

Article

Leakage-Controlled Horizon-Specific Model Selection for Daily Equity Forecasting: An Automated Multi-Model Pipeline

Francisco Augusto Nuñez Perez ^{1,*} , Francisco Javier Aguilar Mosqueda ¹ , Adrian Ramos Cuevas ¹ ,
Jaqueline Muñoz Beltrán ¹  and Jose Cruz Nuñez Perez ² 

¹ Universidad Politécnica de Lázaro Cárdenas, Lázaro Cárdenas 60998, Mexico

² Centro de Investigación y Desarrollo de Tecnología Digital (CITEDI), Instituto Politécnico Nacional, Av. Instituto Politécnico Nacional No. 1310, Tijuana 22435, Mexico; jnunez@ipn.mx

* Correspondence: augusto@uplc.edu.mx; Tel.: +52-753-136-42424

Highlights

What are the main findings?

- A fully automated, leakage-controlled multi-model pipeline is introduced, enforcing strict temporal causality (including explicit purging for multi-day targets) and homologated evaluation across baselines, XGBoost, LSTM, and CNN/TCN.
- On daily data for MT, DELL, and the S&P 500 ($\sim$ SPX) through 3 February 2026, price-level accuracy is similarly strong at $H = 1$, largely due to persistence in the price process, whereas return-space predictive performance remains weak and horizon-dependent.

What are the implications of the main findings?

- Horizon-specific model and lookback selection provide a more robust deployment rule than adopting a single global architecture across horizons and assets.
- The framework outputs auditable artifacts (plots, metrics, predictions, and run manifests), enabling reproducible monitoring and fair comparison under leakage-safe validation.

Abstract

Short-horizon equity forecasting remains challenging because daily prices are noisy, heavy-tailed, and subject to structural breaks and regime shifts. We develop a fully automated, reproducible, and leakage-controlled multi-model pipeline for daily forecasting with horizon-specific configuration selection. The task is formulated as predicting cumulative H -day log-returns from OHLCV-derived information and converting them to implied price forecasts. All model families share a homologated design: causal feature construction, a strictly chronological split with an explicit purging rule to prevent label-window overlap for multi-day targets, training-only robustification (winsorization and adaptive clipping), and a unified metric suite computed consistently in return and price spaces. The framework benchmarks transparent baselines (zero- and mean-return), gradient-boosted trees (XGBoost), and deep temporal models (LSTM and CNN/TCN). Lookback length $L \in \{60, 180, 500\}$ is selected via an internal walk-forward procedure on the pre-evaluation block, and final performance is reported on an external hold-out segment (last 15% of instances). Experiments on daily data for MT, DELL, and the S&P 500 index (through 3 February 2026) show that all families achieve similarly strong price-level fit at $H = 1$, largely driven by persistence in the price process, while separation across families becomes more visible at $H = 5$. However, predictive performance in return space remains weak, with R^2 close to zero or negative, and Diebold–Mariano tests do not provide consistent evidence of statistical superiority over naive benchmarks. Under an operational rule that minimizes hold-out RMSE on the price scale, selected models are asset- and



Academic Editor: Piotr Fiszeder

Received: 21 February 2026

Revised: 12 April 2026

Accepted: 14 April 2026

Published: 20 April 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

horizon-dependent, supporting horizon-wise selection rather than a single global architecture. Overall, the primary contribution lies in the proposed leakage-controlled evaluation and benchmarking framework rather than in demonstrating consistent predictive gains in financial time series forecasting.

Keywords: equity forecasting; time-series prediction; leakage control; purging; walk-forward validation; horizon-specific model selection; OHLCV; technical indicators; LSTM; temporal convolutional networks; XGBoost; out-of-sample evaluation

1. Introduction

Short-horizon equity forecasting remains a difficult problem in applied time-series analysis because daily market data are noisy, heavy-tailed, and repeatedly affected by structural breaks, regime shifts, and evolving market conditions [1,2]. Under such non-stationarity, performance can vary sharply across assets and horizons, and no single modeling paradigm is uniformly superior; moreover, simple combinations or robust baselines can be competitive when regimes change [3].

A large and heterogeneous literature has explored linear and non-linear statistical learning methods and, more recently, deep neural architectures for forecasting prices or returns. Surveys and comparative studies report broad experimentation with artificial neural networks, support vector machines, and LSTM variants [4,5], while architecture-focused contributions propose specialized CNN-(Bi)LSTM pipelines for next-day prediction [6,7] and related deep-learning designs for commodity or index forecasting [8]. Other streams incorporate sentiment from news or social media into recurrent or convolutional models [9], optimize technical analysis rules with machine learning [10], or reinterpret moving-average forecasting as an adaptive weighting problem using attention-based mechanisms [11]. Despite these advances, results are often difficult to compare because studies typically differ in asset universes, horizons, feature definitions, training/validation designs, and reporting conventions, which can blur whether observed gains reflect genuine improvements or evaluation artifacts.

