Adaptive Markets Hypothesis and Conditional Ecology Performance: Evidence from Machine-Learning Market-Timing Strategies

Franklin Allen[1,2,3], Marcin Kacperczyk[1,2,3], and K S Sesh Kumar[1]

[1]Imperial Business School
[2]CEPR
[3]ECGI

February 25, 2026

Abstract

We evaluate traditional and modern machine-learning market-timing strategies from 1970 to 2024 under a common information set, portfolio rule, and realistic transaction costs. Unconditionally, none of the strategies delivers robust abnormal performance relative to buy-and-hold, consistent with the Efficient Markets Hypothesis (EMH). Conditioning on an ex-ante ecology vector capturing crowding, funding, liquidity, and volatility, performance, risk, and cross-strategy co-movement vary systematically, with different model families specializing in distinct habitats. These state-dependent patterns are difficult to reconcile with a purely risk-based EMH but align closely with the Adaptive Markets Hypothesis (AMH).

Keywords: Adaptive Markets Hypothesis, efficient markets, machine learning, non-stationarity, neural architectures, market-timing strategies

1 Introduction

Machine learning has transformed prediction problems in domains such as image recognition and natural language processing, yet its performance in financial markets remains puzzlingly fragile. A large body of work in computer science treats financial time series as another high-dimensional forecasting task and deploys ever more sophisticated architectures, including patch-based time-series transformers, non-stationary transformers, neural hidden Markov models, and meta-learning frameworks, to extract patterns from returns and related signals. At the same time, the asset pricing and empirical finance literatures have only cautiously adopted these tools, often relying on relatively basic recurrent networks and ensemble methods while emphasizing concerns about market efficiency, data-snooping, and backtest overfitting. The striking fact is that, despite this growth in methodological sophistication and computational power, credible evidence of robust, implementable excess returns from machine-learning trading strategies is scarce. The central objective of this paper is to understand why. We argue that modern machine learning struggles in financial markets not because the algorithms lack sophistication, but because they are solving the wrong problem.

Standard machine learning algorithms assume some form of stationarity or slowly drifting non-stationarity that can be tracked with enough data and flexible function classes. Financial markets, by contrast, are evolutionary environments in which profitable trading rules are transient ecological niches: once discovered and scaled, they reshape order flow, prices, and constraints, eroding the very patterns they exploit. In Andrew Lo's *Adaptive Markets Hypothesis* (AMH) (Lo, 2017; Lo and Zhang, 2024), markets are neither permanently efficient nor permanently inefficient; instead, populations of strategies, institutions, and constraints co-evolve, and the risk-return trade-off of any given rule is intrinsically state dependent. The Efficient Markets Hypothesis (EMH) appears as a limiting case of this framework, corresponding to environments in which learning has largely exhausted exploitable opportunities and constraints are stable.

An important implication of this evolutionary perspective is that trading strategies need not follow a one-way lifecycle in which profitability is permanently eliminated once a rule is discovered or disseminated. Instead, under AMH, the effectiveness of a given strategy is inherently episodic: competitive pressure, capital flows, and constraints may erode its profitability in some environments, while shifts in the market ecology can later restore conditions under which the same strategy again becomes viable. This stands in contrast to the canonical EMH view, in which publication or widespread adoption permanently arbitrages away any exploitable regularity.

We test this evolutionary perspective empirically by evaluating a large universe of modern machine learning-based portfolio strategies. Using data for the U.S. equity market from 1970 to 2024, we construct a panel of market-timing rules that allocate between the S&P 500 index and T-bills based on a common information set of lagged returns and publicly observable signals. The forecasting engines underlying these rules range from classical statistical and econometric models, through tree-based and regularized machine learning methods, to recurrent and attention-based neural architectures explicitly designed for non-stationary and regime-switching environments. Crucially, all strategies are implemented through a common mapping from forecasts to portfolio weights, share the same risk-management and cost assumptions, and are evaluated out of sample using realistic trading frictions. This design holds fixed the asset universe and implementation technology and attributes any performance differences to how models process information rather than to differences in leverage, asset coverage, or ad-hoc risk control.

Our analysis exploits this common implementation framework to distinguish between EMH and AMH explanations for ML performance. We proceed in two steps. We first conduct an unconditional evaluation of all strategies and compare them to a simple buy-and-hold allocation in the market index. We then embed these same strategies in an explicit AMH framework and study how their performance, risk, and co-movement vary with a vector of observable market ecology variables, Z_t. The ecology vector comprises measures of crowding, funding conditions, trading environment (liquidity, volatility composition, and trend), and implementation frictions (bid-ask spreads). Under a rich EMH benchmark that allows for time-varying betas, stochastic volatility, and a state-independent but realistic cost model, the ecology vector should have no incremental predictive content for risk-adjusted outcomes. Under AMH, by contrast, the conditional distribution of strategy returns and risks should move systematically with Z_t in ways that can be signed ex ante: crowding and tight funding compress net alpha and worsen tails, favorable trend and liquidity states support higher Sharpe ratios, and illiquid, reversal-prone regimes synchronize strategies and raise co-movement.

The unconditional results strongly favor the EMH benchmark. Across all three model families, average excess returns, alphas, and Sharpe ratios of the machine learning-based timing strategies are, if anything, slightly worse than those of the buy-and-hold benchmark once we account for costs and the substantial dispersion across individual methods. The buy-and-hold portfolio delivers an annualized excess return of about 12% and a Sharpe ratio of around 0.66, whereas the statistical

and econometric models cluster around 6–7% excess returns, classical machine learning methods around 4%, and advanced neural architectures around 5%. Risk-adjusted performance is similarly unimpressive: the cross-sectional distribution of alphas is centered close to zero, and no family delivers a robust, unconditional premium relative to the passive benchmark. Where the strategies do differ is in their risk: many ML-based rules achieve these modest outcomes with substantially lower volatility and tail risk than buy-and-hold, often by de-risking aggressively when their signals are weak. Unconditionally, therefore, the data favor an interpretation in which sophisticated models approximate optimal scaling of market exposure in an almost-efficient market, rather than uncovering a persistent anomaly.

The conditional analysis, however, reveals patterns invisible in these aggregate statistics. We operationalize the AMH by taking the realized path of Z_t as a sequence of observable market states and estimating panel regressions that relate a rich set of strategy-level outcomes—alphas, Sharpe ratios, variance, expected shortfall, maximum drawdown, portfolio weights, betas, and cross-strategy correlations—to the ecology. These regressions include strategy fixed effects and control for market volatility and short-horizon returns, so that the coefficients on Z_t capture how the same strategy's performance and risk change across habitats rather than spurious differences across models or mechanical exposure to standard risk factors. The AMH imposes additional discipline by delivering the following predictions: (1) crowding and tight funding should reduce conditional alpha and Sharpe ratios and raise tail risk and drawdowns; (2) trend-consistent volatility and persistence should improve performance and smooth paths for continuation rules; (3) jump-like, reversal volatility and poor liquidity should worsen tails; and (4) illiquid, crowded, high-volatility states should generate spikes in cross-strategy correlation.

These predictions find strong empirical support. Three main findings emerge. First, conditional performance is strongly state dependent in precisely the sense predicted by AMH. While unconditional alphas are small, we find that Sharpe ratios, CAPM alphas, and risk measures load significantly on the ecology vector with the expected signs. Strategies earn higher risk-adjusted returns in favorable trend, liquidity, and funding environments, and underperform when these conditions deteriorate. The same strategies also adapt their risk taking: betas and portfolio weights rise in favorable habitats and fall when volatility and funding stress surge, while strategies' variance, expected shortfall, and maximum drawdown respond in a coherent way to liquidity, volatility composition, crowding, and funding conditions. These patterns are difficult to reconcile with a purely risk-based EMH

benchmark once we allow for time-varying betas and volatility, but they arise naturally in an AMH world in which different trading constraints evolve.

Second, this state dependence varies sharply across model families, revealing ecological specialization. When we allow the loadings on Z_t to differ by family, we find pronounced heterogeneity in how strategies respond to the same market conditions. The baseline family of statistical and econometric models tends to perform relatively better in low-crowding, low-illiquidity environments and earns the largest premia in high jump-volatility regimes. Further, tree-based and regularized machine learning methods behave more like liquidity providers and are comparatively better positioned when spreads widen or funding conditions tighten. Finally, advanced neural architectures are disproportionately rewarded in high trend-volatility states and are more resilient to adverse funding shocks. Put differently, the same trading opportunity set rewards different types of strategies in different regimes. This family-specific pattern is central for AMH: it shows that machine learning does not fail uniformly, but that different architectures occupy different niches in the market ecology and that their profitability is conditional on the state of that ecology, rather than being fixed.

Third, beyond individual performance, the market ecology also organizes how strategies move together. Average cross-strategy return correlations are substantial even unconditionally, reflecting a shared exposure to the market and to common design choices, but they vary strongly with Z_t. Co-movement rises sharply when trading costs are high, positioning is crowded, trends are strong, and funding and liquidity are strained, and the dispersion of correlations across strategy pairs compresses in those same states. In loose, uncrowded, and microstructurally benign environments, both the level and concentration of correlations decline, allowing a richer diversity of effective niches. These patterns are difficult to generate under an EMH benchmark in which co-movement is orthogonal to the ecology, but follow naturally if common shocks to liquidity, funding, and competition act as habitat-level disturbances in an adaptive market.

These three findings collectively support AMH over EMH, but a natural concern is whether the observed state dependence is genuine or spurious. Our empirical design is explicitly built to address this concern. First, the ecology vector Z_t and its components are specified ex ante, grounded in AMH theory and prior work on liquidity, funding, and crowding, rather than chosen to maximize in-sample fit. Second, all strategies are evaluated strictly out of sample, under a common and realistic cost model and a fixed mapping from forecasts to portfolio weights, which limits the scope

for backtest overfitting at the strategy level. Third, our EMH benchmark is deliberately rich, allowing for time-varying betas, stochastic volatility, and state-independent but nontrivial trading costs.

Importantly, this benchmark delivers a sharp and testable implication in our setting. Because all strategies trade the same market-timing portfolio under identical implementation rules, a purely risk-based explanation predicts that changes in the market ecology should affect strategies in a broadly similar manner once exposure is controlled for. In particular, the incremental explanatory power of the ecology vector Z_t should be largely invariant across forecasting methods. We find instead that the same ecological variables generate systematically different responses across model families in terms of performance, risk, and effective market exposure. Because the traded asset, transaction costs, and forecast-to-portfolio mapping are held fixed, this heterogeneity cannot be attributed to mechanical differences in risk exposure and more likely reflects method-specific interaction between forecasting technology and the evolving market environment. Finally, the same ecology variables jointly explain conditional alphas, Sharpe ratios, downside risk, and cross-strategy co-movement with a coherent sign pattern across equations. This cross-metric and cross-method coherence is difficult to obtain spuriously and is precisely what the Adaptive Markets Hypothesis predicts.

Our findings reconcile two seemingly contradictory views of market efficiency. Unconditionally, modern machine learning-based timing strategies fail to beat a simple buy-and-hold allocation once we impose realistic frictions, lending support to an approximate EMH view at the aggregate level. Conditionally, however, their performance, risk, and interactions vary in systematic and economically intuitive ways with the market ecology, and different strategy families specialize in distinct ecological niches. The evidence thus favors an Adaptive Markets perspective in which efficiency is conditional and evolving rather than absolute. For both communities, progress requires designing methods that are explicitly aware of, and robust to, evolutionary market dynamics—not deploying ever more complex architectures against stationary benchmarks, nor treating machine learning as a black box factor generator.

In summary, we contribute the first systematic, large-scale evidence on when and why advanced ML timing strategies fail to improve on buy-and-hold, and we show that their performance, risk, and co-movement are tightly linked to the trading ecology in ways that are predicted by AMH but invisible in unconditional tests.

Related Literature. Our work contributes to several strands of literature on the limits of trading strategies and market adaptation. Technical analysis dominated trading floors for decades, with practitioners convinced that price patterns could forecast future movements. Early studies considering the effectiveness of this approach suffered from data snooping bias, testing pre-selected technical indicators that may have been chosen based on prior knowledge of their historical performance (Brock et al., 1992; Sullivan et al., 1999; Levich and Thomas, 1993; Neely et al., 1997; Ready, 2002). The emergence of statistical correction methods such as White's Reality Check (White, 2000) and bootstrap-based testing (Sullivan et al., 1999) highlighted the limitations of early approaches. Allen and Karjalainen (1999) employed genetic algorithms to discover trading rules without human preconceptions, eliminating bias toward traditional indicators. They found using realistic trading costs that such trading rules designed to go in and out of the S&P 500 couldn't reliably beat a buy-and-hold strategy.

Quantitative methods emerged as the sophisticated response, promising scientific rigor and statistical validity. Academic finance contributed numerous anomalies seemingly ripe for exploitation: the value premium documented by Fama and French (1993), the momentum effect identified by Jegadeesh and Titman (1993), and dozens of other factors suggest that markets might contain discoverable inefficiencies. Jegadeesh and Titman (2001) confirmed momentum's persistence across decades and asset classes, while Asness et al. (2013) demonstrated international robustness for value and momentum strategies, showing that the initial results were impressive. Baltas (2017) also showed that various enhancements could further improve returns.

Yet these strategies traced the same lifecycle as technical analysis. McLean and Pontiff (2016) documented a sobering finding: strategy returns decline by approximately 58% following academic publication, with decay beginning even before publication as practitioners independently discover similar patterns. The mechanism is predictable: initial exploitation by traders with minimal market impact (Tóth et al., 2011), followed by increasing capital allocation as knowledge spreads, ultimately compressing profit margins until strategies become economically obsolete.

The momentum anomaly exemplifies this trajectory. Once a robust alpha source, Daniel and Moskowitz (2016) showed that momentum strategies now suffer periodic severe reversals, particularly during market recoveries following crises. These reversals occur precisely when momentum positioning becomes crowded, triggering simultaneous unwinding that amplifies losses. Value strategies have similarly experienced unprecedented underperformance recently, sparking debate

over whether this represents permanent arbitrage elimination or temporary cyclical behavior.

Algorithmic trading's rise in the 2000s promised to overcome human limitations through speed, precision, and emotionless execution. Firms invested billions of dollars in infrastructure–microwave networks, co-located servers, optimized code–to gain millisecond advantages. Yet the August 2007 quantitative crisis exposed this approach's fragility (Khandani and Lo, 2011). Strategies successful for years failed simultaneously, not from technical flaws but from excessive convergence on identical opportunities.

Modern machine learning represents the latest attempt to discover patterns beyond human comprehension. These approaches can process alternative data sources, identify non-linear relationships, and theoretically adapt to changing conditions. However, empirical evidence suggests they face identical fundamental challenges. Harvey et al. (2016) catalogued over 316 factors proposed to explain cross-sectional equity returns, many discovered through ML techniques. This proliferation suggests widespread false discovery rather than genuine inefficiencies. Bailey et al. (2017) formalized the mathematics of backtest overfitting, demonstrating how extensive parameter testing, endemic to machine learning, virtually guarantees spuriously profitable strategies even in random data. While Harvey and Liu (2017) and Lopez de Prado (2018) developed frameworks to address these issues, implementation remains inconsistent in practice.

Against this backdrop, our work relates most closely to recent machine learning asset pricing studies such as Gu et al. (2020), Kelly et al. (2024), and Chen et al. (2024). Whereas these papers focus on cross-sectional return prediction and factor pricing, typically evaluating models by improvements in pricing errors or Sharpe ratios of long-short portfolios, we study a canonical time-series market-timing problem in which a single risky asset (the market index) is traded against T-bills under realistic implementation costs. Moreover, while most of the existing empirical asset-pricing literature relies on feed-forward networks, tree ensembles, and relatively simple recurrent models, we import a broader class of recent time-series architectures from the machine-learning literature–non-stationary transformers, patch-based time-series transformers, neural hidden Markov models, meta-learning ensembles, and mixture-of-experts variants–and evaluate them within a unified trading protocol; to our knowledge, these architectures have not been systematically benchmarked in an asset-pricing context. These models are explicitly designed to handle non-stationarity, regime shifts, and long-range temporal dependencies, which are first-order features of financial time series but are only imperfectly captured by shallow architectures, so they constitute a particularly strin-

gent test of the potential and limits of modern ML in market applications. What our Adaptive Markets framework also adds to this literature is an explicit, ex-ante characterization of the market ecology and an empirical demonstration that the conditional performance, risk, and co-movement of ML strategies load systematically on this ecology. In that sense, our results suggest that future work on ML-based factor investing should evaluate not only average pricing improvements, but also how the profitability and risk of ML-managed portfolios vary across funding, liquidity, volatility, and crowding states. More broadly, we contribute to three literatures: (1) market efficiency and the limits of predictability; (2) machine learning in asset pricing; and (3) the empirical validation of AMH.

2 Unconditional Strategies Evaluation

In this section, we conduct a systematic unconditional evaluation of our portfolio strategies. We first group strategies into three categories according to their level of quantitative sophistication. We next introduce our metrics of portfolio evaluation. Finally, we provide summary statistics for each strategy (within a broader family) and relate their performance to that of a simple buy & hold strategy.

2.1 Classification of strategies

Our portfolio strategies focus on timing a broad equity market index, with r_{t+1} being the excess return on the market portfolio between t and $t+1$ net of the risk-free rate. They are implemented as dynamic allocations between the market portfolio and a risk-free asset, based on a common information set derived exclusively from historical OHLC (Open-High-Low-Close) price data.

2.1.1 Strategy taxonomy

Our empirical analysis employs 25 distinct forecasting models spanning three methodological families, each reflecting different levels of sophistication in information processing and adaptation to changing market conditions. Each model m produces at time t either a forecast $\widehat{r}_{t+1}^{(m)}$ or a direct portfolio weight $w_t^{(m)}$ for the market portfolio. We convert forecasts into portfolio weights using fixed mappings that control leverage and cap extreme positions, ensuring all strategies share

common risk-management frameworks. The resulting excess return for model m is

$$r_{t+1}^{(m)} \equiv w_t^{(m)} \, r_{t+1}. \tag{2.1}$$

This design ensures that performance differences across methods reflect differences in how they extract and process information from price data, rather than differences in assets, signal sources, or trading frictions. The progression from simple parametric models to complex adaptive architectures allows us to test whether market efficiency varies systematically with forecasting sophistication, a central prediction of AMH.

Table 1 lists all 25 strategies organized by family. We briefly describe each family's key characteristics. Complete technical specifications, hyperparameter grids, feature definitions, and training procedures are reported in Internet Appendix A.

Statistical and econometric methods (6 strategies). The first family comprises classical time-series and state-space models that rely on explicit parametric structure: ARIMA, Exponential Smoothing, Risk-Adjusted ARIMA, Linear Regression, Kalman Filter, and Hidden Markov Model. These methods use relatively low-dimensional feature sets (typically 1-20 indicators), employ closed-form or iterative parameter estimation, and maintain interpretable coefficient structures linked to economic intuition.

ARIMA models autoregressive and moving-average dynamics supplemented by volatility-based regressors. Exponential Smoothing captures trend and seasonal components with optimized smoothing parameters. Risk-Adjusted ARIMA extends the base specification by modeling residual volatility through GARCH(1,1) and implementing Kelly criterion-inspired position sizing targeting 15% annualized volatility. The Kalman Filter uses state-space dynamics to track time-varying parameters, interpreting filtered states as regime signals. The Hidden Markov Model explicitly estimates three latent market states (bear, neutral, bull) using Gaussian emissions with regime persistence constraints. Linear Regression provides a baseline parametric approach with isotonic calibration to correct prediction bias.

These models represent quantitative approaches available to market participants for several decades and provide natural benchmarks for evaluating whether more sophisticated methods offer incremental predictive power.

Classical machine learning (9 strategies). The second family consists of standard machine learning methods that relax functional-form assumptions while remaining computationally tractable: regularized linear models (Ridge, Lasso, Elastic Net, PCA Regression), non-parametric methods (K-Nearest Neighbors, Support Vector Machine), and tree-based ensembles (Decision Tree, Random Forest, Gradient Boosting).

Regularized regression handles multicollinearity through L1 penalties (Lasso), L2 penalties (Ridge), or combined penalties (Elastic Net), with regularization strength selected via time-series cross-validation. PCA Regression addresses multicollinearity through dimensionality reduction. Non-parametric methods capture local patterns: KNN uses distance-weighted voting among similar historical states, while SVMs employ kernel tricks to identify non-linear decision boundaries. Tree-based ensembles learn complex decision rules through recursive partitioning, with Random Forests using bootstrap aggregation and Gradient Boosting employing sequential error correction.

Classical ML methods typically use 15-30 technical indicators selected for low correlation and strong individual predictive power, with exact counts varying by model to balance expressiveness against computational and statistical efficiency. All apply isotonic calibration to correct systematic forecast bias while preserving rank ordering.

Neural networks and advanced methods (10 strategies). The third family employs deep learning architectures and meta-learning schemes designed to capture complex temporal dependencies, regime-specific behavior, and rapid adaptation to distributional shifts: recurrent architectures (LSTM, Enhanced LSTM), transformer-based models (PatchTST, Temporal Fusion Transformer, Non-Stationary Transformer), hybrid regime-switching models (Neural HMM), and adaptive learning methods (Online Learning, MAML, Robust Mixture of Experts, Ensemble Meta-Learning).

Neural networks leverage the full 101-feature set derived from OHLC data, exploiting their capacity for automatic feature learning. LSTMs use gating mechanisms to capture long-range temporal dependencies, with Enhanced LSTM adding multi-scale encoders and attention mechanisms. Transformer models apply self-attention to adapt feature importance dynamically: PatchTST processes temporal patches at multiple scales, Temporal Fusion Transformer combines variable selection with quantile forecasting, and Non-Stationary Transformer explicitly models distributional shifts through learned normalization and temperature scaling.

Neural HMM extends classical HMM by using a neural network to predict state-conditional Gaus-

sian emission parameters, with regime inference performed via the forward algorithm. Meta-learning approaches focus on rapid adaptation: Online Learning incrementally updates an ensemble while detecting concept drift, MAML learns initialization parameters that enable fast fine-tuning to new conditions, Robust Mixture of Experts combines specialist models through learned context-dependent gating, and Ensemble Meta-Learning dynamically weights predictions from diverse base models.

These advanced methods represent the current frontier of adaptive learning and constitute the closest analogues to sophisticated quantitative strategies employed by institutional investors. Their performance relative to simpler benchmarks speaks directly to whether increased sophistication translates into economically significant market-timing gains.

2.1.2 Common implementation framework

All strategies share four key design elements to ensure comparability:

Information Set. Our models use technical indicators derived exclusively from S&P 500 daily OHLC price data, consistent with weak-form market efficiency. We construct 101 features across seven categories: (1) *Price and Trend* (23 features): moving averages, price ratios, crossovers; (2) *Volatility* (18 features): rolling standard deviations, Bollinger Bands, Average True Range; (3) *Momentum* (11 features): RSI, MACD, rate-of-change measures; (4) *Candlestick Patterns* (6 features): Doji, Hammer, Engulfing patterns; (5) *Support/Resistance* (18 features): Donchian channels, percentile bounds; (6) *Calendar Effects* (12 features): day/month/quarter with cyclical encoding; (7) *Statistical* (13 features): STL decomposition, Fourier cycles, rolling skewness/kurtosis.

Different model families use feature subsets matched to their structure. Time-series models (ARIMA, Exponential Smoothing) operate primarily on return series with limited external regressors. Regime-switching models focus on returns and volatility to identify latent states. Classical ML uses 15-30 core indicators selected for predictive power and low correlation. Neural networks use the full 101-feature set, discovering relevant representations through training. Feature lookback windows range from 5 to 252 days, capturing dynamics at multiple time scales. All features strictly avoid look-ahead bias and are standardized using parameters estimated solely from training data.

Walk-forward validation. We employ strict walk-forward validation to eliminate look-ahead bias and simulate realistic trading conditions. Each model trains on a rolling 5-year window (ap-

proximately 1,260 trading days), generates a one-day-ahead forecast, then advances the window by one day and retrains. This ensures predictions at time t use only information available through $t-1$, producing fully out-of-sample forecasts throughout our evaluation period.

The 5-year window balances competing considerations. Shorter windows provide insufficient observations for stable parameter estimation, particularly for complex models. Longer windows reduce adaptation to evolving market dynamics, which is critical given AMH's prediction that efficiency varies over time. The chosen horizon provides adequate observations for models ranging from simple 6-parameter ARIMA to neural networks with hundreds of parameters, while allowing meaningful adaptation across market cycles.

We select hyperparameters using time-series cross-validation (TSCV) using expanding training sets to prevent information leakage. Daily retraining is particularly important for testing AMH: if market efficiency varies over time as predicted, models must adapt continuously to capture this time-variation.

Position sizing. We map raw forecasts $\widehat{r}_{t+1}^{(m)}$ into portfolio weights $w_t^{(m)} \in [0,1]$ through model-specific rules under returns optimization, or a unified Sharpe-ratio rule that ensures comparability. Under Sharpe optimization (our primary focus), all models compute expected Sharpe ratio $\widehat{S}_t = (\widehat{r}_{t+1} - r_{ft})/\widehat{\sigma}_t$ using 60-day rolling volatility, then convert to positions via $w_t = \min(1, \max(0, k \cdot \widehat{S}_t))$ with scaling factor $k = 0.5$. Positions are set to zero when $\widehat{S}_t < 0.1$ to avoid trading on weak signals.

This uniform approach isolates differences in forecast quality rather than position-sizing heuristics. Alternative mappings under returns optimization (adaptive thresholds, Kelly criterion, regime-based rules) are detailed in the Internet Appendix A.

Transaction costs. All strategies face identical transaction costs of 10 basis points roundtrip, calibrated to realistic bid-ask spreads and market impact for S&P 500 index trading.[1] Signal thresholds and hysteresis filters are designed to ensure strategies remain implementable after accounting for these frictions.

[1] As a robustness check, we also evaluate the strategies under a higher transaction-cost assumption of 25 basis points. The qualitative patterns of state dependence and cross-family heterogeneity remain unchanged.

2.1.3 Implementation principles

To ensure that differences in performance across strategies reflect differences in forecasting technology rather than implementation choices, we impose a common set of design principles across all 25 models. These principles are not intended to optimize any individual strategy, but to enforce temporal discipline, cost awareness, and risk control in a uniform manner, thereby making the strategies comparable and implementable in realistic market conditions. The five principles below summarize the shared constraints under which all models are developed and evaluated.

1. *Strict temporal discipline:* Walk-forward validation with time-series cross-validation prevents look-ahead bias. Feature standardization uses parameters estimated only from training data, refit at each window advance. Isotonic calibration for regularized models uses out-of-fold predictions.

2. *Transaction-cost awareness:* Signal thresholds create buffers relative to trading costs. Hysteresis filters reduce unnecessary turnover. These choices ensure implementability after accounting for realistic frictions.

3. *Volatility-aware risk management:* Adaptive thresholds scale with realized volatility, automatically tightening during stress periods. Risk-Adjusted ARIMA targets constant 15% annualized volatility. Regime-switching models adapt positions through state-probability weighting, reducing exposure when regime uncertainty is high.

4. *Regularization discipline:* Neural networks employ dropout (0.1-0.2), early stopping, and in some cases L1/L2 weight penalties. Linear models use L1/L2/combined penalties with cross-validated regularization strength. Tree-based models limit depth, ensemble size, and minimum samples per split.

5. *Computational tractability:* Architectures balance expressiveness with training efficiency, allowing daily retraining on 5-year windows to complete within reasonable time budgets while maintaining sufficient model capacity.

Internet Appendix A provides full technical specifications for each model; the common principles above ensure that subsequent differences in behavior can be attributed to forecasting architecture rather than implementation details.

2.2 Evaluation metrics

Let $\mathcal{I}_t$ denote the investor's information set at time t. A market-timing strategy S maps signals in $\mathcal{I}_t$ into a (possibly constrained) portfolio weight $w_t \in [\underline{w}, \bar{w}]$ on the risky asset (or factor) with excess return r_{t+1}. The strategy's excess return is

$$r^S_{t+1} \equiv w_t \, r_{t+1}.$$

Risk adjustment uses a one-factor (market) model with factor return f_{t+1}:

$$r^S_{t+1} = \alpha_t + \beta_t f_{t+1} + \varepsilon_{t+1}, \qquad \mathbb{E}[\varepsilon_{t+1} \mid \mathcal{I}_t] = 0,$$

where α_t captures the conditional abnormal return of S given $\mathcal{I}_t$. We define the conditional Sharpe ratio as $\mathrm{SR}_t \equiv \mu^S_t / \sigma^S_t$, where $\mu^S_t \equiv \mathbb{E}[r^S_{t+1} \mid \mathcal{I}_t]$ and $\sigma^S_t \equiv \sqrt{\mathbb{V}ar(r^S_{t+1} \mid \mathcal{I}_t)}$.

We further define risk measures. $\sigma^{S\,2}_t = Var(r^S_{t+1} \mid \mathcal{I}_t)$; $\mathrm{ES}_{q,t} \equiv \mathbb{E}\left[r^S_{t+1} \mid r^S_{t+1} \le q_t\right]$ is tail risk (expected shortfall) at level q; $\mathrm{MDD}_{t,h}$ is horizon-h expected maximum drawdown; β_t is the conditional market beta; $\rho_t(S, S')$ is the conditional cross-strategy correlation.

2.3 Empirical results

Our timing strategies assume that the market portfolio is proxied by the S&P 500 index while the risk-free rate is a 3-month T-bill rate. We further assume a roundtrip cost of trade of 10bps of the transaction value. Tables 2–5 report unconditional performance statistics for the Sharpe-ratio-optimized portfolios implied by each forecasting model, together with the buy-and-hold benchmark in the market index. We report a time-series average and standard deviation of each strategy based on monthly observations. We further report minimum and maximum values of each variable for the time series. Our sample covers the period 1970–2024.

Several patterns stand out. First, none of the model-based timing strategies delivers economically large unconditional outperformance relative to buy-and-hold. Average excess returns, alphas, and Sharpe ratios are, if anything, slightly lower than those of the passive benchmark once we account for the substantial dispersion across methods. Second, there is pronounced heterogeneity within each method family: a small number of individual models achieve buy-and-hold–like risk-adjusted performance, but many others perform poorly. Third, most strategies achieve their modest outcomes with substantially lower volatility than buy-and-hold, consistent with dynamic de-risking in

the absence of strong predictive signals rather than the exploitation of a persistent anomaly.

Table 2 shows that the buy-and-hold portfolio attains an average annualized excess return of about 12%, with substantial time-series variability. By comparison, the mean returns of the statistical/econometric models (Panel A) cluster around 6–7%, with the best-performing specifications, such as Exponential Smoothing and HMM, approaching but not materially exceeding the benchmark.[2] Classical ML methods (Panel B) deliver even lower unconditional returns on average, around 4% per year. Advanced ML architectures (Panel C) do not systematically dominate: the group average is roughly 5%, and while some models such as MAML and Neural HMM come reasonably close to the buy-and-hold mean, many others generate very modest returns. At the family level, therefore, the timing strategies fall short of the unconditional performance of a simple buy-and-hold allocation.

Risk-adjusted performance, measured by CAPM alpha in Table 3, paints a similar picture. The buy-and-hold strategy, by construction, exhibits an annualized alpha close to zero. The cross-sectional averages of alphas within each model family are slightly negative and economically small, on the order of -1 to -2 percentage points per year. Within families, there are a few individual models with mildly positive alphas–for example, Exponential Smoothing among statistical methods or some of the advanced ML specifications–but these are offset by others with substantially negative alphas, such as ARIMA or the Robust Mixture of Experts. Notably, Exponential Smoothing delivers a statistically significant alpha of 1.6% per year ($t \approx 2.2$) and a Sharpe ratio exceeding buy-and-hold (0.69 vs 0.66), demonstrating that the unconditional null of zero alpha can be rejected for select strategies. Overall, the unconditional alpha distribution is centered near zero, and no family of models delivers a systematically positive risk-adjusted premium relative to the benchmark.

The realized Sharpe ratios reported in Table 4 further reinforce this conclusion. The buy-and-hold portfolio attains an average annualized Sharpe ratio of about 0.66. Among the statistical methods, the best performers (Exponential Smoothing and HMM) achieve Sharpe ratios in a similar range, but the cross-sectional average across this family is much lower due to strongly negative outcomes for some specifications such as ARIMA. Classical ML models again cluster around Sharpe ratios close to zero, with several tree- and ensemble-based methods delivering negative Sharpe ratios on average. Advanced ML architectures exhibit somewhat higher Sharpe ratios on average than the classical ML group, and individual models such as MAML, Ensemble Meta learning, and some

[2] All means and standard deviations are annualized; see the Internet Appendix for implementation details.

transformer-based approaches achieve Sharpe ratios that approach, but do not robustly exceed, buy-and-hold. Taken together, the distribution of Sharpe ratios indicates that modern ML is not a free lunch: it produces a few strategies with benchmark-like risk-adjusted performance, but no robust improvement in unconditional Sharpe ratios at the family level.

Finally, Table 5 highlights the risk side of the trade-off. The buy-and-hold portfolio exhibits an average annualized volatility of roughly 16%. In contrast, the model-based timing strategies are substantially less volatile on average. Statistical/econometric methods operate at about half this volatility, classical ML models at roughly one-sixth, and advanced ML architectures sit in between. Several individual strategies, such as Linear Regression or the Nonstationary Transformer, generate portfolios with near-zero volatility, effectively hugging the risk-free asset for extended periods. These patterns are consistent with the Sharpe-ratio-optimization rule scaling down the risky exposure when the model delivers weak or noisy signals. They also explain why modest Sharpe ratios can coexist with low unconditional returns: the strategies often achieve acceptable risk-adjusted performance by taking very little risk, rather than by reliably timing the market.

In summary, the unconditional evaluation in Tables 2–5 yields a clear message. Despite their complexity, the ML-based timing strategies do not generate a stable, unconditional anomaly relative to buy-and-hold. Across families, average returns and alphas are at best comparable to, and often below, the passive benchmark, while Sharpe ratios are similar once one conditions on the realized risk profile. The main systematic difference is that many models deliver substantially lower volatility and tail risk by flexibly de-risking when their signals are weak, which is behavior that is hard to distinguish from optimal scaling under an approximately efficient market. Further, we observe a significant time-series variability in the evaluation metrics for several of the strategies, which suggests that many of the strategies do observe periods of significant outperformance; they just simply do not show consistency in this pattern. Both facts motivate the conditional analysis in the next sections, in which we examine whether any of these models deliver state-dependent improvements in performance that are consistent with AMH rather than the purely risk-based efficient markets benchmark.

3 Adaptive Markets Hypothesis and Testable Predictions

In this section, we adopt the Adaptive Markets Hypothesis (AMH) of Lo (2004), Lo (2017), and Lo and Zhang (2024) as an organizing framework for interpreting how modern machine learning

methods perform within—and adapt to—the evolving market ecology in our portfolio-management setting. AMH views financial markets as an evolving ecology in which populations of strategies, constraints, and trading technologies co-evolve. In this environment, the risk-return trade-off of a given rule is not a fixed property of the data-generating process but depends on the current habitat defined by competition, funding, liquidity, and volatility conditions. EMH arises as a special, knife-edge case in which learning has largely exhausted exploitable patterns and constraints are stable, so that conditional risk-adjusted returns do not vary systematically over time or across market states.

