trials. However, this approach inherently risks backtest overfitting, where a strategy might show exceptional performance in a historical context but fails to generalize to new, unseen data. To quantitatively assess and mitigate this risk, we calculate the Probability of Backtest Overfitting (PBO) using the Combinatorially Symmetric Cross-Validation (CSCV) method as introduced by Bailey et al. [2016]. CSCV provides a more robust measure of a strategy's effectiveness by examining its performance across different segments of market data, allowing us to evaluate the consistency of trial returns both in-sample and out-of-sample.

The CSCV process is outlined in the following steps:

1. Formation of a performance matrix M of size $T \times N$, where each column represents the log returns series for a specific model configuration over T time observations.
2. Partitioning of M into S disjoint submatrices M_s of equal dimensions, with each submatrix being of order $\frac{T}{S} \times N$.
3. Formation of combinations C_S of these submatrices, taken in groups of size $\frac{S}{2}$, yielding a total number of combinations calculated as:

$$\binom{S}{S/2} = \prod_{i=0}^{S/2-1} \frac{S-i}{S/2-i}. \tag{2.26}$$

4. For each combination $c \in C_S$, the following steps are carried out:
   (a) Formation of the training set J and the testing set $\bar{J}$.
   (b) Computation of the performance statistic vectors R and $\bar{R}$ for the training and testing sets, respectively.
   (c) Identification of the optimal model n^* in the training set and determination of its relative rank $\bar{\omega}_c$ in the testing set.
   (d) Definition of the logit $\lambda_c = \log\left(\frac{\bar{\omega}_c}{1-\bar{\omega}_c}\right)$.
5. Finally, the PBO is estimated by calculating the distribution of ranks out-of-sample (OOS) and integrating the probability distribution function $f(\lambda)$ as:

$$\text{PBO} = \int_{-\infty}^{0} f(\lambda) d\lambda. \tag{2.27}$$

   where the PBO represents the probability of in-sample optimal strategies underperforming out-of-sample.

This rigorous statistical approach leading to Eqn. (2.27) allows us to evaluate the extent of overfitting in our strategy development process, ensuring that selected strategies are robust and not merely tailored to historical market idiosyncrasies.

2.5.2. *Probability of False Discovery: The Deflated Sharpe Ratio*

In selecting the optimal strategy from multiple backtest trials, a key concern is the probability of false discovery, which refers to the likelihood that the observed performance of a strategy is due to chance rather than true predictive power. To address this, we use the Deflated Sharpe Ratio (DSR), which extends the Probabilistic Sharpe Ratio (PSR) concept to account for the multiplicity of trials.

The PSR, as introduced by Bailey and Lopez de Prado [2012], adjusts the observed Sharpe Ratio ($\widehat{SR}$) by accounting for the distributional properties of returns, such as skewness and kurtosis. It is calculated as:

$$\widehat{PSR}(SR^*) = Z\left[\frac{(\widehat{SR} - SR^*)\sqrt{T-1}}{\sqrt{1 - \hat{\gamma}_3 \widehat{SR} + \frac{\hat{\gamma}_4 - 1}{4}\widehat{SR}^2}}\right], \tag{2.28}$$

where $Z[.]$ is the cumulative distribution function (CDF) of the standard Normal distribution, T is the number of observed returns, $\hat{\gamma}_3$ is the skewness of the returns, and $\hat{\gamma}_4$ is the kurtosis of the returns. SR^* is a benchmark Sharpe ratio against which $\widehat{SR}$ is compared.

The Deflated Sharpe Ratio (DSR), as introduced by Bailey and López de Prado [2014b], refines the Probabilistic Sharpe Ratio (PSR) as given in Eqn. (2.28) by considering the number of independent trials. This refinement yields a more precise measure of the probability of false discovery when multiple strategies are tested. Specifically, the DSR employs a benchmark Sharpe ratio (SR^*) which is calculated in Eqn. (2.29), that is influenced by the variance of the estimated Sharpe Ratios ($\widehat{SR}_n$) from the trials, the number of trials (N), and incorporates the Euler-Mascheroni constant (γ):

$$SR^* = \sqrt{V\left(\{\widehat{SR}_n\}\right)} \left((1-\gamma)Z^{-1}\left(1 - \frac{1}{N}\right) + \gamma Z^{-1}\left(1 - \frac{1}{N}e^{-1}\right)\right), \tag{2.29}$$

where Z^{-1} is the inverse of the cumulative distribution function (CDF) of the standard normal distribution Z. This adjustment is based on the expectation of the maximum of a sample of IID random variables from the standard normal distribution, which is delineated in Eqn. (2.30):

$$\mathrm{E}[\max\{x_i\}_{i=1,\dots,I}] \approx (1-\gamma)Z^{-1}\left[1-\frac{1}{I}\right] + \gamma Z^{-1}\left[1-\frac{1}{I}\,e^{-1}\right] \leq \sqrt{2\log[I]}, \quad (2.30)$$

with $\gamma \approx 0.57721566$ representing the Euler-Mascheroni constant, and $I \gg 1$ indicating a large number of trials. This formulation, known as the "False Strategy Theorem" Lopez de Prado [2020], informs the calculation of SR^* in the DSR methodology, providing a benchmark against which the observed Sharpe Ratios can be evaluated. The DSR, computed using this adjusted SR^* within the PSR framework, offers a comprehensive assessment of a strategy's true performance by correcting for the inflationary effect of multiple testing and helps distinguish genuine skill from statistical flukes.

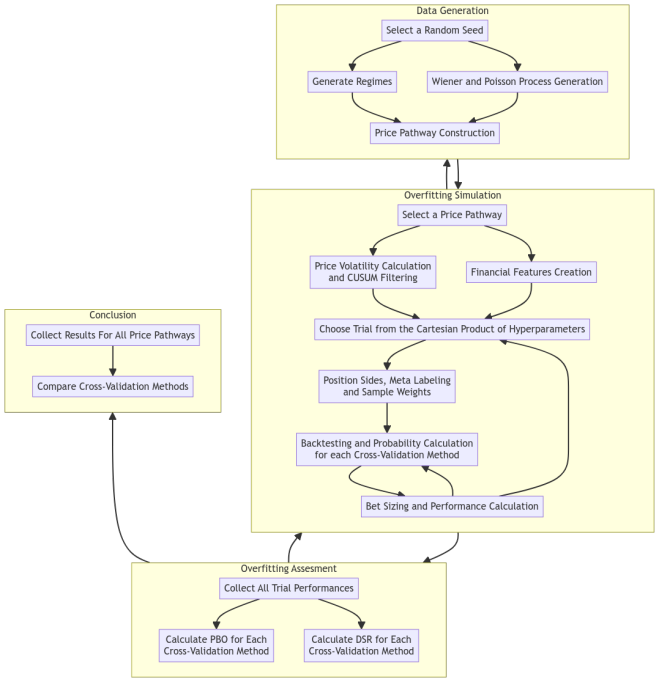

Figure 1: Flow Chart of the Empirical Results Simulation

3. Empirical Results

In this pivotal section of our study, we delve into a comprehensive empirical investigation designed to perform a comparative analysis of cross-validation techniques within a synthetic controlled environment. Our empirical endeavor is meticulously constructed to evaluate the robustness of these techniques against backtest overfitting—a critical pitfall in the development and validation of financial models. The exploration unfolds across a series of simulations replicating market conditions, informed by sophisticated models like the Heston Stochastic Volatility and Merton Jump Diffusion models, offering a rich tapestry of tranquil and turbulent market scenarios. The heart of our inquiry lies in the robustness check against backtest overfitting, ensuring the strategies we assess are not merely artifacts of hindsight bias but can stand the test of uncharted market dynamics. Through our Synthetic Controlled Market Environment lens, 28 strategic trials in 1000 simulations dissect the strengths and weaknesses of a spectrum of out-of-sample testing methodologies. Each technique's ability to identify and negate overfitting is rigorously examined, thereby serving as a crucible for determining the most reliable approach to cross-validation. The results presented here are a testament to the analytical rigor of the study but also form a cornerstone for the selection and implementation of cross-validation techniques in the ever-evolving domain of quantitative finance.

3.1. Implementation and Parameterization of Synthetic Data Models

This subsection delineates the development of a synthetic market environment, utilizing the Heston Stochastic Volatility Model and the Merton Jump Diffusion Model, with parameters reflecting market conditions during tranquil and tumultuous times. Parameters derived from empirical studies offer a faithful representation of historical market behavior, ensuring the robustness of our simulation. We detail the configuration of these models, their integration within a cohesive framework, and the parameter sets governing their behavior, which are crucial for capturing the complex dynamics of financial markets.

3.1.1. Base Model: The Heston Stochastic Volatility Model

The Heston Stochastic Volatility Model in our study is parameterized using "Calm" and "Volatile" market regimes, based on the empirical analysis of the S&P 500 during 2008 and 2011 by Papanicolaou and Sircar [2014] as shown in Table 1. These parameter sets are chosen for their robustness in replicating the real-world market volatility observed in these periods.