A recurring limitation in prior empirical studies is that competing models are often not evaluated under a fully standardized protocol: chronological splits, purging rules for overlapping targets, preprocessing fitted on training data only, and reporting conventions frequently differ across model families. This makes it difficult to determine whether reported gains reflect genuine predictive improvements or differences in evaluation design. The present study addresses this gap by enforcing a homologated protocol across baselines, XGBoost, LSTM, and CNN/TCN, so that cross-model differences are interpreted under a common leakage-controlled framework.

A central methodological issue is that time-series forecasting is especially vulnerable to information leakage and optimistic bias if temporal causality is not enforced throughout preprocessing, model selection, and performance assessment. Small design choices, for example, fitting scalars using future observations, selecting hyperparameters using the evaluation segment, or ignoring the overlap structure of multi-day targets can materially distort out-of-sample conclusions. Recent evidence in AI-driven financial forecasting also highlights that a large share of empirical studies still under-report validation details and often fails to assess temporal robustness across market regimes, which can inflate reported accuracy and yield overstated conclusions [12]. Recent statistical evaluations further emphasize that robust conclusions require homogeneous protocols, careful temporal splitting, principled model comparison, and formal significance assessment when competing fore-

casts are close in accuracy [13,14]. Accordingly, operationally useful forecasting systems should provide auditable, time-respecting validation procedures and transparent baselines, rather than relying on architecture-specific claims.

At the same time, the practical context of equity forecasting increasingly demands not only point forecasts, but also (i) coherent, comparable summaries of out-of-sample performance across horizons and assets; (ii) robust benchmarking against transparent baselines; and (iii) interpretable outputs that can be monitored and audited over time. These requirements are particularly stringent in multi-asset settings, where volatility profiles, liquidity regimes, and shock sensitivities differ across instruments. Interpretability also matters for downstream decision making, as user behavior and financial literacy can influence how forecast information is consumed and acted upon [15]. Moreover, directional evaluation must be treated with care, because small deviations from chance-level hit rates can be difficult to interpret without appropriate accuracy assessment and uncertainty considerations [16]. Any short-horizon improvement in price-level accuracy should nevertheless be interpreted cautiously, since the present study does not evaluate directional predictability, trading rules, transaction costs, or risk-adjusted economic value.

Forecast horizon is a further first-order dimension. Evidence from short-dated S&P 500 index options suggests horizon-dependent risk pricing and steep changes in effective risk aversion across short maturities [17]. Related preferred-habitat mechanisms in term-structure models show that maturity-segmented demand can generate horizon-specific premia [18]. Multi-horizon forecast comparison procedures and heterogeneous-horizon asset-pricing evidence likewise support the view that both predictability and the economic environment can differ by horizon [13,19,20]. These findings motivate horizon-wise evaluation and selection, which is a configuration that performs well at $H = 1$ may be suboptimal at $H = 5$, and vice versa.

Although the framework supports longer horizons, we report results only for $H \in \{1, 5\}$ to focus on short-horizon daily forecasting (next day and one trading week).

Motivated by these considerations, we propose an automated multi-model framework for daily equity forecasting and horizon-specific model selection under a strictly time-respecting, leakage-controlled evaluation design. The framework orchestrates complementary predictors—gradient-boosted decision trees (XGBoost), LSTM networks, convolutional/temporal convolutional models (CNN/TCN), and moving-average baselines—all operating on a common representation derived from daily OHLCV data and causally computed technical features. A key design principle is homologation, where all model families share the same target definition, the same chronological splitting logic with an explicit purging gap, a unified preprocessing schema, and a common metric suite computed consistently in return and implied price spaces. Within this controlled setting, we perform horizon-wise selection based on aggregated out-of-sample evidence, aligning with data-driven selection principles in complex time-series settings [21].

This study is guided by two working hypotheses. First, when heterogeneous predictors are evaluated under a common, rigorous protocol, tree-based ensembles with carefully engineered, causally computed features can match or outperform deep sequence models for short-horizon level forecasting, while deep temporal models may be competitive in calibration for specific assets or horizons. Second, performance is inherently horizon-dependent; therefore, explicit horizon-specific model selection driven by out-of-sample metrics yields more robust and interpretable forecasting rules than adopting a single global architecture across horizons.

The main contributions of this paper are as follows:

- An auditable, end-to-end forecasting pipeline with explicit leakage control (including purging for multi-day targets) and homogeneous evaluation across assets, horizons, and model families;
- A systematic, horizon-wise benchmarking of boosted trees, LSTM, CNN/TCN, and moving-average baselines on a diversified equity universe using only OHLCV-derived information;
- An operational horizon-specific selection procedure supported by unified metric reporting and reproducible artifacts suitable for monitoring and deployment.

The remainder of the paper is organized as follows: Section 2 describes the data, target definition, feature construction, and the leakage-controlled training/validation protocol. Section 3 details the model families and implementation choices. Section 4 reports out-of-sample results and horizon-wise model selection patterns, and Section 5 discusses implications, limitations, and directions for future work.