We do not attempt to estimate a full structural evolutionary model in which the ecology and the population of strategies are jointly determined. Instead, we take the realized path of a parsimonious ecology vector Z_t as a sequence of observable market states and test AMH through its implications for the conditional distribution of strategy outcomes. The key prediction is that the performance, risk, and co-movement of a given rule vary systematically with Z_t in ways that can be signed ex ante. Our EMH benchmark is deliberately rich in that it allows for time-varying betas and stochastic volatility. AMH requires, in addition, that Z_t has incremental predictive content for risk-adjusted outcomes with a coherent sign pattern across performance, risk, market exposures, and co-movement.

3.1 Unified outcome system and EMH benchmark

In our setting, the AMH disciplines the analysis by restricting attention to observable market states that plausibly shift the feasibility and profitability of timing strategies, and by generating qualitative sign predictions for how these states should affect performance, risk, and dependence. We formalize these ideas in a unified outcome system. For each strategy S and month t, we consider the vector of outcomes

$$Y_{S,t} = (\alpha_{S,t},\ SR_{S,t},\ \sigma^2_{S,t},\ ES_{q,S,t},\ MDD_{S,t,h},\ \beta_{S,t},\ w^M_{S,t},\ \rho_t(S,S')),$$

where $\alpha_{S,t}$ is the CAPM alpha, $SR_{S,t}$ the realized Sharpe ratio, $\sigma^2_{S,t}$ the realized variance, $ES_{q,S,t}$ the expected shortfall at tail probability q, $MDD_{S,t,h}$ the maximum drawdown over a fixed horizon h, $\beta_{S,t}$ the market beta, $w^M_{S,t}$ the portfolio weight in the market index, and $\rho_t(S,S')$ the return correlation between strategies S and S'.[3]

We model these outcomes as functions of the ecology vector Z_t through panel regressions of the

[3]For correlations, we work with cross-sectional averages and their dispersion across pairs.

form

$$Y_{S,t+1} = a_S + b'X_t + c'Z_t + \varepsilon_{S,t+1}, \tag{3.1}$$

where a_S are strategy fixed effects, X_t collects standard risk and volatility controls, and Z_t captures the ecology. The EMH benchmark imposes that, once we control for X_t and a state-independent cost model, the ecology has no incremental predictive content for risk-adjusted outcomes:

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1} \mid Z_t]}{\partial Z_t} = 0, \quad \frac{\partial \mathbb{E}[SR_{S,t+1} \mid Z_t]}{\partial Z_t} = 0, \quad \frac{\partial \mathbb{E}[\sigma^2_{S,t+1}, ES_{q,S,t+1}, MDD_{S,t+1,h} \mid Z_t]}{\partial Z_t} = 0,$$

and similarly for $\beta_{S,t+1}$, $w^M_{S,t+1}$, and $\rho_t(S, S')$. Finding $c \neq 0$ in (3.1) after these controls is therefore evidence of AMH-style state dependence rather than a by-product of unmodelled time variation in betas or volatility alone. The joint system allows us to test AMH not only through average performance, but also through the shape of the conditional distribution of returns and the co-movement of strategies.

3.2 Identification and forecasting heterogeneity

The unified outcome system in (3.1) delivers a natural benchmark for distinguishing between risk-based explanations and adaptive specialization. Under a rich conditional version of the Efficient Markets Hypothesis, the ecology vector Z_t may capture time variation in risk premia, volatility, funding conditions, or effective betas. In that case, conditional performance, risk, and dependence measures may vary over time without implying departures from efficiency.

Our setting yields a sharper implication. All strategies in our analysis trade the same market-timing portfolio under identical transaction costs, leverage constraints, information sets, and a common mapping from forecasts to portfolio weights. Consequently, differences across strategies arise exclusively from the forecasting framework used to process information. Under a purely risk-based interpretation, changes in the market ecology should therefore affect strategies in a broadly similar manner once realized exposure is controlled for. In particular, the loadings of performance, risk, and exposure measures on Z_t should be largely invariant across forecasting methods, up to sampling variation.

The Adaptive Markets Hypothesis implies a different pattern. In an evolving market ecology, forecasting technologies differ in their ability to extract information, adapt to non-stationarity, and manage implementation frictions. These differences generate endogenous specialization: the

same ecological state may be favorable for some forecasting frameworks and unfavorable for others, even when the traded asset and cost structure are held fixed. Empirically, this implies that the sensitivity of strategy outcomes to the ecology vector need not be common across methods, but may vary systematically across model families.

Our empirical analysis exploits this distinction. By holding the investment problem constant and varying only the predictive framework, we test whether state dependence reflects common shifts in market risk or method-specific interaction with the evolving ecology. The heterogeneity in ecological loadings across forecasting methods that we document in Section 5 is therefore central for identification and difficult to reconcile with a purely risk-based explanation.

3.3 Ecology channels and qualitative sign predictions

AMH becomes empirically disciplined once we specify the components of Z_t and their predicted effects. We construct Z_t from four blocks: aggregate crowding, funding constraints, the trading environment (liquidity-variance-trend, LVT), and implementation frictions. Section 4 describes their measurement. Here, we summarize the qualitative sign predictions that we test in the outcome system. Internet Appendix B provides the corresponding formal derivative restrictions.

Crowding captures the degree to which capital is concentrated in equity-oriented strategies. When many agents pursue overlapping rules, trades become more synchronized and price impact per unit of flow rises. Under AMH, this compresses conditional outperformance and steepens left tails. We therefore expect higher crowding to lower conditional alpha and Sharpe ratios, to increase return variance, expected shortfall, and drawdown depth, and to raise cross-strategy co-movement as similar strategies are forced to de-risk in tandem. In other words, crowding primarily acts as a fragility amplifier. It may not always destroy gross opportunities, but it makes their net, implementable value more sensitive to shocks.

Funding conditions summarize leverage, balance-sheet space, and financing haircuts faced by intermediaries and leveraged traders. Tighter funding makes it harder to take large positions, raises the shadow cost of risk, and forces de-risking precisely when volatility is elevated. AMH therefore predicts that tighter funding reduces risk-adjusted performance, raises variance, tail losses, and maximum drawdowns, and lowers effective market exposure as strategies cut risk. Episodes of funding stress also tend to synchronize flows and margin calls across strategies, generating correlation spikes when constraints bind.

The LVT block groups market liquidity, variance, and return persistence into a single trading-environment channel. Liquidity governs the mapping from intended trades to executed prices; volatility composition determines whether price changes reflect tradable structure or non-forecastable turbulence; trend measures the persistence of returns. In AMH, conditional performance is highest when volatility is dominated by trend-consistent moves and when liquidity is abundant, because signals are more informative and easier to implement. We therefore expect Sharpe ratios to be higher in liquid, trend-favourable states and lower when volatility is dominated by large, reversal-type jumps. Risk objects respond in parallel: variance, expected shortfall, and drawdowns should worsen in illiquid, jump-dominated, high-volatility regimes and improve in liquid, trend-consistent regimes. Market betas and weights should rise when trends are strong and liquidity is good, and fall when volatility becomes turbulent and implementation risk increases. Volatility and liquidity shocks also act as habitat-level disturbances, raising co-movement when many rules respond in the same direction.

Finally, implementation frictions translate gross edge into net performance. Spreads, depth, and other microstructure costs tend to widen in precisely those states in which volatility and constraints are most severe. AMH implies that higher trading costs directly erode alpha and Sharpe ratios for any given gross signal, and exacerbate variance and tail risk when strategies continue to trade in stressed microstructure conditions. High-cost environments also depress effective betas and market weights as strategies scale back trading to preserve net performance. Because cost spikes are often systemic, they contribute to episodes in which a diverse set of algorithms simultaneously reduces risk, further increasing co-movement.

In combination, these four blocks generate a rich but disciplined system. Under AMH, performance, risk, market exposures, and co-movement should load on Z_t with signs that are largely shared across strategies, and that differ in predictable ways across ecology components. The empirical analysis in Section 5 evaluates these qualitative predictions in the unified outcome system (3.1). A formal statement of the sign restrictions implied by each channel is provided in Internet Appendix B.

4 Measures of the Ecology Vector Z_t and Data Sources

In this section, we operationalize the ecology vector:

$$Z_t = (\text{Crowding}_t,\ \text{Funding}_t, Z_{LVT,t},\ \text{Costs}_t),$$

as postulated in the AMH framework. Each component corresponds to a distinct ecological channel, crowding, funding stress, trading environment, and implementation frictions, and is constructed at the monthly frequency to align with the S&P 500 vs. T-bill timing strategy's rebalancing horizon.

4.1 Crowding: System-Wide Equity Leverage and Positioning

For an aggregate equity-timing strategy, crowding captures how much capital is already committed to equity risk in the U.S. market. We construct three complementary, observable proxies extending back to the early 1970s.

Our primary crowding variable is (1) mutual fund flow crowding (MF Crowding), measured as:

$$\text{MF Crowding}_t = \frac{\sum_{j=1}^{12} \text{Net Flows}_{t-j}^{\text{Equity}}}{\text{AUM}_{t-12}^{\text{Equity}}}.$$

Net Flows are monthly aggregate net flows into U.S. equity mutual funds over the prior 12 months, scaled by 12-month lagged total equity mutual funds assets under management (AUM). The data are obtained from the CRSP Mutual Fund Database.[4] High values of MF Crowding_t indicate recent chase into equities and synchronized positioning.

Our second measure is (2) margin-debt crowding (Margin), defined as:

$$\text{Margin}_t = \frac{\text{Margin Debt}_t}{\text{MktCap}_t^{CRSP}}.$$

We obtain monthly margin debt from the NYSE Factbook (pre-1997) and FINRA (post-1997). Total U.S. equity market capitalization is from CRSP (for common shares, and three main exchanges).

For robustness, we have also used (3) equity-allocation crowding (Eq Share), calculated as:

$$\text{EqShare}_t = \frac{\text{Household equity assets}_t}{\text{Household financial assets}_t}.$$

where both measures of household equity and financial assets are from Federal Reserve Financial Accounts (Z.1, Table B.101). The data are quarterly which we linearly interpolate to monthly.

[4] For observations that are sampled are frequencies lower than monthly, we interpolate intra-quarter observations using linear smoothing.

4.2 Funding constraints: leverage and financial stress

Funding tightness captures how costly or difficult it is for investors to lever equity positions or maintain risk exposures. For a monthly S&P–T-bill strategy, we construct a long-horizon (1970-2024) funding block Z_t^{fund} that combines balance-sheet, market-price, and financial-conditions information. High values correspond to tight and risk-averse markets; low values indicate loose funding and abundant leverage.

Our first measure of funding tightness is (1) Chicago Fed National Financial Conditions Index, its credit part (NFCI Credit). The index, obtained from the Chicago Fed database, captures broad aggregate funding availability. Positive values of the index, NFCI Credit$_t > 0$ imply tight conditions whereas negative values, NFCI Credit$_t < 0$, loose conditions. Since the index is available weekly and our portfolio strategies are analyzed monthly, we take the last observation in each month t.

Our second measure of funding stress is (2) Credit Spread, defined as a difference in yields between Moody's BAA and AAA-rated U.S. corporate debt. Higher credit spreads imply tighter funding, higher leverage cost. As a final measure of funding stress we use previously defined (3) measure of Margin. High values of Margin imply easy funding and aggressive risk taking whereas lower value trigger deleveraging and binding constraints.

Under AMH, tight funding suppresses adaptive risk taking and increases payoff asymmetry for the S&P–T-bill strategy: performance deteriorates and drawdowns deepen when funding is tight, while loose funding supports stronger risk taking. Under EMH, these variables should not forecast risk-adjusted returns.

4.3 Trading environment: liquidity-variance-trend

The trading-environment block Z_{LVT} characterizes the ease with which short-term price movements can be exploited by adaptive timing strategies. It combines liquidity depth, directional volatility composition, and trend persistence.

(i) Liquidity. We use the Pástor–Stambaugh (2003) aggregate liquidity factor as a proxy for overall market liquidity and funding ease. This factor, LIQ_t, captures the sensitivity of stock returns to unexpected order-flow-induced price reversals and reflects time-varying liquidity risk in the U.S. equity market. Conceptually, high (positive) values of LIQ_t indicate abundant market-making capacity and low marginal trading costs, while low or negative realizations correspond

to liquidity dry-ups and constrained intermediation. Under AMH, fluctuations in LIQ_t represent shifts in the adaptive trading environment: a low-liquidity state constrains arbitrage capital and raises the ecological pressure on continuation strategies, whereas a high-liquidity state supports faster information absorption and smoother adaptation. Monthly LIQ_t data are obtained from WRDS.

(ii) Trend-based variance decomposition. Market volatility is one of the most important components of investment opportunities among active investors. We separate realized market return variance into components that help or hurt a continuation rule using the following three-step procedure.

Step 1: Let r_d be the daily log excess return on the S&P 500 (over T-bill), obtained from CRSP.

Step 2: Define a monthly trend direction signal

$$S_t = \text{sign}\left(\sum_{j=1}^{K} r_{t-j}\right), \qquad S_t \in \{-1, +1\},$$

using the past K months' cumulative excess returns (in our case we choose $K = 6$).

Step 3: Decompose daily volatility within month t:

$$\text{Trend Variance}_t = \sum_{d \in t:\, \text{sign}(r_d) = S_t} r_d^2, \qquad \text{Jump Variance}_t = \sum_{d \in t:\, \text{sign}(r_d) \neq S_t} r_d^2,$$

so total realized variance is $\text{RV}_t = \text{Trend Variance}_t + \text{Jump Variance}_t$. Trend Variance_t indicates variance in the trend direction (helpful for continuation). Jump Variance_t indicates variance opposing the trend. Under EMH, conditional on total RV_t, neither component should predict risk-adjusted returns.

(iii) Trend. We measure the persistence of price movements using the cumulative magnitude of past monthly returns, capturing how directional recent trends have been. To measure persistence, we define a 6-month trend signal:

$$\text{Trend}_t = \left| \sum_{j=1}^{6} r_{t-j} \right|,$$

where r_t denotes the monthly excess return on the S&P 500. In our setting, high values of Trend

indicate that returns have moved consistently in one direction, either clear uptrend or downtrend. Low values of Trend indicate that returns have been choppy or net to zero, a weak directional signal and regime uncertainty.

Under AMH, a high value of Trend indicates a stable, momentum-friendly environment that rewards adaptive continuation rules. Low values of Trend reflect evolutionary instability and likely reversals. Under EMH, conditional on volatility, Trend should have no predictive content.

4.4 Implementation frictions: trading-cost states

The implementation-friction captures how expensive it is at time t to turn portfolio signals into actual trades. For the S&P-T-bill strategy, the relevant frictions are the bid-ask spread, the price impact of trading, and a broad indicator of commission regimes. For our application, we estimate transaction costs using bid-ask spreads. Because intraday quotes are unavailable before the mid-1990s, all measures are based on daily CRSP prices, making them feasible from 1970 onward.

Specifically, we use the Roll (1984) spread estimator, which infers bid-ask bounce from daily price changes.

For each stock i in month t with D_t trading days:

$$\widehat{\mathrm{Cov}}_{i,t} = \frac{1}{D_t - 1} \sum_{d=2}^{D_t} \left(\Delta p_{i,d} \Delta p_{i,d-1} \right), \qquad \widehat{\mathrm{Spread}}_{i,t}^{\mathrm{Roll}} = \begin{cases} 2\sqrt{-\widehat{\mathrm{Cov}}_{i,t}}, & \text{if } \widehat{\mathrm{Cov}}_{i,t} < 0, \\ 0, & \text{otherwise.} \end{cases}$$

Here $\Delta p_{i,d}$ denotes the daily price change or daily log return. We aggregate to the market level (e.g., all CRSP common shares or S&P 500 constituents) using mkt-cap weights:

$$\text{Bid-Ask Spread}_t = \sum_i w_{i,t} \widehat{\mathrm{Spread}}_{i,t}^{\mathrm{Roll}}.$$

Higher spreads imply more expensive trading and tighter liquidity. Under AMH, timing-strategy alpha and Sharpe ratios fall when BAS is high. Under EMH, costs should not forecast conditional performance once gross returns are adjusted for realized volatility.

4.5 Summary statistics of the ecology vector Z_t

Table 6 reports summary statistics and the correlation matrix for the components of the ecology vector Z_t at the monthly frequency over our 1970–2024 sample. The variables comprise measures of trading costs and microstructure conditions (Bid-Ask Spread), aggregate crowding (MF Crowding), slow-moving market trend and its volatility decomposition (Trend, Trend Variance, Jump Variance), as well as funding and liquidity proxies (Credit Spread, NFCI Credit, LIQ, Margin).

Panel A shows that all components of Z_t exhibit economically meaningful variation over time. The average quoted bid–ask spread is only around 0.6% of price, with a standard deviation of 0.4% and occasional spikes above 5%, consistent with episodic microstructure stress and changes in trading technology over the sample. The crowding measure is centered essentially at zero with a standard deviation of about 0.02, and a range from approximately -0.15 to 0.11, indicating that the U.S. equity market oscillates between under- and over-weighted equity exposure rather than spending most of the time in a single extreme state. The slow trend variable has a mean of 0.21 with a relatively large standard deviation of 0.09, and a maximum close to 0.57, implying that the market frequently experiences persistent directional moves over horizons relevant for our timing strategies.

The volatility decomposition confirms that a nontrivial fraction of realized variation can be attributed to trend-consistent versus trend-breaking (jump) moves. Both Trend Variance and Jump Variance have small average magnitudes in absolute terms but exhibit fat right tails: their maxima exceed 0.03 and 0.05, respectively. This pattern is consistent with the idea that most months feature modest realized variance, punctuated by occasional episodes in which either continuation or reversal volatility dominates. The funding block displays substantial dispersion as well. The average corporate credit spread is about 1.07 percentage points with a standard deviation of 0.43 and a maximum above 3.3 percentage points, while the NFCI Credit index has mean essentially zero and moves between roughly -3.1 and $+3.4$. These ranges indicate that the sample spans periods of both extremely loose and extremely tight financial conditions. The aggregate liquidity measure (LIQ) has a slightly negative mean and a standard deviation of 0.06, with realizations extending down to roughly -0.46 and up to 0.20, capturing well-known liquidity dry-ups and rebounds. Finally, the margin proxy varies between about 0.006 and 0.022, with a mean of 0.013, indicating that while margin conditions adjust more slowly than credit spreads or NFCI, they nonetheless move meaningfully over time and can tighten substantially in stress episodes.

Panel B documents the correlation structure of Z_t and reveals a clear block structure that is consistent with our conceptual decomposition into trading costs, crowding, trading environment, and funding constraints. Within the Liquidity–Volatility–Trend (LVT) block, the bid–ask spread is strongly correlated with both Trend Variance ($\rho \approx 0.74$) and Jump Variance ($\rho \approx 0.63$), and moderately correlated with the trend level itself ($\rho \approx 0.25$). Trend Variance and Jump Variance are also highly correlated with each other ($\rho \approx 0.81$). These links indicate that episodes with elevated volatility, particularly of the jump/reversal type, tend to coincide with wider spreads and thus more expensive trading, which is precisely the environment in which the Adaptive Markets Hypothesis predicts both higher tail risk and lower net implementable edge for high-turnover strategies.

Liquidity (LIQ) loads with the expected sign against these stress measures. It is negatively correlated with Bid-Ask Spread ($\rho \approx -0.38$), Trend Variance ($\rho \approx -0.35$), and Jump Variance ($\rho \approx -0.42$), and only weakly related to NFCI Credit and MF Crowding. High values of LIQ therefore correspond to tight spreads and relatively subdued volatility, while low LIQ realizations align with illiquidity, wide spreads, and turbulent price dynamics. This pattern supports our interpretation of LIQ as capturing the ease of implementation and the extent to which realized volatility reflects tradable structure versus non-forecastable turbulence.

The funding variables form a distinct but related block. Credit Spread is positively correlated with NFCI Credit ($\rho \approx 0.47$) and with Trend ($\rho \approx 0.41$), reflecting the fact that periods of wide credit spreads tend to coincide with tighter financial conditions more broadly and with protracted valuation dislocations. The margin proxy is positively correlated with Bid–Ask Spread ($\rho \approx 0.24$), NFCI Credit ($\rho \approx 0.20$), and, to a lesser extent, Jump Variance and LIQ. While the magnitudes are modest, they are consistent with the view that margin constraints tighten in states characterized by elevated volatility and financial stress, yet their dynamics are not fully redundant with the other funding measures. Together, Credit Spread, NFCI Credit, and Margin span a multi-dimensional funding-constraints block rather than a single scalar factor.

By contrast, MF Crowding is nearly orthogonal to the rest of Z_t: its correlations with the other variables are all small in magnitude, generally below 0.08 in absolute value. This orthogonality is desirable for identification. Crowding captures slow-moving changes in the aggregate appetite for equity risk that are not mechanically tied to contemporaneous volatility, spreads, or conventional financial-conditions indices. In the AMH framework, this implies that crowding can shift the capacity and risk of timing strategies in ways that are not simply a re-labeling of high-volatility or

tight-funding states.

Overall, the summary statistics in Table 6 show that the ecology vector Z_t exhibits rich time variation and an economically intuitive dependence structure. The LVT and funding blocks are internally coherent and correlated in ways that match the narrative of liquidity and funding cycles, while the crowding measure adds an orthogonal dimension of competitive pressure. At the same time, none of the pairwise correlations is so large as to suggest degeneracy or exact multicollinearity. This combination of strong within-block structure and moderate cross-block dependence justifies our use of residualized versions of Z_t in the state-dependent performance regressions that follow. It also implies that any significant loadings of strategy performance and risk on Z_t can be interpreted as genuine ecological sensitivities, rather than artefacts of a single latent macro factor.

5 Performance of the Timing Strategies and the Adaptive Markets Hypothesis

This section evaluates the out-of-sample performance of the machine-learning market-timing strategies through the lens of AMH. The central implication of the AMH is that abnormal performance is state-dependent: strategies can deliver positive risk-adjusted returns in some market environments but not in others, as the market ecology of competition, funding conditions, and investor behavior evolves over time.

5.1 State dependence of portfolio strategies: empirical specification and results

Before turning to the estimates, it is useful to emphasize that the AMH makes predictions not only about average performance, but about how performance, risk-taking, and dependence respond jointly to changes in the market ecology. We therefore study a system of outcomes that includes risk-adjusted returns, tail risk, market exposure, and cross-strategy co-movement. Examining these objects in parallel allows us to distinguish adaptive specialization from generic time variation in market risk.

To formalize this idea, we relate a rich set of strategy performance measures to our state variables Z_t using the pooled panel regression

$$Y_{i,t+1} = a_i + b' X_t + \beta' Z_t + u_{i,t+1}, \tag{5.1}$$

where $Y_{i,t+1}$ denotes a monthly observed performance statistic for strategy i measured over the evaluation window between end of month t and month $t+1$. We alternately consider (i) CAPM alphas (α); (ii) the realized Sharpe ratio (SR); and a set of risk characteristics: return variance ($\sigma_t^{S\,2}$), expected shortfall ES, portfolio weight in the market index $weight$, market beta (β), and maximum drawdown (MDD). All strategies are constructed from Sharpe-ratio-optimized portfolios. The vector Z_t includes Bid-Ask Spread, $MF\ Crowding$, $Trend$, $Trend\ Variance$, $Jump\ Variance$, $Credit\ Spread$, $NFCI\ Credit$, LIQ, and $Margin$. In addition, we also control for $Mkt\ Volatility$ and $SP\ Ret$. All regressions include strategy fixed effects a_i absorbing any time-invariant heterogeneity across models. We cluster standard errors by year-month. We report the results in Table 7. Accordingly, we interpret these regressions as predictive in information time rather than as structural causal relationships.

State dependence of strategy attributes. The performance regressions show that risk-adjusted returns are clearly conditioned by the market ecology. For CAPM α, several components of Z_t are statistically significant, particularly those associated with funding conditions and the variance composition. F-statistics of the models with either α or Sharpe ratios clearly reject the hypothesis that all coefficients of Z are jointly equal to zero. Tighter credit and balance-sheet conditions are generally associated with lower conditional alphas, while more favorable volatility configurations (for example, states in which volatility is dominated by trend-consistent rather than jump-like moves) are associated with higher alphas. At the same time, the explanatory power is modest and unconditional alphas remain small, consistent with the view that any departures from efficiency are episodic and modest in magnitude. By contrast, realized Sharpe ratios exhibit a much stronger state dependence: multiple elements of Z_t load significantly on SR with an economically coherent pattern across liquidity, funding, and variance variables. Sharpe ratios tend to be higher in liquid, trend-supportive, funding-benign states and lower when volatility is reversal-driven and funding conditions are tight. This is precisely the type of sign pattern anticipated by AMH, even though the role of crowding in SR is more nuanced.

The risk regressions show that the same ecology variables systematically organize the conditional distribution of downside risk, in line with the channels emphasized in AMH. Like before, all F-statistics reject the hypothesis that all coefficients of Z are jointly equal to zero. Return variance is significantly higher when volatility and funding-stress proxies are elevated and significantly lower in more benign volatility and funding environments. Expected shortfall behaves similarly: extreme

losses are more pronounced in states with high volatility and tight funding or margin conditions, and attenuated when volatility is more trend dominated and liquidity conditions are better. Maximum drawdowns also respond strongly to Z_t: drawdowns deepen in high-volatility, funding-stress, and illiquidity states, and are mitigated in more liquid and trend-favorable environments. Crowding plays a visible role in the tails, with more crowded environments generally associated with larger drawdowns or more adverse downside realizations, consistent with emphasis on crowding as a channel for fragility rather than as a pure performance driver. Overall, the ecology vector explains a substantial share of the time-series variation in σ_t^2, ES_t, and MDD_t, and the sign pattern across coefficients is closely aligned with the AMH narratives about how liquidity, funding, and volatility composition translate into risk and tail behavior.

The exposure regressions further reinforce the idea that effective market risk is itself endogenous to the ecology. The portfolio weight in the market index responds significantly to Z_t. Strategies tend to carry more market risk in states characterized by favorable volatility composition and looser funding and less risk when spreads and volatility proxies signal stress. Market beta behaves in a similar way, increasing in more benign funding and volatility states and declining when trading conditions are adverse or implementation costs are high. These patterns are particularly informative given that all strategies share the same mechanical mapping from signals to target weights. The fact that betas and weights still exhibit systematic loadings on Z_t indicates that realized exposures are shaped by the interaction of the common trading rule with the evolving market environment, as predicted by AMH.

Taken together, the results in Table 7 show that performance, risk, and market exposures of timing strategies are jointly and significantly state dependent. Conditional Sharpe ratios, variances, tail risks, drawdowns, and betas all move with the ecology in ways that are broadly consistent with the AMH predictions for funding, liquidity, volatility composition, and costs, while the evidence on crowding is weaker for average performance but clearer for tail behavior. This joint pattern is difficult to reconcile with a purely risk-based EMH benchmark in which betas and risk-adjusted performance are constant and orthogonal to Z_t, and is instead highly consistent with an Adaptive Markets setting in which the profitability and aggressiveness of a given rule depend on the prevailing liquidity, funding, volatility, and crowding conditions.

Cross-strategy co-movement. Table 8 further summarizes how the average correlation of strategy excess returns and the dispersion of those correlations vary with the market ecology. For each month, we compute the full cross-strategy return correlation matrix and extract two objects: the cross-sectional mean of pairwise correlations, which captures the level of co-movement across all timing rules, and the cross-sectional standard deviation of correlations, which captures how heterogeneous or concentrated that co-movement is. We then regress these two series on the ecology vector Z_t, using the same set of monthly state variables as in the previous regressions.

The first thing to note is that the average correlation across all strategies is about 0.42, indicating that even though the models span very different architectures and forecasting philosophies, their realized returns load substantially on a common component. Second, the dispersion of pairwise correlations is around 0.22, so there is meaningful cross-sectional heterogeneity: in typical months some strategies move almost one-for-one with the rest of the universe, while others remain only weakly connected. The question is whether this co-movement structure is itself state dependent in the way predicted by the Adaptive Markets Hypothesis.

The regression results show that the level of co-movement responds strongly and systematically to the ecology variables. Higher bid-ask spreads are associated with significantly higher average correlations. The coefficient of the spread proxy is large and positive. Periods of high trading costs therefore look like one-factor environments in which all timing rules tend to rise or fall together, consistent with synchronized de-risking in stressed microstructure conditions. MF Crowding has a similar effect. The crowding index enters significantly positively in the average-correlation regression, indicating that when aggregate equity positioning is high and strategies overlap more, the cross-section of portfolio returns becomes more tightly coupled. The trend variable also raises co-movement, suggesting that strong, persistent market moves create a common directional force that many strategies ride simultaneously. The liquidity factor LIQ also enters with a positive and statistically significant coefficient.

Funding stress variables further reinforce this picture. Average correlations load positively on the NFCI Credit. In turn, wider corporate credit spreads are associated with lower average correlations. One interpretation of these two seemingly opposing effects is that broad stress in funding markets pushes strategies toward a shared deleveraging mode and hence higher co-movement, whereas idiosyncratic widening of credit spreads can induce more differentiated behavior across models and thus reduce the common component. The margin proxy has an especially strong effect. Its

effect indicates that tighter margin conditions sharply increase the average correlation of strategy returns. This is precisely the pattern one would expect when binding financing constraints force many risk-taking rules to exit simultaneously, turning a diverse ecology of strategies into a highly synchronized deleveraging trade.

The dispersion of correlations displays a generally mirrored pattern, albeit with somewhat weaker explanatory power (adjusted R^2 of 0.16 versus 0.40 for the mean). States that raise average co-movement often compress the cross-sectional spread in correlations, making the correlation structure more homogeneous. Broader funding stress, as measured by NFCI Credit and Margin, has a significantly negative loading for correlation dispersion, while Credit Spread has a significantly positive loading. This contrast again suggests that aggregate tightening of credit conditions makes strategy behavior more homogeneous, whereas widening credit spreads that are not accompanied by a broad stress episode can increase cross-sectional differentiation. For the remaining variables, the point estimates are generally of the expected sign (for example, bid–ask spreads, trend variance, and crowding enter with negative coefficients in the dispersion regression), but are imprecisely estimated once the full ecology vector is included.

Overall, the panel of conditional regressions paints a picture that is very consistent with the Adaptive Markets Hypothesis. The timing strategies do not generate a stable, unconditional anomaly. Instead, their alphas, Sharpe ratios, risk characteristics, and joint co-movement are all strongly state-dependent and move in economically intuitive ways with sentiment, liquidity, funding conditions, and market volatility. The results suggest that the strategies thrive only in particular configurations of the market ecology and that their effectiveness evolves over time as competition, constraints and risk perceptions change. In this sense, the evidence supports an AMH perspective in which market efficiency is conditional and adaptive rather than absolute.

5.2 Identification and ecological specialization across model families

A central implication of the identification discussion in Section 3 is that a purely risk-based interpretation predicts broadly similar sensitivities of strategy outcomes to the ecology vector once exposure is controlled for, because all strategies trade the same market-timing portfolio under identical implementation rules. The Adaptive Markets Hypothesis, by contrast, allows for systematic heterogeneity in ecological loadings across forecasting technologies. We test this implication by interacting each element of Z_t with indicators for the three model families and examining whether

the resulting loadings differ across families.

To assess whether the performance of our strategies varies systematically with market conditions in a way consistent with AMH, we regress the monthly attributes of each strategy on a set of state variables and their interactions with family dummies. The specification in Table 9 includes strategy fixed effects and standard errors clustered by year-month. The omitted category is family 1, so the coefficients of the state variables capture the loading of family 1, while the interaction terms measure how the loading of families 2 and 3 differs from that baseline. This allows us to study whether and how these three broad classes of timing methods specialize in different ecological niches.

The joint F-test is highly significant (Prob $> F = 0.000$), and the regression explains around 17% of the cross-sectional and time-series variation in α (adjusted $R^2 = 0.166$). Thus, even after absorbing strategy fixed effects, the state variables and their method-specific interactions contain non-trivial information about realized performance.

Crowding, liquidity, and order-flow conditions. Several variables that capture crowding, liquidity, and order-flow imbalances display strong method heterogeneity. For the weighted bid–ask spread BAS, family 1 loads negatively (coefficient about -0.84), implying that wider spreads and poorer liquidity are associated with lower alpha. In contrast, the interaction for family 2 is positive and significant, which yields an overall loading for family 2 that is positive. Family 3's interaction is essentially zero, so its loading remains close to that of method 1. Hence, when illiquidity increases and trading becomes more expensive, family 1 is penalized, whereas method 2 actually benefits. This pattern is consistent with family 2 acting more like a liquidity provider or harvesting a liquidity premium, while method 1 requires tight spreads to perform well.

A similar adaptive pattern emerges for the crowding measure MF Crowding. Family 1 is hurt in crowded environments: its coefficient is about -0.09 and significant at the 5% level. However, the interactions for family 2 and family 3 are $+0.15$ and $+0.11$, respectively (both $p < 0.01$). The implied loadings are therefore negative for method 1 but close to zero or mildly positive for families 2 and 3. In other words, the baseline family underperforms precisely when crowding is high, whereas the alternative families are neutral or even slightly rewarded in the same states. This sharp reversal across methods is a direct manifestation of AMH-style state dependence: as markets become more congested and illiquid, some strategies adapt and exploit the new conditions while

others are systematically harmed.

Trend and jump variance. Trend and jump variance, TV and JV, also exhibit clear method-specific effects. For trend variance TV, family 1 has a negative but imprecisely estimated loading (around -1.62), suggesting that higher trend variance does not translate into higher alpha for the baseline method. Family 3, however, has a large and statistically significant positive interaction (about $+2.83$), yielding a net loading that is positive and significant, while family 2's interaction is small and insignificant. Thus, only family 3 is systematically rewarded in states with elevated trend volatility, consistent with a design that better exploits favorable volatility conditions as defined in the main text.

For jump variance JV, family 1 loads positively and significantly (coefficient about $+1.30$), indicating that its alpha is higher when jump variance is elevated. The interactions for families 2 and 3 are negative, so their net loadings remain positive but are smaller than that of family 1. If we interpret JV as adverse or stress-type volatility, all methods earn compensation in turbulent environments, but family 1 is most exposed to and rewarded by jump variance, while families 2 and 3 are somewhat more insulated. This differentiation lines up with an ecological view in which strategies specialize in different parts of the variance spectrum: family 3 is particularly effective in trend-variance regimes, whereas family 1 is more "crisis-leveraged" and earns the largest premia when jump variance is high.