3.1.2. Price Jumps: The Merton Jump Diffusion Model

Incorporating the Merton Jump Diffusion Model into our simulation, we have calibrated parameters for "Calm" and "Volatile" market regimes based on insights from the study of the S&P 500 market by Hanson and Zhu [2004] as shown in Table 1. This parameterization is critical for accurately replicating the jump behavior in asset prices characteristic of varied market volatility conditions as observed in the empirical data.

3.1.3. Modeling Market Anomalies: The Drift Burst Hypothesis

Adopting the Drift Burst Hypothesis model for simulating market anomalies and speculative bubbles, our study

Table 1
Parameterization of the Heston and Merton Jump Diffusion Models for Calm and Volatile Market Regimes

Parameter	Calm Regime	Volatile Regime
Heston Stochastic Volatility		
Expected Return (μ)	0.1	0.1
Mean Reversion Rate (κ)	3.98	3.81
Long-term Variance (θ)	0.029	0.25056
Volatility of Variance (ξ)	0.389645311	0.59176974
Correlation Coefficient (ρ)	-0.7	-0.7
Merton Jump Diffusion		
Jump Intensity (λ)	121	121
Mean of Logarithmic Jump Size (m)	-0.000709	-0.000709
Variance of Logarithmic Jump Size (v)	0.0119	0.0119

aligns with the parameters delineated in the foundational work by Christensen et al. [2022] as shown in Table 2. This model uniquely influences the synthetic market environment by imposing a fixed-length regime characterized by predefined arrays of drift and volatility values. During this regime, the Heston Stochastic Volatility Model operates under these conditions, featuring non-stochastic volatility and the absence of jumps. After the drift burst period, the simulation mandates a transition to a different market regime, ensuring a realistic representation of abrupt market transitions. To ensure computational stability and circumvent potential zero division errors, the drift and volatility values are constant at a specific fraction of the entire duration, corresponding to the explosion filter width.

Table 2
Parameters for the Drift Burst Hypothesis Model

Parameter	Value
Bubble Length (T_{bubble})	5×252 days
Pre-Burst Drift Parameter (a_{before})	0.35
Post-Burst Drift Parameter (a_{after})	-0.35
Pre-Burst Volatility Parameter (b_{before})	0.458
Post-Burst Volatility Parameter (b_{after})	0.458
Drift Burst Intensity (α)	0.75
Volatility Burst Intensity (β)	0.225
Explosion Filter Width	0.1

3.1.4. Market Regime Dynamics: Markov Chain Transition Modeling

In our simulation, the transitions between market regimes are governed by a Markov Chain model, drawing insights from the works of Xie and Deng [2022] and Elliott et al. [2016] on regime-switching Heston models as shown in Table 3. The transition matrix, pivotal to the Markov chain model, is meticulously calibrated based on these references to represent regime shifts accurately, providing a realistic portrayal of market regime dynamics within our synthetic controlled environment.

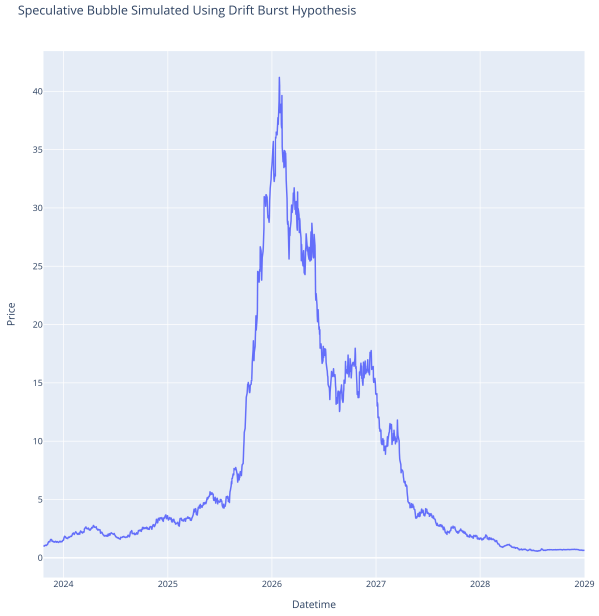

Figure 2: Speculative Bubble Simulated Using Drift Burst Hypothesis

Table 3
Markov Chain Transition Matrix for Market Regimes

From/To	Calm	Volatile	Speculative Bubble
Calm	$1 - \Delta t$	$\Delta t - 0.00001$	0.00001
Volatile	$20\Delta t$	$1 - 20\Delta t - 0.00001$	0.00001
Speculative Bubble	$1 - \Delta t$	Δt	0.0

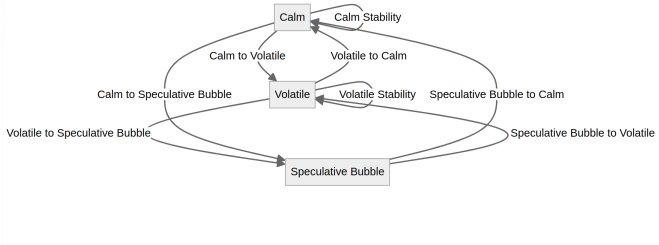

Figure 3: Regime Transition Diagram

3.1.5. Putting Them All Together: Synthetic Controlled Market Environment

In this section, we present the integration of a comprehensive synthetic market environment, utilizing a blend of the Heston Stochastic Volatility and Merton Jump Diffusion models. Our implementation leverages the Python programming language, `numpy` for numerical computations, the `@jit` decorator for performance optimization, and the QuantEcon library's `qe.MarkovChain` for Markov chain generation. Stochastic elements' reproducibility is ensured through `np.`

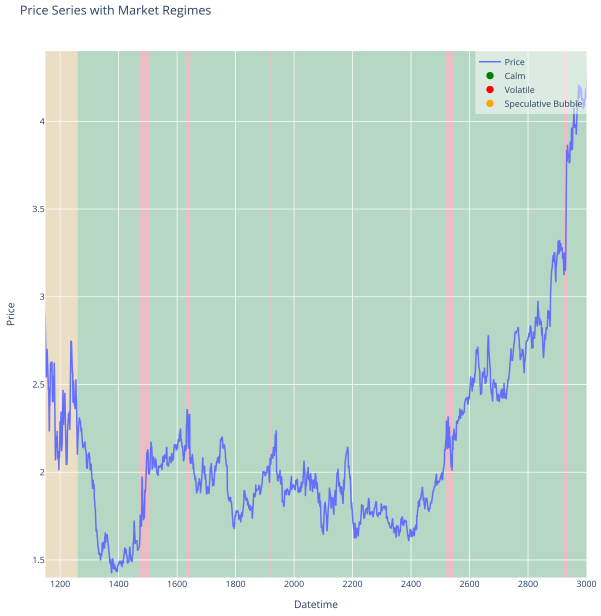

Figure 4: Price Series with Market Regimes

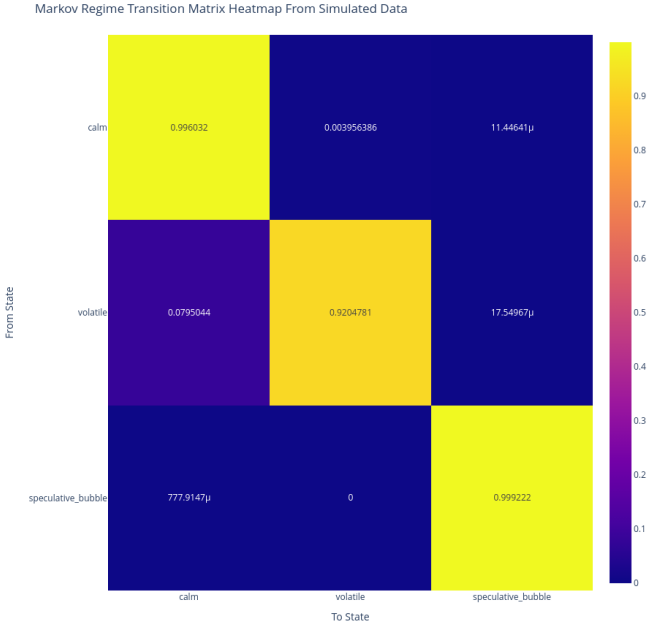

Figure 5: Markov Regime Transition Matrix Heatmap From Simulated Data

`random.default_rng()`.