2. Materials and Methods

2.1. Overall Architecture of the Forecasting Pipeline

We implement a reproducible, fully automated multi-model forecasting pipeline whose core objectives are as follows: (i) acquire and curate daily market data; (ii) transform each asset time series into supervised learning instances for multiple horizons and lookback lengths; (iii) train, validate, and benchmark heterogeneous model families under a unified, leakage-controlled chronological protocol; and (iv) generate and persist current forecasts together with diagnostics, metadata, and error metrics.

All components are implemented in Python 3.10.11 using standard scientific and machine-learning libraries (NumPy, pandas, TensorFlow/Keras, and XGBoost). The pipeline is organized as modular scripts that can be executed sequentially (manual runs) or via a scheduled daily job. Each run produces an auditable set of artifacts written to disk under a fixed directory structure, including updated OHLCV files, serialized model objects, per-run metadata (hyperparameters, feature names and training configuration), validation predictions, future forecasts, and aggregated summary tables.

A key design principle is homologation, where all model families share the same target definition, the same time-respecting splitting logic (with explicit purging to avoid label-window overlap), a common metric suite computed in both return and price spaces, and a standardized artifact schema. This ensures that performance differences are attributable to modeling choices rather than to inconsistent preprocessing or evaluation.

2.2. Data Sources and Universe of Assets

The empirical analysis uses daily OHLCV (open, high, low, close, and volume) time series for three instruments available in the current study universe: DELL (Dell Technologies, US equity), MT (ArcelorMittal, US listing), and the S&P 500 index (downloaded as `~SPX` from Stooq and stored locally as `data_spx500.csv`). Historical prices are obtained through programmatic access to public market-data providers, with Stooq used as the primary source and Yahoo Finance as a fallback for tickers or periods not covered by the primary provider. Experiments are conducted using a nominal start date of 2 January 2018 whenever available. In the current implementation, the two equity series (MT and DELL) effectively follow this recent window design, whereas the final `~SPX` build retains a longer historical span because the local index file preserved the full available provider history during data consolidation. Accordingly, each asset is evaluated under the same leakage-controlled protocol, but cross-asset comparisons should not be interpreted as perfectly matched in calendar span.

All time series are stored locally as CSV files under a unified schema with required columns Date, Open, High, Low, Close, and Volume. The trading-day index excludes weekends and exchange holidays by construction. Core experiments evaluate horizons $H \in \{1, 5\}$ trading days (next day and next week).

Table 1 summarizes the effective sample period and number of observations per asset in the current implementation.

Table 1. Summary of the daily OHLCV datasets used in the current implementation. All models are evaluated at forecast horizons $H \in \{1, 5\}$.

Ticker	Asset	Market	Start Date	End Date (N_{obs})
DELL	Dell Technologies Inc.	US (Equity)	31 December 2018	3 February 2026 (1783)
MT	ArcelorMittal	US (Equity)	2 January 2018	3 February 2026 (2033)
^SPX	S&P 500 index	US (Index)	2 January 2018	3 February 2026 (2033)

2.3. Data Acquisition, Cleaning, and Incremental Updates

Daily OHLCV series are downloaded programmatically using custom Python scripts. All downloads are normalized to the common OHLCV schema, and timestamps are converted to time-zone-naïve dates.

A sanitization routine enforces consistency constraints: dates are parsed into `datetime` objects, sorted in ascending order, and deduplicated by Date. Non-numeric values in OHLCV fields are coerced to missing values. Rows with missing Date or Close are removed. If a Volume column is absent, it is created and set to zero to preserve a uniform schema across assets. Remaining missing OHLCV values are handled by forward fill; any residual missing rows are dropped. When used as an input feature, volume is stabilized via the transform $\log(1 + V_t)$.

Updates are incremental. For each asset, the system reads the existing local CSV file, identifies the last available date, and requests data from the next calendar day up to the current date in the local time zone (America/Mexico_City). Newly downloaded data undergo the same sanitization and are appended to the local file, followed by a Date-keyed deduplication. If a local file is missing or corrupted, the dataset is reconstructed from the nominal start date. This mechanism keeps all assets current while maintaining consistent, well-formed inputs across executions [22].

2.4. Forecasting Tasks and Target Definition

For each asset and horizon H (trading days), the primary forecasting task is to predict the cumulative H -day log-return from time t to $t + H$. Let P_t denote the closing price on trading day t with $P_t > 0$. The one-day log-return is

$$r_t = \log\left(\frac{P_t}{P_{t-1}}\right), \quad t = 2, \dots, T, \quad (1)$$

and the H -day cumulative log-return is

$$y_t^{(H)} = \sum_{k=1}^H r_{t+k} = \log\left(\frac{P_{t+H}}{P_t}\right), \quad t = 1, \dots, T - H. \quad (2)$$

All model families operate in a return-target mode by default, producing a prediction $\hat{y}_t^{(H)}$. The implied forecast on the price scale is recovered through

$$\hat{P}_{t+H} = P_t \exp(\hat{y}_t^{(H)}). \quad (3)$$

2.5. Feature Construction and Supervised Instance Generation

From each cleaned OHLCV series, supervised learning instances are built using a sliding-window scheme. For a lookback length L , each instance uses information available up to the anchor time t to predict the target at $t + H$. The framework supports lookback families $L \in \{60, 180, 500\}$ to capture short-, medium-, and long-memory regimes.