Other macro-financial state variables. The jump variance JV is positively related to performance for family 1 (coefficient about $+1.30$, $p = 0.006$). The interactions for families 2 and 3 are negative but not strongly significant, implying somewhat smaller positive loadings for those methods. The corporate credit spread proxy Credit Spread and the liquidity factor LIQ show no robust relation with alpha for any family: the main coefficients are small and insignificant, and the family interactions are also statistically weak. Similarly, the broad market controls Mkt Volatility and SP Ret are close to zero with tight confidence intervals, suggesting that the strategy alphas are not simply compensation for conventional market volatility or contemporaneous market returns.

Funding conditions play a more prominent role. The margin variable enters with a large and highly significant negative coefficient for family 1 (around -2874), indicating that tighter margin conditions are strongly associated with lower alpha for the baseline. The interactions for families 2 and 3 are positive and significant (roughly $+1013$ and $+1975$), so their implied loadings are still

negative but substantially smaller in absolute value. All three families suffer when margin conditions tighten, but families 2 and 3 are much less exposed than the baseline, with method 3 in particular being considerably more resilient to funding shocks.

Credit conditions, captured by NFCI Credit, also generate strong heterogeneity. For family 1, the loading is negative and precisely estimated (around -0.0054), indicating that increases in this credit indicator reduce alpha. Family 2 has a positive and significant interaction (about $+0.0053$), which nearly offsets the baseline effect and yields an implied loading close to zero, while family 3's interaction is tiny and insignificant. Thus, deteriorating credit conditions are bad for the baseline method but largely neutral for family 2, again highlighting that some methods adapt to adverse environments more effectively than others.

Implications for AMH. Taken together, these results show that the relative performance of the three families is strongly state dependent. For several key variables, including the illiquidity proxy (BAS), the crowding measure (MF Crowding), trend and jump variance (TV and JV), the credit indicator (NFCI Credit), and the funding proxy (Margin), we observe sign reversals or large differences in loadings across families, with many of the interaction terms statistically significant at conventional levels. The same trading opportunity set therefore rewards different methods in different regimes: family 1 tends to perform well in low-crowding, low-illiquidity environments and is highly exposed to bad volatility, while families 2 and 3 are comparatively better positioned in crowded, illiquid markets, under stressed credit conditions, and, for family 3, in trend-volatility regimes.

This pattern is consistent with the central prediction of AMH: strategy profitability is not fixed but depends on an evolving market ecology. As liquidity, crowding, volatility composition, funding, and credit conditions change, some methods adapt and continue to extract rents, whereas others lose their edge. The interaction structure in Table 9 provides direct evidence of such adaptation, with method-specific exposures that shift in ways aligned with AMH rather than with a purely risk-based, time-invariant risk–return trade-off.

5.3 State-dependent performance across model families

Table 9 extends the baseline AMH specification in Tables 7 and 8 by allowing the impact of the ecology vector Z_t on strategy outcomes to differ across model families. Specifically, we interact

each state variable with indicators for the three families of timing rules and consider majority of our dependent variables, including performance (CAPM α), risk (return variance σ^2 and expected shortfall ES), and cross-strategy co-movement (the average and standard deviation of pairwise correlations).

The first column of Table 9 confirms that, even after conditioning on the ecology vector, state dependence in performance is strongly heterogeneous across model families. For the baseline family, which includes the simpler and more traditional timing rules, several elements of Z_t are significantly related to α. For example, higher Trend and Jump Variance are associated with economically and statistically large increases in α, consistent with these strategies exploiting more pronounced directional and discontinuous movements in the market. At the same time, tighter funding conditions–as captured by more binding margin constraints–are also associated with higher contemporaneous α for the baseline family. The interaction terms show that these relationships are far from uniform across families. In general, family-1 models rely on relatively low-dimensional parametric structures. They earn the largest premia in high jump-volatility regimes. This suggests their mechanism is exploitative continuity, that is, they are designed to capture explicit statistical modeling of dynamics like autocorrelation and regime-switching. At the same time, they are hurt in crowded environments and suffer significantly when margin conditions tighten. Because they assume stable statistical properties (stationarity or simple drifting), they lack the flexibility to adapt when many agents pursue overlapping rules, which synchronizes trades and compresses outperformance.

Family 2 strategies, which rely on more flexible but still relatively structured machine-learning methods, typically attenuate the sensitivity of α to the LVT block (trend and jump variance) and to funding variables. They behave like a liquidity provider. By relaxing functional-form assumptions and allowing for high-dimensional nonlinearities and interactions, these models can capture the predictable price pressure that occurs when liquidity is strained. Unlike family 1, these models are relatively neutral to crowding (loadings close to zero). Their mechanism of recursive partitioning allows them to find small, non-linear niches that remain profitable even when broad market funding is tight, effectively attenuating the sensitivity to funding shocks that cripple simpler models.

Family 3 strategies, which employ the most advanced architectures, often exhibit loadings on Z_t that are opposite in sign to the baseline. In several cases (e.g., jump variance and margin), the sum of the baseline coefficient and the family interaction is close to zero or even negative, indicating that the same ecological state that is favorable for simpler rules can be neutral or unfavorable for

the most complex models. In general, these architectures are designed for contextual adaptation. They are disproportionately rewarded in trend-volatility regimes because they can capture complex long-range temporal dependencies and meta-learn across tasks. This family is the most resilient to funding shocks (Margin and Credit). For example, the Non-stationary Transformer separates relative patterns from absolute levels, allowing it to maintain predictive power even as distributions shift during turbulent periods. In AMH terms, these architectures exhibit a comparative advantage in environments characterized by stronger trend persistence and higher trend-consistent volatility, consistent with their design for contextual adaptation and non-stationarity.

The joint significance tests at the bottom of the table reject the null of identical ecology loadings across families, reinforcing the view that the AMH operates through a rich, model-specific pattern of ecological sensitivities.

Columns (2) and (3) show that the ecology variables also have systematic, family-dependent effects on risk. For the baseline family, higher market volatility and looser margin constraints are associated with significantly higher return variance and expected shortfall, consistent with the idea that strategies that remain active in stressed environments bear more systematic and tail risk. Trend variance and jump variance exhibit the opposite pattern. For the baseline strategies, periods of high trend (jump) variance are associated with higher (lower) realized variance and higher (lower) ES, suggesting that these rules successfully increase (cut) risk exposure when directional (discontinuous) moves intensify. The interaction terms again reveal substantial heterogeneity. Family 2 strategies tend to dampen the sensitivity of variance and ES to higher bid–ask spreads, trend, and tighter funding conditions, whereas family 3 strategies reduce the sensitivity of variance and tail risk to credit spreads but often remain more exposed to margin constraints. In other words, the same ecological state that generates attractive alphas for some families may do so by accepting more variance and tail risk, while other families generate their alphas in states where they can simultaneously reduce risk. This multidimensional state dependence is exactly the type of trade-off emphasized by the AMH, that is, performance, risk, and ecological conditions are intertwined, and the balance differs across evolutionary families of trading strategies.

Columns (4) and (5) document equally rich state dependence in cross-strategy co-movement. The average correlation of strategy returns and the dispersion of these correlations respond strongly to the crowding and funding components of Z_t. For the baseline family, tighter funding conditions are associated with positive changes in average correlations and a statistically significant reduction

in the standard deviation of pairwise correlations, indicating that strategies become more homogeneous in stressed markets. Similarly, episodes of elevated trend variance and high margin utilization are associated with a sharp rise in average correlations for the baseline family, but the effect is significantly reduced for families 2 and 3, suggesting that the former family of models tends to load more heavily on common risk components when the environment becomes more challenging. By contrast, in more benign liquidity and funding conditions, the dispersion of correlations is higher, and the more complex methods appear to span a broader set of risk dimensions. These patterns are consistent with the AMH view that competition and funding constraints compress the effective opportunity set, inducing convergence of strategies in bad times and greater diversity in good times.

Taken together, the results in Table 9 demonstrate that state dependence in our setting is not confined to average performance. The ecology vector Z_t jointly shapes α, risk, and cross-strategy co-movement in ways that differ systematically across model families. In some environments, such as those characterized by high trend variance, more crowding, and benign funding conditions advanced machine-learning strategies can deliver higher α without proportionately higher risk, while maintaining relatively low co-movement with other rules. In other environments–particularly those featuring tight margins, high jump variance, and high liquidity–the same strategies either fail to generate incremental α or do so at the cost of higher variance, larger tail risk, and greater co-movement with competing rules. This rich pattern of family-specific ecological sensitivities is precisely what the AMH would predict. The profitability and risk profile of a given algorithm depend on the evolving market ecology, and different *species* of strategies thrive in different niches.

6 Conclusion

This paper has examined the performance of a broad universe of modern machine learning-based market-timing strategies through the lens of the Adaptive Markets Hypothesis. We assembled a large panel of strategies that time the U.S. equity market by dynamically allocating between the S&P 500 and T-bills, based on a common information set and a shared implementation and cost model. The forecasting engines span classical statistical and econometric methods, tree-based and regularized machine learning models, and advanced recurrent and attention-based architectures explicitly designed for non-stationary environments. This design isolates differences in how models process information from differences in leverage, asset coverage, or ad hoc risk management, and allows us to interpret cross-strategy variation in performance and risk in economic rather than

purely algorithmic terms.

Our findings deliver a nuanced message. Unconditionally, a simple value-weighted buy-and-hold allocation in the market index remains a formidable benchmark that modern machine learning-based timing strategies do not surpass once realistic frictions and risk adjustments are imposed, lending support to an approximate EMH at the aggregate level. Conditionally, however, performance, risk, and co-movement vary in systematic and economically interpretable ways with the market ecology, and different model families specialize in different habitats. The evidence therefore favors an Adaptive Markets perspective in which efficiency is conditional, episodic, and shaped by the evolving interaction of strategies, funding, liquidity, and competition.

An important implication of this perspective is that the lifecycle of trading strategies need not be monotonic. The absence of robust unconditional outperformance does not imply that a strategy is permanently arbitraged away once discovered, disseminated, or scaled. Instead, our results are consistent with an adaptive environment in which the same forecasting rules can move in and out of effectiveness as liquidity conditions, funding constraints, volatility composition, and competitive pressure evolve. This episodic pattern of profitability contrasts with a strict interpretation of EMH, under which exploitable strategies are eliminated once and for all, but arises naturally in an Adaptive Markets setting.

These results suggest several directions for future research. First, evaluation of trading strategies, particularly those based on machine learning, should be explicitly conditional on the market ecology. Rather than asking whether a model beats the market on average, future work should characterize which states of Z_t support positive conditional premia for which strategy classes, and how these premia evolve as capital reallocates across models and asset classes. Second, the ecology vector we construct is a reduced-form summary of deeper forces. Embedding it and the strategies in a structural adaptive model in which Z_t and the population of rules co-evolve would allow sharper welfare and systemic-risk implications to be drawn. Third, our results point toward the design of ecology-aware machine learning methods that incorporate Z_t directly into their objectives, recognize regime shifts, and adapt capacity and risk-taking to crowding, funding, and liquidity conditions. For both practitioners and regulators, understanding how such adaptive algorithms interact with market-wide constraints and liquidity provision is likely to be central to assessing the stability of increasingly automated financial markets.

In summary, the historical progression of trading strategies highlights a central lesson for evalu-

ating contemporary approaches. Technological sophistication does not prevent strategy decay; if anything, it may accelerate it by enabling faster discovery and replication. The method of pattern discovery—whether through human insight, statistical analysis, or machine learning—matters less than the erosion of profitability through competitive exploitation and changing constraints. Markets adapt to successful strategies, altering the environment in which they operate. Any realistic assessment of machine learning's role in asset management must therefore be grounded in a framework that recognizes these adaptive dynamics.

Table 1: Portfolio Strategies by Methodological Family

Statistical & Econometric	**Classical ML**	**Neural & Advanced**
ARIMA	Ridge Regression	LSTM
Exponential Smoothing	Lasso Regression	Enhanced LSTM
Risk-Adjusted ARIMA	Elastic Net	PatchTST
Linear Regression	PCA Regression	Temporal Fusion Transformer
Kalman Filter	K-Nearest Neighbors	Non-Stationary Transformer
Hidden Markov Model	Support Vector Machine	Neural HMM
	Decision Tree	Online Learning
	Random Forest	MAML
	Gradient Boosting	Robust Mixture of Experts
		Ensemble Meta-Learning

Table 2: Summary statistics of annualized returns

Strategy	Mean	SD	Min	Max
Buy and Hold	0.1206	0.1528	-0.5218	0.5261
Panel A: Statistical Methods				
ARIMA	-0.0117	0.0705	-0.3988	0.1977
Exponential Smoothing	0.0735	0.0650	-0.1184	0.3221
HMM	0.1129	0.1528	-0.3569	0.5197
Kalman Filter	0.0738	0.1773	-0.7513	0.5310
Linear Regression	0.0413	0.0322	-0.0060	0.1422
Risk-adjusted ARIMA	0.0442	0.0370	-0.0510	0.1403
Total	0.0557	0.0891	-0.7513	0.5310
Panel B: Classical ML				
Decision Tree	0.0331	0.0494	-0.1280	0.2094
Elastic Net	0.0427	0.0351	-0.0753	0.1598
Gradient Boosting	0.0431	0.0339	-0.0437	0.1384
KNN	0.0400	0.0328	-0.0317	0.1436
Lasso	0.0404	0.0355	-0.1018	0.1321
PCA Regression	0.0442	0.0703	-0.1931	0.3336
Random Forest	0.0360	0.0438	-0.0904	0.1723
Ridge	0.0388	0.0726	-0.2474	0.2656
SVM	0.0495	0.0371	-0.0619	0.2318
Total	0.0409	0.0481	-0.2474	0.3336
Panel C: Advanced ML Methods				
Enhanced LSTM	0.0335	0.1020	-0.2699	0.4102
Ensemble Meta	0.0499	0.0537	-0.1612	0.2244
LSTM	0.0393	0.0582	-0.1985	0.2800
MAML	0.1046	0.1520	-0.4850	0.5330
Neural HMM	0.0678	0.0942	-0.2674	0.3242
Nonstationary Transformer	0.0427	0.0323	-0.0009	0.1407
Online Learning	0.0403	0.0368	-0.0351	0.1412
PatchTST	0.0455	0.0481	-0.1712	0.1621
Robust Mixture of Experts	0.0098	0.0900	-0.2606	0.2465
TFT	0.0622	0.0863	-0.2335	0.3633
Total	0.0496	0.0863	-0.4850	0.5331

Table 3: Summary statistics of annualized alphas

Strategy	Mean	SD	Min	Max
Buy and Hold	0.0001	0.0253	-0.1066	0.0847
Panel A: Statistical Methods				
ARIMA	-0.0688	0.0499	-0.3458	0.0886
Exponential Smoothing	0.0155	0.0446	-0.1440	0.2291
HMM	-0.0043	0.0685	-0.2060	0.3140
Kalman Filter	-0.0412	0.1101	-0.5514	0.3295
Linear Regression	-0.0029	0.0067	-0.0433	0.0124
Risk-adjusted ARIMA	-0.0007	0.0160	-0.0468	0.0641
Total	-0.0171	0.0493	-0.5514	0.3295
Panel B: Classical ML				
Decision Tree	-0.0167	0.0338	-0.1590	0.0977
Elastic Net	-0.0040	0.0194	-0.1344	0.0755
Gradient Boosting	-0.0087	0.0114	-0.0470	0.0179
KNN	-0.0057	0.0134	-0.0655	0.0211
Lasso	-0.0050	0.0165	-0.1492	0.0275
PCA Regression	-0.0196	0.0550	-0.2759	0.2393
Random Forest	-0.0162	0.0255	-0.1243	0.0616
Ridge	-0.0255	0.0564	-0.3265	0.1294
SVM	-0.0021	0.0112	-0.0973	0.0962
Total	-0.0115	0.0327	-0.3265	0.2393
Panel C: Advanced ML Methods				
Enhanced LSTM	-0.0461	0.0822	-0.3975	0.1750
Ensemble Meta	-0.0029	0.0381	-0.1534	0.1432
LSTM	-0.0098	0.0399	-0.2438	0.1162
MAML	-0.0123	0.0402	-0.1731	0.0967
Neural HMM	-0.0267	0.0496	-0.2103	0.0679
Nonstationary Transformer	-0.0001	0.0017	-0.0353	0.0032
Online Learning	-0.0070	0.0147	-0.0698	0.0416
PatchTST	-0.0083	0.0314	-0.1246	0.1024
Robust Mixture of Experts	-0.0448	0.0698	-0.3319	0.1710
TFT	-0.0124	0.0737	-0.2451	0.2179
Total	-0.0170	0.0506	-0.3975	0.2393

Table 4: Summary statistics of annualized Sharpe ratios

Strategy	Mean	SD	Min	Max
Buy and Hold	0.6619	1.0093	-2.7826	3.5707
Panel A: Statistical Methods				
ARIMA	-1.2421	1.0194	-4.1728	1.6901
Exponential Smoothing	0.6856	0.9589	-1.8759	2.8520
HMM	0.5949	1.1120	-2.2977	3.5572
Kalman Filter	0.3916	1.0852	-3.1551	3.3333
Linear Regression	-0.1153	0.9879	-3.0714	2.3900
Risk-adjusted ARIMA	0.0584	1.2128	-2.9378	3.5234
Total	0.0622	1.0627	-4.1728	3.5707
Panel B: Classical ML				
Decision Tree	-0.2819	1.1688	-2.9925	3.6142
Elastic Net	0.0471	0.9445	-2.4645	2.8622
Gradient Boosting	0.0463	1.0393	-4.1408	3.3766
KNN	-0.1127	1.0481	-3.2672	2.3055
Lasso	-0.0798	0.9235	-2.6289	2.9447
PCA Regression	-0.0813	1.0798	-2.6728	2.8806
Random Forest	-0.2788	1.0528	-3.0807	2.8444
Ridge	-0.1500	1.0245	-3.6693	2.5564
SVM	0.1823	1.0261	-2.6624	3.0071
Panel C: Advanced ML Methods				
Enhanced LSTM	0.0225	0.9987	-2.7604	3.0281
Ensemble Meta	0.4433	1.0146	-1.9424	3.0827
LSTM	0.0937	0.9300	-2.1433	2.3246
MAML	0.5457	1.0148	-2.8072	3.5707
Neural HMM	0.3385	1.0529	-2.4479	3.0385
Nonstationary Transformer	0.2816	1.1281	-3.2556	3.7508
Online Learning	-0.1578	1.0523	-3.1539	2.0845
PatchTST	0.1152	0.8730	-1.9157	2.3198
Robust Mixture of Experts	-0.7067	1.3858	-4.2293	4.0317
TFT	0.3083	1.0339	-2.5613	3.5658
Total	0.1284	1.1096	-4.2293	4.0317

Table 5: Summary statistics of annualized volatility

Strategy	Mean	SD	Min	Max
Buy and Hold	0.1583	0.0675	0.0684	0.4537
Panel A: Statistical Methods				
ARIMA	0.0486	0.0245	0.0203	0.1582
Exponential Smoothing	0.0473	0.0240	0.0091	0.1485
HMM	0.1441	0.0507	0.0684	0.3961
Kalman Filter	0.1389	0.0442	0.0694	0.3287
Linear Regression	0.0055	0.0043	0.0001	0.0248
Risk-adjusted ARIMA	0.0081	0.0068	0.0000	0.0347
Total	0.0654	0.0258	0.0000	0.4537
Panel B: Classical ML				
Decision Tree	0.0258	0.0178	0.0003	0.0860
Elastic Net	0.0141	0.0208	0.0007	0.1135
Gradient Boosting	0.0181	0.0117	0.0031	0.0590
KNN	0.0102	0.0071	0.0004	0.0384
Lasso	0.0080	0.0130	0.0000	0.0915
PCA Regression	0.0518	0.0369	0.0041	0.1886
Random Forest	0.0237	0.0132	0.0034	0.0710
Ridge	0.0554	0.0386	0.0046	0.2191
SVM	0.0120	0.0356	0.0001	0.2220
Total	0.0243	0.0296	0.0000	0.2220
Panel C: Advanced ML Methods				
Enhanced LSTM	0.0971	0.0426	0.0464	0.2655
Ensemble Meta	0.0429	0.0459	0.0020	0.2821
LSTM	0.0353	0.0476	0.0004	0.2706
MAML	0.1542	0.0672	0.0658	0.4517
Neural HMM	0.0897	0.0249	0.0429	0.1859
Nonstationary Transformer	0.0010	0.0010	0.0000	0.0191
Online Learning	0.0151	0.0096	0.0041	0.0702
PatchTST	0.0416	0.0239	0.0003	0.1309
Robust Mixture of Experts	0.0459	0.0170	0.0178	0.1083
TFT	0.0928	0.0384	0.0311	0.2843
Total	0.0616	0.0573	0.0000	0.4517

Table 6: Summary statistics and correlation matrix of market ecology variables

Panel A: Summary statistics

Variables	Mean	SD	Min	Max
Bid-Ask Spread	0.0062	0.0038	0.0010	0.0575
MF Crowding	0.0016	0.0205	-0.1548	0.1140
Trend	0.2071	0.0899	0.0458	0.5688
Trend Volatility	0.0012	0.0022	0.0001	0.0325
Jump Volatility	0.0013	0.0032	0.0001	0.0554
Credit Spread	1.0691	0.4281	0.5500	3.3800
NFCI Credit	0.0014	1.0088	-3.1226	3.42962
LIQ	-0.0280	0.0632	-0.4617	0.1980
Margin	0.0128	0.0034	0.0059	0.0215

Panel B: Correlation matrix

	Bid-Ask Spread	MF Crowding	Trend	Trend Volatility	Jump Volatility	Credit Spread	NFCI Credit	LIQ	Margin
Bid-Ask Spread	1.0000	0.0275	0.2512	0.7382	0.6255	0.1114	0.2222	-0.3751	0.2415
MF Crowding	0.0275	1.0000	-0.0590	-0.0086	0.0780	-0.0779	-0.0079	-0.0294	0.0522
Trend	0.2512	-0.0590	1.0000	0.2342	0.1594	0.4127	0.2025	-0.1383	-0.0958
Trend Volatility	0.7382	-0.0086	0.2342	1.0000	0.8123	0.3077	0.2243	-0.3451	0.1858
Jump Volatility	0.6255	0.0780	0.1594	0.8123	1.0000	0.2095	0.1543	-0.4198	0.1395
Credit Spread	0.1114	-0.0779	0.4127	0.3077	0.2095	1.0000	0.4699	-0.1052	-0.0886
NFCI Credit	0.2222	-0.0079	0.2025	0.2243	0.1543	0.4699	1.0000	-0.0174	0.1969
LIQ	-0.3751	-0.0294	-0.1383	-0.3451	-0.4198	-0.1052	-0.0174	1.0000	0.1288
Margin	0.2415	0.0522	-0.0958	0.1858	0.1395	-0.0886	0.1969	0.1288	1.0000

Table 7: Strategy attributes and ecology variables

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
	Performance			Risk		Exposures	
VARIABLES	α	SR	σ^2	ES	MDD	Mkt Weight	β
Bid-Ask Spread	-0.405*	13.067	0.087	0.033	-0.249	-0.436	-1.360***
	(0.221)	(9.203)	(0.062)	(0.035)	(0.337)	(0.399)	(0.498)
MF Crowding	0.015	5.766***	-0.000	-0.005	-0.093***	0.034	0.077
	(0.028)	(1.327)	(0.006)	(0.003)	(0.029)	(0.047)	(0.079)
Trend	0.017	0.494	0.002	0.003	0.036**	0.026	0.001
	(0.014)	(0.444)	(0.003)	(0.002)	(0.016)	(0.020)	(0.029)
Trend Variance	-0.262	-31.583	0.316**	0.255***	2.710***	-0.205	-0.270
	(0.443)	(21.633)	(0.135)	(0.076)	(0.749)	(0.777)	(1.179)
Jump Variance	0.609***	18.939	-0.077	-0.068*	-0.504*	0.988***	1.211
	(0.235)	(12.545)	(0.086)	(0.037)	(0.304)	(0.370)	(0.818)
Credit Spread	0.004**	-0.085	0.001*	-0.000	0.002	-0.014***	-0.008**
	(0.002)	(0.074)	(0.001)	(0.000)	(0.003)	(0.003)	(0.003)
NFCI Credit	-0.003***	-0.142***	-0.000*	0.000**	0.001	-0.000	0.003***
	(0.001)	(0.027)	(0.000)	(0.000)	(0.001)	(0.001)	(0.001)
LIQ	0.015*	1.242***	0.000	-0.002	-0.027**	0.008	-0.006
	(0.009)	(0.390)	(0.002)	(0.001)	(0.012)	(0.016)	(0.019)
Margin	-1.778***	7.057	0.417***	0.379***	2.117***	2.479***	0.432
	(0.174)	(7.123)	(0.042)	(0.023)	(0.201)	(0.375)	(0.388)
Mkt Volatility	-0.047	-8.355***	0.191***	0.112***	0.797***	-1.062***	-1.204***
	(0.066)	(2.708)	(0.022)	(0.012)	(0.093)	(0.117)	(0.178)
SP Ret	-0.011	4.625***	-0.000	-0.003*	-0.046***	0.062**	0.079**
	(0.014)	(0.640)	(0.003)	(0.001)	(0.015)	(0.024)	(0.031)
Constant	0.005	0.201	-0.010***	-0.003***	-0.026***	0.270***	0.304***
	(0.003)	(0.161)	(0.001)	(0.001)	(0.005)	(0.006)	(0.007)
Strategy fixed effects	Yes	Yes	Yes	Yes	Yes	Yes	Yes
F-statistic	17.52	15.66	37.60	91.50	69.20	40.95	28.25
p-value	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Observations	16,175	16,175	16,175	16,175	16,175	16,175	16,175
R-squared	0.150	0.207	0.484	0.690	0.557	0.892	0.847

Robust standard errors in parentheses.

*** p<0.01, ** p<0.05, * p<0.1

Table 8: Cross-strategy return correlations and ecology variables

VARIABLES	(1) Avg Correlations	(2) St dev Correlations
Bid-Ask Spread	2.729**	-0.535
	(1.215)	(0.377)
MF Crowding	0.356**	-0.058
	(0.138)	(0.045)
Trend	0.132**	-0.038**
	(0.053)	(0.015)
Trend Variance	0.280	-0.995
	(2.366)	(0.685)
Jump Variance	0.305	0.182
	(1.405)	(0.310)
Credit Spread	-0.078***	0.013***
	(0.008)	(0.003)
NFCI Credit	0.009***	-0.006***
	(0.003)	(0.001)
LIQ	0.118**	-0.024*
	(0.046)	(0.014)
Margin	8.148***	-0.127
	(0.814)	(0.307)
Mkt Volatility	0.084	-0.115
	(0.323)	(0.083)
SP Ret	0.096	-0.000
	(0.060)	(0.021)
Constant	0.356***	0.228***
	(0.017)	(0.005)
F-statistic	49.37	9.10
p-value	0.000	0.000
Observations	647	647
R-squared	0.404	0.160

Robust standard errors in parentheses.
*** p<0.01, ** p<0.05, * p<0.1

Table 9: Strategy attributes and cross-strategy correlations between ecology variables and model families

VARIABLES	(1) α	(2) σ^2	(3) ES	(4) Avg Correlations	(5) St dev Correlations
Bid-Ask Spread	-1.141*	0.244**	0.099*	-2.376	-1.295*
	(0.583)	(0.114)	(0.052)	(1.546)	(0.735)
Family 2×Bid-Ask Spread	1.816**	-0.299**	-0.169**	5.800***	0.328
	(0.732)	(0.128)	(0.066)	(1.849)	(0.670)
Family 3×Bid-Ask Spread	0.206	-0.123	-0.014	6.512***	-1.305*
	(0.659)	(0.112)	(0.051)	(1.861)	(0.757)
MF Crowding	-0.069	0.011	-0.001	0.224	-0.159**
	(0.044)	(0.008)	(0.004)	(0.220)	(0.080)
Family 2×MF Crowding	0.135***	-0.002	0.000	-0.119	0.062
	(0.047)	(0.007)	(0.004)	(0.206)	(0.093)
Family 3×MF Crowding	0.089**	-0.026	-0.011	0.175	0.050
	(0.041)	(0.020)	(0.009)	(0.210)	(0.095)
Trend	0.070***	0.013***	0.008***	0.110	0.005
	(0.026)	(0.004)	(0.002)	(0.073)	(0.028)
Family 2×Trend	-0.082***	-0.034***	-0.018***	0.050	-0.048**
	(0.024)	(0.004)	(0.002)	(0.056)	(0.023)
Family 3×Trend	-0.059***	0.003	0.003	0.018	-0.055**
	(0.022)	(0.006)	(0.002)	(0.058)	(0.025)
Trend Variance	-3.362***	0.522**	0.342***	8.091**	-0.861
	(1.275)	(0.234)	(0.107)	(3.171)	(1.688)
Family 2×Trend Variance	3.183**	-0.352	-0.167	-4.342	-1.350
	(1.574)	(0.236)	(0.114)	(4.081)	(1.387)
Family 3×Trend Variance	4.884***	-0.198	-0.068	-9.597**	1.992
	(1.415)	(0.330)	(0.147)	(4.062)	(1.703)
Jump Variance	2.807***	-0.334***	-0.162***	1.303	-0.607
	(0.480)	(0.111)	(0.052)	(1.605)	(0.637)
Family 2×Jump Variance	-2.793***	0.166	0.065	-1.110	0.754
	(0.630)	(0.113)	(0.056)	(1.550)	(0.692)
Family 3×Jump Variance	-2.980***	0.494**	0.176*	0.263	0.656
	(0.562)	(0.204)	(0.097)	(1.911)	(0.710)
Credit Spread	-0.001	0.004***	0.001**	-0.081***	0.023***
	(0.004)	(0.001)	(0.000)	(0.012)	(0.005)
Family 2×Credit Spread	0.005	-0.004***	-0.001***	-0.011	0.001
	(0.005)	(0.001)	(0.000)	(0.014)	(0.005)
Family 3×Credit Spread	0.007	-0.003***	-0.002***	0.006	0.003
	(0.004)	(0.001)	(0.000)	(0.015)	(0.006)
NFCI Credit	-0.005***	-0.000*	0.000	0.013***	-0.006***
	(0.001)	(0.000)	(0.000)	(0.005)	(0.002)
Family 2×NFCI Credit	0.005***	0.000	-0.000	0.001	-0.001
	(0.001)	(0.000)	(0.000)	(0.006)	(0.002)
Family 3×NFCI Credit	0.000	0.000*	0.000***	-0.012**	-0.002
	(0.001)	(0.000)	(0.000)	(0.005)	(0.002)
LIQ	0.019	0.001	-0.000	0.041	-0.049*
	(0.021)	(0.003)	(0.002)	(0.077)	(0.030)
Family 2×LIQ	0.025	0.000	-0.000	0.173**	-0.004
	(0.025)	(0.004)	(0.002)	(0.086)	(0.030)
Family 3×LIQ	-0.034	-0.002	-0.003**	0.072	-0.007
	(0.025)	(0.003)	(0.001)	(0.089)	(0.031)
Margin	-3.323***	0.534***	0.381***	7.300***	-1.085*
	(0.402)	(0.082)	(0.037)	(1.299)	(0.604)
Family 2×Margin	1.513***	-0.155*	0.085*	-4.603***	-0.461
	(0.526)	(0.091)	(0.046)	(1.519)	(0.595)
Family 3×Margin	2.503***	-0.151***	-0.080***	0.745	-0.539
	(0.450)	(0.049)	(0.027)	(1.414)	(0.636)
Mkt Volatility	-0.047	0.191***	0.112***	0.020	-0.349**
	(0.066)	(0.022)	(0.012)	(0.394)	(0.145)
SP Ret	-0.011	-0.000	-0.003*	0.142*	-0.039
	(0.014)	(0.003)	(0.001)	(0.076)	(0.031)
Constant	0.005	-0.010***	-0.003***	0.430***	0.239***
	(0.003)	(0.001)	(0.001)	(0.019)	(0.008)
Strategy fixed effects	Yes	Yes	Yes	Yes	Yes
F-statistic	10.95	26.02	49.45	12.16	10.05
p-value	0.000	0.000	0.000	0.000	0.000
Observations	16,175	16,175	16,175	1,943	1,943
R-squared	0.166	0.516	0.712	0.179	0.139

Robust standard errors in parentheses.
*** p<0.01, ** p<0.05, * p<0.1

References

Allen, F. and Karjalainen, R. (1999). Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51(2):245–271.

Asness, C. S., Moskowitz, T. J., and Pedersen, L. H. (2013). Value and momentum everywhere. *Journal of Finance*, 68(3):929–985.

Bailey, D. H., Borwein, J. M., de Prado, M. L., and Zhu, Q. J. (2017). The probability of backtest overfitting. *Journal of Computational Finance*, 20(4):39–69.

Baltas, N. (2017). Optimizing cross-asset carry. In *Factor Investing*, pages 317–364. Elsevier.

Brock, W., Lakonishok, J., and LeBaron, B. (1992). Simple technical trading rules and the stochastic properties of stock returns. *Journal of Finance*, 47(5):1731–1764.

Chen, L., Pelger, M., and Zhu, J. (2024). Deep learning in asset pricing. *Management Science*, 70(2):714–750.

Daniel, K. and Moskowitz, T. J. (2016). Momentum crashes. *Journal of Financial Economics*, 122(2):221–247.

Fama, E. F. and French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56.

Gu, S., Kelly, B., and Xiu, D. (2020). Empirical asset pricing via machine learning. *Review of Financeial Studies*, 33(3): 2223–2273.

Harvey, C. R. and Liu, Y. (2017). Backtesting. *Journal of Portfolio Management*, 42(1):13–28.

Harvey, C. R., Liu, Y., and Zhu, H. (2016). ... and the cross-section of expected returns. *Review of Financial Studies*, 29(1):5–68.

Jegadeesh, N. and Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance*, 48(1):65–91.

Jegadeesh, N. and Titman, S. (2001). Profitability of momentum strategies: An evaluation of alternative explanations. *The Journal of Finance*, 56(2):699–720.

Kelly, B., Malamud, S., and Zhou, K. (2024). The virtue of complexity in return prediction. *Journal of Finance*, 79(3): 459–503.

Khandani, A. E. and Lo, A. W. (2011). What happened to the quants in august 2007? evidence from factors and transactions data. *Journal of Financial Markets*, 14(1):1–46.

Levich, R. M. and Thomas, L. R. (1993). The significance of technical trading-rule profits in the foreign exchange market: A bootstrap approach. *Journal of International Money and Finance*, 12(5): 451–474.

Lo, A. W. (2004). The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. *Journal of Portfolio Management*, 30(5): 15–29.

Lo, A. W. (2017). *Adaptive Markets: Financial Evolution at the Speed of Thought*. Princeton University Press.

Lo, A. W. and Zhang, R. (2024). *The Adaptive Markets Hypothesis: An Evolutionary Approach to Understanding Financial System Dynamics*. OUP Oxford.

Lopez de Prado, M. (2018). *Advances in Financial Machine Learning*. John Wiley & Sons.