1000 price paths are generated, each simulating 40 years of market data, equivalent to 40×252 business days. The time step for each simulation is $\Delta t = \frac{1}{252}$. Initially, 1000 unique random seeds are generated, which is the foundation for the price path simulations. The simulations adhere to the following equations, as detailed in Eqn. (2.9) and Eqn. (2.10):

$$\Delta S_t = \left(\mu - \frac{1}{2}v_t - \lambda\left(m + \frac{v^2}{2}\right)\right) S_t \Delta t + \sqrt{v_t} S_t Z \sqrt{\Delta t} + Y \Delta N(t),$$

$$\Delta v_t = \kappa(\theta - v_t)\Delta t + \xi\sqrt{v_t}(\rho_\epsilon \epsilon_t^P + \sqrt{1-\rho_\epsilon^2}\epsilon_t^v)\sqrt{\Delta t}.$$

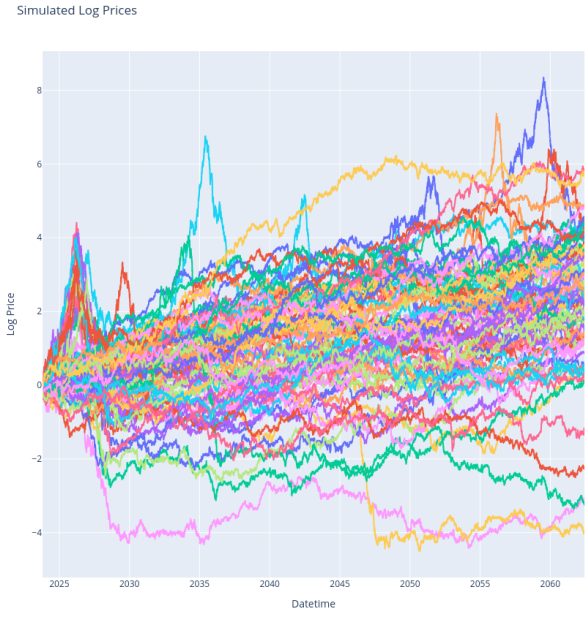

Figure 6: Simulated Log Prices

The synthesized log returns, encapsulating 1000 pathways over 40 years of market dynamics, are comprehensively summarized in Table 4, presenting the mean, standard deviation, skewness, and excess kurtosis of returns across different market regimes. Across all regimes, the returns exhibit a slight negative skewness and a notable excess kurtosis, suggesting a leptokurtic distribution more prone to extreme events than a normal distribution. The 'Calm' regime presents a relatively higher mean and lower volatility, indicating more stable market conditions. Conversely, the 'Volatile' and 'Bubble' regimes manifest heightened volatility and negative means, with the 'Bubble' regime showing the largest standard deviation and negative mean, characterizing periods of significant market stress and potential downturns.

The Q-Q plots of log returns in Figure 7 illustrate regime-specific distributions against a theoretical normal distribution. The 'All' category shows significant tail deviations, indicating outlier presence. The 'Calm' regime aligns more closely with normality except in the tails, hinting at occasional extremes. The 'Volatile' regime's plot diverges more noticeably in the tails, typical of unstable market periods. The 'Speculative Bubble' displays steep slopes and marked tail divergence, characteristic of the rapid price swings during speculative phases. These plots underscore the distinct

distributional features of each regime, from relative stability in 'Calm' conditions to the pronounced tail risks in 'Volatile' and 'Speculative Bubble' scenarios.

Table 4
Descriptive Statistics of Log Returns Overall and For Each Regime

Regimes	Mean	Std.	Skewness	Excess Kurtosis
All	0.000 230	0.018 216	−0.124 832	5.164 035
Calm	0.000 306	0.014 699	−0.096 815	3.487 078
Volatile	−0.000 131	0.033 051	−0.011 123	0.248 862
Bubble	−0.000 753	0.040 550	−0.054 046	1.120 036

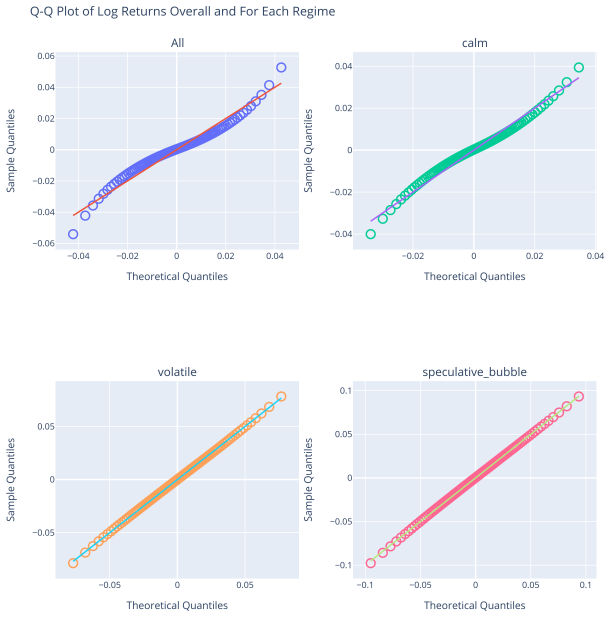

Figure 7: Q-Q Plot of Log Returns Overall and For Each Regime

3.2. Implementation of the Financial Machine Learning Strategy Components

Our comprehensive financial machine-learning strategy systematically integrates analytical components for robust and dynamic market analysis. This encompasses a meticulous assessment of market volatility, precise event-based sampling, strategic determination of trade directionality, application of advanced meta-labeling techniques, allocation of sample weights, and selection of financial features. Our methodology incorporates Fractional Differentiation to achieve stationarity without compromising memory, alongside the utilization of technical analysis indicators for enhanced market insight. The strategy intricately balances risk and opportunity through optimal bet sizing, grounded in probabilistic assessments from meta-labeling and the uniqueness of trade labels. Each component, from volatility assessment to bet sizing, is methodically implemented and parameterized, ensuring a harmonious integration that fortifies our model's predictive accuracy and adaptability to the nuanced dynamics of financial markets.

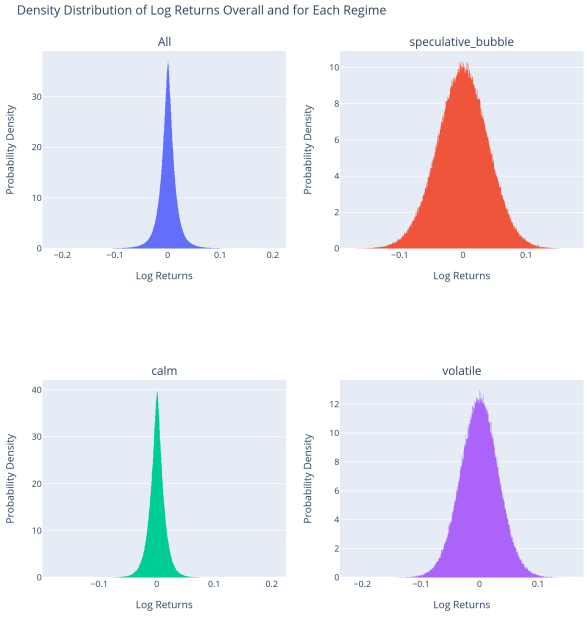

Figure 8: Density Distribution of Log Returns Overall and for Each Regime

3.2.1. *Volatility Assessment and Event-Based Sampling*

Our quantitative analysis adopts an Exponentially Weighted Moving Average (EWMA) approach to assess daily volatility, σ_t. Utilizing `pandas.Series.ewm` with a span of 100 days, we accurately capture the evolving market volatility. This calculated σ_t forms the basis for our dynamic threshold in the symmetric Cumulative Sum (CUSUM) filter Lopez de Prado [2018] applied to log prices. Specifically, we set the threshold at $1.8\sigma_t$, which is instrumental in resampling the data for identifying position opening days. This methodology ensures a data-driven, responsive sampling process, effectively aligning our trading strategy with prevailing market volatility and capturing significant price movements.

3.2.2. *Determining Trade Directionality*

In our trading strategy, trade directionality is determined using a momentum strategy based on simple moving averages, calculated via `pandas.Series.rolling`. We define a short-term moving average $\mathrm{MA}_{\text{short}}(y_t) = \frac{1}{N_{fast}} \sum_{i=0}^{N_{fast}-1} y_{t-i}$ and a long-term moving average $\mathrm{MA}_{\text{long}}(y_t) = \frac{1}{N_{slow}} \sum_{i=0}^{N_{slow}-1} y_{t-i}$, where N_{fast} and N_{slow} represent the window sizes for the respective averages. Trade positions are initiated based on the crossover of these averages post a CUSUM event: a long position when $\mathrm{MA}_{\text{short}}(y_t)$ exceeds $\mathrm{MA}_{\text{long}}(y_t)$, suggesting upward momentum, and a short position when $\mathrm{MA}_{\text{short}}(y_t)$

falls below $\mathrm{MA}_{\text{long}}(y_t)$, indicative of downward momentum. This approach aligns trading actions with the prevailing market trend, as reflected in the price momentum.

3.2.3. *Meta-Labeling Strategy*

Incorporating Lopez de Prado [2018]'s meta-labeling with the triple-barrier method, our strategy evaluates trades post-momentum-based direction determination. This process crucially informs position sizing decisions and enhances trade selection accuracy. The triple-barrier method applies two horizontal barriers for profit-taking and stop-loss, set with dynamic volatility-adjusted thresholds of $0.5\sigma_t$ and $1.5\sigma_t$ respectively, and a vertical barrier with a 20 working days expiration time. The outcome of a trade is determined as follows: hitting the upper (profit-taking) barrier results in a label of 1 for successful trades while reaching the lower (stop-loss) barrier first assigns a label of -1 for unsuccessful trades. If neither horizontal barrier is hit within the vertical time frame, the trade is evaluated based on the sign of the return at the end of this period.