In the current implementation, the feature set is composed of causally computed technical descriptors derived from the close series (and, when enabled, a stabilized volume transform). Specifically, we compute the following: one-, 5-, and 20-day log returns; simple momentum ratios; rolling volatility proxies (standard deviation of 1-day log returns); relative moving-average deviations; and the log-close level. Rolling indicators are computed without backward imputation; warm-up periods required by rolling operators are handled by discarding initial rows with undefined indicators, thereby preventing any look-ahead bias.

Let $\mathbf{x}_s \in \mathbb{R}^d$ denote the feature vector at time s . A windowed instance is

$$\mathbf{X}_t = \begin{bmatrix} \mathbf{x}_{t-L+1}^\top \\ \vdots \\ \mathbf{x}_t^\top \end{bmatrix} \in \mathbb{R}^{L \times d}, \quad t = L, \dots, T - H. \quad (4)$$

For computational efficiency and determinism, windows are generated with a vectorized stride-based routine (NumPy `sliding_window_view`) and a robust axis-order correction to guarantee the canonical shape $(n_{\text{samples}}, L, d)$.

2.6. Scaling and Robustification (Training-Only)

To stabilize training under heavily tailed returns, we apply robustification steps using training data only. Training targets are winsorized at symmetric quantiles (e.g., $q = 0.995$) to reduce the influence of extreme outliers. In addition, predicted log-returns are clipped to an adaptive bound determined from the training distribution of $|y|$ (quantile-based, capped at a maximum absolute value) to prevent implausible implied prices.

For models sensitive to feature scaling (neural networks and, optionally, tree-based models in flattened form), inputs are normalized using parameters fitted exclusively on the training portion of each time-respecting split [23]. The implemented normalization is a per-feature min-max transform fitted on training rows up to the last training anchor date, then applied unchanged to tuning, hold-out, and future-forecast windows. For tabular learners on flattened windows (e.g., XGBoost), an additional standardization (z-score) may be applied after flattening, again fitted on training data only.

2.7. Model Families

The pipeline orchestrates five model families spanning complementary inductive biases and complexity levels:

- Baselines (mean-return and zero-return). Two transparent baselines are included: (i) a historical-mean predictor that outputs the mean of the (winsorized) training targets, and (ii) a zero-return predictor ($\hat{y} = 0$), which implies $\hat{P}_{t+H} = P_t$.
- Gradient-boosted trees (XGBoost). For each asset, lookback L and horizon H , an ensemble of regression trees is trained using `xgboost.train`. The input vector is formed by flattening the L -step multivariate feature window into a tabular vector. The objective is configured to be robust (e.g., pseudo-Huber), with early stopping performed on an internal time-respecting split. Related studies apply XGBoost to financial time-series prediction and explore hybrid variants with generative adversarial

networks [24], while stacking ensembles combining tree-based models and recurrent architectures have been proposed in stock-index prediction settings [25,26].

- Long short-term memory networks (LSTM). The recurrent architecture processes sequences of length $L \in \{60, 180, 500\}$ of multivariate technical features. Training uses early stopping on an internal tuning segment carved from the training portion, retaining the epoch with the lowest tuning loss. Loss functions are robust (Huber) to reduce sensitivity to outliers.
- Convolutional and temporal convolutional networks (CNN/TCN). Convolutional models operate on causal sliding windows of technical features. A causal one-dimensional convolution is followed by a small stack of dilated temporal convolutional blocks with residual connections, and a pooling head maps the representation to a scalar log-return. Training uses the same time-respecting tuning logic and robust losses.

Hyperparameter Configuration and Tuning Protocol

Hyperparameter tuning was performed exclusively for non-baseline models (XGBoost, LSTM, and CNN/TCN) using Bayesian optimization with Optuna's Tree-structured Parzen Estimator (TPE) sampler. The optimization procedure was strictly confined to the pre-evaluation segment under a time-respecting walk-forward validation scheme.

For each asset, prediction horizon ($H \in \{1, 5\}$), model family, and candidate lookback window ($L \in \{60, 180, 500\}$), hyperparameters were calibrated by minimizing return-space forecasting error (RMSE_y) across rolling validation folds. The lookback length itself was then selected separately within the pre-evaluation block using the walk-forward procedure described below. This ensured that model calibration remained aligned with the primary target variable of the study.