McLean, R. D. and Pontiff, J. (2016). Does academic research destroy stock return predictability? *Journal of Finance*, 71(1): 5–32.

Neely, C. J., Weller, P. A., and Dittmar, R. (1997). Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32(4):405–426.

Ready, M. J. (2002). Profits from technical trading rules. *Financial Management*, 31(3):43–61.

Sullivan, R., Timmermann, A., and White, H. (1999). Data-snooping, technical trading rule performance, and the bootstrap. *Journal of Finance*, 54(5):1647–1691.

Tóth, B., Lempérière, Y., Deremble, C., de Lataillade, J., Kockelkoren, J., and Bouchaud, J.-P. (2011). Anomalous price impact and the critical nature of liquidity in financial markets. *Physical Review X*, 1(2):021006.

White, H. (2000). A reality check for data snooping. *Econometrica*, 68(5):1097–1126.

Internet Appendix A: Portfolio Strategy Implementation Guide

Abstract

This appendix details the implementation of each forecasting strategy from the main paper. For each strategy, we describe the underlying theory and financial interpretation, explain implementation choices including hyperparameter selection with justification, and provide practical notes to facilitate replication.

Internet Appendix Contents

Internet Appendix List of Tables

IA1 Introduction and Framework

This appendix describes the implementation pipeline for trading strategies used to test the Adaptive Market Hypothesis. The framework comprises two stages.

Stage 1: Expected Return Forecasting. Sections IA3–IA6 introduce four model classes for forecasting next-period returns $\hat{r}_{t+1}$:

- **Section IA3 (Benchmark)**: Buy-and-hold strategy as reference point.
- **Section IA4 (Statistical Models)**: ARIMA, Exponential Smoothing, Kalman Filter, and HMM.
- **Section IA5 (Classical Machine Learning)**: Ridge, Lasso, SVM, Random Forest, and related methods.
- **Section IA6 (Neural Networks)**: LSTM, Transformer-based models, and Meta-Learning.

Stage 2: Signal Construction. Section IA7 describes converting return forecasts into trading signals using two optimization schemes:

- **Return-Focused**: Threshold rules maximizing expected returns.
- **Sharpe-Focused**: Risk-adjusted optimization targeting maximum expected Sharpe ratio.

All strategies share a unified evaluation framework:

1. **Walk-Forward Evaluation**: Five-year rolling training window.
2. **Features**: Daily OHLC-derived features for S&P 500 (Section IA2).
3. **Forecast Horizon**:
    - *Daily*: Classical ML models (Ridge, Lasso, Elastic Net, PCA Regression, SVM, KNN, Decision Tree, Random Forest, Gradient Boosting, Linear Regression).
    - *Monthly*: Time-series models (ARIMA, Exponential Smoothing, Kalman Filter, HMM) and neural networks (LSTM, Transformers, Meta-Learning).
4. **Sample Period**: January 1970–December 2024.

IA1.1 Validation Framework

Rigorous validation is critical for assessing true predictive power in financial forecasting. We employ time-series cross-validation for hyperparameter tuning and walk-forward evaluation with five-year rolling windows for out-of-sample testing. This approach eliminates look-ahead bias and data leakage, ensuring reliable performance estimates across different market regimes. Section IA9 details the validation framework, cross-validation design, and evaluation metrics.

IA2 Feature Engineering

This section describes how we transform raw OHLC (Open, High, Low, Close) price data for the S&P 500 into 101 engineered features that serve as inputs to our forecasting models. These features are organized into seven categories, each capturing different aspects of market behavior: price levels and trends, volatility and risk, momentum and direction, candlestick patterns, support/resistance

levels, calendar effects, and higher-order statistics. The goal is to provide models with a comprehensive representation of market state at each point in time.

IA2.1 Feature Categories Overview

IA2.1.1 Price Features (23 features)

Price features describe the current level of the market relative to its recent history. They help models identify whether prices are elevated or depressed, trending or mean-reverting, and how current levels compare to key moving averages that traders commonly monitor.

- **Log Price** (1): $\log(P_t)$ to normalize scale effects.
- **Price Z-Scores** (3): $z_t = \frac{P_t - \mu_n}{\sigma_n}$ for $n \in \{20, 60, 250\}$ days.
- **Normalized Price** (1): $\frac{P_t}{MA_{250}(P_t)}$ to remove long-term trends.
- **Moving Averages** (4): MA_n for $n \in \{5, 10, 20, 50\}$ days.
- **MA Slopes** (4): $\frac{MA_n(P_t) - MA_n(P_{t-5})}{MA_n(P_{t-5})}$ for $n \in \{5, 10, 20\}$.
- **Price Distance from MA** (3): $\frac{P_t - MA_n}{MA_n}$ for $n \in \{10, 20, 50\}$.
- **Price-to-MA Ratios** (3): $\frac{P_t}{MA_n}$ for $n \in \{10, 20, 50\}$.
- **Heikin-Ashi Candles** (4): HA_Open, HA_High, HA_Low, HA_Close to reduce noise.

The normalization using the 250-day moving average plays a key role:

$$\text{Normalized Price} = \frac{P_t}{MA_{250}(P_t)}. \tag{IA2.1}$$

This transformation removes long-horizon trends and improves stationarity, which is particularly important for many statistical and econometric models. By expressing prices as a ratio to their yearly average, we make features more comparable across different time periods and price levels—a 5% deviation from the 250-day MA means the same thing in 1980 as in 2024, even though absolute price levels differ dramatically.

The simple moving average (SMA) over n periods is computed as:

$$MA_n(P_t) = \frac{1}{n} \sum_{i=1}^{n-1} P_{t-i}. \tag{IA2.2}$$

It provides a smoothed estimate of the price by averaging the most recent n observations. We use shorter windows (e.g., $n = 5$) to capture quick price changes, but they are noisier estimates. We use longer windows (e.g., $n = 50$) for more stable estimates but lag behind current prices. This creates a natural tradeoff: short-window MAs react quickly but can whipsaw during choppy markets, while long-window MAs are more reliable trend indicators but respond slowly to genuine regime changes.

Heikin-Ashi Formulas. Heikin-Ashi (literally "average bar" in Japanese) candles smooth price action by averaging OHLC values across consecutive periods, making trends easier to identify and reducing the impact of single-day volatility spikes. This is particularly useful for models that struggle with noisy, erratic price movements.

$$\text{HA\_Close}_t = \frac{O_t + H_t + L_t + C_t}{4}, \tag{IA2.3}$$

$$\text{HA\_Open}_t = \frac{O_{t-1} + C_{t-1}}{2}, \tag{IA2.4}$$

$$\text{HA\_High}_t = \max(H_t, \text{HA\_Open}_t, \text{HA\_Close}_t), \tag{IA2.5}$$

$$\text{HA\_Low}_t = \min(L_t, \text{HA\_Open}_t, \text{HA\_Close}_t), \tag{IA2.6}$$

with initialization $\text{HA\_Open}_0 = O_0$. We use a simplified, non-recursive definition of HA_Open based on the previous standard candle's open and close, rather than the traditional recursive formulation $(\text{HA\_Open}_{t-1} + \text{HA\_Close}_{t-1})/2$. This choice avoids look-ahead bias in walk-forward testing while preserving similar smoothing properties.

IA2.1.2 Volatility Features (18 features)

Volatility features quantify market risk and uncertainty. They are essential for position sizing (riskier markets warrant smaller positions), regime detection (bull markets typically show lower volatility than bear markets), and forecast calibration (predictions are less reliable when volatility is elevated).

- **Returns** (1): Daily log returns $r_t = \ln(P_t/P_{t-1})$.
- **Annualized Volatility** (4): $\sigma_n \cdot \sqrt{252}$ for $n \in \{5, 10, 20, 60\}$ days.
- **Raw Volatility** (4): σ_n for $n \in \{5, 10, 20, 60\}$.
- **Average True Range** (2): ATR_n for $n \in \{14, 20\}$ days, where $TR = \max(H - L, |H - C_{prev}|, |L - C_{prev}|)$.
- **ATR Z-Score** (1): ATR standardized using a 60-day rolling mean and standard deviation.
- **Bollinger Bands** (4): BB_Upper, BB_Lower, BB_Width, BB_Position.
- **Volatility Z-Scores** (2): Standardized volatility for 20- and 60-day windows.

Bollinger Bands Formulas. Bollinger Bands construct a volatility-scaled envelope around price, with bands typically set at ±2 standard deviations from the 20-day moving average. Prices near the upper band suggest potential overbought conditions, while prices near the lower band suggest oversold conditions—though these should not be interpreted as mechanical buy/sell signals, since strong trends can keep prices "riding the bands" for extended periods:

$$\text{BB\_Upper}_t = MA_{20}(P_t) + 2 \cdot \sigma_{20}(P_t), \tag{IA2.7}$$

$$\text{BB\_Lower}_t = MA_{20}(P_t) - 2 \cdot \sigma_{20}(P_t), \tag{IA2.8}$$

$$\text{BB\_Width}_t = \frac{\text{BB\_Upper}_t - \text{BB\_Lower}_t}{MA_{20}(P_t)}, \tag{IA2.9}$$

$$\text{BB\_Position}_t = \frac{P_t - MA_{20}(P_t)}{2 \cdot \sigma_{20}(P_t)}, \tag{IA2.10}$$

where MA_{20} denotes the 20-day simple moving average and σ_{20} the corresponding rolling standard deviation. BB_Width reflects volatility, while BB_Position measures the relative location of the price within the bands, with values outside ± 1 indicating band breakouts.

IA2.1.3 Trend and Momentum Features (11 features)

Momentum features capture whether prices are accelerating in a particular direction and how strong that directional movement is. Unlike simple price changes, momentum indicators use multi-period comparisons and smoothing to filter out noise and identify persistent trends.

- **RSI** (1): Relative Strength Index.
- **MACD** (3): MACD line, Signal line, and Histogram.
- **Momentum** (3): $M_n = \frac{P_t - P_{t-n}}{P_{t-n}} \times 100$ for $n \in \{5, 10, 20\}$.
- **Price Position** (1): Location within the 52-week high–low range.
- **MA Crossovers** (2): Binary indicators of fast/slow MA cross events.
- **Trend Strength** (1): Composite measure based on MA alignment and RSI.

RSI Formulas. The Relative Strength Index evaluates recent price changes to detect overbought or oversold conditions. It compares the magnitude of recent gains to recent losses, producing a bounded oscillator between 0 and 100:

$$\Delta_t = P_t - P_{t-1}, \tag{IA2.11}$$

$$\text{gain}_t = \max(\Delta_t, 0), \tag{IA2.12}$$

$$\text{loss}_t = |\min(\Delta_t, 0)|, \tag{IA2.13}$$

$$\overline{\text{gain}}_{14} = \frac{1}{14} \sum_{i=0}^{13} \text{gain}_{t-i}, \tag{IA2.14}$$

$$\overline{\text{loss}}_{14} = \frac{1}{14} \sum_{i=0}^{13} \text{loss}_{t-i}, \tag{IA2.15}$$

$$RS = \frac{\overline{\text{gain}}_{14}}{\overline{\text{loss}}_{14}}, \tag{IA2.16}$$

$$RSI = 100 - \frac{100}{1 + RS}. \tag{IA2.17}$$

Gains and losses are calculated using a simple moving average. RSI lies between 0 and 100, where levels above 70 traditionally indicate overbought territory and below 30 indicate oversold territory. However, these thresholds should be interpreted with caution: in strong bull markets, RSI can remain above 70 for extended periods without signaling an imminent reversal.

MACD Formulas. The Moving Average Convergence Divergence (MACD) indicator tracks the relationship between two exponential moving averages to identify changes in momentum strength and direction. When the MACD line crosses above its signal line, it suggests strengthening upward momentum; crossovers below suggest weakening or reversing momentum.

The indicator is calculated using exponential moving averages (EMAs) with smoothing factor $\alpha = \frac{2}{n+1}$:

$$EMA_t = \alpha \cdot P_t + (1 - \alpha) \cdot EMA_{t-1}. \tag{IA2.18}$$

The MACD components are defined as:

$$\text{MACD}_t = EMA_{12}(P_t) - EMA_{26}(P_t). \tag{IA2.19}$$
$$\text{Signal}_t = EMA_9(\text{MACD}_t). \tag{IA2.20}$$
$$\text{Histogram}_t = \text{MACD}_t - \text{Signal}_t. \tag{IA2.21}$$

Positive MACD values indicate stronger short-term momentum relative to the long term, while the histogram reflects changes in momentum intensity.

Price Position. The 52-week price position scales the current price relative to its 52-week price range, making it a simple measure of whether the market is trading near recent highs (values near 1), recent lows (values near 0), or somewhere in between:

$$\text{Price\_52W\_Position}_t = \frac{P_t - L_{252}}{H_{252} - L_{252}}. \tag{IA2.22}$$

MA Crossovers. Binary variables flag crossover events, which many technical traders view as potential trend change signals:

$$\text{MA\_5\_10\_Cross}_t = \mathbf{1}[(MA_5^t > MA_{10}^t) \wedge (MA_5^{t-1} \leq MA_{10}^{t-1})]. \tag{IA2.23}$$
$$\text{MA\_10\_20\_Cross}_t = \mathbf{1}[(MA_{10}^t > MA_{20}^t) \wedge (MA_{10}^{t-1} \leq MA_{20}^{t-1})]. \tag{IA2.24}$$

Trend Strength. A summary indicator of overall trend alignment, aggregating signals from multiple moving averages and RSI into a single score between 0 (all bearish) and 1 (all bullish):

$$\text{Trend\_Strength}_t = \frac{\mathbf{1}[MA_5^t > MA_{10}^t] + \mathbf{1}[MA_{10}^t > MA_{20}^t] + \mathbf{1}[RSI_t > 50]}{3}. \tag{IA2.25}$$

IA2.1.4 Candlestick Patterns (6 features)

Candlestick patterns encode classical technical analysis heuristics into quantitative features. These patterns are widely monitored by discretionary traders and are thought to signal potential reversals or continuations. While their predictive power is debated in academic literature, including them allows models to learn whether these patterns contain information beyond what is already captured in price and volume data.

Define:

$$\text{Body}_t = |C_t - O_t|, \tag{IA2.26}$$
$$\text{UpperShadow}_t = H_t - \max(C_t, O_t), \tag{IA2.27}$$
$$\text{LowerShadow}_t = \min(C_t, O_t) - L_t. \tag{IA2.28}$$

The following binary indicators are computed:

Doji Pattern. A doji occurs when the opening and closing prices are nearly identical, suggesting indecision in the market:

$$\text{Doji}_t = \mathbf{1}[\text{Body}_t < 0.1 \times \overline{\text{Body}}_{20}]. \tag{IA2.29}$$

Hammer Pattern. A hammer features a long lower shadow and small body near the top, potentially signaling a bullish reversal after a decline:

$$\text{Hammer}_t = \mathbf{1}[(\text{LowerShadow}_t > 2 \cdot \text{Body}_t) \wedge (\text{UpperShadow}_t < 0.5 \cdot \text{Body}_t) \wedge (C_t > O_t)]. \tag{IA2.30}$$

Shooting Star. The inverse of a hammer—a long upper shadow suggests rejection of higher prices and potential bearish reversal:

$$\text{ShootingStar}_t = \mathbf{1}[(\text{UpperShadow}_t > 2 \cdot \text{Body}_t) \wedge (\text{LowerShadow}_t < 0.5 \cdot \text{Body}_t) \wedge (C_t < O_t)]. \tag{IA2.31}$$

Bullish Engulfing. Occurs when a bullish candle completely engulfs the previous bearish candle, suggesting strong buying pressure:

$$\text{BullishEngulf}_t = \mathbf{1}[(C_t > O_t) \wedge (C_{t-1} < O_{t-1}) \wedge (\text{Body}_t > 1.5 \cdot \text{Body}_{t-1})]. \tag{IA2.32}$$

Bearish Engulfing. The bearish counterpart—a large down day that engulfs the previous up day:

$$\text{BearishEngulf}_t = \mathbf{1}[(C_t < O_t) \wedge (C_{t-1} > O_{t-1}) \wedge (\text{Body}_t > 1.5 \cdot \text{Body}_{t-1})]. \tag{IA2.33}$$

Pattern Score. A simple count of how many candlestick patterns are active, providing a single measure of pattern intensity:

$$\text{Score}_t = \sum_{p \in \{\text{Doji,Hammer,ShootingStar,BullishEngulf,BearishEngulf}\}} p_t. \tag{IA2.34}$$

IA2.1.5 Channel and Support/Resistance Features (18 features)

These features capture price channels and key support/resistance levels. The intuition is that prices often oscillate within identifiable ranges, and the position within these ranges may predict future movements—prices near the bottom of a channel may be more likely to rise, while prices at the top may face resistance.

- **Donchian Channels** (12): For $n \in \{10, 20, 50\}$ days, we compute four components:
  - High: $\max(H_n)$,
  - Low: $\min(L_n)$,
  - Width: $(High - Low)/P_t$,
  - Position: $(P_t - Low)/(High - Low)$.

- **Price Channels** (6): For $n \in \{20, 50\}$ days, each channel includes three quantities:

$$\text{Channel\_High}_{n,t} = Q_{0.8}\left(C_{t-n+1}, \ldots, C_t\right), \tag{IA2.35}$$

$$\text{Channel\_Low}_{n,t} = Q_{0.2}\left(C_{t-n+1}, \ldots, C_t\right), \tag{IA2.36}$$

$$\text{Channel\_Position}_{n,t} = \frac{C_t - \text{Channel\_Low}_{n,t}}{\text{Channel\_High}_{n,t} - \text{Channel\_Low}_{n,t}}, \tag{IA2.37}$$

  where $Q_{0.8}$ and $Q_{0.2}$ represent the 80th and 20th percentiles over the rolling window. In contrast to Donchian Channels, which are based on absolute maxima and minima, these Price Channels use percentiles to reduce the impact of extreme outliers. Channel_Position is typically between 0 (near the 20th percentile) and 1 (near the 80th percentile), while values outside $[0, 1]$ indicate that the price is moving beyond the usual range.

IA2.1.6 Time-Based Features (12 features)

Calendar effects are well-documented in financial markets—the January effect, month-end rebalancing flows, and quarter-end window dressing can all create predictable patterns. These features allow models to learn whether certain days, weeks, or months systematically differ in their return distributions.

- **Basic Calendar** (5): DayOfWeek, Month, Quarter, DayOfMonth, WeekOfYear.
- **Cyclical Encoding** (4): sin and cos transforms applied to DayOfWeek and Month.
- **Market Regime Flags** (3): IsMonthEnd, IsYearEnd, IsQuarterEnd (binary).

Cyclical Encoding. The time variables are cyclical by nature (for example, the transition from Sunday to Monday, or December to January), so treating them as ordinary integers can miss this structure. For instance, if we encode months as integers 1-12, a linear model would treat the distance between December (12) and January (1) as 11 units, when they are actually adjacent. Therefore, we apply sine and cosine transformations to map these periodic variables onto a unit circle:

$$\text{DayOfWeek\_Sin}_t = \sin\left(\frac{2\pi \cdot d_t}{7}\right), \tag{IA2.38}$$

$$\text{DayOfWeek\_Cos}_t = \cos\left(\frac{2\pi \cdot d_t}{7}\right), \tag{IA2.39}$$

$$\text{Month\_Sin}_t = \sin\left(\frac{2\pi \cdot m_t}{12}\right), \tag{IA2.40}$$

$$\text{Month\_Cos}_t = \cos\left(\frac{2\pi \cdot m_t}{12}\right), \tag{IA2.41}$$

where $d_t \in \{0, 1, \ldots, 6\}$ is the day-of-week index (Monday = 0) and $m_t \in \{1, 2, \ldots, 12\}$ is the month number. Using both sine and cosine retains full phase information and ensures that adjacent periods are represented as nearby points.

Market Regime Flags. These are simple binary variables to capture calendar-driven market effects:

$$\text{IsMonthEnd}_t = \mathbf{1}[\text{day}_t \geq 28], \tag{IA2.42}$$

$$\text{IsYearEnd}_t = \mathbf{1}[\text{month}_t = 12], \tag{IA2.43}$$

$$\text{IsQuarterEnd}_t = \mathbf{1}[\text{month}_t \mod 3 = 0]. \tag{IA2.44}$$

They are meant to reflect potential behaviors such as month-end rebalancing, year-end tax-loss selling, and quarter-end reporting dynamics.

IA2.1.7 Statistical Features (13 features)

This group contains higher-order statistical moments and decompositions that go beyond simple means and standard deviations. Skewness and kurtosis capture asymmetry and tail behavior—features that are particularly important in financial markets, where extreme events occur more frequently than Gaussian distributions would predict. The Fourier and STL decompositions attempt to isolate cyclical and seasonal patterns from the underlying trend.

- **STL Decomposition** (3): Trend (20-day MA), Seasonal (day-of-week pattern), Residual.
- **Fourier Cycle** (2): sin and cos terms based on the dominant price cycle estimated from FFT.
- **Rolling Skewness** (4): Skewness of prices (20, 60-day) and skewness of returns (20, 60-day).
- **Rolling Kurtosis** (2): Price kurtosis over 20 and 60-day windows.
- **Normalized Returns** (2): Rolling z-score of returns for 20 and 60-day windows.

STL Decomposition. We implement a simplified version of Seasonal and Trend decomposition using LOESS (STL), which separates prices into a smooth trend component, a day-of-week seasonal pattern, and a residual:

$$\text{STL\_Trend}_t = \frac{1}{20}\sum_{i=0}^{19} P_{t-i}, \tag{IA2.45}$$

$$\text{STL\_Seasonal}_t = \bar{P}_{d_t} \quad \text{where } \bar{P}_d = \frac{1}{|D_d|}\sum_{j \in D_d} P_j, \tag{IA2.46}$$

$$\text{STL\_Residual}_t = P_t - \text{STL\_Trend}_t - \text{STL\_Seasonal}_t, \tag{IA2.47}$$

where D_d denotes the set of all observations whose day-of-week equals d, so the seasonal term represents the average price level for each weekday. This decomposition helps models distinguish persistent trends from recurring weekly patterns and transient noise.

Fourier Cycle Features. We use Fourier methods to capture the dominant cyclical structure in prices—essentially asking what is the typical length of a market cycle? and then creating features that track phase within that cycle:

1. Remove slow-moving trends using a 50-day moving average:

$$\tilde{P}_t = P_t - \frac{1}{50}\sum_{i=0}^{49} P_{t-i}. \tag{IA2.48}$$

2. Apply the Fast Fourier Transform (FFT) on the most recent 252 trading days:

$$\hat{P}_k = \sum_{n=0}^{N-1} \tilde{P}_n \cdot e^{-2\pi i k n/N}. \tag{IA2.49}$$

3. Select the dominant frequency f^* as the one with the largest amplitude (excluding the DC term):

$$f^* = \underset{k\in\{1,\ldots,N/2\}}{\arg\max} |\hat{P}_k|. \tag{IA2.50}$$

4. Convert this into the dominant period $T^* = 1/f^*$ and clamp it to $[5, 60]$ days.

5. Build the cycle features using T^*:

$$\text{Price\_Cycle\_Sin}_t = \sin\left(\frac{2\pi \cdot t}{T^*}\right), \tag{IA2.51}$$

$$\text{Price\_Cycle\_Cos}_t = \cos\left(\frac{2\pi \cdot t}{T^*}\right). \tag{IA2.52}$$

Rolling Skewness. Skewness summarizes distribution asymmetry. For window size n, we compute:

$$\text{Skew}_n = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s}\right)^3, \tag{IA2.53}$$

where $\bar{x}$ is the sample mean and s is the sample standard deviation. Negative skewness implies a heavier left tail (more extreme negative outcomes), while positive skewness implies a heavier right tail. In equity markets, negative skewness is common—large down days are more frequent and more severe than large up days—a pattern often called the crash premium.

Rolling Kurtosis. Kurtosis measures tail heaviness, and we report excess kurtosis relative to a normal distribution:

$$\text{Kurt}_n = \frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s}\right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}. \tag{IA2.54}$$

Positive excess kurtosis indicates heavier tails than Gaussian behavior, meaning extreme values are more likely. This is a common empirical property of financial returns and is one reason why risk models based on normal distributions systematically underestimate the probability of large losses.

Normalized Returns. Returns are also standardized using rolling windows $n \in \{20, 60\}$:

$$\text{Normalized\_Returns}_{n,t} = \frac{r_t - \bar{r}_{n,t}}{\sigma_{n,t}}, \tag{IA2.55}$$

where r_t is the return at time t, $\bar{r}_{n,t} = \frac{1}{n}\sum_{i=0}^{n-1} r_{t-i}$ is the rolling mean, and $\sigma_{n,t}$ is the rolling standard deviation. This feature indicates how unusual the current return is compared to recent history. A normalized return of +2.5 means today's return is 2.5 standard deviations above the recent average—a statistically significant positive move that may signal momentum or an overreaction.

IA2.2 Feature Scaling

We standardize all features in a pre-processing step prior to training so that all variables are on comparable scales:

$$z = \frac{x - \mu}{\sigma}. \tag{IA2.56}$$

Critical Rule: The scaler is fit ONLY using training data in order to avoid any look-ahead bias. The same fitted scaling parameters are then reused to transform the test data. This ensures that no information from the test period—not even basic statistics like means and standard deviations—can leak into the training process.

Alternative scaling approaches are also applied in specific situations:

- **MinMaxScaler**: $x' = \frac{x - x_{min}}{x_{max} - x_{min}}$ for features that are naturally bounded.
- **RobustScaler**: $x' = \frac{x - Q_{50}}{Q_{75} - Q_{25}}$ for features that are sensitive to outliers.

IA2.3 Model-Specific Feature Sets

Different model classes use different sets of the features defined above:

- **Time Series Models** (ARIMA, Exponential Smoothing): These models are applied directly to returns, with optional external predictors such as volatility regime indicators and ATR.
- **Classical ML Models** (Ridge, Lasso, SVM, etc.): These models use a selected set of roughly 25–30 technical indicators, including MA ratios, momentum measures, RSI, MACD, and volatility-related features.
- **Neural Networks** (LSTM, Transformers): These models take all 101 features as input, leveraging deep learning's capacity to learn useful structure from high-dimensional feature representations.
- **Regime-Switching Models** (HMM, Kalman): These models mainly use returns and volatility signals to infer latent market states.

Please refer to corresponding model-specific sections below for more implementation details of exact features used by each model. Each of the features is constructed only from OHLC data as shown in the previous subsections.

IA3 Benchmark Strategy

IA3.1 Buy-and-Hold

The Buy-and-Hold strategy is used as a passive reference benchmark. It keeps a fixed 100% exposure to the S&P 500 index for the entire sample period, and serves as the standard point of comparison for all active trading strategies.

Implementation: This strategy does not involve any model estimation or forecasting. The market portfolio is held continuously without any rebalancing decisions.

IA4 Statistical/Econometric Methods

In this section, we discuss classical statistical and econometric methods, covering time-series models, state-space approaches, and regression techniques. These methods have been widely used with theoretical guarantees, and interpretable parameters.

IA4.1 ARIMA: Autoregressive Integrated Moving Average

IA4.1.1 Theoretical Foundation

The ARIMA(p, d, q) model, originally introduced in Box and Jenkins (1970), is still one of the standard tools in quantitative finance for short-horizon forecasting. It captures two common empirical patterns observed in financial return series: (1) weak yet persistent autocorrelation structures that can be leveraged for prediction, and (2) shocks tend to decay over time instead of remaining permanent.

ARIMA models returns as a linear function of lagged values and past forecast errors, while optionally applying differencing to deal with non-stationarity. The general form is:

$$\phi(B)(1-B)^d r_t = \theta(B)\epsilon_t, \tag{IA4.1}$$

where r_t is the return at time t, B is the backshift operator ($Br_t = r_{t-1}$), d is the differencing order, $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is white noise, and $\phi(B)$ and $\theta(B)$ are polynomial operators:

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p. \tag{IA4.2}$$

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q. \tag{IA4.3}$$

Expanding the ARIMA(p, d, q) specification yields:

$$(1-B)^d r_t = c + \sum_{i=1}^{p} \phi_i (1-B)^d r_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t. \tag{IA4.4}$$

For most financial return series, which are often close to stationary, setting $d = 0$ or $d = 1$ is usually enough. We estimate parameters using maximum likelihood, and model selection is performed using information criteria such as AIC/BIC.

Financial Interpretation. The autoregressive (AR) part reflects momentum-like behavior, where past returns can contain information about future returns. The moving-average (MA) part describes how shocks are absorbed over time; a positive θ implies that positive surprises can carry over into later returns. In practice, ARIMA is designed to exploit these weak predictive relationships, while differencing helps keep the model stable even when price levels trend or experience structural breaks.

A central assumption is that return dynamics are approximately linear and that the underlying autocorrelation pattern remains reasonably stable within the training window. This seems to be a good assumption for indices such as the S&P 500, where aggregation across many stocks can smooth and partially linearize behavior, although the assumption can fail during extreme stress periods when strong non-linearities appear.

IA4.1.2 Implementation Details

Table IA1: ARIMA Hyperparameters

Parameter	**Value**	**Justification**
Max AR order (p)	3	Upper bound for auto-selection; sufficient for daily returns
Max MA order (q)	3	Upper bound for auto-selection; balances shock memory vs. overfit
Max differencing (d)	2	Upper bound; handles unit roots if present
Seasonal component	None	Daily returns show minimal seasonality
Model selection	Auto (AIC)	Stepwise search selects optimal (p, d, q) within bounds

Feature Set: External Regressors for Volatility Regimes. The primary input is the log return series. Pure ARIMA models only the univariate return series itself, capturing linear autocorrelation patterns. However, financial returns exhibit volatility clustering and regime changes that pure ARIMA cannot represent—volatility today predicts volatility tomorrow, and high-volatility periods often coincide with different return dynamics.

We therefore employ ARIMAX (ARIMA with eXogenous regressors), using a specialized set of 8 externally computed volatility indicators rather than the general 101-feature set described in Section IA2. This design choice matches ARIMA's time-series forecasting structure while adding regime awareness that allows the model to adjust forecasts when market conditions shift. Although some of these variables (such as ATR and realized volatility) overlap conceptually with features in the broader set, they are computed independently within the ARIMA pipeline to maintain methodological consistency:

- **ATR Regime**: Average True Range classification (high/normal/low volatility based on 60-day median)
- **Volatility Regime Signal**: Ratio of 20-day realized volatility to 60-day baseline
- **ATR Percent**: ATR as percentage of price, $\text{ATR}_t/P_t \times 100$
- **High-Low Range Percent**: Daily range as percentage of price, $(H_t - L_t)/P_t \times 100$

- **Realized Volatility (5-day)**: Annualized 5-day rolling standard deviation of returns
- **Garman-Klass Volatility**: Efficient volatility estimator using OHLC data (20-day average)
- **Volatility-Adjusted Trend**: MA slope normalized by volatility
- **Z-Score**: Standardized deviation of price from 20-day moving average

Garman-Klass Volatility. The Garman-Klass estimator provides a more efficient volatility estimate than the standard close-to-close measure by incorporating intraday information from OHLC prices. By using the full daily range (high minus low) and the open-close relationship, it captures price movement that occurs within the trading day, typically reducing estimation variance by 5–7× for the same sample size **?**. The estimator is defined as:

$$\sigma^2_{GK} = \frac{1}{2}\ln\left(\frac{H_t}{L_t}\right)^2 - (2\ln 2 - 1)\ln\left(\frac{C_t}{O_t}\right)^2, \tag{IA4.5}$$

where H_t, L_t, O_t, C_t are the high, low, open, and close prices. The annualized version used as a regressor is:

$$\text{GK\_Vol}_t = \sqrt{\max(\sigma^2_{GK}, 0) \times 252}, \tag{IA4.6}$$

and in practice we use a 20-day rolling mean (GK_Vol_20d) as the final regressor to reduce noise.

Algorithm:

1. Compute log returns: $r_t = \ln(P_t/P_{t-1})$.
2. Compute the 8 external volatility regressors using only data available up to time t (no look-ahead).
3. Fit AutoARIMA on the training window:
   - Search over (p, d, q) orders within bounds using stepwise AIC minimization
   - Include external regressors in the regression component
   - Estimate parameters via maximum likelihood
4. Generate 1-step ahead forecast: $\hat{r}_{t+1}$.
5. Convert forecast to signal:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1)
   - SR-optimized: Sharpe-based method (Section IA7.2)

Practical Notes. We use the `statsforecast` implementation of AutoARIMA, which searches model orders using a stepwise routine that usually converges in about 2–5 iterations rather than running a full grid search. The ARIMA model is refit at every walk-forward step so it can track changing dynamics over time. External regressors add regime awareness that a pure ARIMA structure cannot represent on its own, but avoiding look-ahead bias is essential—all regressors are computed strictly from information available up to time t. The stepwise search procedure evaluates

candidate models by adding/removing one AR or MA term at a time, selecting the specification with the lowest AIC at each stage.

IA4.2 Exponential Smoothing

IA4.2.1 Theoretical Foundation

Exponential smoothing produces forecasts by assigning exponentially decaying weights to past observations (Holt, 1957; Brown, 1959; Winters, 1960). Even though the method is simple, it aligns with a natural financial intuition: the most recent information should matter more, while older observations should still contribute instead of being discarded. Because of the weighting structure, the method adapts to evolving market conditions without needing an explicit regime-detection mechanism.

The simple exponential smoothing (SES) forecast is:

$$\hat{r}_{t+1} = \alpha r_t + (1-\alpha)\hat{r}_t, \tag{IA4.7}$$

where $\alpha \in (0,1)$ is the smoothing parameter. Equivalently, the forecast can be expressed as a weighted sum over the entire history:

$$\hat{r}_{t+1} = \alpha \sum_{j=0}^{t-1} (1-\alpha)^j r_{t-j} + (1-\alpha)^t l_0. \tag{IA4.8}$$

When trend and seasonality are present, Holt-Winters extends SES by introducing level (l_t), trend (b_t), and seasonal (s_t) states:

$$l_t = \alpha(r_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}), \tag{IA4.9}$$
$$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}, \tag{IA4.10}$$
$$s_t = \gamma(r_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}, \tag{IA4.11}$$

where m is the seasonal period and α, β, γ are smoothing parameters for the level, trend, and seasonal components.

Financial Interpretation. α is the parameter that balances between sensitivity and smoothness. If α is large (close to 1), the model reacts strongly to recent moves, which can be beneficial in trending markets. If α is small, forecasts become steadier and less sensitive to noise, which can work better in mean-reverting or sideways regimes. The Holt-Winters extension is useful when returns contain calendar-driven patterns.

Compared with ARIMA (Section IA4.1), exponential smoothing avoids selecting lag orders, which can make it more robust when the true lag structure is uncertain. Smoothing parameters are estimated directly from the data rather than chosen via a combinatorial search across many model orders. That said, the method implicitly assumes relatively smooth evolution, and can lag during abrupt transitions, large shocks, or crash-like events where dynamics shift suddenly.