3.2.4. *Sample Weight Allocation*

Our financial model calculates sample weights based on the uniqueness and magnitude of log returns within the meta-labeled data. For each label i, a concurrency count c_t and a binary indicator array $\{1_{t,i}\}$ are used to determine overlapping intervals. The preliminary weight $\tilde{w}_i$ is computed as $\left\| \sum_{t=t_{i,0}}^{t_{i,1}} \frac{r_{t-1,t}}{c_t} \right\|$. These weights are then normalized, giving $w_i = \frac{\tilde{w}_i}{\sum_{j=1}^{I} \tilde{w}_j}$, ensuring a balanced impact of each observation on the model. This approach emphasizes learning from distinct market events, thus refining the model's accuracy.

3.2.5. *Feature Selection for Financial Modeling*

We meticulously select features in our financial modeling process to ensure robust predictive capability. The Fractional Differentiation (FracDiff) feature is pivotal in this endeavor. We used the fixed-width window fractional differentiation approach to set the weight-loss threshold at 0.01. The differentiation order is incrementally determined using steps of size 0.1, with a p-value threshold of 0.05 for the ADF test implemented using the `statsmodels.tsa.stattools` module with a maximum lag of `maxlag=1`, balancing memory retention with the attainment of stationarity in the series. We leverage the `ta` Python library to construct Technical Analysis features, utilizing its default configurations to derive a spectrum of indicators. We apply specific thresholds for some indicators to enhance their interpretability:

1. **ADX Strength:** A threshold of 25 distinguishes between strong and weak trends.
2. **RSI Signal:** Thresholds of 30 and 70 identify overbought and oversold conditions.
3. **CCI Signal:** Thresholds of -100 and 100 signal potential market reversals.
4. **Stochastic Signal:** Thresholds of 20 and 80 indicate overbought and oversold conditions.

This methodical approach to feature selection, blending fractional differentiation with technical analysis, enables our model to capture intricate market dynamics effectively. The average Pearson correlation between the 22 features extracted from each of the 1000 generated price pathways is demonstrated in Figure 9.

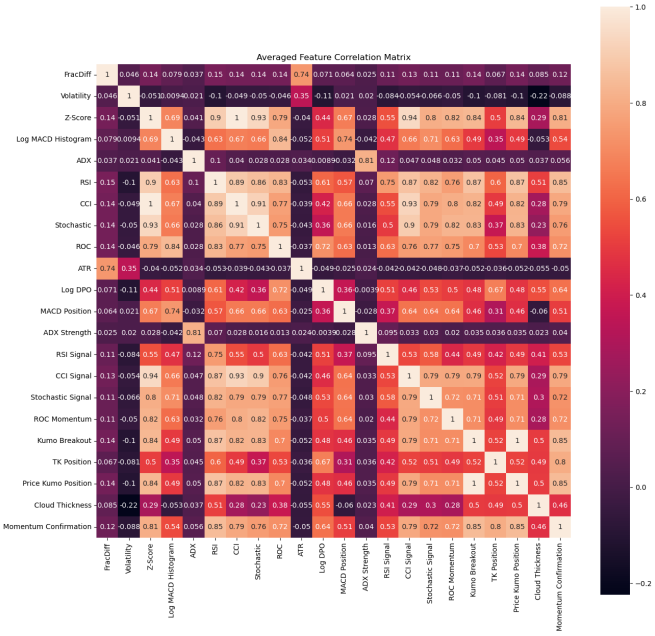

Figure 9: Average Feature Correlation Matrix

3.2.6. *Optimal Bet Sizing*

In our model, bet sizing is calibrated using probabilities from the meta-labeling strategy and the uniqueness of each label. For each label x, we test the hypothesis $H_0 : p[x = 1] = \frac{1}{2}$ using the statistic $z = \frac{p[x=1]-\frac{1}{2}}{\sqrt{p[x=1](1-p[x=1])}}$. The bet size m is determined as $m = 2Z[z] - 1$, where $Z[\cdot]$ is the cumulative distribution function of the standard normal distribution. To mitigate look-ahead bias, we shift the bet size time series by one day, then calculate the daily strategy return by multiplying each bet size with the corresponding daily return and position side. The bet size is then readjusted for the next trading day based on the forthcoming meta-label or liquidated as needed. Aggregate bet sizing at any timestamp t is computed as $m_t = \frac{\sum_{i=1}^{I} m_i 1_{t,i}}{\sum_{i=1}^{I} 1_{t,i}}$, averaging all active bets at that time. This approach ensures dynamic adaptation of bet sizes to the evolving market conditions, aligning with the probabilities and uniqueness of trade signals.

3.3. Design and Parameter Dynamics of Trial Simulations

In our empirical exploration, we methodically design 28 strategy trials by manipulating two core components: the parameters of the momentum cross-over strategy and the configurations of various machine learning models. We create

a comprehensive array of trials by alternating between different sets of rolling window sizes in the momentum strategy and a diverse range of hyperparameters in the machine learning models. This approach allows us to assess the performance impact of these variables under varying market conditions and model specifications. Each trial represents a unique combination of these configurations, providing us with a broad spectrum of insights into the dynamics of our financial strategy.

3.3.1. Momentum Cross-Over Strategy Variations

Our financial model evaluates momentum cross-over strategy variations by altering the moving averages' rolling window sizes. We test four distinct configurations: (5, 10), (20, 50), (50, 100), and (70, 140), representing various pairs of fast and slow-moving average window sizes. These trials systematically examine the strategy's performance under diverse temporal dynamics.

3.3.2. Machine Learning Models Variations

Our strategy explores various machine learning models to predict meta-labels, each with specific parameter configurations. Certain hyperparameters are explored with a single candidate value, while others are tested across multiple values, ensuring all cases are comprehensively utilized in our strategy trials. The configurations are strategically chosen to heighten the potential for overfitting. We have

1. **k-Nearest Neighbors (k-NN):** Implemented via `sklearn.neighbors.KNeighborsClassifier`, with neighbors parameter varied as `n_neighbors : [1, 2, 3]`. The data is standardized using `sklearn.preprocessing.StandardScaler` within a custom pipeline extending `sklearn.pipeline.Pipeline`, which incorporates sample weights.
2. **Decision Tree:** Utilized through `sklearn.tree.DecisionTreeClassifier`, with parameters set to `min_samples_split : [2]` and `min_samples_leaf : [1]`.
3. **XGBoost:** Executed using `xgboost.XGBClassifier`, with parameters including `n_estimators : [1000]`, `max_depth : [1000000000]`, `learning_rate : [1, 10, 100]`, `subsample : [1.0]`, and `colsample_bytree : [1.0]`.

3.4. Out-of-Sample Testing via Cross-Validation

In our quantitative finance framework, we apply a comprehensive suite of cross-validation techniques to conduct out-of-sample testing, employing the robust `CrossValidatorController` for initializing different validation methods. This includes K-Fold, Walk-Forward, Purged K-Fold, and Combinatorial Purged Cross-Validation, each specifically adapted to the challenges of financial time series data. Utilizing `CrossValidator.backtest_predictions`, we generate backtest paths for each cross-validation method, comprising probabilities corresponding to the meta-labels. For labels encountered across multiple backtest paths, we average their probabilities, creating a consolidated measure that informs subsequent strategy performance calculations. Integrating traditional and innovative cross-validation methodologies, this meticulous approach ensures robustness and accuracy in comparing our out-of-sample testing procedures.

3.4.1. Implementation of K-Fold Cross-Validation

Our implementation of K-Fold Cross-Validation (KFold) in financial modeling utilizes the `KFold` class within the `CrossValidatorController` framework. Configured with `n_splits=4`, this approach partitions the dataset into four distinct segments, adhering to the conventional methodology of KFold. Each segment sequentially serves as a test set, while the remaining data forms the training set. This structure is pivotal in our financial time series analysis, where it is crucial to avoid look-ahead bias and maintain the chronological integrity of data.

```
CrossValidatorController(
    'kfold',
    n_splits=4,
).cross_validator
```

Given the nature of financial data, characterized by temporal dependencies, our KFold implementation is tailored to respect these sequences, ensuring more accurate and realistic model validation. This adherence to the time series structure in our KFold setup underscores our commitment to rigorous, temporally-aware analytical practices in financial modeling.

3.4.2. Implementation of Walk-Forward Cross-Validation

Implemented using the `WalkForward` class, our Walk-Forward Cross-Validation (WFCV) employs the `CrossValidatorController` with `n_splits=4`, indicating a division of the dataset into four sequential segments. This ensures chronological training and testing phases, which is crucial for maintaining temporal integrity in financial data analysis. This approach, emphasizing the sequence and structure of data, mirrors real-world financial market dynamics and is key to achieving a realistic assessment of model performance. The specific parameterization of WFCV underscores our commitment to temporal consistency and robust validation in financial modeling.

```
CrossValidatorController(
    'walkforward',
    n_splits=4,
).cross_validator,
```

By aligning model evaluation with the chronological progression of market data, this configuration enhances the reliability and relevance of our strategy assessments.