To preserve strict temporal causality and avoid information leakage, the final external hold-out set was never used during hyperparameter tuning or model selection. All optimization steps were completed exclusively within the pre-evaluation block.

Baseline models (BASE_ZERO and BASE_MEAN) do not involve hyperparameter optimization and were evaluated directly under the same leakage-controlled framework. Lookback selection remained horizon-specific and was determined within the pre-evaluation period.

All preprocessing parameters were computed using training information only, including winsorization thresholds, adaptive clipping bounds, and feature scaling parameters. For auditability and reproducibility, each run stores the selected hyperparameters, model configuration, and protocol settings in machine-readable artifacts (e.g., `config.json`) together with hold-out predictions and plots. Additional implementation details and representative configurations are summarized in Appendix A (Table A1).

2.8. Training Protocol, Walk-Forward Selection, and Leakage Control

All models are trained and evaluated using a strict chronological protocol designed to avoid look-ahead bias. For each asset, lookback L , and horizon H , the ordered supervised instances are split into the following: (i) a pre segment containing the first 85% of windows and (ii) a final hold-out segment containing the remaining 15%. To prevent leakage induced by overlapping label windows ($y_t^{(H)}$ depends on P_{t+H}), an explicit purging rule is enforced. The training portion includes only windows whose label horizon ends strictly before the first anchor of the hold-out segment.

Within the pre segment, the pipeline performs a walk-forward evaluation scheme to support both hyperparameter calibration for non-baseline models and horizon-specific selection of the lookback length. Walk-forward schemes are a standard way to evaluate models under evolving market conditions while maintaining strict temporal causality, and

have been used in LSTM-based stock forecasting settings [27]. Specifically, a sequence of expanding-window folds is created, where each fold trains on an initial block and validates on the subsequent block, advancing forward in time by a fixed step. This yields fold-wise performance estimates that reflect time variation and regime shifts.

In the current implementation, the walk-forward selector uses an initial training fraction of 0.60 of the pre-evaluation block, a validation fraction of 0.12, and a forward step of 0.06, yielding between 3 and 6 expanding-window folds depending on the effective sample size after lookback and horizon constraints. If the initial step does not produce the minimum required number of folds, the step is internally reduced to increase fold count while preserving the same expanding-window logic. These settings are fixed across runs and ensure a deterministic internal selection of the lookback length before final model fitting.

After lookback selection, a final model is trained on the full pre segment using early stopping and an internal tuning slice carved from the end of the training portion (time-respecting). A light bias correction can be applied by estimating the mean prediction error on the internal tuning slice and adding the negative of this bias to the hold-out and future predictions. All transformations (winsorization parameters, clipping bounds, and scaling parameters) are computed using training and internal tuning data only.

Compact Pipeline Summary

For each asset and forecast horizon H : (1) construct causally computed supervised windows for each $L \in \{60, 180, 500\}$; (2) split the ordered instances into a pre-evaluation segment (85%) and an external hold-out segment (15%) with explicit purging at the boundary; (3) within the pre-evaluation segment, apply an expanding-window walk-forward selector to choose the lookback length for each model family; (4) train the final model on the pre-evaluation segment using a time-respecting internal validation slice and early stopping; (5) generate predictions on the external hold-out segment; (6) compute metrics in target and implied-price spaces; and (7) store models, predictions, metrics, plots, and machine-readable run configuration files as auditable artifacts.

For every combination of asset, horizon, lookback, and model family, the pipeline stores: (i) serialized model objects and metadata; (ii) hold-out predictions in return and price spaces; (iii) per-run metric files (CSV/JSON); and (iv) run manifests linking each trained configuration to its artifacts (models, metrics, predictions, and plots).

2.9. Evaluation Metrics

For every asset, horizon, and model, we compute metrics on the final hold-out segment in both target space (y) and implied price space (P). The core suite includes

- RMSE and R^2 in target and price spaces;
- SMAPE (scale-free relative error) in target and price spaces;
- LogRMSE on prices, computed as RMSE on $\log(P)$ to emphasize relative price accuracy;
- NRMSE on prices, defined as $\text{RMSE}/\mathbb{E}[P]$ over the evaluated segment for comparability across instruments.

2.10. Model Comparison, Horizon-Specific Selection, and Future Forecast Generation

All per-run metric files are aggregated into unified summary tables indexed by (asset, horizon, model family, lookback). Within each model family and horizon, candidate lookback values are evaluated within the pre-evaluation block under the walk-forward protocol. For non-baseline models, hyperparameters are calibrated via Bayesian optimization the best-performing configurations are identified by their price-space errors (e.g., LogRMSE and RMSE) together with stability indicators. After training, each model configuration generates a one-step-ahead future forecast using the most recent available date as the anchor time t .

The target date is computed as the H -th subsequent trading day, and the predicted price $\hat{P}_{t+H}$ is stored together with the last observed price P_t and identifying tags (model family and lookback). A dedicated aggregation step consolidates the latest forecasts into a single comparison table indexed by (asset, H , target date), maintained both as a full historical log and as a compact “latest” view (one entry per asset and horizon).