IA4.2.2 Implementation Details

Algorithm:

1. Compute log returns from price data: $r_t = \ln(P_t/P_{t-1})$.
2. Initialize level l_0 as mean of first seasonal period, trend b_0 as initial slope estimate.
3. For each observation in training window, update state equations (additive Holt-Winters):
   - Update level: $l_t = \alpha(r_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1})$.
   - Update trend: $b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$.
   - Update seasonal: $s_t = \gamma(r_t - l_t - b_{t-1}) + (1-\gamma)s_{t-m}$.
4. Generate forecast: $\hat{r}_{t+1} = l_t + b_t + s_{t+1-m}$.
5. Convert forecast to signal:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Table IA2: Exponential Smoothing Hyperparameters

Parameter	**Value**	**Justification**
Trend component	Additive	Linear trends avoid noise amplification
Seasonal component	Additive	Consistent with trend specification
Seasonal period	12	Captures monthly calendar effects
Smoothing parameters	Grid search	α, β, γ optimized via MSE minimization

Feature Set: Unlike regression-style models, exponential smoothing is applied directly to the return series and does not require external predictors. The temporal components, i.e, level, trend and seasonality are used to generate the expected returns.

Parameter Estimation: The smoothing parameters (α, β, γ) are chosen through a grid search that minimizes the sum of squared one-step-ahead forecast errors over the training window. Each parameter is searched over $(0.01, 0.99)$, and the parameter combination that yields the lowest MSE is selected. The method also automatically determines whether the data supports a simple (level-only), double (level + trend), or triple (level + trend + seasonality) specification based on autocorrelation structure.

Practical Notes: The seasonal period is set to 12 (monthly) to represent calendar effects; when working with daily observations, this corresponds to roughly 21 trading days per month. The implementation automatically selects the appropriate smoothing form—simple, double, or triple exponential smoothing—based on observed series characteristics.

Design Choice: Undamped variant. In general, a damped version of exponential smoothing is used with the following updates using a damping factor ϕ:

- level: $l_t = \alpha(r_t - s_{t-m}) + (1-\alpha)(l_{t-1} + \phi b_{t-1})$.
- trend: $b_t = \beta(l_t - l_{t-1}) + (1-\beta)\phi b_{t-1}$.

- seasonal: $s_t = \gamma(r_t - l_t - \phi b_{t-1}) + (1-\gamma)s_{t-m}$.

We use an undamped Holt-Winters specification (damping factor ϕ=1) rather than a damped variant. Although damped exponential smoothing ($\phi < 1$) is commonly preferred for long-horizon forecasts to prevent excessive trend extrapolation, the undamped version is suitable here for multiple reasons:

- **Short forecast horizon**: Forecasts are one-step-ahead (next-day). For $h = 1$, the damping effect $\phi^h \approx \phi$ has little influence on accuracy.
- **Frequent retraining**: Walk-forward validation refits the model often, so parameters can update as market conditions shift rather than relying on damping for trend decay.
- **Model parsimony**: Using fewer parameters reduces overfitting risk in noisy financial settings, and undamped ϕ simplifies estimation without hurting short-horizon forecasting.
- **Momentum capture**: In strong trends, undamped forms track persistent directional movement more closely, which fits a momentum-sensitive trading use case.

IA4.3 Risk-Adjusted ARIMA

IA4.3.1 Theoretical Foundation

Risk-Adjusted ARIMA improves standard ARIMA (Section IA4.1) by adding GARCH-style volatility modeling and Kelly criterion-inspired position sizing. The motivation is a practical weakness of plain ARIMA: it provides point forecasts of returns but does not directly incorporate current market risk. As a result, it can suggest large exposures during volatile periods, exactly when risk management would usually reduce risk.

Key Distinction from Standard ARIMA. Although both approaches rely on ARIMA(p, d, q) for forecasting returns, they differ at the signal-generation stage:

- **Standard ARIMA**: Uses external volatility regressors (ATR, Garman-Klass, etc.) *within* the forecasting model, and then produces signals using a straightforward threshold rule.
- **Risk-Adjusted ARIMA**: Fits a pure ARIMA to returns, then models the residual volatility using GARCH, and finally uses Kelly criterion-inspired sizing to form trading positions.

A GARCH(1,1) model is fitted to the ARIMA residuals to represent time-varying volatility (Engle, 1982; Bollerslev, 1986):

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2, \tag{IA4.12}$$

where ϵ_t are ARIMA residuals, ω is a constant, α captures the effect of recent shocks (ARCH), and β captures volatility persistence (GARCH).

Kelly Criterion-Inspired Position Sizing. The trading position scales with forecasted return and estimated volatility following a Kelly-style structure (Kelly, 1956; Thorp, 2006):

$$\text{Position}_t = \frac{\hat{r}_{t+1}}{\hat{\sigma}_t^2} \cdot \frac{\sigma_{\text{target}}}{\hat{\sigma}_t}, \tag{IA4.13}$$

where $\hat{r}_{t+1}$ is the ARIMA forecast, $\hat{\sigma}_t$ is annualized GARCH conditional volatility, and $\sigma_{\text{target}} = 0.15$ (15% annualized) is the volatility target. The first factor $\hat{r}_{t+1}/\hat{\sigma}_t^2$ corresponds to a Kelly-*inspired* sizing term, while the second factor rescales exposure to target a fixed volatility level. Note that this formulation is not strictly Kelly-optimal.

Since expected returns are forecast at a daily frequency while volatility is annualized, this formulation implicitly shrinks position sizes relative to a unit-consistent daily Kelly rule, providing a conservative adjustment for estimation error in short-horizon forecasts.

Financial Interpretation. The volatility adjustment changes behavior in several important ways:

1. **Volatility normalization**: Positions shrink in high-volatility environments, preventing oversized exposure during turbulent periods. A 1% expected return is not equivalent when daily volatility is 0.5% versus 3%.
2. **Risk-adjusted returns**: The sizing term resembles a Sharpe-like scaling, which aligns the signal more closely with utility-based portfolio choices.
3. **Dynamic position sizing**: Higher uncertainty leads naturally to smaller effective positions, reducing exposure precisely when risk is elevated.
4. **Volatility targeting**: The approach aims to keep risk exposure roughly stable at 15% annualized volatility across regimes.

IA4.3.2 Implementation Details

Table IA3: Risk-Adjusted ARIMA Hyperparameters

Parameter	Value	Justification
ARIMA order	$(p, d, q) \leq (3, 2, 3)$	Automatic selection via AIC
GARCH order	$(1, 1)$	Standard specification for daily data
Volatility target	15% annualized	Common institutional target
Max position size	1.0	Full investment constraint
Log returns	Yes	Ensures additivity and normality

Algorithm:

1. Compute log returns: $r_t = \ln(P_t/P_{t-1})$.
2. Fit ARIMA(p, d, q) model on training window returns (order selected by AIC).
3. Extract ARIMA residuals: $\epsilon_t = r_t - \hat{r}_t$.
4. Fit GARCH(1,1) model on residuals to obtain conditional volatility $\hat{\sigma}_t$.
5. Generate return forecast: $\hat{r}_{t+1}$ from ARIMA model.
6. Compute Kelly-inspired position size:
   - Base position: $p_{\text{base}} = \hat{r}_{t+1}/\hat{\sigma}_t^2$.

- Volatility adjustment: $p_{\text{adj}} = p_{\text{base}} \cdot (\sigma_{\text{target}}/\hat{\sigma}_t)$.
- Final position: $p_t = \text{clip}(p_{\text{adj}}, -1, 1)$.

7. Convert continuous position to trading signal:
   - Returns-optimized: Kelly criterion-inspired positioning(Eq IA4.13).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: In contrast to standard ARIMA, which incorporates 8 external regressors, Risk-Adjusted ARIMA uses only the return series itself. Volatility information is introduced through the GARCH model fitted to residuals, rather than via external regressors inside the forecasting equation. This structure allows independent modeling of expected returns and risk.

Practical Notes: This risk adjustment adjusts the behavior in stressed markets. In periods such as the 2008 financial crisis or the March 2020 COVID crash, a plain ARIMA model can produce large positive forecasts during short rebounds, which may translate into long exposure. Volatility scaling reduces those positions when volatility increases. In stable markets, the same mechanism increases exposure, so even moderate forecasts can turn into strong signals when risk-adjusted opportunities are more favorable.

IA4.4 Linear Regression

IA4.4.1 Theoretical Foundation

Linear regression is a classical technique in quantitative finance and econometrics, widely used for interpreting factor exposures and forecasting returns (Shalev-Shwartz and Ben-David, 2014). Although it is sometimes grouped with machine learning, linear regression is fundamentally a statistical/econometric method that existed much before modern ML. We therefore place it in this section to reflect its basis in statistical inference, closed-form solutions, and well-studied asymptotic behavior—features that differentiate it from more iterative ML approaches.

In trading contexts, linear models forecast future returns from technical indicators (such as moving averages and momentum), price-derived signals (such as RSI and Bollinger Bands), and volatility measures.

$$r_{t+1} = \alpha + \sum_{i=1}^{k} \beta_i X_{i,t} + \epsilon_{t+1}, \tag{IA4.14}$$

where r_{t+1} denotes future returns (typically log returns for stationarity), $X_{i,t}$ are predictors at time t (including technical indicators, and volatility variables), α is an intercept term (expected return when predictors are zero), β_i are coefficients capturing sensitivity to each predictor, and ϵ_{t+1} represents unexplained variation.

The OLS estimator minimizes squared residuals and yields:

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \mathbf{r}, \tag{IA4.15}$$

Financial Interpretation. Each coefficient β_i has an immediate interpretation: it measures the change in expected return for a one-unit change in the corresponding predictor while holding other predictors fixed. For example, a positive momentum coefficient suggests continuation in winners, while a negative coefficient on a price-to-moving-average ratio is consistent with mean reversion.

Linear regression is transparent and interpretable but also has a limitation. Linear models cannot represent many non-linear effects that appear in markets, such as asymmetric volatility impacts or threshold-type behavior where predictors matter only beyond certain levels.

IA4.4.2 Implementation Details

Table IA4: Linear Regression Hyperparameters

Parameter	Value	Justification
Method	OLS	Baseline without regularization
Training window	5 years	Standard walk-forward window
Signal threshold	0.005	Minimum forecast for position
OOF calibration	Isotonic regression	Improves prediction reliability
Adaptive thresholds	Enabled	Volatility-scaled signal generation

Feature Set: The model uses roughly 20 OHLC-based technical indicators:

- **Price/Trend**: Moving averages (5, 10, 20, 50-day), price-to-MA ratios, 52-week price position
- **Momentum**: RSI (14-day), rate of change (5, 10, 20-day), MACD (line, signal, histogram)
- **Volatility**: Rolling standard deviation (5, 20, 60-day), Bollinger Bands (upper, lower, middle, width, position), price high-low ratio

Algorithm:

1. Standardize features to zero mean and unit variance.
2. Collect out-of-fold (OOF) predictions (Section IA9.3) using 5-fold time-series cross-validation (Section IA9.2).
3. Fit isotonic regression calibrator on OOF predictions (Section IA9.4).
4. Fit final OLS regression on full 5-year training window.
5. Generate 1-step ahead forecast: $\hat{r}_{t+1} = \hat{\alpha} + \sum_i \hat{\beta}_i X_{i,t}$.
6. Apply isotonic calibration to improve prediction reliability.
7. Convert forecast to signal:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

OOF Isotonic Calibration: To improve forecast reliability, we apply isotonic regression calibration trained on out-of-fold predictions. During training, 5-fold time-series cross-validation produces

predictions where each fold is predicted by a model that was not trained on that fold. An isotonic (monotonic) mapping is then fitted to transform raw predictions into calibrated forecasts, correcting systematic bias without introducing leakage.

Practical Notes: Linear regression provides the baseline for the regularized variants (Ridge, Lasso, Elastic Net). Because OLS can become unstable when predictors are correlated, its coefficients may show high variance out-of-sample. With a 5-year training window, there are roughly 1,260 observations, which is typically enough for stable estimation with around 15–20 features. The isotonic calibration step further helps by correcting consistent bias patterns in the raw model outputs.

IA4.5 Kalman Filter

IA4.5.1 Theoretical Foundation

The Kalman Filter is a recursive algorithm for estimating unobserved states that evolve over time and are observed through noisy measurements (Kalman, 1960; Kalman and Bucy, 1961; Harvey, 1989). In finance, it is commonly applied to smooth noisy return series, estimate time-varying parameters, and construct forecasting models that adapt gradually as market conditions change.

In contrast to discrete regime-switching models, the Kalman Filter assumes that the latent state evolves continuously over time. Model parameters therefore drift rather than switching abruptly between a small number of regimes, which is often a more realistic description of financial markets.

A state-space model of the Kalman filter assumes linear dynamics under Gaussian noise assumption. It is defined by the following two equations.

State transition equation:

$$\beta_t = F\beta_{t-1} + \eta_t, \quad \eta_t \sim \mathcal{N}(0, Q). \tag{IA4.16}$$

Observation equation:

$$r_t = H_t\beta_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, R). \tag{IA4.17}$$

The Kalman Filter produces estimates of the latent state sequentially over time. The notation $t \mid s$ is used to indicate that an estimate at time t is constructed using only information that is available up to time s. In particular, quantities indexed by $t \mid t-1$ are based solely on information observed up to the previous period, while quantities indexed by $t \mid t$ incorporate the current observation r_t.

Prediction step:

$$\hat{\beta}_{t|t-1} = F\hat{\beta}_{t-1|t-1}, \tag{IA4.18}$$

$$P_{t|t-1} = FP_{t-1|t-1}F^T + Q. \tag{IA4.19}$$

The predicted state $\hat{\beta}_{t|t-1}$ therefore represents the model's best guess of the state at time t before observing the return r_t, while $P_{t|t-1}$ measures the uncertainty associated with this prediction.

Update step:

$$K_t = P_{t|t-1}H_t^T \left(H_tP_{t|t-1}H_t^T + R\right)^{-1}, \tag{IA4.20}$$

$$\hat{\beta}_{t|t} = \hat{\beta}_{t|t-1} + K_t \left(r_t - H_t\hat{\beta}_{t|t-1}\right), \tag{IA4.21}$$

$$P_{t|t} = (I - K_tH_t)\, P_{t|t-1}. \tag{IA4.22}$$

After observing r_t, the filter updates the state estimate to $\hat{\beta}_{t|t}$. The associated covariance matrix $P_{t|t}$ reflects the remaining uncertainty after incorporating the new information.

The matrix $P_{t|s}$ is the conditional covariance matrix of the state estimation error,

$$P_{t|s} = \mathbb{E}\Big[(\beta_t - \hat{\beta}_{t|s})(\beta_t - \hat{\beta}_{t|s})^T \mid \text{information available up to time } s\Big],$$

and provides a quantitative measure of how uncertain the estimate of β_t is at each point in time.

Financial interpretation. The Kalman gain K_t governs the trade-off between the model prediction and the new observation. When the predicted uncertainty $P_{t|t-1}$ is large, the filter responds more strongly to the incoming data. When observation noise R is large, updates are more conservative. This adaptive behavior allows the filter to track slow structural changes while remaining robust to short-term noise.

In trading applications, the filtered state $\hat{\beta}_{t|t}$ can be interpreted directly as a signal, while the covariance $P_{t|t}$ provides a real-time measure of estimation uncertainty that can be used for risk-aware position sizing. The two assumptions of linear dynamics and Gaussian noise are the limitations of the Kalman filter, although there are more general extensions available at the cost of increased complexity.

IA4.5.2 Implementation Details

Table IA5: Kalman Filter Hyperparameters

Parameter	**Value**	**Justification**
Transition coefficient (F)	1	Random walk assumption (univariate state)
Observation coefficient (H)	1	Direct observation of returns
Process noise variance (Q)	0.01	Low variance allows smooth state evolution
Observation noise variance (R)	1.0	Higher variance reflects noisy return observations
EM iterations	10	For parameter estimation

Algorithm:

1. **Initialization**: Set initial state estimate $\hat{\beta}_0 = 0$ and covariance $P_0 = 1.0$.
2. **Parameter Estimation**: Estimate Q and R using Expectation-Maximization (EM) on the training sample:
   - E-step: Run the Kalman smoother to infer latent states.
   - M-step: Update noise covariances based on smoothed residuals.
   - Iterate for 10 EM cycles or until convergence.
3. **Filtering**: For each new return observation r_t:
   - Predict: $\hat{\beta}_{t|t-1} = F\hat{\beta}_{t-1|t-1}$, $P_{t|t-1} = FP_{t-1|t-1}F^T + Q$.
   - Update: Compute K_t and update the state estimate.

4. **State Probability Mapping**: Convert filtered state $\hat{\beta}_{t|t}$ into soft regime probabilities (see below).
5. **Signal Generation**: Convert to trading signal:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: The Kalman Filter is applied directly to returns and does not use external predictors. In effect, the return series itself is the input, and the filter separates persistent structure from transient noise to generate a smoothed estimate. This makes it related to exponential smoothing (Section IA4.2), but the effective smoothing weights vary over time through the Kalman gain.

State-Space Interpretation: With $F = H = 1$, the model implies:

- The latent "true" return follows a random walk: $\beta_t = \beta_{t-1} + \eta_t$.
- Observed returns are noisy measurements of the state: $r_t = \beta_t + \epsilon_t$.
- The ratio $Q/R = 0.01$ indicates observation noise is $100\times$ larger than the state innovation scale.

This setup produces strong smoothing: the filter places more trust in its prior estimate than in any single return observation, while still allowing gradual adjustments when trends shift.

State Probability Mapping: The Kalman filter uses soft probabilities over three regimes (bear, neutral, bull), similar in spirit to the HMM method. This results in smoother position transitions and reduces abrupt switching. The mapping uses thresholds $\tau_{\text{upper}} = 0.002$ and $\tau_{\text{lower}} = -0.002$ as follows:

- **Bull territory** ($\hat{\beta}_{t|t} > \tau_{\text{upper}}$): Probability shifts toward bull state smoothly as the estimate rises above the threshold.
- **Bear territory** ($\hat{\beta}_{t|t} < \tau_{\text{lower}}$): Probability shifts toward bear state smoothly as the estimate falls below the threshold.
- **Neutral territory** ($\tau_{\text{lower}} \leq \hat{\beta}_{t|t} \leq \tau_{\text{upper}}$): Linear interpolation between states.

The final allocation is a probability-weighted combination:

$$w_t = 0.0 \cdot p_{\text{bear}} + 0.6 \cdot p_{\text{neutral}} + 1.0 \cdot p_{\text{bull}}. \tag{IA4.23}$$

This produces continuous weights in $[0, 1]$ that reflects regime confidence, reducing jumpiness of hard rule-based strategies while still reacting to meaningful trend changes.

Practical Notes: A key advantage of the Kalman Filter over simple moving-average smoothing is its explicit treatment of uncertainty. The Kalman gain K_t changes automatically: when state uncertainty is large relative to observation noise, the filter reacts more to new data; when observations are noisy relative to state uncertainty, it relies more on its prior. This adaptivity makes the method robust without manual tuning across regimes. Still, the linear Gaussian assumptions can be violated during crash periods, where returns may be fat-tailed and dynamics can become strongly non-linear.

IA4.6 Hidden Markov Model (HMM)

IA4.6.1 Theoretical Foundation

Hidden Markov Models (HMMs) provide a standard probabilistic framework for regime identification (Rabiner, 1989). HMMs are the canonical regime-switching approach in quantitative finance, and their neural variants exist (see Section IA6.1), we include the classical HMM here under statistical/econometric methods because it has established theory, closed-form inference procedures, and a transparent probabilistic structure.

The model assumes that observed returns are generated by an unobserved (hidden) state process that follows a first-order Markov chain—the current regime depends only on the previous regime, not on the entire past history. This memoryless structure keeps inference tractable while still capturing the empirical fact that regimes typically persist.

An HMM is defined by:

Transition Probabilities:

$$P(S_t = j|S_{t-1} = i) = a_{ij}. \tag{IA4.24}$$

Emission Probabilities: Under Gaussian emissions, each state produces returns from its own distribution:

$$P(r_t|S_t = i) = \mathcal{N}(\mu_i, \sigma_i^2), \tag{IA4.25}$$

where μ_i and σ_i^2 are state-specific mean and variance. In financial terms, a "bull" state would typically correspond to $\mu > 0$ with moderate σ, while a "crisis" state could correspond to $\mu < 0$ with high σ.

Model parameters are estimated using the Baum-Welch algorithm (an EM procedure), and the Viterbi algorithm is used to recover the most likely hidden-state path given the observed return sequence.

Financial Interpretation. HMMs can be used to identify regimes such as "risk-on" versus "risk-off," to adjust exposure based on volatility-driven states, and to generate forecasts conditional on the inferred regime. A key benefit is that regime characteristics are learned directly from the data, rather than being imposed via hand-chosen thresholds.

At the same time, HMMs usually assume fixed transition probabilities and fixed emission distributions, which can limit flexibility if the regime dynamics themselves shift structurally over time. They can also have difficulty representing slow transitions, since regimes are modeled as discrete jumps between states.

IA4.6.2 Implementation Details

Feature Set:

- Daily returns: $r_t = (P_t - P_{t-1})/P_{t-1}$
- Rolling volatility: 20-day rolling standard deviation

Algorithm:

1. Prepare features: compute daily returns and 20-day rolling volatility.

Table IA6: HMM Hyperparameters

Parameter	Value	Justification
Number of states	3	Bear/Neutral/Bull regime taxonomy
Covariance type	Diagonal	Computational efficiency
Max iterations	100	Sufficient for EM convergence
Min state spacing	10 days	Prevents regime whipsaws

2. Fit Gaussian HMM using Baum-Welch (EM) algorithm for 100 iterations.
3. Identify state characteristics by computing mean return per state.
4. Create state mapping: sort states by mean return (bear=lowest, neutral=middle, bull=highest).
5. For prediction: compute state probabilities $P(S_t = s|r_{1:t})$ using forward algorithm.
6. Apply minimum state spacing filter (10 days) to prevent regime whipsaws.
7. Calculate weighted position using state probabilities (see below).
8. Convert to trading signal:
   - Returns-optimized: Static Threshold method (Section IA7.1.2).
   - SR-optimized: Sharpe-based method (Section IA7.2).

State Mapping: After estimation, states are ranked by their mean return so that they can be labeled consistently as bear (lowest mean), neutral (middle mean), and bull (highest mean). This removes dependence on the arbitrary labeling produced by EM.

State Probability Mapping: Exposure is determined using soft state probabilities instead of hard regime assignments, which leads to smoother transitions:

$$w_t = \sum_{s \in \{0,1,2\}} P(S_t = s|r_{1:t}) \cdot \text{pos}_s, \tag{IA4.26}$$

where $\text{pos}_s \in \{0.0, 0.6, 1.0\}$ correspond to bear, neutral, and bull states. This produces continuous weights in $[0, 1]$ that represent confidence across regimes.

Minimum State Spacing: To reduce excessive switching and regime "whipsaws," the implementation enforces a minimum holding time of 10 days in any state before a transition is allowed. This lowers turnover from noisy state estimates while still allowing the model to react to real regime shifts.

Practical Notes: The HMM provides an interpretable regime classification mechanism. Unlike methods that rely mainly on adaptive thresholds, the HMM uses probabilistic state inference to drive position sizing, with static thresholds applied only at the final discretization step. Using 3 states captures the standard bear/neutral/bull structure while keeping estimation manageable.

IA5 Classical Machine Learning Models

In this section, we describe the classical machine learning approaches that explore cross-sectional information through technical indicators and engineered features. Unlike the statistical and econometric models, most of these methods reply on iterative optimization algorithms, explicit hyperparameter tuning via cross-validation, and prioritize predictive accuracy over interpretability. Most of these methods are widely adopted in quantitative trading because they are flexible, computationally efficient, and capable of modeling non-linear relationships.

IA5.1 Ridge Regression

IA5.1.1 Theoretical Foundation

Linear regression (Section IA4.4) suffers from instability in the financial settings, where multicollinearity and overfitting are pervasive due to heavy reuse of similar technical indicators (e.g., overlapping moving averages or momentum measures). In order to overcome this, regularized linear models extend ordinary least squares by penalizing coefficient magnitudes (Hoerl and Kennard, 1970; Tibshirani, 1996; Zou and Hastie, 2005).

The ridge regression applies an L2 penalty that shrinks coefficients towards zero:

$$\hat{\boldsymbol{\beta}}^{\text{ridge}} = \arg\min_{\boldsymbol{\beta}} \left\{ \sum_{t=1}^{T} (r_t - X_t^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}. \tag{IA5.1}$$

This yields the closed-form solution:

$$\hat{\boldsymbol{\beta}}^{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T \mathbf{r}, \tag{IA5.2}$$

where $\lambda \geq 0$ controls the strength of regularization. Unlike Lasso, the L2 penalty does not force coefficients exactly to zero, so all predictors remain in the model.

Financial Interpretation. Ridge regression performs well even when predictors are highly correlated. The L2 penalty stabilizes the inversion of $X^T X$, which can become ill-conditioned under multicollinearity. In practice, this is common when using families of related indicators, such as moving averages at different horizons or multiple momentum measures.

Rather than arbitrarily favoring one correlated predictor, ridge distributes weight across them, producing smoother and more stable forecasts out-of-sample. Although this introduces some bias, the reduction in variance generally improves predictive performance in low signal-to-noise environments typical of financial returns.

IA5.1.2 Implementation Details

Algorithm:

1. Standardize all features to zero mean and unit variance.
2. Select λ using 5-fold time-series cross-validation (Section IA9.2), collecting out-of-fold predictions (Section IA9.3).

3. Fit isotonic regression calibrator on OOF predictions (Section IA9.4).
4. Fit ridge regression on the full training window: $\hat{\boldsymbol{\beta}} = (X^T X + \lambda I)^{-1} X^T \mathbf{r}$.
5. Generate forecasts and apply isotonic calibration: $\hat{r}_{t+1} = f_{\text{iso}}(\hat{\alpha} + \sum_i \hat{\beta}_i X_{i,t})$.
6. Convert calibrated forecasts into trading signals:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Table IA7: Ridge Regression Hyperparameters

Parameter	**Value**	**Justification**
Regularization λ	CV-optimized	Selected via time-series CV from $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
CV splits	5	Time-series cross-validation
Calibration	Isotonic	Out-of-fold calibration for improved reliability

Feature Set: Identical to the linear regression feature set (Section IA4.4), including moving averages, price-to-MA ratios, momentum indicators (ROC, RSI), volatility metrics (rolling standard deviation, Bollinger Bands), and trend indicators such as MACD. Ridge regression is particularly appropriate given the strong correlations within this set.

Calibration Procedure: Full details of isotonic calibration using out-of-fold predictions are provided in Section IA9.4.

Practical Notes: In most financial forecasting tasks, ridge regression outperforms plain OLS due to its favorable bias–variance trade-off. Its closed-form solution keeps computation fast even with moderately large feature sets. Importantly, λ is always re-selected via cross-validation for each training window rather than fixed globally, allowing the model to adapt to changing market conditions.

IA5.2 Lasso Regression

IA5.2.1 Theoretical Foundation

The Lasso (Least Absolute Shrinkage and Selection Operator) employs L1 regularization, which encourages sparsity by driving some coefficients exactly to zero (Tibshirani, 1996). Unlike ridge regression (Section IA5.1), Lasso simultaneously regularizes and performs feature selection:

$$\hat{\boldsymbol{\beta}}^{\text{lasso}} = \arg\min_{\boldsymbol{\beta}} \left\{ \sum_{t=1}^{T} (r_t - X_t^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \tag{IA5.3}$$

Financial Interpretation. Financial datasets often contain many potential predictors, while only a subset carries genuine predictive content. Lasso is appealing in this setting because it automatically filters out irrelevant features. Sparse models are also easier to interpret and monitor operationally.

However, when predictors are strongly correlated, Lasso can be unstable. It tends to select one variable from a correlated group and drop the rest, and the choice can vary across training windows. For example, among multiple momentum horizons that convey similar information, Lasso may alternate which one it selects, leading to inconsistent factor exposures over time.

IA5.2.2 Implementation Details

Table IA8: Lasso Regression Hyperparameters

Parameter	**Value**	**Justification**
Regularization λ	CV-optimized	Selected via time-series CV from $\{0.0001, 0.001, 0.01, 0.1, 1.0\}$
CV splits	5	Time-series cross-validation
Calibration	Isotonic	Out-of-fold calibration for improved reliability

Algorithm:

1. Standardize features to ensure comparable L1 penalties.
2. Select λ via 5-fold time-series cross-validation (Section IA9.2), collecting out-of-fold predictions (Section IA9.3).
3. Fit isotonic calibration on OOF predictions (Section IA9.4).
4. Fit Lasso using coordinate descent on the full training window (maximum 10,000 iterations).
5. Record non-zero coefficients for interpretability analysis.
6. Generate forecasts and apply isotonic calibration.
7. Convert calibrated forecasts into signals:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Same OHLC-derived indicator set as Linear Regression (Section IA4.4). Lasso is particularly useful when only a small subset of these indicators is expected to be informative.

Sparsity Monitoring: The count of non-zero coefficients is tracked across training windows to assess the stability of feature selection. Predictors that remain selected across many windows are considered more robust.

Calibration: Refer to Section IA9.4. Calibration is especially important for Lasso because coefficient shrinkage tends to compress prediction magnitudes, leading to systematic underestimation.

Practical Notes: Lasso's automatic feature selection improves interpretability but can introduce instability when predictors are correlated. The coordinate descent solver converges quickly, making Lasso computationally efficient. Adaptive thresholds are scaled down by a factor of 10 (0.0003/0.00015 versus 0.003/0.0015) to compensate for the reduced forecast magnitudes induced by L1 shrinkage.

IA5.3 Elastic Net

IA5.3.1 Theoretical Foundation

Elastic Net blends the L1 penalty of Lasso (Section IA5.2) with the L2 penalty of Ridge regression (Section IA5.1) (Zou and Hastie, 2005):

$$\hat{\boldsymbol{\beta}}^{\text{elastic}} = \arg\min_{\boldsymbol{\beta}} \left\{ \sum_{t=1}^{T} (r_t - X_t^T \boldsymbol{\beta})^2 + \lambda \left[\alpha \sum_{j=1}^{p} |\beta_j| + (1-\alpha) \sum_{j=1}^{p} \beta_j^2 \right] \right\}, \tag{IA5.4}$$

where $\alpha \in [0, 1]$ controls the relative contribution of the L1 and L2 penalties. A value of $\alpha = 0.5$ assigns equal weight to both.

Financial Interpretation. Elastic Net mitigates the instability of Lasso in the presence of correlated predictors while preserving sparsity. The L2 component encourages grouped selection of correlated indicators, while the L1 component still eliminates irrelevant features. This makes Elastic Net particularly suitable for technical indicator sets with moderate to strong correlations.

IA5.3.2 Implementation Details

Table IA9: Elastic Net Hyperparameters

Parameter	**Value**	**Justification**
Regularization λ	CV-optimized	Selected from $\{0.0001, 0.001, 0.01, 0.1, 1.0\}$
L1 ratio α	CV-optimized	Selected from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$
CV splits	5	Time-series cross-validation
Calibration	Isotonic	Out-of-fold calibration for improved reliability

Algorithm:

1. Standardize features.
2. Jointly select λ and α via time-series cross-validation (Section IA9.2), collecting OOF predictions (Section IA9.3).
3. Fit isotonic calibrator on OOF predictions (Section IA9.4).
4. Fit Elastic Net using coordinate descent with warm starts.
5. Generate calibrated forecasts.
6. Convert calibrated forecasts into signals:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Same OHLC-derived technical indicators as Linear Regression. In correlated feature environment, which is generally the case in financial time series, Elastic net is a good choice as it is a combination of stability due to ridge component and sparsity due to Lasso component.

Parameter Selection: Both λ and α are selected via grid search using time-series cross-validation. The combination minimizing cross-validated MSE is chosen. See Section IA9.5 for full validation details.

Practical Notes: Elastic Net often provides the best balance among linear regularized models for financial data. While it introduces an extra hyperparameter, the resulting stability and interpretability justify the added complexity. As with Lasso, adaptive thresholds are reduced by $10\times$ to account for coefficient shrinkage.

IA5.4 PCA Regression

IA5.4.1 Theoretical Foundation

Principal Component Analysis (PCA) regression mitigates multicollinearity by first transforming the original predictors into a set of orthogonal principal components and then performing regression on these components (Pearson, 1901; Hotelling, 1933). This is especially useful in financial applications, where large collections of technical indicators are often highly correlated, causing standard regression methods to become unstable.

Let the feature matrix be $X \in \mathbb{R}^{T\times p}$, where T denotes the number of observations and p the number of predictors:

$$\Sigma = \frac{1}{T}X^T X = V\Lambda V^T, \tag{IA5.5}$$

where V contains the eigenvectors of the covariance matrix and Λ is a diagonal matrix of eigenvalues. The first k eigenvectors correspond to the directions of highest variance, and projecting onto them yields:

$$Z = XV_k. \tag{IA5.6}$$

Regression is then carried out using these reduced features:

$$r_{t+1} = \alpha + \sum_{j=1}^{k} \gamma_j Z_{j,t} + \epsilon_{t+1}. \tag{IA5.7}$$

Financial Interpretation. PCA regression offers several practical benefits in financial settings. By removing multicollinearity, it stabilizes estimation that would otherwise be problematic for ordinary least squares. Reducing the dimensionality of the predictor space also acts as an implicit form of regularization, helping to control model complexity and limit overfitting.

In some cases, the resulting components have intuitive interpretations. For equity data, the first principal component often reflects broad market exposure, while later components may be linked to volatility or momentum effects. PCA assumes that direction of largest variance is most informative for prediction. However, this is not generally true. Predictive signals can reside in lower-variance components that are discarded. Additionally, the eigen-decomposition can be sensitive to outliers, making careful preprocessing important.

Table IA10: PCA Regression Hyperparameters

Parameter	Value	Justification
Principal components k	CV-optimized	Selected from $\{3, 5, 10, 15, 20, 30\}$ via TSCV
CV splits	5	Time-series cross-validation
Standardization	Yes	Necessary for meaningful PCA results
Calibration	Isotonic	Out-of-fold calibration to improve reliability

IA5.4.2 Implementation Details

Algorithm:

1. Standardize all input features with zero mean and unit variance.
2. Select the optimal number of components k using 5-fold time-series cross-validation (Section IA9.2), collecting out-of-fold predictions (Section IA9.3).
3. Fit an isotonic regression calibrator on the OOF predictions (Section IA9.4).
4. Perform PCA with the chosen k components on the full training dataset.
5. Project the original features into the principal component space:

$$Z = XV_k.$$

6. Fit an OLS regression model using the transformed features.
7. Generate a one-step-ahead forecast and apply isotonic calibration:

$$\hat{r}_{t+1} = f_{\text{iso}}(\hat{\alpha} + \sum_{j=1}^{k} \hat{\gamma}_j Z_{j,t}).$$

8. Convert the calibrated forecast into a trading signal:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1)
   - SR-optimized: Sharpe-based method (Section IA7.2)

Feature Set: The model uses the same OHLC-derived technical indicators as Linear Regression (Section IA4.4), including momentum ratios, volatility metrics, and trend indicators. PCA is particularly useful in this context because many of these indicators exhibit strong multicollinearity, such as closely related moving-average ratios.