3.4.3. Implementation of Purged K-Fold Cross-Validation

The implementation of Purged K-Fold Cross-Validation in our framework leverages the `PurgedKFold` class through the `CrossValidatorController`, specifically tailored for financial time series data. Configured with `n_splits=4`, an embargo rate of `embargo=0.02`, and time-based partitioning, this approach rigorously maintains the integrity of the temporal order. The initialization parameters ensure that the dataset is

divided into four contiguous segments, each representing a distinct period in time.

```
CrossValidatorController(
    'purgedkfold',
    n_splits=4,
    times=times,
    embargo=0.02
).cross_validator
```

This structure is instrumental for mitigating information leakage and lookahead biases by purging training data that overlaps with the validation period and implementing an embargo period. Such modifications are crucial in financial modeling, where the chronological sequence of data plays a pivotal role in the validity and realism of backtesting results. Our Purged K-Fold setup, therefore, ensures a more authentic and reliable assessment of the model's predictive capabilities.

3.4.4. Implementation of Combinatorial Purged Cross-Validation

Our implementation of Combinatorial Purged Cross-Validatic (CPCV) is realized using the `CombinatorialPurged` class, orchestrated through the `CrossValidatorController`. Tailored for financial time series analysis, CPCV is initialized with `n_splits=8` and `n_test_groups=2`, signifying that the dataset is divided into eight non-overlapping groups with two groups designated for testing in each combinatorial split. Additionally, an embargo rate of `embargo=0.02` is applied to mitigate information leakage further. This setup is encapsulated in the following configuration:

```
CrossValidatorController(
    'combinatorialpurged',
    n_splits=8,
    n_test_groups=2,
    times=times,
    embargo=0.02
).cross_validator,
```

The CPCV approach, with its combinatorial nature, ensures a thorough and diversified examination of the model's performance across multiple backtest paths, effectively addressing overfitting concerns prevalent in financial modeling. The integration of purging and embargoing within this framework further bolsters the temporal integrity of the validation process, making CPCV a robust tool for assessing predictive models in the dynamic environment of financial markets. The selection of parameters in our implementation reflects a deliberate balance between comprehensive backtesting and computational feasibility.

3.5. Comparative Assessment of Out-of-Sample Testing Techniques

In our study, we conduct a detailed comparative assessment of various out-of-sample testing methodologies, focusing on reducing the likelihood of backtest overfitting within our collection of 28 strategy trials. The analysis is methodically structured to encompass a holistic evaluation of the entire performance timeline and an annualized, segmented examination. Each year is meticulously analyzed, considering 252 trading days per segment. This dual-faceted analysis offers insights into the strategies' overall and specific yearly performances and serves as a litmus test for the effectiveness of different out-of-sample testing techniques in curbing overfitting. The scatter plot in Figure 10 illustrates a negligible correlation of -0.03 between the Probability of Backtest Overfitting (PBO) and the Best Trial Deflated Sharpe Ratio (DSR) Test Statistic in the overall analysis, signaling their independence as evaluative tools. Their independence is instrumental, as it implies a multi-faceted assessment of backtest validity, combining robustness checks against overfitting with adjustments for multiple hypothesis testing, thereby enriching the strategy selection process with diverse yet complementary reliability metrics.

Figure 10: Probability of Backtest Overfitting vs Best Trial Deflated Sharpe Ratio Test Statistic with the Correlation of -0.03

3.5.1. Implementation of Combinatorially Symmetric Cross-Validation (CSCV)

For assessing the Probability of Backtest Overfitting (PBO) in our 28 strategy trials, we implement the Combinatorially Symmetric Cross-Validation (CSCV) using Python's `numpy` library. The CSCV method is applied to a matrix of strategy returns, which represents the log returns for different model configurations across various time observations. We utilize `n_partitions = 16` to divide the performance matrix into an equal number of disjoint submatrices, ensuring a balanced evaluation across multiple data segments. Our evaluation metric, the Sharpe ratio, is computed through a custom

function to measure the performance of strategies over an annual risk-free rate of 0.05. The `probability_of_backtest_overfitting` function synthesizes this data, estimating the PBO and producing an array of logit values. This procedure thoroughly compares each strategy's in-sample and out-of-sample performance, which is crucial for identifying overfitting and ensuring the robustness of our trading strategies against future market scenarios.

3.5.2. *Utilization of the Deflated Sharpe Ratio in False Discovery Analysis*

In our analysis of 28 trading strategy trials, we employ the Deflated Sharpe Ratio (DSR) to critically assess the likelihood of false discoveries. This evaluation begins with computing the Sharpe Ratios for each strategy, using an annual risk-free rate of 0.05, to identify the best-performing trial. Subsequently, we calculate the DSR, considering the skewness, kurtosis of log returns, and the variance of Sharpe Ratios across all trials. Crucially, we focus on the test statistic derived from the DSR calculation rather than its value post-application in the normal cumulative distribution function (CDF). This approach allows us to more accurately discern between strategies that exhibit predictive skill and those that may have performed well by chance, ensuring a more robust and reliable selection of the optimal trading strategy.

3.6. Analytical Approaches for Out-of-Sample Testing Results Evaluation

A meticulous statistical examination of out-of-sample testing results is vital for validating the robustness of cross-validation methods against overfitting. In this subsection, we implement a comprehensive suite of non-parametric statistical tests, augmented by multivariate analysis, to rigorously compare the distributional characteristics of metric values derived from different cross-validation techniques. Utilizing Python libraries `scipy.stats`, `scikit_posthocs`, and `sklearn`, we perform the Kruskal-Wallis H Test for a global understanding of distributional differences, followed by pairwise comparisons via Dunn's Test for specific methodological distinctions. Furthermore, we conduct Principal Component Analysis (PCA) to assess the independence of simulation outputs, which provides a deeper insight into the interdependencies of the backtest overfitting metrics across our simulation trials. Together, these analytical strategies ensure a robust evaluation of the cross-validation methods' stability, reliability, and independence, painting a comprehensive picture of their performance in financial strategy validation.

3.6.1. *Assessing Distributional Variance Across Methods*

The Kruskal-Wallis H Test, a non-parametric method for determining stochastic dominance among multiple groups, evaluates the null hypothesis that the distributions of metric values across different cross-validation methods are identical. Unlike parametric counterparts, it does not necessitate the assumption of normally distributed data. Significant results from this test suggest at least one group's distribution differs from others. Should the test yield significance, we

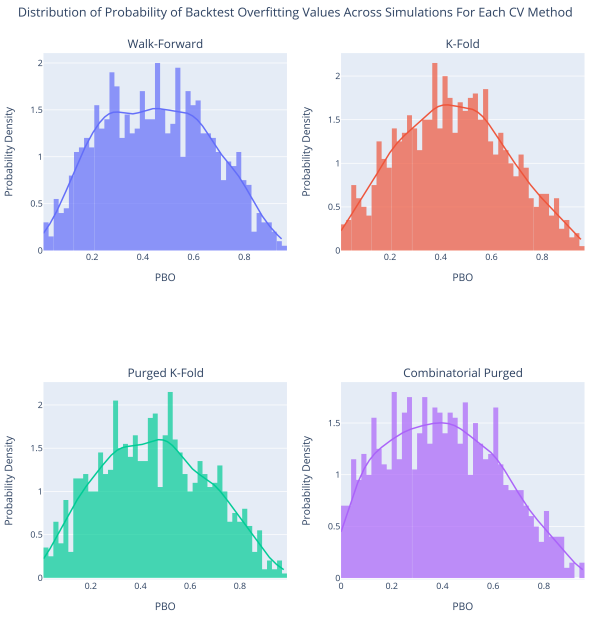

Figure 11: Distribution of Probability of Backtest Overfitting Values Across Simulations For Each Cross-Validation Method

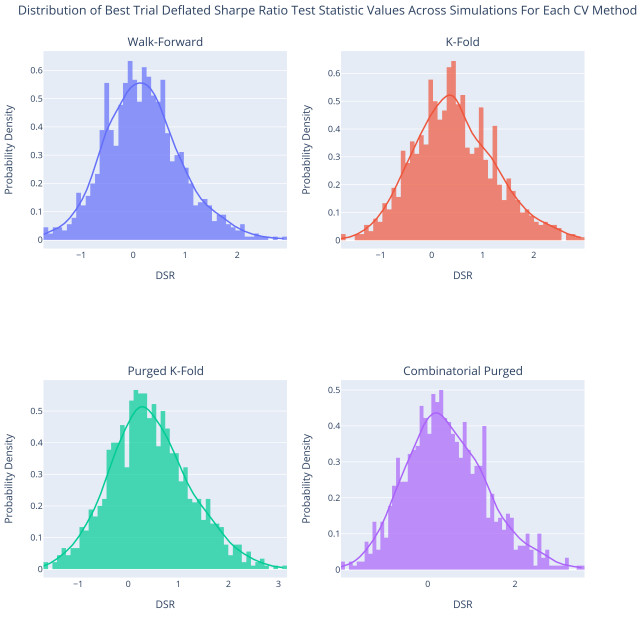

Figure 12: Distribution of Best Trial Deflated Sharpe Ratio Test Statistic Values Across Simulations For Each Cross-Validation Method

calculate the effect size using $\eta^2 = \frac{H-(k-1)}{N-k}$, where H is the Kruskal-Wallis statistic, k represents the number of groups, and N is the total number of observations. This statistic delineates the proportion of total variance in the metric values the cross-validation method explains, providing insights be-

yond statistical significance to the magnitude of differences observed.