2.11. Software Implementation and Automation

The pipeline can be executed automatically via a scheduler that triggers a daily job at a fixed local time. Each execution performs the following: (i) incremental update (or repair) of local datasets; (ii) training and hold-out evaluation of all model families under the current configuration; (iii) generation of new future predictions for each asset and horizon; and (iv) aggregation of metrics and forecasts into unified summary tables. All artifacts (datasets, models, metadata, hold-out predictions, metric summaries, manifests, and forecast tables) are written to disk under a fixed directory structure to ensure traceability and reproducibility. Experiments are configured via lightweight settings controlling the asset universe, horizons, lookback lengths, and training hyperparameters, enabling repeated experimentation and deployment without modifying the core code base.

3. Results

Predictive performance in return space remains weak across all assets, with R^2 values close to zero or negative. Diebold–Mariano tests do not provide consistent evidence of statistical superiority over naive benchmarks. These results indicate that apparent accuracy in price space, largely driven by persistence in the price process, does not translate into genuine predictive skill.

This section reports out-of-sample performance on the external evaluation block (the last 15% of the supervised instances), using a horizon-dependent purging gap of length H trading days between the training and evaluation segments. All results correspond to the datasets available up to 3 February 2026.

The empirical universe in the current implementation comprises three instruments: ArcelorMittal (MT), Dell Technologies (DELL), and the S&P 500 index ($\sim$ SPX). For each model family and each (asset, H), the lookback length $L \in \{60, 180, 500\}$ is selected using an internal walk-forward procedure on the pre-evaluation block, prioritizing stability on the price scale by minimizing the mean LogRMSE (with mean R^2 as a secondary tie-breaker). After selecting L , we report performance on the external hold-out block.

Unless stated otherwise, metrics are computed on the implied price scale, RMSE is measured in price units, SMAPE is reported in percent, and R^2 is computed on the price scale. Given the small universe and the robustification mechanisms used in the pipeline (winsorization and adaptive clipping applied using training information only), no additional stability filtering is applied in this section.

For compactness, the main summary tables report Baseline (0) as the most transparent operational reference. The historical-mean baseline is nevertheless part of the evaluated benchmark set, is documented in Appendix B (Figures A1–A30) through the complete external hold-out plots, and is preserved in the stored run artifacts.

3.1. Next-Day Forecasts ($H = 1$)

At $H = 1$, all learning-based families and the transparent baselines deliver very similar price-level accuracy on this three-instrument universe. Price-space R^2 remains uniformly high, and relative errors are low, reflecting the strong persistence of daily price levels at the next-day horizon. Differences in RMSE and SMAPE across model families are therefore small and should be interpreted as marginal within this limited universe.

These high R^2 values are largely driven by the strong persistence of financial price series and should not be interpreted as evidence of true predictive ability. Accordingly, price-level metrics are treated as descriptive rather than indicative of genuine forecasting performance. However, return-space R^2 remains close to zero across all assets, indicating limited predictive power.

Cross-asset comparisons at $H = 1$ require care because RMSE is scale-dependent. In particular, the larger RMSE observed for $\hat{\text{SPX}}$ mainly reflects the substantially higher price level of the index, rather than poorer relative predictive performance. Accordingly, cross-asset interpretation should rely primarily on scale-free or normalized measures such as SMAPE, NRMSE, and LogRMSE. Under these relative metrics, next-day errors remain of comparable order across the evaluated assets, so the high absolute RMSE of the index should be read mainly as a scale effect rather than as evidence that $\hat{\text{SPX}}$ is intrinsically harder to forecast in relative terms. Table 2 reports the corresponding results.

Table 2. Out-of-sample next-day ($H = 1$) performance across model families for the three-instrument universe. For each (ticker, model), the table reports the selected lookback $L \in \{60, 180, 500\}$ obtained by internal walk-forward stability selection, and the corresponding external hold-out metrics on the price scale.

Ticker	Model	Lookback L	RMSE	SMAPE (%)	R^2
DELL	Baseline (0)	60	3.639	2.259	0.9647
DELL	CNN/TCN	60	3.616	2.254	0.9650
DELL	LSTM	60	3.635	2.257	0.9648
DELL	XGB	60	3.932	2.387	0.9588
MT	Baseline (0)	500	0.623	1.510	0.9910
MT	CNN/TCN	500	0.624	1.512	0.9909
MT	LSTM	500	0.623	1.509	0.9910
MT	XGB	500	0.641	1.540	0.9905
$\hat{\text{SPX}}$	Baseline (0)	60	29.914	0.782	0.9996
$\hat{\text{SPX}}$	CNN/TCN	60	29.881	0.780	0.9996
$\hat{\text{SPX}}$	LSTM	60	29.889	0.779	0.9996
$\hat{\text{SPX}}$	XGB	60	30.613	0.805	0.9996

3.2. One-Week-Ahead Forecasts ($H = 5$)

As expected, errors increase at $H = 5$ as uncertainty accumulates over the weekly horizon. Relative calibration (SMAPE) and fit (R^2) remain strong on the price scale in this universe, with the LSTM family providing the best mean SMAPE at $H = 5$ in this run, and CNN/TCN remaining competitive. However, return-space predictive performance remains weak, with R^2 values close to zero or negative, so these price-space differences should not be interpreted as evidence of robust predictive superiority. The boosted-tree configuration reported here is consistently weaker on MT and DELL at $H = 5$ in both RMSE and SMAPE.