Component Selection: Rather than retaining components based on a fixed explained-variance threshold (e.g., 90%), this implementation selects the number of components through cross-validation, using out-of-sample prediction error as the criterion. This avoids assuming that high-variance directions are necessarily the most predictive.

Calibration: A detailed description of the isotonic calibration procedure is provided in Section IA9.4. The calibration step learns a monotonic mapping from raw model outputs to calibrated forecasts using out-of-fold predictions.

Practical Notes: Principal components are recomputed at each rolling training window to reflect evolving correlation structures in the data. While the leading component often represents broad market movement, the interpretation of later components can change over time, complicating ex-post attribution. The explained variance ratios are recorded for monitoring purposes but are not used to determine k; instead, cross-validated mean squared error drives component selection.

IA5.5 K-Nearest Neighbors

IA5.5.1 Theoretical Foundation

The K-Nearest Neighbors (KNN) approach performs forecasting through non-parametric pattern matching rather than by fitting a global model (Fix and Hodges, 1989; Cover and Hart, 1967). Instead of assuming a specific functional relationship between predictors and returns, the method searches for historical market situations that resemble the current state and forms a prediction by averaging the outcomes that followed those similar cases. This reflects the common intuition that while markets do not repeat exactly, similar conditions often lead to comparable outcomes.

Given a query point x_t representing the current feature vector, the algorithm identifies the k closest historical observations according to a distance metric $d(\cdot, \cdot)$:

$$\mathcal{N}_k(x_t) = \{x_{(1)}, x_{(2)}, \ldots, x_{(k)}\}, \tag{IA5.8}$$

where the neighbors are ordered by increasing distance $d(x_t, x_{(i)})$. The return forecast is then computed as a weighted average of the subsequent returns associated with these neighbors:

$$\hat{r}_{t+1} = \sum_{i=1}^{k} w_i \cdot r_{(i)+1}, \tag{IA5.9}$$

where $r_{(i)+1}$ denotes the return observed after neighbor $x_{(i)}$, and the weights w_i may be uniform or inversely proportional to distance.

Financial Interpretation. KNN offers several appealing properties for financial forecasting. Because it does not impose a parametric structure, it can capture complex and highly non-linear relationships between indicators and returns. Its local nature allows different regions of the feature space to exhibit distinct predictive behavior. Importantly, KNN implicitly conditions on market regimes: by focusing only on historically similar states, the model naturally adapts its forecasts to the prevailing market environment.

The main drawbacks are the curse of dimensionality and computational expense. The distances between points become less informative as the number of features grow. This weakens the notion of nearest neighbors. As a result, careful feature selection and, in some cases, dimensionality reduction (for example via PCA, Section IA5.4) are critical for effective use of KNN. The choice of k governs the bias–variance trade-off: small values of k emphasize local structure but can overfit noise, while larger values yield smoother predictions that may overlook regime-specific effects.

IA5.5.2 Implementation Details

Algorithm:

Table IA11: KNN Hyperparameters

Parameter	Value	Justification
Neighbors (K)	CV-optimized from {3, 5, 7, 10, 15}	Time-series CV selects K to balance local pattern capture and noise sensitivity
Distance metric	Euclidean	Standard choice for normalized feature spaces
Weighting	CV-optimized from {uniform, distance}	Distance weighting assigns greater influence to closer neighbors
CV splits	3	Time-series cross-validation for hyperparameter tuning

1. Standardize all features to zero mean and unit variance, which is essential for meaningful distance calculations.
2. **Time-Series CV Optimization:** Perform a grid search over $K \in \{3, 5, 7, 10, 15\}$ and weighting schemes in {uniform, distance} using 3-fold time-series cross-validation (Section IA9.2), selecting the configuration that minimizes MSE.
3. **OOF Calibration:** Collect out-of-fold predictions during cross-validation and fit an isotonic regression calibrator (Section IA9.4).
4. For each test observation x_t, compute the Euclidean distance to all observations in the training set.
5. Identify the K nearest neighbors based on the CV-selected parameters.
6. Retrieve the next-day returns $r_{(i)+1}$ associated with each neighbor.
7. Compute the raw forecast using the selected weighting scheme (either uniform averaging or distance-weighted averaging).
8. Apply isotonic calibration to the raw predictions.
9. Convert the calibrated forecast into a trading signal using adaptive thresholds (Section IA7.1.1).

Feature Set: The model uses a carefully selected subset of eight distance-friendly features tailored for KNN: Price_MA_5_Ratio, Price_MA_10_Ratio, Price_MA_20_Ratio, Momentum_5, Momentum_10, Momentum_20, Volatility_20, and RSI. The curse of dimensionality does not really effect as the number of features we use are limited. Standardization is crucial, as features with larger scales would otherwise dominate the distance computation.

Practical Notes: The computational cost of the algorithm is $O(n \cdot d)$, where n is the number of training samples and d is the feature dimension. With the 5-year rolling training windows used here (approximately 1260 trading days), this cost remains manageable. The cross-validated choices of K and weighting strike a balance between capturing local market structure and avoiding excessive sensitivity to noise: smaller K values emphasize localized patterns but are noisier, while larger K values yield smoother yet potentially less responsive forecasts.

Out-of-fold isotonic calibration (Section IA9.3) improves forecast reliability by mapping raw neighbor-averaged predictions to calibrated return estimates. Trading signals are generated using standard

adaptive thresholds (Section IA7.1.1), with base upper and lower thresholds of 0.003 and 0.0015, respectively.

IA5.6 Support Vector Machine

IA5.6.1 Theoretical Foundation

Support Vector Machines (SVMs) overcome several shortcomings of purely linear models by using the *kernel trick*, which implicitly maps input features into a higher-dimensional space where complex, non-linear relationships can be represented as linear structures (Vapnik, 1995). SVMs model non-linear dependencies while retaining the benefits of convex optimization, such as a unique global optimum.

In the regression setting, SVM aims to find a function that deviates from observed returns by no more than a tolerance level ϵ, while remaining as flat (low-complexity) as possible:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \tag{IA5.10}$$

subject to:

$$r_i - w^T\phi(x_i) - b \leq \epsilon + \xi_i, \tag{IA5.11}$$

$$w^T\phi(x_i) + b - r_i \leq \epsilon + \xi_i^*, \tag{IA5.12}$$

$$\xi_i, \xi_i^* \geq 0. \tag{IA5.13}$$

The primal problem involves the mapping $\phi(x)$ which can be computationally intractable. The "kernel trick" emerges from the Lagrangian dual formulation, where the optimization depends only on inner products of the transformed features:

$$\max_{\alpha,\alpha^*} -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i, x_j) - \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} r_i(\alpha_i - \alpha_i^*), \tag{IA5.14}$$

subject to:

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0, \tag{IA5.15}$$

$$0 \leq \alpha_i, \alpha_i^* \leq C. \tag{IA5.16}$$

The dual variables α_i and α_i^* are Lagrange multipliers associated with the ϵ-insensitive loss constraints. Specifically, $\alpha_i > 0$ when observation i lies above the ϵ-tube (underpredicted by more than ϵ), while $\alpha_i^* > 0$ when observation i lies below the tube (overpredicted by more than ϵ). Points inside the tube have $\alpha_i = \alpha_i^* = 0$ and do not influence predictions. The box constraint $\alpha_i, \alpha_i^* \leq C$ limits the influence of any single observation, providing robustness to outliers.

The kernel function $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$ computes inner products in the transformed space without explicitly computing the (potentially infinite-dimensional) mapping ϕ. The prediction function becomes:

$$\hat{r}_{t+1} = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x_t) + b, \tag{IA5.17}$$

where only training points with non-zero $(\alpha_i - \alpha_i^*)$ (the "support vectors") contribute to predictions.

A commonly used kernel in financial applications is the Radial Basis Function (RBF):

$$K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2). \tag{IA5.18}$$

Financial Interpretation. The RBF kernel emphasizes local structure in the feature space: observations that are close in terms of their technical indicators exert stronger influence on each other's predictions. The parameter γ determines how localized this effect is. Large values of γ restrict influence to very similar historical patterns, while smaller values allow a broader range of past observations to contribute.

The regularization parameter C governs the trade-off between model complexity and tolerance for prediction errors. In noisy financial data, moderate values of C help avoid fitting spurious fluctuations. Although SVMs require careful hyperparameter tuning via cross-validation and can be computationally demanding, they offer a powerful and robust way to model non-linear relationships without explicitly engineering interaction terms.

IA5.6.2 Implementation Details

Table IA12: SVM Hyperparameters

Parameter	**Value**	**Justification**
Kernel	CV-optimized	Selected from {rbf, linear} via TSCV
Regularization (C)	CV-optimized	Selected from {0.1, 1.0, 10.0} via TSCV
CV splits	3	Time-series cross-validation
Calibration	Isotonic	Out-of-fold calibration for improved reliability

Algorithm:

1. Standardize all features to have zero mean and unit variance.
2. Choose the optimal kernel type and regularization parameter C using 3-fold time-series cross-validation (Section IA9.2), while collecting out-of-fold predictions (Section IA9.3).
3. Fit an isotonic regression calibrator on the OOF predictions (Section IA9.4).
4. Train the SVM with the selected hyperparameters on the full training dataset.
5. The optimization identifies support vectors, which define the ϵ-insensitive regression tube.
6. Generate predictions by computing kernel similarities to the support vectors:

$$\hat{r}_{t+1} = \sum_i \alpha_i K(x_t, x_i) + b.$$

7. Apply isotonic calibration to the raw predictions.
8. Convert calibrated forecasts into trading signals:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: In contrast to other classical machine learning models that use the full indicator set, the SVM relies on a **reduced subset of six features** to control computational cost:

- RSI (momentum oscillator)
- MACD (trend indicator)
- BB_Position (relative position within Bollinger Bands)
- Price_MA_20_Ratio (price relative to the 20-day moving average)
- Volatility_20 (20-day rolling volatility)
- Momentum_10 (10-day momentum)

This subset captures core momentum, trend, and volatility information while substantially reducing training time, which is important given the cubic complexity of SVM training.

Calibration: Refer to Section IA9.4 for a detailed discussion. Isotonic calibration improves forecast reliability by learning a monotonic correction using out-of-fold predictions.

Practical Notes: Training an SVM with an RBF kernel scales as $O(n^3)$ in the number of samples, making it significantly slower than linear models on large datasets. Using a reduced feature set (6 instead of roughly 20 features) and fewer cross-validation splits (3 instead of 5) helps keep computation manageable. For the 5-year rolling windows employed here (approximately 1,260 observations), training remains feasible but noticeably slower than Ridge or Lasso. As with Lasso and Elastic Net, SVM employs adaptive thresholds that are scaled down by a factor of 10 (base upper: 0.0003, base lower: 0.00015) to reflect the smaller magnitude of its predictions.

IA5.7 Decision Tree

IA5.7.1 Theoretical Foundation

Decision trees offer an interpretable, rule-based framework that closely resembles the if–then logic commonly used by discretionary traders (Breiman, 2001). The model recursively partitions the feature space using simple decision rules, producing explicit conditions similar to those found in traditional technical analysis. For instance, a rule might state: "If the 50-day moving average crosses above the 200-day moving average and volume exceeds its 30-day average, then predict positive returns."

At each node in the tree, the algorithm selects the feature and split point that minimize impurity:

$$\text{Split} = \arg\min_{j,t} \left[\frac{n_L}{n} \text{MSE}_L + \frac{n_R}{n} \text{MSE}_R \right], \tag{IA5.19}$$

where n_L and n_R denote the number of samples in the left and right child nodes, respectively, and MSE represents variance within each node. The tree is built by greedy splits that yield the largest reduction in prediction error.

Financial Interpretation. One of the main advantages of decision tree is its transparency. Each forecast can be traced back through a sequence of decision rules, making it straightforward to understand and justify model behavior—an important consideration for risk management and regulatory contexts. Trees also naturally capture threshold effects and feature interactions without requiring explicit interaction terms.

The main weakness is their tendency to overfit, particularly in noisy financial datasets. Small changes in the training data can lead to large differences in tree structure. Constraints such as limiting depth or requiring a minimum number of samples per node help, but single trees still exhibit high variance. Ensemble methods (Sections IA5.8 and IA5.9) are designed to address this issue.

IA5.7.2 Implementation Details

Table IA13: Decision Tree Hyperparameters

Parameter	**Value**	**Justification**
Maximum depth	CV-optimized	Selected from {3, 5, 10, 15, None} via TSCV
Min samples split	CV-optimized	Selected from {2, 5, 10} via TSCV
CV splits	3	Time-series cross-validation
Criterion	MSE	Standard regression objective

Algorithm:

1. Select optimal values for max_depth and min_samples_split using 3-fold time-series cross-validation (Section IA9.2).
2. Train the Decision Tree with the selected hyperparameters on the full training dataset.
3. Starting at the root, evaluate all candidate features and thresholds to compute MSE reduction.
4. Choose the split that minimizes weighted child-node MSE.
5. Recursively grow the tree until stopping criteria are met.
6. Assign each leaf node a prediction equal to the mean return of samples within that leaf.
7. For a new observation x_t, traverse the tree to the corresponding leaf and output its prediction.
8. Convert the forecast into a trading signal:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: The model uses the same OHLC-derived technical indicators as Linear Regression (Section IA4.4). Although decision trees are invariant to feature scaling, standardization is applied for consistency across models.

Practical Notes: Tree depth is cross-validated rather than fixed, allowing the model to adapt to varying market conditions. Shallow trees (depth 3–5) yield interpretable structures with roughly 8–32 leaf nodes, each corresponding to a distinct market scenario. Example rules might include:

"If RSI > 70 and volatility exceeds $1.5\times$ its average, predict negative returns." Unlike regularized regression models and SVMs, decision trees do not employ isotonic calibration, as their piecewise-constant outputs are already well-calibrated within each leaf. Despite their interpretability, individual trees remain high-variance models; ensemble methods such as Random Forests mitigate this limitation.

IA5.8 Random Forest

IA5.8.1 Theoretical Foundation

Random Forests build on decision trees (Section IA5.7) by using bootstrap aggregation, or bagging, to reduce variance (Breiman, 2001). A large number of trees are trained on bootstrap samples drawn with replacement from the training data, and each tree considers only a random subset of features at every split:

$$\hat{f}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(x), \tag{IA5.20}$$

where B is the total number of trees and $\hat{f}_b(x)$ denotes the prediction of tree b. Averaging across trees substantially lowers variance while preserving the ability to model non-linear relationships.

Financial Interpretation. Random Forests are well-suited for trading applications because they naturally handle diverse feature types and capture complex interactions without explicit feature engineering. The random feature selection at each split decorrelates the trees, making the ensemble more robust than simply averaging identical models. For example, the model can learn that momentum signals behave differently under high- versus low-volatility regimes.

Random Forests also provide feature importance measures, offering insight into which indicators drive predictions. However, they cannot extrapolate beyond the range of historical data; when the market enters an unseen regime, predictions are limited to interpolation within past observations.

IA5.8.2 Implementation Details

Table IA14: Random Forest Hyperparameters

Parameter	**Value**	**Justification**
Number of trees	CV-optimized	Selected from {50, 100, 200} via TSCV
Maximum depth	CV-optimized	Selected from {5, 10, 15, None} via TSCV
CV splits	3	Time-series cross-validation
Max features	sqrt(p)	Standard random subspace size

Algorithm:

1. Use 3-fold time-series cross-validation (Section IA9.2) to select the optimal number of trees and maximum depth.
2. Train the Random Forest with the selected hyperparameters:

(a) For each tree $b = 1, \ldots, B$, draw a bootstrap sample from the training data.

(b) Grow a decision tree on the bootstrap sample, considering only $\sqrt{p}$ randomly selected features at each split.

(c) Continue splitting until the maximum depth or minimum sample constraints are met.

3. Aggregate predictions across trees:

$$\hat{r}_{t+1} = \frac{1}{B} \sum_{b=1}^{B} T_b(x_t).$$

4. Convert the ensemble forecast into a trading signal:

   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Identical OHLC-derived technical indicators as Linear Regression (Section IA4.4). While Random Forests are largely insensitive to feature scaling, features are standardized for consistency.

Practical Notes: Random Forests provide an internal validation mechanism through out-of-bag (OOB) error estimation, where each tree is evaluated on samples not included in its bootstrap set. Feature importance scores, typically based on mean decrease in impurity, highlight influential indicators but tend to favor high-cardinality features and should be interpreted cautiously. Like decision trees, random forests do not apply isotonic calibration, as averaging across many trees already yields well-calibrated predictions.

IA5.9 Gradient Boosting

IA5.9.1 Theoretical Foundation

Gradient Boosting constructs an ensemble in a sequential manner, with each new model trained to correct the residual errors of the existing ensemble (Friedman, 2001). Instead of fitting independent learners and averaging them, the method incrementally refines predictions:

$$F_m(x) = F_{m-1}(x) + \nu \cdot h_m(x), \tag{IA5.21}$$

where F_m denotes the ensemble after m iterations, h_m is the m-th weak learner (typically a shallow decision tree), and ν is the learning rate that scales each update. Each weak learner is obtained by minimizing:

$$h_m = \arg\min_h \sum_{i=1}^{n} L(r_i, F_{m-1}(x_i) + h(x_i)). \tag{IA5.22}$$

Financial Interpretation. Gradient Boosting is particularly effective at uncovering complex, subtle patterns by iteratively focusing on observations that previous models predicted poorly. Using shallow trees (often depth 3–5) prevents individual learners from overfitting, while the cumulative effect of many small corrections yields a strong predictor.

The learning rate ν serves as a key regularization parameter. Smaller values slow down learning but generally improve robustness, requiring more trees to achieve similar performance. This gradual learning process reduces the influence of any single tree and allows the ensemble to generalize better. Modern implementations such as XGBoost and LightGBM introduce additional regularization mechanisms, further enhancing performance in noisy financial environments.

IA5.9.2 Implementation Details

Table IA15: Gradient Boosting Hyperparameters

Parameter	**Value**	**Justification**
Number of estimators	CV-optimized	Selected from {50, 100, 200} via TSCV
Learning rate (ν)	CV-optimized	Selected from {0.01, 0.05, 0.1} via TSCV
Maximum depth	CV-optimized	Selected from {3, 5} via TSCV
CV splits	3	Time-series cross-validation
Monotonic constraints	Enabled	Constraints based on financial intuition

Algorithm:

1. Specify monotonic constraints informed by financial reasoning.
2. Select optimal hyperparameters using 3-fold time-series cross-validation (Section IA9.2).
3. Train the HistGradientBoostingRegressor with the selected parameters and constraints:
   (a) Initialize predictions with a constant value: $F_0(x) = \bar{r}$.
   (b) For each iteration $m = 1, \ldots, M$, compute pseudo-residuals, fit a constrained tree, and update the ensemble.
4. Produce the final forecast: $\hat{r}_{t+1} = F_M(x_t)$.
5. Convert the forecast into a trading signal:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Monotonic Constraints: The implementation relies on sklearn's HistGradientBoostingRegressor, which supports monotonic constraints to encode domain knowledge:

- **Increasing (+1)**: Momentum and MACD features, reflecting the expectation that stronger momentum implies higher returns.
- **Decreasing (−1)**: RSI, capturing the idea that overbought conditions tend to precede lower returns.
- **Unconstrained (0)**: BB_Position, volatility, and other features without clear monotonic relationships.

Feature Set: The same OHLC-based technical indicators used in Linear Regression (Section IA4.4) are employed. Because gradient boosting incorporates regularization through shrinkage and tree structure, features are standardized to maintain consistency.

Practical Notes: The learning rate is selected via cross-validation from {0.01, 0.05, 0.1}. Smaller values lead to more conservative updates and generally improve robustness at the cost of additional trees. Limiting tree depth ensures that each weak learner captures only simple structure, forcing the ensemble to build complexity incrementally. The imposed monotonic constraints prevent economically implausible relationships—such as higher momentum predicting lower returns—thereby improving both stability and interpretability. As with decision trees and random forests, gradient boosting does not use isotonic calibration, since the sequential correction mechanism already yields well-calibrated predictions.

IA6 Neural Networks and Advanced Methods

This section discusses neural network–based models and more advanced techniques that extend beyond traditional approaches. The methods are grouped into three categories: neural network models, which focus on deep learning architectures for sequential data; neural regime-switching models, which combine regime identification with neural networks; and meta-learning and ensemble methods, which aim to adapt across tasks or aggregate multiple predictive models.

IA6.1 Neural HMM

IA6.1.1 Theoretical Foundation

The neural hidden Markov model (Neural HMM) generalizes the classical HMM framework (see Section IA4.6) by replacing simple parametric emission distributions with neural networks. While classical HMMs assume returns within each regime follow a fixed Gaussian distribution $\mathcal{N}(\mu_s, \sigma_s^2)$ with constant parameters, Neural HMMs use a neural network to compute regime-specific distribution parameters as a function of current market conditions. This preserves the probabilistic regime-switching structure while leveraging the representational power of deep learning (Bishop, 1994):

$$P(r_t \mid S_t = s) = \text{NeuralNet}_{\theta_s}(r_t, x_t), \tag{IA6.1}$$

where x_t represents an extended feature vector that goes beyond raw returns, including technical indicators and volatility-related measures. The neural network outputs state-conditional Gaussian parameters $(\mu_s(x_t), \sigma_s(x_t))$ that adapt to current market conditions, enabling the model to capture complex and highly non-linear relationships within each latent regime.

Financial Interpretation. Classical HMMs suffer from a critical limitation in financial applications: they assume returns within each regime follow a fixed Gaussian distribution. This means that once the model identifies "bull market regime," it expects returns to follow the same distribution regardless of whether volatility is at 10% or 30%, whether earnings season is approaching, or whether technical momentum is strong or weak.

Neural HMMs address this rigidity by making emission parameters conditional on market state. For example, within the "bull regime," the neural network might predict:

- Higher expected returns μ_{bull} when RSI shows momentum and volatility is low
- Lower expected returns but wider dispersion σ_{bull} when volatility spikes even though the regime hasn't switched

- Different parameters based on moving-average alignment or earnings calendar proximity

This hybrid structure combines the best of both approaches: regime inference still uses the classical forward–backward algorithm (which is efficient and theoretically well-understood), while emission distributions gain the flexibility to represent complex within-regime dynamics. In practice, this allows the model to respond more quickly to changing market conditions without requiring an explicit regime switch, which can be slow to detect.

Limitations. Neural HMMs retain the fundamental constraint of all regime-based models: they can only recognize regimes similar to those observed during training. When markets enter genuinely novel conditions—such as the initial COVID-19 crash in March 2020, which combined pandemic uncertainty with oil price shocks and unprecedented policy responses—the model may struggle to assign observations to any known regime, leading to unstable or unreliable predictions. The neural parameterization helps within known regimes but does not resolve the out-of-sample regime problem.

IA6.1.2 Implementation Details

Table IA16: Neural HMM Hyperparameters

Parameter	Value	Justification
Number of states	3	Bull, Bear, High-Volatility taxonomy captures primary market regimes
Sequence length	15 days	Approximately 3 trading weeks; balances context vs. stationarity
Emission network	[64, 32] units	Two-layer MLP per state; sufficient capacity without overfitting
Transition network	[32, 16] units	Learns state transition probabilities from features
Dropout rate	0.2	Regularization to prevent overfitting on noisy returns
Epochs	15	Limited training prevents memorization of noise
Regime threshold	0.6	Minimum posterior probability for confident regime assignment
Volatility window	20 days	Approximately 1 trading month for risk scaling

Why Three States? Financial markets are commonly described using a three-regime taxonomy: bull markets (positive drift, moderate volatility), bear markets (negative drift, elevated volatility), and high-volatility regimes (uncertain drift, extreme volatility often seen during crises). This classification aligns with practitioner intuition and is parsimonious enough to estimate reliably while capturing the primary modes of market behavior. More states would increase model complexity and risk overfitting, while fewer states would force the model to conflate economically distinct regimes.

Algorithm:

1. Initialize the HMM transition matrix and one neural emission network per latent state.

2. Train the model jointly via variational inference:
   - **E-step**: Compute state posteriors $P(S_t \mid r_{1:T}, x_{1:T})$ using the forward–backward algorithm with current neural emission parameters
   - **M-step**: Update neural network parameters θ_s via gradient descent to maximize expected log-likelihood under posterior state assignments
3. Each emission network outputs the Gaussian parameters (μ_s, σ_s) for its corresponding state s as a function of current features x_t.
4. During inference, forward probabilities $P(S_t \mid r_{1:t})$ are computed using the learned emissions.
5. Generate regime-specific return forecasts and combine them by weighting with posterior state probabilities:
$$\hat{r}_{t+1} = \sum_{s=1}^{3} P(S_t = s \mid r_{1:t}) \cdot \mu_s(x_t) \tag{IA6.2}$$
6. Map regime probabilities to portfolio positions based on confidence:
   - If regime confidence ≥ 0.6: Bull→1.0, Bear→0.0, High_Vol→0.0
   - If regime confidence < 0.6: apply return-based fallback (Equation IA6.3)

Feature Set. The input features consist of both raw return information and higher-level technical indicators:

- Daily returns and 20-day rolling volatility
- Momentum indicators: RSI (14-day) and MACD components
- Moving-average ratios: MA_20 and MA_50 relative to current price
- Regime-related signals: price Z-scores, Bollinger Band breakouts, moving-average crossovers, and candlestick engulfing patterns

Architecture Details. For each latent state $s \in \{1, 2, 3\}$, the emission model is implemented as a two-layer MLP with 64 and 32 hidden units, ReLU activations, and a dropout rate of 0.2. The network takes the feature vector x_t as input and outputs $(\mu_s, \log \sigma_s)$ to parameterize a Gaussian emission distribution for that state. Using $\log \sigma_s$ as output ensures positivity of the standard deviation.

A separate two-layer transition network with [32, 16] units learns state transition probabilities as a function of features, allowing transition dynamics to adapt to market conditions rather than remaining fixed.

Confidence-Based Position Mapping. A regime confidence threshold of 0.6 is applied to avoid trading on uncertain regime classifications. When the maximum posterior probability $\max_s P(S_t = s \mid r_{1:t}) \geq 0.6$, positions are assigned deterministically based on the most likely regime. When confidence is below this threshold, a fallback mechanism generates signals based on the expected return forecast relative to a threshold $\tau = 0.005$:

$$\text{Position} = \begin{cases} 1.0 & \text{if } \hat{r}_{t+1} > \tau, \\ 0.6 & \text{if } -\tau \leq \hat{r}_{t+1} \leq \tau, \\ 0.0 & \text{if } \hat{r}_{t+1} < -\tau. \end{cases} \tag{IA6.3}$$

The intermediate 0.6 position for neutral predicted returns (between $\pm\tau$) reduces whipsaw trading when neither regime classification nor return forecasts provide clear directional signals. This conservative stance acknowledges model uncertainty and limits turnover during ambiguous market conditions.

Practical Notes. Training is performed using the Adam optimizer with a learning rate of 0.001. Early stopping with a patience of 15 epochs and learning-rate reduction on plateau are applied to stabilize training. The objective function is the negative log-likelihood computed via the forward algorithm, which efficiently marginalizes over latent state sequences. Limiting training to 15 epochs helps prevent overfitting to noise in daily returns, while the 15-day sequence length provides approximately three weeks of historical context for regime inference without requiring the model to maintain stationarity over excessively long windows.

IA6.2 Long Short-Term Memory (LSTM)

IA6.2.1 Theoretical Foundation

Financial time series exhibit temporal dependencies that are difficult to capture with feedforward neural networks. Recurrent neural networks (RNNs) introduce hidden states to model such dependencies, but they often suffer from vanishing gradients when learning long-term relationships. Long short-term memory (LSTM) networks address this issue through gating mechanisms that explicitly regulate information flow (Hochreiter and Schmidhuber, 1997).

Forget Gate:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f). \tag{IA6.4}$$

Input Gate:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i). \tag{IA6.5}$$

Output Gate:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o). \tag{IA6.6}$$

Cell State Update:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tanh(W_C[h_{t-1}, x_t] + b_C). \tag{IA6.7}$$

Hidden State:

$$h_t = o_t \odot \tanh(C_t), \tag{IA6.8}$$

where σ denotes the sigmoid activation function, $\odot$ represents element-wise multiplication, and $[\cdot, \cdot]$ indicates vector concatenation.

Information Flow. The gates work together through a two-stage process:

Stage 1: Cell State Update. The cell state C_t aggregates past and present information:

$$C_t = \underbrace{f_t \odot C_{t-1}}_{\text{retained memory}} + \underbrace{i_t \odot \tilde{C}_t}_{\text{new information}}, \tag{IA6.9}$$

where $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$ is the candidate new information. The forget gate $f_t \in (0, 1)$ acts as a element-wise filter on previous memory—values near 0 erase that dimension, values near 1 preserve it. The input gate i_t similarly filters how much of each candidate dimension enters the cell state.

Stage 2: Hidden State Output. The hidden state h_t is a filtered view of the cell state:

$$h_t = o_t \odot \tanh(C_t). \tag{IA6.10}$$

The output gate o_t determines which dimensions of the (squashed) cell state are exposed as output. This hidden state h_t serves two purposes: (1) it feeds into the next time step as context, and (2) it connects to the dense layers for prediction.

The cell state C_t acts as a "memory highway"—information can flow across many time steps with minimal transformation (just multiplicative gating), avoiding the vanishing gradient problem of classical RNNs. The hidden state h_t is the "working memory" exposed at each step.

Financial Interpretation. The gating structure of the LSTM can be interpreted as a dynamic filtering system. The forget gate determines which historical information is no longer relevant, such as outdated momentum signals. The input gate decides whether new observations—like sudden price jumps—should be stored in memory. The output gate controls which parts of the internal state are exposed for prediction at each time step.

The cell state acts as a long-term memory channel, allowing information to persist across many periods with minimal degradation. This is particularly useful in financial contexts, where signals from several weeks ago (e.g., trend breaks or support violations) may still influence current market behavior. LSTMs naturally balance short-term noise and long-term structure, making them well suited for modeling financial time series with multiple temporal horizons.

IA6.2.2 Implementation Details

Algorithm:

1. Create OHLC-derived technical features from price data.
2. Construct input sequences: for each day t, create feature matrix $X_t \in \mathbb{R}^{L \times F}$ where $L = 15$ is the lookback window and F is the number of features.
3. Normalize features using StandardScaler fitted on training data.
4. Build sequential architecture and process sequences:
   - **LSTM layer (64 units):** For each sequence $X_t \in \mathbb{R}^{L \times F}$, the LSTM processes time steps $\ell = 1, \ldots, L$ sequentially. At each step, the gates aggregate information as described in the Information Flow section above:

$$h_\ell = \text{LSTM}(x_\ell, h_{\ell-1}, C_{\ell-1}) \quad \text{for } \ell = 1, \ldots, L. \tag{IA6.11}$$

Table IA17: LSTM Hyperparameters

Parameter	**Value**	**Justification**
Lookback window	15 days	Roughly three trading weeks of context
LSTM units	[64]	Single-layer model for efficiency
Dropout rate	0.2	Regularization against overfitting
Dense units	10	Output projection layer
Batch size	32	Standard mini-batch size
Epochs	20	With early stopping enabled
Learning rate	0.001	Default for Adam optimizer
Early stopping	10 epochs	Stops training once validation loss stagnates

The final hidden state $h_L \in \mathbb{R}^{64}$ summarizes the entire 15-day sequence.

- **Dropout (0.2):** Randomly set zeros for 20% of h_L during training to prevent overfitting.
- **Dense layer (10 units, ReLU):** Projects h_L to a 10-dimensional representation: $z = \text{ReLU}(W_1 h_L + b_1)$.
- **Output layer (1 unit, linear):** Produces the return prediction: $\hat{r}_{t+1} = W_2 z + b_2$.

5. Train using MSE loss with Adam optimizer (learning rate 0.001)
6. Apply early stopping (patience 10) and learning rate reduction on plateau.
7. Generate return predictions $\hat{r}_{t+1}$ from the final dense layer.
8. Convert predictions to positions using threshold $\tau = 0.005$:

$$\text{Position} = \begin{cases} 1.0 & \text{if } \hat{r}_{t+1} > \tau, \\ 0.6 & \text{if } -\tau \leq \hat{r}_{t+1} \leq \tau, \\ 0.0 & \text{if } \hat{r}_{t+1} < -\tau. \end{cases} \tag{IA6.12}$$

Feature Set: The model uses OHLC-based technical indicators (see Section IA2):

- Normalized prices and returns
- RSI (14-day) and MACD for momentum and trend
- Bollinger Band position and width
- Moving-average ratios over 5, 10, and 20 days
- ATR and stochastic oscillator
- Candlestick patterns such as Doji and Hammer

Practical Notes: The single-layer architecture with 64 units balances expressiveness with computational efficiency. The 15-day lookback captures approximately three trading weeks of context.

Early stopping avoids overfitting to noise in financial returns as they have low signal to noise ratio. The three-level position sizing (1.0/0.6/0.0) reduces turnover compared to binary signals while maintaining directional exposure.

IA6.3 Enhanced LSTM

IA6.3.1 Theoretical Foundation

The enhanced LSTM builds on the standard LSTM architecture (Section IA6.2) by adapting to financial time series challenges (Bahdanau et al., 2014; Qin et al., 2017):

1. **Multi-Time frame Modeling**: Three different LSTM encoders process short (5-day), medium (10-day), and long (15-day) horizons respectively.
2. **Attention Mechanism**: Learns the importance and relevance of historical observations dynamically.
3. **Batch Normalization**: Improves training stability in dense layers.
4. **Multi-Scale Feature Fusion**: Combines representations from all time frames.

The attention mechanism computes a weighted combination of past hidden states:

$$\text{context}_t = \sum_{i=1}^{T} \alpha_i h_i, \quad \alpha_i = \frac{\exp(e_i)}{\sum_j \exp(e_j)}, \tag{IA6.13}$$

where $e_i = v^T \tanh(W_h h_i + W_s s_t + b)$ measures the relevance of past state h_i for the current prediction.

Financial Interpretation. Attention allows the model to selectively focus on historically similar market situations. For example, during periods of elevated volatility, the network may emphasize past high-volatility episodes, while in trending environments it may attend more strongly to previous trend phases. This flexibility is difficult to achieve with purely sequential models.

The multi-scale design reflects the fact that financial markets move and react differently at different frequencies. Daily noise, weekly mean-reversion, and monthly momentum effects coexist and interact. By explicitly modeling multiple time horizons, the enhanced LSTM captures these dynamics without forcing them into a single temporal representation.