3.6.2. *Delineating Distinct Distributions via Pairwise Comparisons*

We proceed with Dunn's Test for pairwise comparisons upon detecting significant variance in distributions with the Kruskal-Wallis H Test. This method pinpoints the specific cross-validation methods with statistically discernible differences. Dunn's Test is adept for multiple comparisons, applying the Bonferroni correction to adjust the significance threshold, thus controlling the family-wise error rate and reinforcing the validity of the inferential distinctions among pairs.

3.6.3. *Principal Component Analysis for Simulation Independence*

To evaluate the correlation and ascertain the independence of our 1000 Simulation, we conducted a Principal Component Analysis (PCA) on the annualized metric values, assessing backtest overfitting. By examining the Cumulative Explained Variance by PCA Components for each CV method, we could discern the degree of correlation among trials. A higher explained variance by fewer principal components indicates a stronger correlation and less independence between the trials, which is critical in understanding the diversification benefits of our strategy portfolio. Our PCA implementation utilized Python's `sklearn.pipeline.Pipeline`, incorporating a `sklearn.preprocessing.StandardScaler` to normalize the data, a simple average imputer `sklearn.impute.SimpleImputer` for handling missing values, and a `sklearn.decomposition.PCA` to perform the decomposition, thereby providing a quantitative assessment of the simulations' interdependencies.

3.7. Disclosure of Empirical Findings

In this subsection, we unveil the empirical findings derived from our meticulous analysis of backtest overfitting within various cross-validation frameworks. Our investigation meticulously scrutinizes the Probability of Backtest Overfitting (PBO) and the Best Trial Deflated Sharpe Ratio (DSR) Test Statistic, employing robust statistical methods to discern significant disparities and temporal variabilities across multiple validation techniques. The ensuing results illuminate the comparative resilience of these methods to the perils of overfitting and offer a nuanced understanding of their temporal behavior, thereby guiding the strategic selection of the most robust and stable cross-validation approaches in financial strategy development. This disclosure is anchored in a profound commitment to empirical rigor to bolster the integrity of model validation processes within quantitative finance.

3.7.1. *Overall Assessment of Backtest Overfitting*

Our comparative analysis of the Probability of Backtest Overfitting (PBO) across various cross-validation techniques revealed significant statistical differences, as visualized in Figure 13. The 'Walk-Forward' approach exhibited the highest median PBO value of 0.451437, suggesting a potentially higher risk of overfitting than other methods. The Kruskal-Wallis test indicated significant discrepancies among the groups ($p = 7.05 \times 10^{-9}$, $\eta^2 = 0.01022$), underscoring the presence of at least one method with a distinct PBO distribution. Dunn's pairwise comparison, as presented in Table 5, further corroborated significant distinctions: 'Combinatorial Purged' demonstrated a markedly lower PBO compared to both 'K-Fold' ($p = 4.20 \times 10^{-6}$) and 'Purged K-Fold' ($p = 3.32 \times 10^{-7}$), as well as against 'Walk-Forward' ($p = 1.09 \times 10^{-6}$), implying a superior efficacy in mitigating the risk of overfitting. Meanwhile, 'K-Fold' and 'Purged K-Fold' were statistically indistinguishable from 'Walk-Forward', suggesting similar PBO profiles between these methods. These insights are essential for strategically selecting cross-validation methodologies in quantitative finance models, aiming to reduce the probability of overfitting while ensuring robust predictive performance.

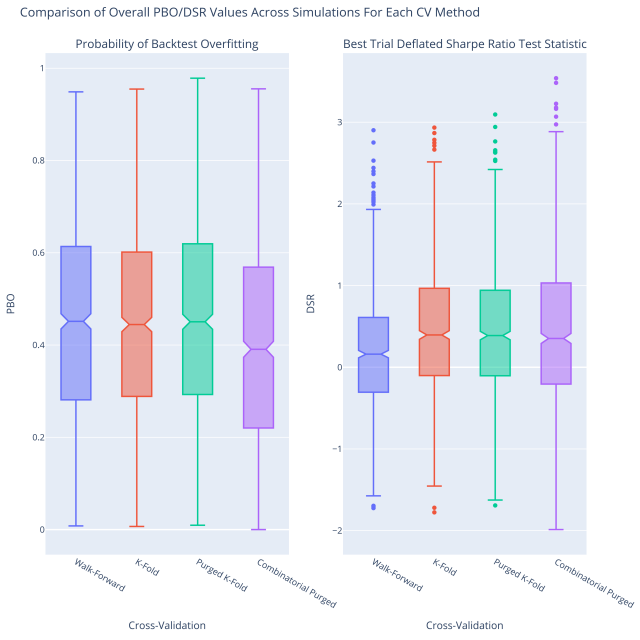

Figure 13: Comparison of Overall Probability of Backtest Overfitting and Best Trial Deflated Sharpe Ratio Test Statistic Values Across Simulations For Each Cross-Validation Method

Our simulations' non-parametric analysis of the Best Trial Deflated Sharpe Ratio (DSR) Test Statistic values revealed distinct statistical characteristics among the various cross-validation methods. As illustrated in Figure 13, the distribution of DSR values for the 'Walk-Forward' methodology markedly differed from those of other methods, with a notably lower median value of 0.160818. The Kruskal-Wallis test confirmed significant disparities across the distributions ($p = 5.0367 \times 10^{-15}$, $\eta^2 = 0.017$), suggesting at least one method deviates from the others in terms of DSR values. Subsequent pairwise comparisons using Dunn's Test, detailed in Table 6, identified 'Walk-Forward' as significantly differ-

Table 5
Distributions Comparison for Probability of Backtest Overfitting Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	7.05e-09	0.01022

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	4.20e-06	Yes
Combinatorial Purged vs. Purged K-Fold	3.32e-07	Yes
Combinatorial Purged vs. Walk-Forward	1.09e-06	Yes
K-Fold vs. Purged K-Fold	1.0	No
K-Fold vs. Walk-Forward	1.0	No
Purged K-Fold vs. Walk-Forward	1.0	No

ent from both 'K-Fold' and 'Purged K-Fold' ($p = 8.32 \times 10^{-12}$ and $p = 1.42 \times 10^{-11}$, respectively), while 'Combinatorial Purged' did not exhibit significant differences from 'K-Fold' and 'Purged K-Fold'. These statistical insights are crucial for recognizing each cross-validation method's relative effectiveness and characteristics in minimizing overfitting while striving for optimal Sharpe ratio performance.

Table 6
Distributions Comparison for Best Trial Deflated Sharpe Ratio Test Statistic Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	5.0367e-15	0.0174

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	1.0	No
Combinatorial Purged vs. Purged K-Fold	1.0	No
Combinatorial Purged vs. Walk-Forward	3.21-09	Yes
K-Fold vs. Purged K-Fold	1.0	No
K-Fold vs. Walk-Forward	8.32e-12	Yes
Purged K-Fold vs. Walk-Forward	1.42e-11	Yes

3.7.2. Temporal Variability of Overfitting Assessment

The annual Efficiency Ratio, defined as $\frac{\sigma^2}{\mu^2}$, was calculated for the Probability of Backtest Overfitting (PBO) across various cross-validation methods to assess the relative variability of PBO through time. As depicted in Figure 14, 'Walk-Forward' displayed the highest median Efficiency Ratio value of 0.224821, suggesting a higher variance to the mean PBO value than the other methods. The Kruskal-Wallis test revealed highly significant differences in the Efficiency Ratio distributions across methods ($p = 1.426\times10^{-76}$, $\eta^2 = 0.09$), indicating varying levels of PBO stability. Dunn's Test results, presented in Table 7, showed significant differences between 'Combinatorial Purged' and both 'K-Fold' ($p = 8.14\times10^{-23}$) and 'Purged K-Fold' ($p = 9.29\times10^{-22}$), as well as 'Walk-Forward' ($p = 1.06\times10^{-7}$). Additionally, 'K-Fold' and 'Walk-Forward' demonstrated a significant variance in their Efficiency Ratios ($p = 2.15 \times 10^{-54}$), as did 'Purged