Horizon effects and differing behavior of RMSE vs. SMAPE. As the horizon increases, absolute errors (RMSE) naturally accumulate in price units. For individual stocks (DELL and MT), both RMSE and SMAPE increase, consistent with higher uncertainty at longer horizons. For the index, RMSE can increase substantially while SMAPE remains stable (or even improves) because SMAPE is a *relative* error—when the underlying level is large and the index aggregates idiosyncratic noise, the same absolute deviation may represent a small percentage error. This also helps explain why R^2 may remain high for $\hat{\text{SPX}}$ across horizons, where the index level is highly persistent and smoother due to cross-sectional aggregation, whereas single stocks contain more idiosyncratic variation that can reduce explained variance as H grows. Therefore, horizon-dependent comparisons are

most informative when interpreted jointly through scale-dependent (RMSE) and scale-free (SMAPE/NRMSE/LogRMSE) metrics.

Table 3 reports the corresponding results.

Table 3. Out-of-sample one-week-ahead ($H = 5$) performance across model families for the three-instrument universe. The lookback L is selected by internal walk-forward stability selection, and metrics are reported on the external hold-out block on the price scale.

Ticker	Model	Lookback L	RMSE	SMAPE (%)	R^2
DELL	Baseline (0)	60	9.112	5.845	0.7046
DELL	CNN/TCN	60	9.209	6.012	0.6978
DELL	LSTM	60	9.068	5.764	0.7083
DELL	XGB	60	10.171	8.079	0.6330
MT	Baseline (0)	500	1.738	4.144	0.7716
MT	CNN/TCN	500	1.744	4.383	0.7703
MT	LSTM	180	1.736	4.121	0.7718
MT	XGB	180	1.986	5.278	0.7014
^SPX	Baseline (0)	60	61.328	0.750	0.9960
^SPX	CNN/TCN	60	61.030	0.582	0.9996
^SPX	LSTM	60	61.194	0.737	0.9985
^SPX	XGB	60	68.246	0.142	0.9944

3.3. Horizon-Specific Model Choice

We now apply the operational selection rule used to recommend a deployable forecaster per (asset, H): *minimize external hold-out RMSE on the price scale*, with SMAPE used only as a tie-breaker. Table 4 summarizes the selected model for each asset and horizon.

Table 4. Horizon-specific model selection under the operational rule: minimize external hold-out RMSE on the price scale (SMAPE as tie-breaker).

Ticker	H	Selected Model	Lookback L	RMSE	SMAPE (%)	R^2
MT	1	LSTM	500	0.623	1.509	0.9910
DELL	1	CNN/TCN	60	3.616	2.254	0.9650
^SPX	1	CNN/TCN	60	29.881	0.780	0.9996
MT	5	LSTM	180	1.736	4.121	0.7718
DELL	5	LSTM	60	9.068	5.764	0.7083
^SPX	5	CNN/TCN	60	61.030	0.582	0.9996

Under this RMSE-on-price criterion, the selected family is horizon- and asset-dependent. LSTM is selected for MT at both horizons and for DELL at $H = 5$, while CNN/TCN is selected for ^SPX at both horizons and for DELL at $H = 1$. In this run, neither baseline nor XGB is selected by the operational rule for any (asset, H) pair. However, this selection should be interpreted as an operational price-space choice rather than as evidence of consistent predictive superiority in return space.

These findings are further reinforced by additional assets reported in Appendix C (including Bitcoin, Tables A2–A5), where return-space R^2 remains close to zero or negative and Diebold–Mariano tests do not indicate consistent statistical superiority over naive benchmarks. This cross-asset evidence confirms that the limited predictive power observed is not specific to the main equity universe but reflects a broader empirical pattern.

3.4. Statistical Comparison: Diebold–Mariano Tests

To assess whether the performance differences across model families are statistically meaningful, we conduct Diebold–Mariano (DM) tests of equal predictive accuracy on

the full aligned forecast sample available for each asset–horizon–lookback configuration. For each asset and forecast horizon H , we compare model A against a reference model B (XGBoost in our main comparisons) using two loss functions: squared error in price space, $SE(P)$, and squared error in log-price space, $SE(\log P)$.