IA6.3.2 Implementation Details

Algorithm:

1. Extract features at multiple time scales, creating sequences $X^{(5)}, X^{(10)}, X^{(15)}$ for 5, 10, and 15-day windows respectively.
2. For each time frame $k \in \{5, 10, 15\}$, process through dedicated LSTM to obtain hidden states $h_1^{(k)}, h_2^{(k)}, \ldots, h_{T_k}^{(k)}$ where T_k is the window length.
3. Compute attention scores for each time frame using the learned parameters:

$$e_i^{(k)} = v^T \tanh(W_h h_i^{(k)} + W_s s_t + b). \tag{IA6.14}$$

Table IA18: Enhanced LSTM Hyperparameters

Parameter	**Value**	**Justification**
Multi-scale windows	5, 10, 15 days	Captures short, mid, and longer term effects
LSTM units	[24, 16, 16]	Dedicated capacity per time frame
Attention units	12	Attention layer dimension
Dense units	[24, 12]	Gradual dimensionality reduction
Dropout rate	0.2	Regularization
L1/L2 regularization	0.0001	Mild weight decay
Epochs	8	Faster convergence due to complexity
Batch size	16	Smaller batches improve generalization
Early stopping patience	5	Aggressive overfitting control

4. Convert scores to attention weights via softmax: $\alpha_i^{(k)} = \exp(e_i^{(k)}) / \sum_j \exp(e_j^{(k)})$.
5. Generate context vector for each time frame as attention-weighted sum:

$$\text{context}^{(k)} = \sum_{i=1}^{T_k} \alpha_i^{(k)} h_i^{(k)}. \tag{IA6.15}$$

6. Concatenate multi-time frame context vectors:

$$\text{context}_{\text{fused}} = [\text{context}^{(5)}; \text{context}^{(10)}; \text{context}^{(15)}].$$

7. Pass through dense layers with batch normalization:

$$z = \text{BN}(\text{ReLU}(W_1 \cdot \text{context}_{\text{fused}} + b_1)).$$

8. Generate return prediction: $\hat{r}_{t+1} = W_2 z + b_2$.
9. Convert calibrated forecasts into signals:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Identical to the standard LSTM feature set, but computed separately at multiple time horizons to capture cross-frequency behavior.

Practical Notes: Due to its higher complexity, the enhanced LSTM requires strong regularization. Dropout, L1/L2 penalties, and early stopping work together to limit overfitting. Although training time is roughly two to three times longer than that of a standard LSTM, the model captures interactions across time scales that single-resolution models typically miss.

IA6.4 PatchTST

IA6.4.1 Theoretical Foundation

PatchTST extends transformer-based models to time series by borrowing the patching concept from vision transformers (Nie et al., 2023). Instead of processing individual time steps, the input sequence is divided into overlapping temporal patches:

$$\mathbf{P} = \{p_1, p_2, \ldots, p_N\}, \quad p_i \in \mathbb{R}^{L_p \times D}, \tag{IA6.16}$$

where L_p denotes the patch length, D the feature dimension, and patches overlap according to stride S. Self-attention is then applied across patches:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{IA6.17}$$

Why Patching? Standard transformers applied to time series face a computational bottleneck: self-attention has $O(L^2)$ complexity, where L is sequence length. For a 15-day lookback window with hourly data, this becomes prohibitively expensive. Patching reduces the effective sequence length from L individual time steps to $N = \lfloor (L - L_p)/S \rfloor + 1$ patches, dramatically improving efficiency. For example, with $L = 15$ days, $L_p = 4$, and stride $S = 2$, the sequence length drops from 15 to 6 patches, reducing attention operations by over 60%.

Beyond computational gains, patching provides an inductive bias aligned with financial market behavior: adjacent days often share local structure (e.g., a 3-day consolidation pattern or a 4-day momentum surge), and treating these as coherent units may be more natural than forcing the model to reconstruct these patterns from individual days.

Financial Interpretation. Each patch represents a short segment of market behavior—typically 2–4 consecutive trading days. Attention across patches allows the model to learn how local patterns relate over time. For example:

- A consolidation phase captured in patch p_3 (days 5–8) may be linked to a breakout in patch p_5 (days 9–12)
- A volatility spike in patch p_2 might predict mean-reversion behavior in subsequent patches
- Multi-day accumulation patterns can be detected as patch-level features rather than daily noise

This architecture performs well when market behavior evolves gradually and exhibits recurring local structures—for instance, during trending markets where momentum builds over several days. However, during abrupt regime changes (e.g., a sudden policy announcement causing a gap opening), the assumption of stable local patterns may break down. The model may struggle to adapt quickly because it represents days in fixed-size chunks rather than reacting to individual shocks.

IA6.4.2 Implementation Details

Algorithm:

Table IA19: PatchTST Hyperparameters

Parameter	**Value**	**Justification**
Lookback window	15 days	Aligned with other neural models; approximately 3 trading weeks
Forecast horizon	3 days	Short-term outlook (next week)
Patch size	4	Captures multi-day patterns (roughly 1 trading week)
Stride	2	50% overlap reduces boundary effects and information loss
Model dimension	64	Patch embedding size; balances capacity and efficiency
Attention heads	8	Multi-head self-attention captures diverse temporal patterns
Transformer layers	3	Encoder depth; sufficient for learning patch relationships
FFN dimension	128	Feed-forward capacity (2× model dimension)
Dropout	0.1	Moderate regularization for noisy financial data
Batch size	32	Training efficiency and GPU utilization
Epochs	10	Typical for transformer fine-tuning; early stopping applied
Learning rate	0.001	Adam optimizer default
Early stopping patience	15	Prevents overfitting to training noise

1. Divide input sequence $X \in \mathbb{R}^{L \times D}$ into N overlapping patches with patch length $L_p = 4$ and stride $S = 2$:

$$\mathbf{P} = \{p_1, p_2, \ldots, p_N\}, \quad p_i \in \mathbb{R}^{L_p \times D}, \quad N = \left\lfloor \frac{L - L_p}{S} \right\rfloor + 1. \tag{IA6.18}$$

2. Embed each patch via linear projection: $z_i = W_{\text{embed}} \cdot \text{flatten}(p_i) + b_{\text{embed}}$, where $z_i \in \mathbb{R}^{d_{\text{model}}}$ and $d_{\text{model}} = 64$.

3. Add learnable positional encodings: $z'_i = z_i + \text{PE}_i$.

4. Process through $L_{\text{enc}} = 3$ transformer encoder layers. Each layer applies multi-head self-attention across patches:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{IA6.19}$$

   where $Q = ZW_Q$, $K = ZW_K$, $V = ZW_V$, and $Z = [z'_1; z'_2; \ldots; z'_N]$ is the matrix of patch embeddings.

5. Apply feed-forward network after each attention layer: $\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$.

6. Pool final patch representations: $h = \frac{1}{N}\sum_{i=1}^{N} z_i^{(L_{\text{enc}})}$.
7. Project to return prediction: $\hat{r}_{t+1} = W_{\text{out}}h + b_{\text{out}}$.
8. Convert calibrated forecasts into signals:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1)
    - SR-optimized: Sharpe-based method (Section IA7.2)

Feature Set. The same OHLC-based indicators used for LSTM models (Section IA6.2) are used here, but organized into patches so that each local temporal block is treated as a single unit. Features include price ratios, momentum indicators (RSI, MACD), volatility measures (rolling standard deviation, Bollinger Bands), and trend signals.

Practical Notes. By operating on patches rather than individual time steps, PatchTST reduces the effective sequence length processed by attention, improving computational efficiency while preserving temporal information. The 50% overlap between patches (stride = 2, patch length = 4) minimizes boundary effects and ensures that every day appears in multiple patches, reducing information loss at patch boundaries. This design is particularly effective when markets display stable short-term structures that repeat across time, such as multi-day momentum trends or consolidation patterns.

IA6.5 Temporal Fusion Transformer (TFT)

IA6.5.1 Theoretical Foundation

The temporal fusion transformer (TFT) is built around the principle that not all information is equally important at every moment (Lim et al., 2021). Unlike standard neural networks that apply fixed weights to all features, TFT uses dynamic, context-dependent feature selection and attention mechanisms. The model learns which technical indicators, price patterns, or calendar effects matter most given current market conditions, allowing it to emphasize relevant signals while filtering out noise.

Activation Functions. Two specialized activation mechanisms form the building blocks of TFT's architecture:

ELU (Exponential Linear Unit) maintains smooth gradients for negative inputs, avoiding the "dying ReLU" problem where neurons permanently stop learning:

$$\text{ELU}(x) = \begin{cases} x & \text{if } x > 0, \\ \alpha(e^x - 1) & \text{if } x \leq 0, \end{cases} \tag{IA6.20}$$

where $\alpha = 1.0$ by default. Unlike ReLU, ELU outputs negative values when x is negative, pushing the average activation closer to zero. This zero-centered distribution often accelerates training in deep networks by reducing internal covariate shift.

GLU (Gated Linear Unit) adds learnable, input-dependent gating that controls information flow:

$$\text{GLU}(x) = \sigma(W_g x + b_g) \odot (W_o x + b_o), \tag{IA6.21}$$

where σ is the sigmoid function producing values in $[0, 1]$, and $\odot$ denotes element-wise multiplication. The sigmoid gate acts as a soft switch: values near 0 block the signal, while values near 1 allow full transmission. This mechanism enables the network to dynamically suppress irrelevant features and amplify important ones based on current input.

Gated Residual Network (GRN). The core computational block in TFT combines non-linear transformations, residual connections, and gating to preserve important information while filtering noise:

$$\eta_1 = \text{ELU}(W_1 x + b_1), \tag{IA6.22}$$

$$\eta_2 = W_2 \eta_1 + b_2, \tag{IA6.23}$$

$$\text{GRN}(x) = \text{LayerNorm}\big(x + \text{GLU}(\eta_2)\big), \tag{IA6.24}$$

where the GLU gate modulates the transformed signal η_2 before it is added to the residual connection. The residual path ensures gradients flow easily during backpropagation, while the gate learns when to suppress or emphasize different components of the signal.

Variable Selection Network (VSN). TFT learns time-varying feature importance through a variable selection network. Each input variable is processed through its own GRN and then weighted using a softmax:

$$\xi_j^{(t)} = \text{GRN}_j(x_j^{(t)}, c), \quad v_j^{(t)} = \frac{\exp(\xi_j^{(t)})}{\sum_{i=1}^{F} \exp(\xi_i^{(t)})}, \quad \tilde{x}^{(t)} = \sum_{j=1}^{F} v_j^{(t)} \cdot \text{GRN}_j(x_j^{(t)}), \tag{IA6.25}$$

where $x_j^{(t)}$ is the j-th feature at time t, c is a context vector capturing global state, and $v_j^{(t)} \in [0, 1]$ represents the learned importance weight for that variable. The weighted sum $\tilde{x}^{(t)}$ becomes the effective input used by downstream layers.

Interpretable Multi-Head Attention. To capture temporal relationships, TFT applies scaled dot-product attention over the LSTM-encoded sequence:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V, \tag{IA6.26}$$

where Q, K, and V are learned projections of the LSTM hidden states. With H attention heads, the model can focus on different temporal patterns simultaneously:

$$\text{MultiHead}(Q, K, V) = [\text{head}_1; \ldots; \text{head}_H] W_O, \quad \text{head}_h = \text{Attention}(QW_Q^h, KW_K^h, VW_V^h). \tag{IA6.27}$$

Quantile Forecasting. Instead of predicting a single point estimate, TFT outputs multiple quantiles $q \in \{0.1, 0.5, 0.9\}$, representing pessimistic, median, and optimistic return scenarios:

$$\hat{y}_{t+\tau}^{(q)} = W_q \cdot \text{GRN}(\phi_t) + b_q, \tag{IA6.28}$$

where ϕ_t is the decoder representation and each quantile has its own output layer. This distributional forecast captures uncertainty: narrow quantile ranges indicate high confidence, while wide ranges signal elevated uncertainty requiring more conservative position sizing.

Financial Interpretation. TFT's design aligns naturally with financial market dynamics, where relevant information shifts with market regime:

- **Earnings season**: Company fundamentals and analyst revisions receive higher VSN weights
- **FOMC meetings**: Macro indicators (inflation, employment) and bond yields become more important
- **Quiet summer trading**: Technical patterns and seasonal effects dominate
- **Crisis periods**: Volatility measures and risk-off indicators receive elevated attention

The variable selection weights and attention patterns provide transparency into model reasoning, which is valuable for risk management and regulatory compliance. Quantile outputs enable risk-aware trading: when the 10th–90th percentile range is narrow (e.g., [+0.2%, +0.4%]), the forecast is confident and positions can be sized aggressively. When the range is wide (e.g., [-1.5%, +2.0%]), uncertainty is high and positions should be reduced or avoided. This explicit uncertainty quantification distinguishes TFT from models that output only point forecasts.

IA6.5.2 Implementation Details

Table IA20: TFT Hyperparameters

Parameter	Value	Justification
Lookback window	15 days	Fixed historical window (approximately 3 trading weeks)
Forecast horizon	3 days	Short-term prediction (next trading week)
Hidden size	64	Balanced model capacity across components
LSTM layers	2	Encoder–decoder depth; captures sequential dependencies
Attention heads	4	Multiple temporal patterns without excessive computation
Dropout rate	0.1	Light regularization; financial data already noisy
Batch size	32	Training efficiency and stable gradient estimates
Epochs	10	Typical training length with early stopping
Learning rate	0.001	Adam optimizer default; reduced on plateau
Quantiles	[0.1, 0.5, 0.9]	Lower, median, upper bounds for uncertainty
Early stopping patience	15	Prevents overfitting to training noise

Algorithm:

1. For each input feature $x_j^{(t)}$, compute its importance using the Variable Selection Network:

$$v_j^{(t)} = \frac{\exp(\text{GRN}_j(x_j^{(t)}, c))}{\sum_{i=1}^{F} \exp(\text{GRN}_i(x_i^{(t)}, c))}. \tag{IA6.29}$$

2. Build the weighted feature vector:

$$\tilde{x}^{(t)} = \sum_{j=1}^{F} v_j^{(t)} \cdot \mathrm{GRN}_j(x_j^{(t)}). \tag{IA6.30}$$

3. Pass the sequence through $L = 2$ stacked LSTM layers:

$$h_t^{(\ell)} = \mathrm{LSTM}^{(\ell)}(h_{t-1}^{(\ell)}, \tilde{x}^{(t)}), \quad \ell = 1, \ldots, L. \tag{IA6.31}$$

4. Apply interpretable multi-head attention with $H = 4$ heads:

$$\mathrm{MultiHead}(H^{(L)}) = [\mathrm{head}_1; \ldots; \mathrm{head}_H]W_O, \tag{IA6.32}$$

$$\mathrm{head}_h = \mathrm{Attention}(H^{(L)}W_Q^h, H^{(L)}W_K^h, H^{(L)}W_V^h), \tag{IA6.33}$$

where $H^{(L)} = [h_1^{(L)}; h_2^{(L)}; \ldots; h_T^{(L)}]$ is the matrix of final LSTM hidden states.

5. Process the attention output through a GRN:

$$\phi_t = \mathrm{GRN}(\mathrm{MultiHead}(H^{(L)})). \tag{IA6.34}$$

6. Generate quantile predictions for $q \in \{0.1, 0.5, 0.9\}$:

$$\hat{r}_{t+1}^{(q)} = W_q \cdot \phi_t + b_q. \tag{IA6.35}$$

7. Use the median $\hat{r}_{t+1}^{(0.5)}$ as the primary trading signal (see Section IA7). Optionally, use the quantile range $[\hat{r}_{t+1}^{(0.1)}, \hat{r}_{t+1}^{(0.9)}]$ to scale position size: narrow ranges indicate high confidence (increase exposure), while wide ranges indicate uncertainty (reduce exposure or stay neutral).

Feature Set. The model supports both static variables (e.g., asset class, sector) and dynamic time-varying features. For our S&P 500 forecasting application, we use only dynamic features: OHLC-based technical indicators as described in Section IA6.2, including price ratios, momentum indicators (RSI, MACD), volatility measures, and trend signals.

Practical Notes. The variable-selection and attention weights make TFT relatively interpretable compared to black-box neural networks, showing which inputs matter at each point in time. This transparency aids model debugging, feature engineering, and risk management oversight. Quantile outputs provide explicit uncertainty quantification, supporting risk-aware position sizing without requiring separate volatility forecasting models.

However, this flexibility comes at a cost: TFT is architecturally complex, involving multiple specialized components (VSN, GRN, multi-head attention, quantile regression). It typically requires more training data and longer training times compared to simpler architectures like standard LSTMs. The additional hyperparameters (number of GRN layers, attention heads, quantile levels) also expand the tuning search space. Despite these costs, TFT's ability to adaptively select relevant features and quantify uncertainty makes it well-suited for financial forecasting, where the signal-to-noise ratio is low and regime-dependent feature importance is critical.

IA6.6 Non-Stationary Transformer

IA6.6.1 Theoretical Foundation

Non-stationary transformers are designed to deal with the fact that financial time series do not follow fixed statistical distributions over time. Instead of assuming stationarity, the model explicitly separates stable structural relationships from changing distributional properties (Liu et al., 2022). In simple terms, it tries to keep what is relatively constant (such as how assets move compared to each other) while adapting to what changes (such as volatility levels or mean returns). This is done through a modified attention mechanism:

$$\text{Attention}_{\text{NS}}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k} \cdot \tau_t}\right) V \cdot \delta_t, \tag{IA6.36}$$

where τ_t is a learned temperature parameter that reacts to the current volatility of the series and controls how sharp or smooth the attention distribution is, while δ_t is a de-stationary scaling factor that compensates for shifts in the overall data distribution.

Financial Interpretation. In financial markets, relative relationships are often more stable than absolute values. For example, technology stocks usually remain more volatile than utility stocks, often by roughly 1.5 to 2 times, even though overall market volatility may rise or fall. The non-stationary transformer can learn and preserve these relative patterns while still adjusting to whether the market is in a calm or turbulent phase. This allows it to keep making useful predictions even when the magnitude of returns changes.

However, this design also has limits. It assumes that relative relationships remain stable. If a structural break occurs, such as a shift in monetary policy that reverses the relationship between growth and value stocks, the model cannot adapt properly. In such cases the problem is no longer just a change in distribution, but a change in the relationships themselves, which this architecture is not built to handle.

IA6.6.2 Implementation Details

Algorithm:

1. For an input sequence $X \in \mathbb{R}^{L \times D}$, compute its mean and variance:

$$\mu_t = \frac{1}{L} \sum_{i=1}^{L} x_i, \qquad \sigma_t^2 = \frac{1}{L} \sum_{i=1}^{L} (x_i - \mu_t)^2. \tag{IA6.37}$$

2. Normalize the sequence to remove non-stationarity:

$$\tilde{X} = \frac{X - \mu_t}{\sigma_t}.$$

3. Embed the normalized input:

$$Z = \tilde{X} W_{\text{embed}} + \text{PE},$$

where PE denotes positional encoding.

Table IA21: Non-Stationary Transformer Hyperparameters

Parameter	**Value**	**Justification**
Model dimension	128	Larger capacity for non-stationary dynamics
Attention heads	8	d_{model}/heads = 16 per head
Encoder layers	4	Deeper encoder for extracting patterns
Decoder layers	2	Lighter decoder focused on prediction
Feedforward dimension	512	Four times the model size
Lookback window	10 days	Length of input sequence
Forecast horizon	3 days	Short multi-step forecast
Dropout rate	0.1	Standard regularization
Batch size	32	Training batch size
Epochs	10	Standard training duration
Learning rate	0.001	Typical Adam learning rate
Early stopping patience	12	Limits overfitting
Volatility adjustment	Enabled	Adaptive threshold scaling (0.5×)

4. Compute the adaptive temperature from the input variance:

$$\tau_t = f_\tau(\sigma_t^2) = \text{MLP}(\sigma_t^2) \in \mathbb{R}^+. \tag{IA6.38}$$

5. Apply the non-stationary attention with temperature scaling:

$$\text{Attention}_{\text{NS}}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k} \cdot \tau_t}\right) V, \tag{IA6.39}$$

   where larger τ_t values (during high-variance periods) lead to smoother, less peaky attention weights.

6. Compute the de-stationary projection factor:

$$\delta_t = g_\delta(\mu_t, \sigma_t) = \text{MLP}([\mu_t; \sigma_t]).$$

7. Rescale the decoder output back to the original data distribution:

$$\hat{Y} = \delta_t \odot \text{Decoder}(Z^{(L)}) \cdot \sigma_t + \mu_t.$$

8. Produce the next-step return forecast $\hat{r}_{t+1}$ from the final layer.

9. Apply the volatility-adjusted decision threshold for signal generation (see Section IA7).

Feature Set. The model uses the same OHLC-based technical indicators as the LSTM model (Section IA6.2). Thanks to the non-stationary attention mechanism, it automatically adapts to changes in the distribution of these features, without needing an explicit regime-switching model.

Practical Notes. The architecture includes learned temperature scaling inside the StatisticalAttention layer so that attention reacts to changing variance. When volatility is high, the temperature increases and the attention becomes softer, which reduces sensitivity to extreme observations. Adaptive instance normalization is blended with standard layer normalization using a learnable mixing weight, giving the model extra flexibility when handling non-stationary data. Finally, the volatility-adjusted trading threshold is tied to recent market volatility using a 20-day window and a 0.5× multiplier, which helps to avoid excessive false signals during unstable periods.

IA6.7 Online Learning

IA6.7.1 Theoretical Foundation

Online learning updates parameters step-by-step as new observations arrive, rather than retraining a model from the beginning each time. This incremental updating allows the model to track slow market drift, including shifting correlations, evolving volatility, and changing factor relationships (Bifet and Gavaldà, 2007; Shalev-Shwartz, 2012):

$$\theta_{t+1} = \theta_t - \eta_t \nabla L(x_t, r_t, \theta_t), \tag{IA6.40}$$

where η_t is the learning rate and ∇L is the loss gradient with respect to the current parameters.

Financial Interpretation. At every step, the model checks how wrong it was on the newest data point, computes the direction that would reduce that error, and then moves a small distance in that direction. It is similar to a trader who makes small tweaks after each trading day instead of rebuilding the whole strategy.

The learning rate η_t controls how fast the model adapts. A large η_t helps when markets change quickly, but it can also cause the model to chase noise. A small η_t keeps the system stable, but it may respond too slowly and miss short-lived shifts. In practice, the best adaptation speed is itself state-dependent: quiet markets favor slow learning, and transition periods favor faster learning. This turns the problem into a meta-issue—the model needs a good rule for how quickly it should learn.

The low signal-to-noise ratio in markets makes this difficult. It is often not clear whether prediction errors come from real structural change that requires updating, or from temporary volatility bursts that disappear quickly. Online methods can therefore struggle to separate true drift from short-term randomness.

IA6.7.2 Implementation Details

Algorithm:

1. Initialize an ensemble consisting of `SGDRegressor`, `PassiveAggressiveRegressor`, and `RandomForestRegres`
2. Warm-up: train on the first 50 samples using `partial_fit` (SGD, PA) and `fit` (RF)
3. For each incoming observation:
    - Produce an ensemble forecast as a weighted average of model outputs
    - Every 10 steps: update SGD and PA via `partial_fit`, and retrain RF using the most recent 100 samples

Table IA22: Online Learning Hyperparameters

Parameter	Value	Justification
Ensemble methods	SGD, PA, Online RF	Use multiple online learners for robustness
SGD learning rate	Adaptive	Automatically adjusts step size
SGD regularization (α)	0.0001	L2 penalty strength
PA regularization (C)	1.0	Controls aggressiveness
RF estimators	10	Lightweight random forest size
Drift detection window	100	Error comparison horizon
Drift threshold	2.0	Drift flagged at 2 standard deviations
Adaptation rate	0.01	Speed of weight updates
Update frequency	10 steps	Balance responsiveness and stability
Warmup samples	50	Initial fitting before forecasts
Memory buffer size	1000	Upper bound on retained samples

- Detect concept drift by comparing recent prediction errors to historical errors through a z-score test
- If drift is detected, adjust ensemble weights using recent performance

4. Generate OHLC-based features together with regime indicators (Bull/Bear/High_Vol)
5. Convert predictions into trading signals (Section IA7)

Feature Set: OHLC-based features with explicit regime indicators:

- Core features: returns, RSI (14-day), MACD, ATR, and candlestick shadow measures
- Regime indicators: Bull (Price > MA_50 and slope > 0), Bear (opposite), High_Volatility (ATR > 90th percentile)
- Lagged features: returns, RSI, MACD, ATR, and shadows with lags 1, 2, and 5
- Feature selection: keep the top 20 features ranked by correlation with the target

Practical Notes: Using three distinct learners (SGD, Passive-Aggressive, and Random Forest) improves robustness through diversity. Drift detection relies on a z-score approach with a threshold of 2.0, comparing recent error behavior against a historical baseline. When drift is triggered, the ensemble re-weights models to emphasize those performing better recently. The memory buffer of 1000 samples provides enough history for RF retraining while limiting the influence of very old data.

IA6.8 MAML: Model-Agnostic Meta-Learning

IA6.8.1 Theoretical Foundation

Model agnostic meta learning (MAML) aims to learn parameters that can adapt rapidly when conditions change, rather than optimizing only for performance in the current environment. In

other words, it tries to learn how to learn. The objective is (Finn et al., 2017):

$$\theta^* = \arg\min_{\theta} \sum_{\tau \sim p(\tau)} L_\tau(\theta - \alpha \nabla_\theta L_\tau(\theta)), \tag{IA6.41}$$

This formulation is complex but the intuition is direct: instead of finding parameters that perform best immediately, MAML finds an initialization that becomes strong after only a few gradient updates on new data. Tasks τ are sampled from a distribution $p(\tau)$, and the learned parameters θ^* are those that adapt well across many tasks. The method has a two-level optimization structure:

1. **Inner Loop**: quickly adapt to one task or regime using a small number of gradient steps.
2. **Outer Loop**: update the initialization so that this fast adaptation works well for many tasks.

Financial Interpretation. In financial terms, MAML avoids starting from scratch whenever market structure changes. Instead, it begins from a meta-learned starting point that is already shaped to adapt quickly. Over long training histories, the model absorbs general patterns of market shifts, e.g., volatility jumps, sector rotations and liquidity events. This helps it converge faster when similar transitions happen again.

A simple analogy is training a trader not to memorize one fixed strategy, but to become skilled at learning new strategies quickly. The meta-initialization encodes transferable knowledge about market behavior across regimes.

Still, MAML depends on the assumption that future market changes resemble transitions present in training. Under truly new conditions—negative rates, pandemic shutdowns, or new market microstructure—the learned prior can become harmful, pushing the model to adapt in the wrong direction because it is biased toward familiar patterns.

IA6.8.2 Implementation Details

Table IA23: MAML Hyperparameters

Parameter	**Value**	**Justification**
Meta learning rate	0.001	Outer-loop step size (Adam)
Inner learning rate	0.01	Faster task adaptation (10× larger)
Inner loop steps	5	Trade-off between adaptation and noise-fitting
Hidden units	[64, 32, 16]	Three-layer MLP with tapering width
Lookback window	15 days	Feature sequence length
Dropout rate	0.1	Light regularization
Meta batch size	16	Number of tasks per meta-update
Support set size	50	Samples used for adaptation
Query set size	20	Samples used for meta-gradient evaluation
Meta epochs	100	Meta-training iterations
Early stopping patience	10	Avoids meta-level overfitting

Algorithm:

1. Define tasks using overlapping rolling windows (support set: 50 samples, query set: 20 samples)
2. Construct a 3-layer MLP base learner; inputs are flattened features from the lookback window
3. For each meta-epoch (up to 100, with early stopping patience 10):
    - Sample 16 tasks from the task distribution
    - For each task: clone the model and run 5 SGD updates on the support set (inner loop)
    - Evaluate the adapted model on the query set and compute the meta-loss
    - Update the shared initialization using Adam in the outer loop
4. During inference, apply the meta-learned model directly to produce predictions
5. Convert calibrated forecasts into signals:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Multi-horizon OHLC-based features:

- Moving-average ratios and slopes across 5d, 21d, 60d, 252d, plus crossover signals
- Z-scores computed at 5d, 21d, 60d, and 252d horizons
- RSI at 14d and 21d, and a MACD suite (standard 12–26–9 plus long-term 26–52–18)
- Bollinger Band position and width at 20d and 60d
- Candlestick patterns: Bullish, Bearish, Doji, Hammer, Shooting Star, Spinning Top
- Multi-time frame momentum and volatility measures (5d, 21d, 60d)
- ATR (14d)

All features are flattened across the lookback window before being passed into the MLP.

Practical Notes: The implementation follows first-order MAML (FOMAML), meaning gradients are computed through the adaptation step without explicitly calculating second-order derivatives. Tasks are constructed from overlapping rolling windows with a 25% step size to ensure exposure to varied market states. Early stopping with patience 10 is applied to prevent meta-overfitting. The learned initialization captures broad market dynamics and is intended to transfer across regimes.

IA6.9 Robust Mixture of Experts

IA6.9.1 Theoretical Foundation

Mixture of Experts (MoE) models keep several specialized predictors and then decide, at each point in time, which ones should matter more given the current market setting (Jacobs et al., 1991; Shazeer et al., 2017). The combined forecast is formed as a weighted sum of expert outputs:

$$\hat{y} = \sum_{i=1}^{n} g_i(x, \theta_g) \cdot f_i(x, \theta_i), \tag{IA6.42}$$

where the gating functions $g_i(\cdot)$ are produced by a gating network and satisfy $\sum_i g_i = 1$, and $f_i(\cdot)$ are the predictions from expert i. Conceptually, each expert can become good at a different type of market: one for trending phases, another for mean-reversion, and another for volatility-driven behavior. The gating model learns to detect which situation is happening and then activates or blends the relevant experts.

To make the system more robust, dropout is applied in the gating network. This limits over-dependence on any single expert, encourages redundancy, and reduces the chance that the gating collapses into always selecting one expert.

Financial Interpretation. The advantage of expert specialization is that each model can learn its preferred market pattern without being confused by signals from incompatible environments. For example, a linear expert can focus on simple trend-following relations, a neural expert can represent non-linear interactions, and a tree-based expert can capture regime-like conditional rules. The gating network then learns how to recognize present conditions and mix these outputs in a sensible way.

Using dropout in the gating layers (0.2) is especially helpful when regimes change quickly: it discourages the gating system from committing too hard to one expert and helps avoid expert collapse. In addition, training experts on disjoint partitions of the dataset increases diversity and pushes the experts toward genuinely different specializations.

IA6.9.2 Implementation Details

Table IA24: Robust Mixture of Experts Hyperparameters

Parameter	**Value**	**Justification**
Number of experts	3	Linear, NN, Random Forest (Section IA5.8)
Linear expert α	1.0	Ridge penalty strength
NN expert units	[16, 8]	Compact neural expert
NN expert epochs	5	Short training per expert
NN expert dropout	0.2	Regularization inside NN expert
RF estimators	20	Small forest size
RF max depth	3	Shallow trees to reduce overfit
Gating network	[16, 8] units	Lightweight gating model
Gating dropout	0.2	Reduces expert collapse risk
Gating epochs	8	Enough iterations to learn weights
Batch size	32	Mini-batch training

Algorithm:

1. Split the training sample into disjoint parts, assigning one part to each expert.
2. Train the experts separately on their assigned splits: Linear (Ridge), Neural Network, and Random Forest.
3. Compute expert predictions over the full training sample.
4. Train the gating network using both input features and expert predictions, minimizing MSE.
5. During inference, generate all expert predictions and compute softmax gating weights.
6. Produce the combined forecast:

$$\hat{r}_{t+1} = \sum_i g_i(x_t) \cdot f_i(x_t).$$

7. Convert calibrated forecasts into signals:
   - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
   - SR-optimized: Sharpe-based method (Section IA7.2).

   (Section IA7).

Feature Set: All experts and the gating model share the same full OHLC-derived indicator set (Section IA6.2). The gating network takes as input both the original feature vector and the experts' predictions, so it can learn which expert is more trustworthy under the current market context.

Expert Diversity: Experts are selected from different model families—parametric linear methods, neural networks, and tree-based models (Section IA5.8)—to encourage error decorrelation and improve ensemble robustness.

Practical Notes: The implementation relies on standard softmax gating (no temperature scaling), letting the gating network learn blending weights directly from data. Enforcing disjoint training partitions supports specialization and helps prevent one expert from dominating across all conditions. Dropout at 0.2 is used in both the neural expert and the gating network, providing regularization and encouraging diverse predictions. In practice, this design performs best when different market states favor different modeling styles: the linear expert tends to behave well in stable trend periods, the neural expert can pick up non-linear interactions, and the random forest expert can exploit regime-like conditional structures.

IA6.10 Ensemble Meta-Learning

IA6.10.1 Theoretical Foundation

Ensemble Meta-Learning combines forecasts from multiple base models using weights that evolve over time according to recent predictive quality (Bates and Granger, 1969; Wolpert, 1992). Rather than using fixed averaging weights, the method updates model weights based on its recent past performance:

$$\hat{y}_t = \sum_{m=1}^{M} w_m^{(t)} \cdot f_m(x_t), \tag{IA6.43}$$

where $f_m(x_t)$ is the output of model m at time t, and $w_m^{(t)}$ are dynamic weights estimated from recent errors. A common update rule is inverse-error weighting:

$$w_m^{(t)} = \frac{1/(\bar{e}_m + \epsilon)}{\sum_{j=1}^{M} 1/(\bar{e}_j + \epsilon)}, \tag{IA6.44}$$

where $\bar{e}_m$ is the mean absolute error of model m over a rolling performance window, and ϵ is a small constant included to avoid division-by-zero and improve numerical stability.

Financial Interpretation. This framework shifts weight toward models that are currently working, without requiring explicit regime labels. If markets are trending and one model tends to forecast trends well, it will naturally gain more weight. If conditions become range-bound and mean reversion dominates, models specialized in that setting will start receiving more influence.

For financial prediction this is appealing because it acts like automatic, data-driven model selection. It reduces dependence on a single potentially misspecified model by leaning toward those that have been accurate recently. At the same time, minimum weight constraints ensure that no model is completely shut out, which is helpful if a regime change occurs and a previously weak model suddenly becomes relevant.

To increase the benefit of combination, the base learners are chosen from different methodological families—here, an LSTM model, an ARIMA model, and an Exponential Smoothing model—so their errors are less correlated and aggregation is more valuable.

IA6.10.2 Implementation Details

Table IA25: Ensemble Meta-Learning Hyperparameters

Parameter	**Value**	**Justification**
Base models	LSTM, ARIMA, ES	Sections IA6.2, IA4.1, IA4.2
Ensemble method	Performance-weighted	Dynamic inverse-error reweighting
Performance window	40 days	About two months of recent errors
Reweight frequency	10 days	Update weights roughly bi-weekly
Minimum weight	0.1	Keeps diversity in the mix
Risk adjustment	True	Volatility-scaled signal thresholds
Volatility window	20 days	Window used for threshold scaling
Signal threshold	0.005	Base cutoff for position changes

Algorithm:

1. Fit the base learners (LSTM, ARIMA, Exponential Smoothing) using the training data.
2. Produce in-sample predictions from each base model.
3. Start a performance tracker that stores rolling prediction errors for each model.
4. Initialize weights uniformly across models.