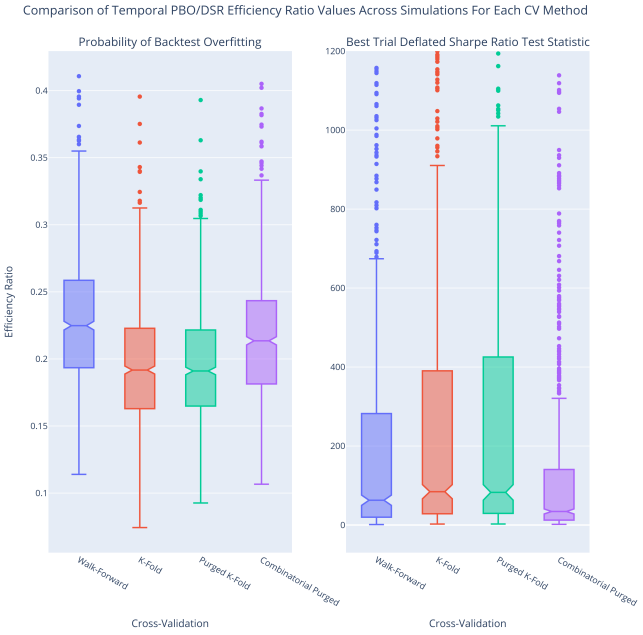

Figure 14: Comparison of Temporal Probability of Backtest Overfitting and Best Trial Deflated Sharpe Ratio Test Statistic Efficiency Ratio Values Across Simulations For Each Cross-Validation Method

K-Fold' and 'Walk-Forward' ($p = 9.60 \times 10^{-54}$). These findings underscore the importance of considering the Efficiency Ratio when evaluating the consistency of PBO over time, with 'Walk-Forward' showing the greatest variability and, thus, potentially, the least stability in PBO values year over year.

Table 7
Distributions Comparison for Probability of Backtest Overfitting Efficiency Ratio Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	1.426e-76	0.08876

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	8.14e-23	Yes
Combinatorial Purged vs. Purged K-Fold	9.29e-22	Yes
Combinatorial Purged vs. Walk-Forward	1.06e-07	Yes
K-Fold vs. Purged K-Fold	1.0	No
K-Fold vs. Walk-Forward	2.15e-54	Yes
Purged K-Fold vs. Walk-Forward	9.60e-54	Yes

The annual Efficiency Ratio $\frac{\sigma^2}{\mu^2}$ for the Best Trial Deflated Sharpe Ratio (DSR) Test Statistic values was scrutinized to evaluate the variability of the DSR through time for each cross-validation method. Figure 14 illustrates the distributions of these ratios, with 'Combinatorial Purged' showing a notably lower median Efficiency Ratio of 34.30, suggesting greater efficiency in DSR performance. In stark

contrast, 'K-Fold' and 'Purged K-Fold' showed higher median values of 84.36 and 82.59, respectively, indicating less efficiency. The Kruskal-Wallis test underscored significant differences in efficiency across methods ($p = 1.43 \times 10^{-76}$, $\eta^2 = 0.0888$). According to Dunn's Test results shown in Table 8, 'Combinatorial Purged' demonstrated statistically significant higher efficiency when compared to both 'K-Fold' ($p = 8.14 \times 10^{-23}$) and 'Purged K-Fold' ($p = 9.29 \times 10^{-22}$), as well as 'Walk-Forward' ($p = 1.06 \times 10^{-7}$). Conversely, 'K-Fold' and 'Walk-Forward' showed no significant difference in their efficiency ($p = 2.15 \times 10^{-54}$), similar to 'Purged K-Fold' versus 'Walk-Forward' ($p = 9.60 \times 10^{-54}$). These findings are instrumental for discerning the most efficient cross-validation method regarding DSR variability, which is crucial for achieving stable performance in financial machine-learning applications.

Table 8
Distributions Comparison for Best Trial Deflated Sharpe Ratio Test Statistic Efficiency Ratio Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	1.43e-76	0.0888

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	8.14e-23	Yes
Combinatorial Purged vs. Purged K-Fold	9.29e-22	Yes
Combinatorial Purged vs. Walk-Forward	1.06e-07	Yes
K-Fold vs. Purged K-Fold	1.00	No
K-Fold vs. Walk-Forward	2.15e-54	Yes
Purged K-Fold vs. Walk-Forward	9.60e-54	Yes

3.7.3. Temporal Stationarity of Overfitting Assessment

In our annual time series analysis of the Probability of Backtest Overfitting (PBO), the Augmented Dickey-Fuller (ADF) test statistic values were utilized to examine the stationarity of the PBO through time. These values are depicted in Figure 15 and quantitatively analyzed in Table 9. The 'Walk-Forward' method exhibited a markedly higher median ADF value of -2.41, indicating less stationarity and greater trend presence than other methods. The Kruskal-Wallis test yielded a significant result ($p < 0.01$, $\eta^2 = 0.55$), implying substantial differences in the time series characteristics among the methods. Dunn's Test revealed that the 'Walk-Forward' method's ADF values were significantly different from those of 'K-Fold', 'Purged K-Fold', and 'Combinatorial Purged' (all with $p = 0.0$), underscoring its distinct behavior in terms of stationarity. These findings suggest that while 'Walk-Forward' might be more prone to exhibit trends in PBO over time, the other methods did not show significant differences among themselves, indicating similar levels of stationarity in their respective PBO values.

The stationarity of the annual Best Trial Deflated Sharpe Ratio (DSR) Test Statistic values was assessed using the Augmented Dickey-Fuller (ADF) test, with the distributions visualized in Figure 15 and the statistical analysis detailed in

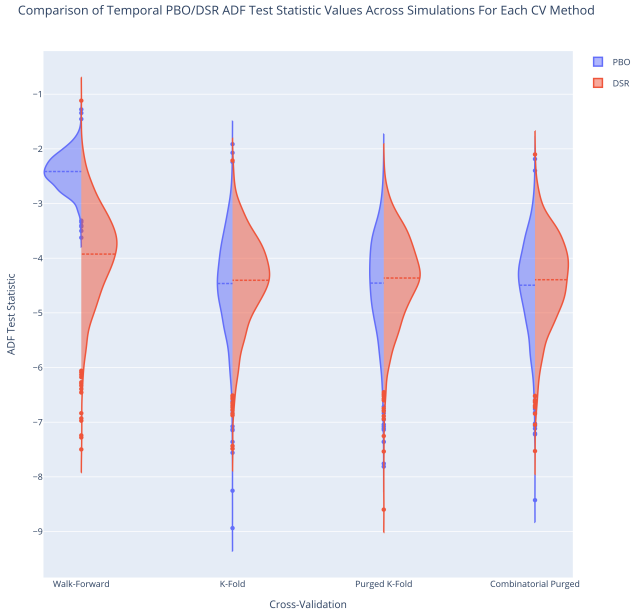

Figure 15: Comparison of Temporal Probability of Backtest Overfitting and Best Trial Deflated Sharpe Ratio Test Statistic ADF Test Statistic Values Across Simulations For Each Cross-Validation Method

Table 9
Distributions Comparison for Probability of Backtest Overfitting ADF Test Statistic Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	0.0	0.55106

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	1.0	No
Combinatorial Purged vs. Purged K-Fold	1.0	No
Combinatorial Purged vs. Walk-Forward	0.0	Yes
K-Fold vs. Purged K-Fold	1.0	No
K-Fold vs. Walk-Forward	0.0	Yes
Purged K-Fold vs. Walk-Forward	0.0	Yes

Table 10. The 'Walk-Forward' approach demonstrated a higher median ADF value of -3.86, suggesting a weaker presence of stationarity compared to the more negative ADF values of the other methods, which implies a stronger rejection of the unit root and thus a stronger indication of stationarity. The Kruskal-Wallis test provided extremely significant evidence of distributional differences among the methods ($p = 2.01 \times 10^{-50}$, $\eta^2 = 0.059$). Dunn's Test further identified significant differences between 'Walk-Forward' and all other methods, with 'Walk-Forward' being less stationary compared to 'K-Fold' ($p = 2.38 \times 10^{-37}$), 'Purged K-Fold' ($p = 1.65 \times 10^{-31}$), and 'Combinatorial Purged' ($p = 9.81 \times 10^{-36}$). These results indicate that 'Walk-Forward' may be less suitable for strategies that require a consistent DSR over time,

while the other cross-validation methods do not exhibit significant differences regarding stationarity in their DSR values.

Table 10
Distributions Comparison for Best Trial Deflated Sharpe Ratio Test Statistic ADF Test Statistic Values Across Simulations For Each Cross-Validation Method

Test	P-Value	Effect Size (η^2)
Kruskal Wallis	2.01e-50	0.05853

Dunn's Test	p-Value	Significant
Combinatorial Purged vs. K-Fold	1.0	No
Combinatorial Purged vs. Purged K-Fold	1.0	No
Combinatorial Purged vs. Walk-Forward	9.81e-36	Yes
K-Fold vs. Purged K-Fold	1.0	No
K-Fold vs. Walk-Forward	2.38e-37	Yes
Purged K-Fold vs. Walk-Forward	1.65e-31	Yes

3.7.4. Correlation of Overfitting Assessments Across Simulations

Our Principal Component Analysis (PCA) investigation into the correlation between different overfitting metrics across simulations revealed notable patterns of dependency. As depicted in the PCA cumulative explained variance plots (Figure 16), both the Probability of Backtest Overfitting (PBO) and the Best Trial Deflated Sharpe Ratio (DSR) Test Statistic values for the 'Walk-Forward' method are characterized by a higher explained variance with fewer principal components. This pattern indicates a lower level of simulation result independence, suggesting that the performance metrics of the 'Walk-Forward' method are more interrelated than those of other cross-validation methods. Such a correlation structure within the 'Walk-Forward' simulations may imply an inherent bias or systemic influence affecting the simulations, an essential consideration for strategy validation and the selection of robust cross-validation methodologies.