5. At inference time, gather the current predictions from all base models.
6. Every 10 days, recompute weights using the inverse-error rule over the last 40 days.
7. Enforce a minimum weight of 0.1, then renormalize weights so they sum to 1.
8. Form the combined forecast:
$$\hat{r}_{t+1} = \sum_m w_m^{(t)} \cdot f_m(x_t).$$
9. Convert calibrated forecasts into signals:
    - Returns-optimized: Adaptive Threshold method (Section IA7.1.1).
    - SR-optimized: Sharpe-based method (Section IA7.2).

Feature Set: Each base model uses their corresponding feature set, as described in Sections IA6.2, IA4.1, and IA4.2 respectively.

Practical Notes: The minimum weight floor (0.1) ensures that all models remain part of the final forecast, even if one has recently performed poorly. This improves robustness when a currently downweighted model becomes useful again after a market shift. We update the weights every 10 days to strike a balance between adaptability ($>$ 10 days might not adapt) and instability ($<$ 10 days might not be stable). Finally, risk adjustment scales the signal threshold using recent volatility, meaning that in turbulent periods a larger predicted return is required to trigger a position change, reducing noise-driven trades.

IA7 Signal Generation Framework

All forecasting models described in Sections IA4–IA6 produce a one-step-ahead expected return, denoted by $\hat{r}_{t+1}$. This section describes the process of converting raw return forecasts to actual trading decisions, expressed as portfolio weights in the interval $[0, 1]$, where 0 means the portfolio is fully in cash and 1 means fully invested in equities.

We use two different optimization criteria in our work:

- **Returns-Optimized** (Section IA7.1): Forecasts are converted into positions using threshold-based rules (optionally adjusted for risk). Three variants are supported: Adaptive Threshold (default), and Static Threshold.
- **Sharpe-Optimized** (Section IA7.2): Forecasts are first turned into expected Sharpe ratios and then mapped into exposure levels. This rule is applied in the same way for all models.

IA7.1 Returns-Optimized Signal Generation

Under the returns-optimized objective, the predicted return $\hat{r}_{t+1}$ is mapped directly into a portfolio position. There are two main variants that give discrete and continuous positions from forecast.

IA7.1.1 Adaptive Threshold Method

This rule is used as the default position sizing mechanism for most baseline forecasting models, including ARIMA, Exponential Smoothing, and Linear Regression. Rather than switching between a

few discrete portfolio states, it produces **smooth, continuous portfolio weights** whose sensitivity automatically adapts to the prevailing market volatility, which helps limit unnecessary trading and sudden jumps in exposure.

Volatility-dependent thresholds. Trading thresholds are adjusted based on the current 21-day realized volatility σ_t:

$$U_t = \max(\tau_{\text{base}}^{\text{upper}},\, k_{\text{upper}} \cdot \sigma_t), \tag{IA7.1}$$

$$L_t = \max(\tau_{\text{base}}^{\text{lower}},\, k_{\text{lower}} \cdot \sigma_t), \tag{IA7.2}$$

where the fixed minimum thresholds are $\tau_{\text{base}}^{\text{upper}} = 0.003$ and $\tau_{\text{base}}^{\text{lower}} = 0.0015$, while the volatility scaling coefficients are $k_{\text{upper}} = 0.06$ and $k_{\text{lower}} = 0.03$.

Choice of parameters. The base thresholds correspond to roughly 30 and 15 basis points, which is about 0.3 and 0.15 standard deviations of a typical daily S&P 500 return assuming $\sigma_{\text{daily}} \approx 1\%$. These values act as a minimum filter against noise, ensuring that very small forecasts are ignored even when markets are quiet.

The volatility multipliers were chosen so that volatility-driven scaling only becomes important in stressed markets. The point at which the volatility-scaled term overtakes the base threshold is

$$\sigma_t = \frac{\tau_{\text{base}}^{\text{upper}}}{k_{\text{upper}}} = \frac{0.003}{0.06} = 0.05,$$

which corresponds to roughly 5% daily volatility, or VIX values around 50. Below this level, the fixed thresholds dominate, while above it the thresholds expand with market turbulence. This behavior is illustrated below:

Daily Volatility	**Upper Threshold U_t**	**Regime**
1% (normal, VIX ∼15)	$\max(0.003, 0.0006) = 0.003$	Base dominates
2% (elevated, VIX ∼30)	$\max(0.003, 0.0012) = 0.003$	Base dominates
5% (crisis, VIX ∼50)	$\max(0.003, 0.003) = 0.003$	Crossover point
8% (extreme, VIX ∼80)	$\max(0.003, 0.0048) = 0.0048$	Volatility-driven

The upper and lower multipliers differ by a factor of two ($k_{\text{upper}}/k_{\text{lower}} = 2$), which introduces an intentional asymmetry. Stronger evidence is therefore required to justify a fully invested long position than to stay neutral, reflecting a cautious stance when working with noisy forecasts.

Model-dependent scaling. Regularization causes models like SVM, Lasso and Elastic Net to produce very small magnitudes of forecasts. For these models, the base thresholds are reduced by a factor of ten to $\tau_{\text{base}}^{\text{upper}} = 0.0003$ and $\tau_{\text{base}}^{\text{lower}} = 0.00015$ so that their signals are treated on a comparable scale.

Regime classification. Using these thresholds, the forecasts are mapped into one of three regimes:

Table IA26: Regime classification based on forecasts.

Condition	Regime	Portfolio Weight Range
$\hat{r}_{t+1} > U_t$	Long	(0.6, 1.0)
$L_t \leq \lvert\hat{r}_{t+1}\rvert \leq U_t$	Neutral	(0.4, 0.8)
$\hat{r}_{t+1} < -U_t$	Cash	(0.0, 0.2)

Z-score driven position selection. After the regime is selected, the exact portfolio weight inside that range is determined using a standardized score that measures how far the forecast lies beyond the threshold:

$$z_t = \frac{\hat{r}_{t+1} - \text{sign}(\hat{r}_{t+1})\, U_t}{\tau_t\, \sigma_t},$$

where

$$\tau_t = \max(10^{-6},\, 0.8\, \sigma_{\text{pred},t}),$$

and $\sigma_{\text{pred},t}$ denotes the 252-day rolling standard deviation of the model's forecasts.

The final allocation is then computed as

$$w_t = \bar{w}_{\text{bucket}} + 0.5 \cdot \text{clip}(z_t, -1, 1) \cdot \Delta w_{\text{bucket}},$$

with $\bar{w}_{\text{bucket}}$ being the midpoint of the selected range and Δw_{bucket} its width.

Hysteresis filter. The portfolio is only rebalanced when the newly computed weight differs from the current allocation by at least 10 percentage points. We implement this with the help of a hysteresis filter with parameter $h = 0.10$.

Models using this rule. This adaptive threshold mechanism is used by ARIMA, Exponential Smoothing, Linear Regression, and most classical machine learning benchmark models.

IA7.1.2 Static Threshold Method

The static threshold rule produces **discrete** positions in {0, 0.6, 1.0} based on fixed cutoffs:

$$w_t = \begin{cases} 1.0 & \text{if } \hat{r}_{t+1} > \tau, \\ 0.6 & \text{if } \lvert\hat{r}_{t+1}\rvert \leq \tau, \\ 0.0 & \text{if } \hat{r}_{t+1} < -\tau, \end{cases} \tag{IA7.3}$$

with $\tau = 0.005$ (0.5% per day). The neutral weight of 0.6 reflects a small long bias consistent with the equity risk premium.

Limitation. A single fixed threshold does not adapt well to changing market regimes.

IA7.2 Sharpe-Optimized Signal Generation

When Sharpe optimization is selected, all models use the same risk-adjusted sizing rule. Model-specific mappings are ignored, and signals are created only from expected Sharpe ratios.

IA7.2.1 Theoretical idea

From mean-variance theory, the optimal risky allocation is proportional to the expected Sharpe ratio:

$$w_t^* \propto \frac{\mathbb{E}[r_{t+1}] - r_f}{\sigma_t}. \tag{IA7.4}$$

IA7.2.2 Implementation

Step 1: Expected Sharpe.

$$\hat{S}_t = \frac{\hat{r}_{t+1} - r_{f,t}}{\hat{\sigma}_t}, \tag{IA7.5}$$

where $\hat{\sigma}_t$ is a 60-day rolling volatility estimate.

Step 2: Convert to position.

$$w_t = \min\Big(1, \max\Big(0, k \cdot \hat{S}_t\Big)\Big), \tag{IA7.6}$$

with $k = 0.5$. If $\hat{S}_t < 0.1$, the position is set to zero.

Table IA27: Sharpe-to-position mapping ($k = 0.5$)

Expected Sharpe	**Position**
< 0.1	0%
0.5	25%
1.0	50%
1.5	75%
≥ 2.0	100%

Key property. Because this rule is applied at the runner level, all forecasting models follow exactly the same position sizing whenever Sharpe optimization is used.

IA7.3 Summary of Signal Generation by Model

Table IA28: Signal generation by model and objective

Model	**Returns-Optimized**	**SR-Optimized**	**Type**
ARIMA	Adaptive Threshold	Sharpe-based	Continuous
Exponential Smoothing	Adaptive Threshold	Sharpe-based	Continuous
Linear Regression	Adaptive Threshold	Sharpe-based	Continuous
Risk-Adjusted ARIMA	Kelly Criterion inspired	Sharpe-based	Continuous

Design rationale. Using a model-specific layer for returns optimization and a shared layer for Sharpe optimization serves two goals:

1. Returns-optimized rules allow each model to use a mapping suited to its structure (for example, GARCH-based Kelly criterion inspired sizing for Risk-Adjusted ARIMA).
2. Sharpe-optimized rules enforce identical sizing across models, making performance comparisons depend only on forecast quality rather than on different trading heuristics.

IA8 Returns Calculation

This section explains how portfolio returns are computed in the backtest, including where the underlying inputs come from (prices, dividends, and cash yields) and how those pieces are combined into daily portfolio performance.

IA8.1 Data Sources

Returns are built using three main datasets:

1. **Yahoo Finance S&P 500 Price Index (^GSPC).** This source provides daily S&P 500 price levels *excluding* dividends. It is used as input to generate trading signals and to compute the price component of equity returns.
2. **Kenneth French Daily Factors.** This dataset supplies two important inputs:
   - *Market returns:* Daily S&P 500 *total* returns (dividends reinvested). These series are used for benchmark comparisons and CAPM-based analysis.
   - *Risk-free rate:* This is used to calculate the interest on the cash part of the portfolio capital.

   Source: Kenneth French Data Library relies on CRSP as the underlying data provider.
3. **CRSP S&P 500 Monthly Data.** Monthly S&P 500 returns with and without dividends. The monthly dividend yield is calculated as:

$$d_m = \text{ret} - \text{retx}, \tag{IA8.1}$$

   where `ret` is the total return including dividends and `retx` is the return excluding dividends. The resulting dividend yield is applied to strategy portfolios at each month-end.

IA8.2 Portfolio Value Computation

The backtest tracks the portfolio using three state variables: the number of shares held (S_t), the cash balance (C_t), and the total portfolio value (V_t). At any date t, total value is defined as:

$$V_t = S_t \cdot P_t + C_t, \tag{IA8.2}$$

where P_t is the current S&P 500 price level.

IA8.3 Daily Cash Returns

Cash holdings earn the daily risk-free rate. Before computing the daily portfolio return, the cash balance is grown by the risk-free return:

$$C_t \leftarrow C_t \cdot (1 + r_t^f), \tag{IA8.3}$$

where r_t^f is the daily risk-free rate taken from the Kenneth French daily factors. This step ensures that cash positions reflect a realistic alternative investment rather than implicitly earning zero.

IA8.4 Daily Portfolio Returns

The daily portfolio return is defined as the percent change in total portfolio value, with cash already updated for its risk-free accrual:

$$r_t^{\text{portfolio}} = \frac{V_t - V_{t-1}}{V_{t-1}}. \tag{IA8.4}$$

With this definition, both components of performance are naturally captured: changes in the equity portion ($S_t \cdot P_t$) and changes in the cash portion (through the updated C_t).

IA8.5 End-of-Month Dividend Reinvestment

Dividend income is added at the end of each month through share reinvestment. On the final trading day of month m, the additional shares received are:

$$\Delta S_m = \frac{S_{T_m} \cdot P_{T_m} \cdot d_m}{P_{T_m}} = S_{T_m} \cdot d_m, \tag{IA8.5}$$

where:

- T_m is the last trading day of month m.
- S_{T_m} is the number of shares held at the end of that month.
- d_m is the monthly dividend yield computed from CRSP S&P 500 data (see Eq IA8.1).
- ΔS_m is the number of new shares obtained from dividend reinvestment.

After applying dividends, the holdings are updated as $S_{T_m} \leftarrow S_{T_m} + \Delta S_m$, and the portfolio value is recomputed so that dividend income is fully reflected.

IA8.6 Benchmark Returns

The buy-and-hold benchmark is calculated using the exact same mechanics as other strategies:

- Trading is based on Yahoo Finance S&P 500 price data (no dividends).
- Dividends are applied using CRSP dividend yields via end-of-month share reinvestment.
- The benchmark always stays 100% invested in equities (Signal = 1.0 at all times).

This setup keeps the comparison fair, because the benchmark and all active strategies follow identical return accounting rules. The Kenneth French daily factors - market return series is used separately for CAPM-based alpha and beta estimations.

IA9 Validation Framework

This section describes the validation procedures applied across all strategy implementations. In financial forecasting, validation is not just a box to check: without a strict procedure, backtests can easily overfit and look impressive only because the evaluation leaked information from the future. The purpose of the framework here is to ensure that reported results reflect genuine predictive skill rather than artifacts of the testing design.

IA9.1 Walk-Forward Validation

Walk-forward validation is the primary evaluation protocol used for every strategy. Rather than relying on a single train/test split, it simulates real trading by repeatedly retraining and forecasting forward through time:

1. Train the model using all observations available up to time t.
2. Produce a forecast for the next period $t + 1$.
3. Store the forecast *before* observing the realized outcome.
4. Advance the window and repeat.

IA9.1.1 Rolling Window Implementation

All strategies use a fixed-length rolling training window of 5 years, which corresponds to approximately 1,260 trading days:

$$\mathcal{D}_{\text{train}}^{(t)} = \{(X_i, y_i) : t - 1260 \leq i < t\} \tag{IA9.1}$$

Table IA29: Toy example to illustrate generating predictions for the first few trading days of January 2020 for rolling window walk-forward validation.

Prediction Date	**Training Period (5-year rolling)**	**Action**
Jan 2, 2020	Jan 2, 2015 – Jan 1, 2020	Train model, predict Jan 2 return
Jan 3, 2020	Jan 3, 2015 – Jan 2, 2020	Retrain, predict Jan 3 return
Jan 6, 2020	Jan 6, 2015 – Jan 3, 2020	Retrain, predict Jan 6 return
⋮	⋮	⋮

One of the most important constraint of any backtesting mechanism is that the train phase has absolutely no clue of the data in the test phase. Any form of such information leakage causes look-ahead bias, leaving us with unrealistically optimistic estimates. This is done by predicting the data at a point in time only by using the data available upto that point in time.

Models using walk-forward. All 26 strategies are evaluated with walk-forward validation.

IA9.2 Time-Series Cross-Validation

Hyperparameters are tuned in each rolling training window using time series cross validation (TSCV). Note that this is different from k-fold cross-validation as this preserves temporal ordering of data.

IA9.2.1 Implementation

For a training window containing n observations, TSCV with k splits constructs folds where the training set expands over time:

$$\text{Split 1:} \quad \text{Train: } [1, n_1], \quad \text{Val: } [n_1 + 1, n_2] \tag{IA9.2}$$

$$\text{Split 2:} \quad \text{Train: } [1, n_2], \quad \text{Val: } [n_2 + 1, n_3] \tag{IA9.3}$$

$$\vdots$$

$$\text{Split } k: \quad \text{Train: } [1, n_{k-1}], \quad \text{Val: } [n_{k-1} + 1, n] \tag{IA9.4}$$

Table IA30: Toy example for a window of 1000 observations using 5 splits to illustrate time-series cross validation.

Split	Train Range	Val Range	Train/Val Size
1	Days 1–333	Days 334–466	333 / 133
2	Days 1–466	Days 467–600	466 / 134
3	Days 1–600	Days 601–733	600 / 133
4	Days 1–733	Days 734–866	733 / 133
5	Days 1–866	Days 867–1000	866 / 134

For each candidate hyperparameter configuration (for example, $\lambda \in \{0.001, 0.01, 0.1, 1.0\}$), the model is trained on each fold's training segment and evaluated on the corresponding validation segment. The optimal hyperparameters are selected using the lowest average validation error across splits.

Why not standard k-fold? Standard k-fold typically shuffles data. In financial time series, shuffling leaks information from the future into the training process. Because returns and volatility are autocorrelated, even indirect leakage (for instance, training on observations from a later volatility regime) can seriously bias evaluation and make results look much better than they would be in live trading.

Models using TSCV. Regularized models such as Ridge (Section IA5.1), Lasso (Section IA5.2), Elastic Net (Section IA5.3) use TSCV for hyperparameter selection.

IA9.3 Out-of-Fold Predictions

Out-of-fold (OOF) predictions are the validation-set forecasts produced during cross-validation. They are useful because each OOF prediction is generated by a model that did not train on that observation, so the full set of OOF predictions acts as an approximation to out-of-sample behavior.

IA9.3.1 Collection Process

After choosing hyperparameters using TSCV, OOF predictions are collected by running the cross-validation procedure again using the selected settings:

1. For each split, fit the model (with the selected hyperparameters) on the training fold.
2. Predict the outcomes in the corresponding validation fold.
3. Save predictions together with realized outcomes.
4. Concatenate predictions across splits to form a single OOF prediction set.

Table IA31: Toy example using the same 1000 observations with 5 splits for out of fold predictions

Split	OOF Range	Explanation
1	Days 334–466	Model trained on Days 1–333 forecasts Days 334–466
2	Days 467–600	Model trained on Days 1–466 forecasts Days 467–600
3	Days 601–733	Model trained on Days 1–600 forecasts Days 601–733
4	Days 734–866	Model trained on Days 1–733 forecasts Days 734–866
5	Days 867–1000	Model trained on Days 1–866 forecasts Days 867–1000
Combined OOF	Days 334–1000 (667 total predictions)	

The first 333 observations do not receive OOF predictions because they always fall inside the training portion in this expanding/forward setup. Every one of the remaining 667 observations receives exactly one OOF forecast from a model that never trained on it.

Why OOF predictions matter. OOF predictions give an unbiased basis for estimating performance and are also required for downstream calibration steps like isotonic regression (Section IA9.4). If calibration were fit using in-sample predictions, the calibrator would itself overfit the training noise and would not generalize reliably.

Models using OOF predictions. Lasso (Section IA5.2) and Elastic Net (Section IA5.3) store OOF predictions in order to run isotonic calibration.

IA9.4 Isotonic Calibration

Isotonic calibration takes the raw model predictions and reshapes them so they better match realized returns on average. It learns a monotonic mapping, so larger predicted values still correspond to larger realized returns on average. This preserves the ranking of predictions while correcting their scale, making the outputs more reliable and easier to interpret.

IA9.4.1 Theoretical Foundation

Isotonic regression finds a non-decreasing function f that minimizes squared error:

$$\hat{f} = \arg \min_{f \text{ non-decreasing}} \sum_{i=1}^{n} (y_i - f(\hat{y}_i))^2 \tag{IA9.5}$$

The monotonicity constraint guarantees that ordering is preserved, while allowing the magnitude of predictions to be adjusted to better match observed outcomes.

IA9.4.2 Implementation with OOF Predictions

To avoid leakage, calibration is trained using only OOF predictions:

1. Collect OOF forecasts $\{\hat{y}_i^{\text{OOF}}\}$ and the corresponding realized outcomes $\{y_i\}$ during TSCV.
2. Fit isotonic regression: $\hat{f} \leftarrow .\text{IsotonicRegression}(\hat{y}^{\text{OOF}}, y)$.
3. Refit the final forecasting model on the full training window.
4. For a new prediction $\hat{y}_{\text{new}}$, apply calibration:

$$\hat{y}_{\text{calibrated}} = \hat{f}(\hat{y}_{\text{new}}).$$

Table IA32: Example of conversion from OOF predictions to calibrated predictions using isotonic calibration.

OOF Prediction	Actual Return	Calibrated Prediction
−0.002	−0.008	−0.006
−0.001	−0.003	−0.002
0.000	0.001	0.000
0.001	0.005	0.003
0.002	0.010	0.007

In this example, the calibration learns that the model systematically underestimates the magnitude of returns (a pattern that is consistent with coefficient shrinkage under L1 regularization). The mapping expands predictions so they better align with realized outcomes, while still keeping the monotone ordering intact.

Why isotonic calibration helps regularized models. Regularized models shrink coefficients to zero, which also compresses predicted returns towards zero. This can reduce variance and improve generalization, but it introduces bias magnitude of forecasts. Isotonic calibration corrects this bias while retaining the benefits of regularization.

Models using isotonic calibration. Lasso (Section IA5.2) and Elastic Net (Section IA5.3) apply OOF isotonic calibration by default.

IA9.5 Validation Summary

Overall, the validation setup can be viewed as a set of nested layers:

Complete example. For Elastic Net producing a forecast for January 2, 2020:

1. **Walk-forward:** set the training window to Jan 2015 – Dec 2019.
2. **TSCV:** within that window, run 5-split TSCV to choose λ and α.

Table IA33: Summary of the validation setup.

Layer	Method	Role
Outer loop	Walk-forward (Section IA9.1)	Emulates live trading and blocks look-ahead
Inner loop	Time-series CV (Section IA9.2)	Tunes hyperparameters inside each training window
Calibration	OOF + isotonic (Sections IA9.3, IA9.4)	Improves prediction reliability while avoiding leakage

3. **OOF collection:** rerun TSCV with the chosen parameters to generate OOF predictions.
4. **Isotonic fit:** train the isotonic calibrator using OOF predictions and realized returns.
5. **Final model:** refit Elastic Net on the full training window with the selected parameters.
6. **Predict:** generate the raw return prediction for Jan 2, 2020.
7. **Calibrate:** apply the isotonic mapping to produce the calibrated prediction.
8. **Signal:** convert the calibrated output into a trading signal (Section IA7).
9. **Advance:** move to Jan 3, 2020 and repeat the entire procedure.

This layered structure ensures that hyperparameters, calibration, and forecasts are all produced and evaluated without using future information.

IA10 Summary of All Strategies

Table IA34: Complete Strategy Summary

No.	Strategy	Category	Key Hyperparameters
1	Buy-and-Hold	Benchmark	—
2	ARIMA	Statistical/Econometric	$p \leq 3$, $d \leq 2$, $q \leq 3$, CV-optimized
3	Exponential Smoothing	Statistical/Econometric	Additive trend/seasonal, period 12
4	Risk-Adjusted ARIMA	Statistical/Econometric	ARIMA + GARCH(1,1) volatility
5	Linear Regression	Statistical/Econometric	OLS with isotonic calibration
6	Kalman Filter	Statistical/Econometric	$Q = 0.01$, $R = 1.0$, EM iterations 10
7	HMM	Statistical/Econometric	3 states, 10-day min spacing
8	Ridge	Classical ML	$\lambda \in \{0.001, ..., 1000\}$, CV-optimized
9	Lasso	Classical ML	$\lambda \in \{0.0001, ..., 1.0\}$, CV-optimized
10	Elastic Net	Classical ML	λ, α CV-optimized
11	PCA Regression	Classical ML	$k \in \{3, 5, 10, 15, 20, 30\}$, CV-optimized
12	K-Nearest Neighbors	Classical ML	$K \in \{3, 5, 7, 10, 15\}$, CV-optimized
13	SVM	Classical ML	Kernel $\in$ {rbf, linear}, C CV-optimized
14	Decision Tree	Classical ML	Depth $\in \{3, 5, 10, 15, \text{None}\}$, CV-optimized
15	Random Forest	Classical ML	Trees $\in \{50, 100, 200\}$, CV-optimized
16	Gradient Boosting	Classical ML	Monotonic constraints, CV-optimized
17	Neural HMM	Advanced	3 states, [64,32]/[32,16] networks
18	LSTM	Advanced	64 units, 15-day lookback
19	Enhanced LSTM	Advanced	[24,16,16] units, 5/10/15-day windows
20	PatchTST	Advanced	Patch 4, stride 2, 8 heads, 3 layers

Table IA34 – continued from previous page

No.	Strategy	Category	Key Hyperparameters
21	TFT	Advanced	4 heads, hidden 64, 2 LSTM layers
22	Non-Stationary Transformer	Advanced	4 heads, 2 layers, d=128
23	Online Learning	Advanced	SGD + PA + RF ensemble, drift detection
24	MAML	Advanced	[64,32,16] MLP, 5 inner steps
25	Robust Mixture of Experts	Advanced	3 experts, [16,8] gating
26	Ensemble Meta-Learning	Advanced	LSTM + ARIMA + ES, perf-weighted

Internet Appendix B: Formal AMH sign map

This appendix records the formal derivative-based sign restrictions that define our Adaptive Markets Hypothesis (AMH) test. For each strategy S and month t, we consider the outcome vector

$$Y_{S,t} = (\alpha_{S,t},\ SR_{S,t},\ \sigma^2_{S,t},\ ES_{q,S,t},\ MDD_{S,t,h},\ \beta_{S,t},\ w^M_{S,t},\ \rho_t(S,S')),$$

and the ecology vector Z_t comprising Crowding, Funding, Liquidity-Variance-Trend (LVT) variables, and trading costs. We denote by $\mathbb{E}[\cdot \mid Z_t]$ the conditional expectation given Z_t, and by $\partial/\partial Z_t$ partial derivatives with respect to components of Z_t.

IB1.1 Unified EMH benchmark

Our EMH benchmark permits time-varying betas, stochastic volatility, and a state-independent cost model calibrated from microstructure. It rules out only incremental predictive content of the ecology vector for risk-adjusted outcomes:

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1} \mid Z_t]}{\partial Z_t} = 0, \quad \frac{\partial \mathbb{E}[SR_{S,t+1} \mid Z_t]}{\partial Z_t} = 0, \quad \frac{\partial \mathbb{E}[\sigma^2_{S,t+1}, ES_{q,S,t+1}, MDD_{S,t+1,h} \mid Z_t]}{\partial Z_t} = 0, \tag{IB1.1}$$

and

$$\frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial Z_t} = 0, \quad \frac{\partial \rho_t(S,S')}{\partial Z_t} = 0. \tag{IB1.2}$$

Finding nonzero coefficients on Z_t in the unified outcome system after controlling for time-varying betas, volatility, and state-independent costs is therefore interpreted as evidence for AMH-style state dependence.

IB1.2 Crowding

Let Crowding_t summarize the degree to which capital is concentrated in equity-oriented strategies. Under AMH, greater crowding compresses conditional edge and synchronizes flows, altering both performance and the shape of risk.

Performance (alpha and Sharpe). Crowding lowers net implementable edge through price impact and adverse selection:

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1} \mid Z_t]}{\partial \text{Crowding}_t} < 0, \qquad \frac{\partial \mathbb{E}[SR_{S,t+1} \mid Z_t]}{\partial \text{Crowding}_t} < 0. \tag{IB1.3}$$

Each unit of capital moves prices more, so the same signal buys less edge and costs more to execute due to decreasing returns to scale.

Tails and drawdown depth. Synchronized positioning steepens left tails and deepens drawdowns:

$$\frac{\partial \mathbb{E}[\sigma^2_{S,t+1} \mid Z_t]}{\partial \text{Crowding}_t} > 0, \quad \frac{\partial \mathbb{E}[ES_{q,S,t+1} \mid Z_t]}{\partial \text{Crowding}_t} > 0, \quad \frac{\partial \mathbb{E}[MDD_{S,t+1,h} \mid Z_t]}{\partial \text{Crowding}_t} > 0. \tag{IB1.4}$$

In shocks, many portfolios attempt to exit simultaneously; inventory constraints and impact amplify losses, pushing mass from the center to the left tail.

Co-movement. Common constraints and overlapping books raise co-movement across strategies:

$$\frac{\partial \rho_t(S,S')}{\partial \text{Crowding}_t} > 0. \tag{IB1.5}$$

Crowding predicts correlation spikes as risk is taken off (or added) in tandem.

IB1.3 Funding stress

Let Funding_t summarize financing and balance-sheet constraints faced by intermediaries and leveraged traders (e.g., funding spreads, haircuts, prime-broker leverage, margin conditions). Higher values indicate tighter funding.

Performance. Tighter funding compresses implementable edge for capital-intensive strategies and raises shorting and financing costs:

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1} \mid Z_t]}{\partial \text{Funding}_t} < 0, \qquad \frac{\partial \mathbb{E}[SR_{S,t+1} \mid Z_t]}{\partial \text{Funding}_t} < 0. \tag{IB1.6}$$

Risk and tails. Funding stress worsens variance and downside risk through forced de-risking and fire sales:

$$\frac{\partial \mathbb{E}[\sigma^2_{S,t+1} \mid Z_t]}{\partial \text{Funding}_t} > 0, \quad \frac{\partial \mathbb{E}[ES_{q,S,t+1} \mid Z_t]}{\partial \text{Funding}_t} > 0, \quad \frac{\partial \mathbb{E}[MDD_{S,t+1,h} \mid Z_t]}{\partial \text{Funding}_t} > 0. \tag{IB1.7}$$

Exposures and co-movement. Funding constraints lower effective risk-bearing capacity and synchronize portfolio adjustments:

$$\frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial \text{Funding}_t} < 0, \qquad \frac{\partial \rho_t(S, S')}{\partial \text{Funding}_t} > 0. \tag{IB1.8}$$

IB1.4 Trading environment: liquidity, volatility composition, and trend

Let the LVT block comprise liquidity (LIQ), trend variance and jump variance, and a slow-moving trend level. We denote by LVT_t the vector of these components.

Performance. Continuation-type timing rules perform best when volatility is dominated by trend-consistent moves and liquidity is abundant. AMH predicts

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1}, SR_{S,t+1} \mid Z_t]}{\partial \text{Trend Variance}_t} > 0, \quad \frac{\partial \mathbb{E}[\alpha_{S,t+1}, SR_{S,t+1} \mid Z_t]}{\partial \text{Jump Variance}_t} < 0, \quad \frac{\partial \mathbb{E}[\alpha_{S,t+1}, SR_{S,t+1} \mid Z_t]}{\partial \text{LIQ}_t} > 0, \tag{IB1.9}$$

with the sign on the slow trend level depending on the strategy's directional tilt.

Risk and tails. Illiquid, jump-dominated, high-volatility regimes raise variance and downside risk:

$$\frac{\partial \mathbb{E}[\sigma^2_{S,t+1}, ES_{q,S,t+1}, MDD_{S,t+1,h} \mid Z_t]}{\partial \text{Jump Variance}_t} > 0, \qquad \frac{\partial \mathbb{E}[\sigma^2_{S,t+1}, ES_{q,S,t+1}, MDD_{S,t+1,h} \mid Z_t]}{\partial \text{LIQ}_t} < 0. \tag{IB1.10}$$

Trend variance enters with a positive sign for variance but an ambiguous sign for downside risk, reflecting the trade-off between smoother paths within regimes and sharper reversals at boundaries.

Exposures and co-movement. Strategies scale up risk in benign, trend-consistent environments and scale down in turbulent regimes:

$$\frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial \text{Trend Variance}_t} > 0, \quad \frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial \text{Jump Variance}_t} < 0, \quad \frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial \text{LIQ}_t} > 0, \tag{IB1.11}$$

and shocks to LVT variables raise co-movement:

$$\frac{\partial \rho_t(S, S')}{\partial \text{Jump Variance}_t} > 0, \qquad \frac{\partial \rho_t(S, S')}{\partial \text{LIQ}_t} < 0. \tag{IB1.12}$$

IB1.5 Implementation frictions

Let Cost_t denote trading costs, proxied by the quoted bid–ask spread and closely related microstructure variables.

Performance. Higher costs directly erode net performance for any given gross signal:

$$\frac{\partial \mathbb{E}[\alpha_{S,t+1} \mid Z_t]}{\partial \text{Cost}_t} < 0, \qquad \frac{\partial \mathbb{E}[SR_{S,t+1} \mid Z_t]}{\partial \text{Cost}_t} < 0. \tag{IB1.13}$$

Risk, exposures, and co-movement. Cost spikes tend to coincide with microstructure stress and de-risking:

$$\frac{\partial \mathbb{E}[\sigma^2_{S,t+1}, ES_{q,S,t+1}, MDD_{S,t+1,h} \mid Z_t]}{\partial \text{Cost}_t} \geq 0, \qquad \frac{\partial \mathbb{E}[\beta_{S,t+1}, w^M_{S,t+1} \mid Z_t]}{\partial \text{Cost}_t} < 0, \tag{IB1.14}$$

and

$$\frac{\partial \rho_t(S, S')}{\partial \text{Cost}_t} > 0. \tag{IB1.15}$$

Taken together, these channel-specific sign restrictions define the AMH alternative to the EMH benchmark in (IB1.1). Our empirical tests in the main text evaluate the extent to which realized coefficients on the ecology variables in the unified outcome system align with this sign map.

References

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Bates, J. M. and Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Society*, 20(4):451–468.

Bifet, A. and Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 443–448.

Bishop, C. M. (1994). Mixture density networks. Technical Report NCRG/94/004, Aston University.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327.

Box, G. E. P. and Jenkins, G. M. (1970). *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Brown, R. G. (1959). *Statistical Forecasting for Inventory Control*. McGraw-Hill Book Company, Inc, New York, 1st edition.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, 50(4):987–1007.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1126–1135.

Fix, E. and Hodges, J. L. (1989). Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical report, International Statistical Review.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Harvey, A. C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, Cambridge.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Holt, C. C. (1957). Forecasting seasonals and trends by exponentially weighted moving averages. *Office of Naval Research Memorandum*, (52).

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45.

Kalman, R. E. and Bucy, R. S. (1961). New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 83(1):95–108.

Kelly, J. L. (1956). A new interpretation of information rate. *Bell System Technical Journal*, 35(4):917–926.

Lim, B., Arik, S. O., Loeff, N., and Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764.

Liu, Y., Wu, H., Wang, J., and Long, M. (2022). Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35:9881–9893.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. *International Conference on Learning Representations (ICLR)*, pages 1–27.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Qin, Y., Song, D., Cheng, H., Cheng, W., Jiang, G., and Cottrell, G. W. (2017). A dual-stage attention-based recurrent neural network for time series prediction. IJCAI'17, page 2627–2633. AAAI Press.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Shalev-Shwartz, S. (2012). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G., and Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*.

Thorp, E. O. (2006). The kelly criterion in blackjack, sports betting, and the stock market. *International Journal of Game Theory*, 10(1):35–44. A version of his work from the early 1960s.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer.

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science*, 6(3):324–342.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2):241–259.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.