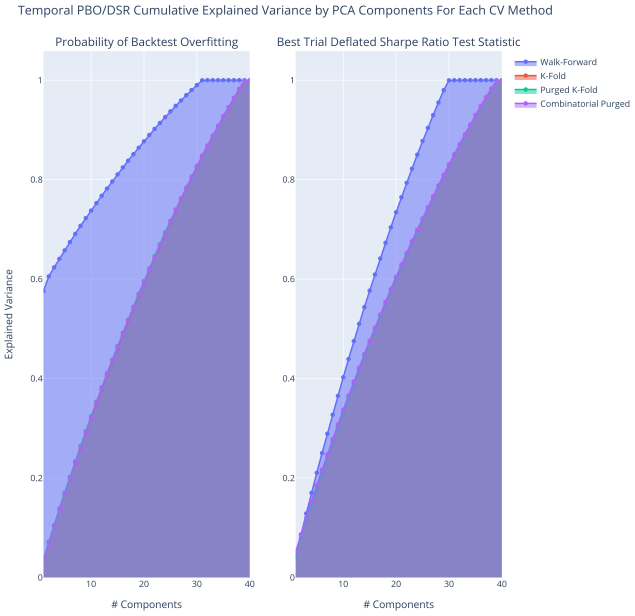

Figure 16: Temporal Probability of Backtest Overfitting and Best Trial Deflated Sharpe Ratio Test Statistic Cumulative Explained Variance by PCA Components For Each Cross-Validation Method

4. Discussion

In assessing backtest overfitting, we observed notable disparities across various cross-validation techniques. The 'Walk-Forward' approach exhibited the highest Probability of Backtest Overfitting (PBO), signaling a heightened risk of overfitting. In contrast, the 'Combinatorial Purged' method significantly outperformed others like 'K-Fold' and 'Purged K-Fold', demonstrating its effectiveness in reducing overfitting risks. The Deflated Sharpe Ratio (DSR) Test Statistic evaluation highlighted distinct performance variations among the methods. 'Walk-Forward' showed a markedly lower median DSR, suggesting heightened false discovery probability. In comparison, 'Combinatorial Purged' aligned closely with 'K-Fold' and 'Purged K-Fold', indicating a more balanced approach in achieving optimal performance while mitigating overfitting.

Our analysis of the Efficiency Ratio for the Probability of Backtest Overfitting (PBO) revealed 'Walk-Forward' as having the highest median value, indicating greater temporal variability and reduced stability. 'Combinatorial Purged', however, displayed a notably lower Efficiency Ratio, suggesting enhanced temporal stability and consistency in performance. When evaluating the Efficiency Ratio for the DSR Test Statistic, 'Combinatorial Purged' exhibited a notably lower median value, implying greater efficiency and stability in its DSR performance over time. This contrasted with 'K-Fold' and 'Purged K-Fold', which showed higher median values, indicating reduced efficiency and potential variability in DSR performance.

The temporal stationarity analysis of PBO, using the Augmented Dickey-Fuller (ADF) test, revealed that the 'Walk-Forward' method exhibited less stationarity, indicating a greater presence of trends in its PBO over time. Other methods, including 'Combinatorial Purged', displayed more consistent stationarity levels, suggesting more reliable performance. In assessing the stationarity of the DSR Test Statistic values, 'Walk-Forward' demonstrated weaker stationarity, as indicated by its higher median ADF value. This contrasted with other methods, which showed stronger indications of stationarity, implying a more stable and consistent rejection of unit root in their DSR values over time.

Our Principal Component Analysis (PCA) on the correlation between different overfitting metrics across simulations highlighted a unique pattern for the 'Walk-Forward' method, characterized by a higher explained variance with fewer principal components. This pattern suggests a lower level of result independence, indicating potential biases or

systemic influences in the 'Walk-Forward' method.

5. Conclusions

Our investigation into cross-validation methodologies in financial modeling has revealed critical insights, especially the superiority of the 'Combinatorial Purged' method in minimizing overfitting risks. This method outperforms traditional approaches like 'K-Fold', 'Purged K-Fold', and notably 'Walk-Forward' in terms of both the Probability of Backtest Overfitting (PBO) and the Deflated Sharpe Ratio (DSR) Test Statistic. 'Walk-Forward', in contrast, shows limitations in preventing false discovery and exhibits greater temporal variability and weaker stationarity from temporal assessment of these methodologies using the Efficiency Ratio and the Augmented Dickey-Fuller (ADF) test, raising concerns about its reliability. On the other hand, 'Combinatorial Purged' demonstrates enhanced stability and efficiency, proving to be a more reliable choice for financial strategy development. The choice between 'Purged K-Fold' and 'K-Fold' requires caution, as they show no significant performance difference, and 'Purged K-Fold' may reduce the robustness of training data for out-of-sample testing. These findings significantly contribute to quantitative finance, providing a robust framework for cross-validation that aligns theoretical robustness with practical reliability. They underscore the need for tailored evaluation methods in an era of complex algorithms and large datasets, guiding decision-making in a data-driven financial world. Future research should extend these findings to real-world market conditions to enhance their applicability and generalizability.

References

[1] David H Bailey and Marcos Lopez de Prado. The sharpe ratio efficient frontier. *Journal of Risk*, 15(2):13, 2012.

[2] David H Bailey and Marcos López de Prado. The deflated sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality. *Journal of Portfolio Management*, 40(5):94–107, 2014b.

[3] David H Bailey, Jonathan M Borwein, Marcos López de Prado, and Qiji Jim Zhu. Pseudomathematics and financial charlatanism: The effects of backtest over fitting on out-of-sample performance. *Notices of the AMS*, 61(5):458–471, 2014a.

[4] David H Bailey, Jonathan Borwein, Marcos Lopez de Prado, and Qiji Jim Zhu. The probability of backtest overfitting. *Journal of Computational Finance, forthcoming*, 2016.

[5] Carlo Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, 8:3–62, 1936.

[6] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[8] Kim Christensen, Roel Oomen, and Roberto Renò. The drift burst hypothesis. *Journal of Econometrics*, 227(2):461–497, 2022. ISSN 0304-4076.

[9] Olive Jean Dunn. Multiple comparisons using rank sums. *Technometrics*, 6(3):241–252, 1964.

[10] Robert J Elliott, Katsumasa Nishide, and Carlton-James U Osakwe. Heston-type stochastic volatility with a markov switching regime. *Journal of Futures Markets*, 36(9):902–919, 2016.

[11] Eugene F. Fama and Marshall E. Blume. Filter rules and stock-market trading. *The Journal of Business*, 39(1):226–241, 1966. ISSN 00219398, 15375374.

[12] Evelyn Fix and Joseph Lawson Hodges. Nonparametric discrimination: consistency properties. *Randolph Field, Texas, Project*, pages 21–49, 1951.

[13] James D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[14] Floyd B Hanson and Zongwu Zhu. Comparison of market parameters for jump-diffusion distributions using multinomial maximum likelihood estimation. In *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, volume 4, pages 3919–3924. IEEE, 2004.

[15] Steven L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6(2):327–343, 1993.

[16] Ulrich Homm and Jörg Breitung. Testing for speculative bubbles in stock markets: a comparison of alternative methods. *Journal of Financial Econometrics*, 10(1):198–231, 2012.

[17] Kin Lam and HC Yam. Cusum techniques for technical trading in financial markets. *Financial Engineering and the Japanese Markets*, 4:257–274, 1997.

[18] Marcos Lopez De Prado. The future of empirical finance. *Journal of Portfolio Management*, 41(4), 2015.

[19] Marcos Lopez de Prado. *Advances in financial machine learning*. John Wiley & Sons, 2018.

[20] Marcos Lopez de Prado. *Machine learning for asset managers*. Cambridge University Press, 2020.

[21] Robert C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1):125–144, 1976.

[22] Andrew Papanicolaou and Ronnie Sircar. A regime-switching heston model for vix and s&p 500 implied volatilities. *Quantitative Finance*, 14(10):1811–1827, 2014.

[23] Michael Schatz and Didier Sornette. Inefficient bubbles and efficient drawdowns in financial markets. *International Journal of Theoretical and Applied Finance*, 23(07):2050047, 2020.

[24] Laerd Statistics. Kruskal-wallis h test using spss statistics. *Statistical tutorials and software guides*, 2015.

[25] Yurong Xie and Guohe Deng. Vulnerable european option pricing in a markov regime-switching heston model with stochastic interest rate. *Chaos, Solitons & Fractals*, 156:111896, 2022.