Let $d_t = L_{A,t} - L_{B,t}$ denote the loss differential. Hence, $\bar{d} < 0$ indicates that model A achieves lower loss (better accuracy) than model B on average. Because multi-step horizons induce serial correlation in forecast errors, we report two-sided DM p -values computed with a heteroskedasticity- and autocorrelation-consistent (HAC) variance estimator (Newey–West) using lag $q = H - 1$ to account for overlapping multi-step errors. Table 5 summarizes the pairwise DM results.

Table 5. Diebold–Mariano tests of equal predictive accuracy on the full aligned forecast sample available for each asset–horizon–lookback configuration. Loss differentials are defined as $d_t = L_A - L_B$; thus, $\bar{d} < 0$ indicates that model A has lower loss (better) than model B . Two-sided p -values are reported; HAC variance uses a Newey–West estimator with lag $q = H - 1$ to account for overlapping multi-step errors.

Asset	Loss	Comparison (A vs. B)	T	$\bar{d}$	DM	p -Value
Panel A: $H = 1$						
MT	$SE(\log P)$	CNN/TCN vs. XGBoost	708	−0.000013	−1.628	0.1035
MT	$SE(\log P)$	LSTM vs. XGBoost	708	−0.000015	−1.810	0.0702
DELL	$SE(\log P)$	CNN/TCN vs. XGBoost	252	−0.000120	−2.499	0.0125
DELL	$SE(\log P)$	LSTM vs. XGBoost	252	−0.000119	−2.562	0.0104
^SPX	$SE(\log P)$	CNN/TCN vs. XGBoost	5928	−0.000005	−2.927	0.0034
^SPX	$SE(\log P)$	LSTM vs. XGBoost	5928	−0.000005	−2.923	0.0035
MT	$SE(P)$	CNN/TCN vs. XGBoost	708	−0.020879	−2.652	0.0080
MT	$SE(P)$	LSTM vs. XGBoost	708	−0.023196	−2.749	0.0060
DELL	$SE(P)$	CNN/TCN vs. XGBoost	252	−2.248760	−2.559	0.0105
DELL	$SE(P)$	LSTM vs. XGBoost	252	−2.243996	−2.680	0.0074
^SPX	$SE(P)$	CNN/TCN vs. XGBoost	5928	−43.871512	−4.182	<0.0001
^SPX	$SE(P)$	LSTM vs. XGBoost	5928	−43.823051	−4.189	<0.0001
Panel B: $H = 5$						
MT	$SE(\log P)$	CNN/TCN vs. XGBoost	708	−0.000319	−2.634	0.0084
MT	$SE(\log P)$	LSTM vs. XGBoost	708	−0.000693	−3.565	0.0004
DELL	$SE(\log P)$	CNN/TCN vs. XGBoost	233	−0.003629	−3.599	0.0003
DELL	$SE(\log P)$	LSTM vs. XGBoost	233	−0.003749	−3.869	0.0001
^SPX	$SE(\log P)$	CNN/TCN vs. XGBoost	5910	−0.000078	−5.617	<0.0001
^SPX	$SE(\log P)$	LSTM vs. XGBoost	5910	−0.000077	−5.182	<0.0001
MT	$SE(P)$	CNN/TCN vs. XGBoost	708	−0.422343	−2.578	0.0099
MT	$SE(P)$	LSTM vs. XGBoost	708	−0.876202	−3.176	0.0015
DELL	$SE(P)$	CNN/TCN vs. XGBoost	233	−54.764040	−3.166	0.0015
DELL	$SE(P)$	LSTM vs. XGBoost	233	−56.183425	−3.369	0.0008
^SPX	$SE(P)$	CNN/TCN vs. XGBoost	5910	−705.874581	−5.858	<0.0001
^SPX	$SE(P)$	LSTM vs. XGBoost	5910	−676.377521	−5.546	<0.0001

Overall, the DM results corroborate that several of the performance gaps observed in RMSE/SMAPE are not merely descriptive. In particular, at $H = 5$ most comparisons against XGBoost are statistically significant on both $SE(P)$ and $SE(\log P)$ for the evaluated assets, consistent with the clearer separation across model families at the weekly horizon. At $H = 1$, significance is more mixed for some cases, which aligns with the strong persistence of daily price levels and the smaller expected performance separation at the shortest horizon.

Overall, across all assets and horizons, predictive performance in return space remains weak, and Diebold–Mariano tests do not provide consistent evidence of statistical superiority of machine learning models over naive benchmarks. Additional appendix results for assets such as Bitcoin further reinforce this interpretation, where return-space predictability remains negligible or negative, and any statistical advantages are not robust enough to support a general claim of model superiority.

Taken together, these results indicate that the predictive performance in the return space is consistently weak across all assets and horizons, and no model family demonstrates robust statistical superiority over naive benchmarks.

4. Discussion

The results highlight a practical point that is central to this study, when heterogeneous predictors are compared under a consistent, leakage-controlled chronological protocol with purging and homologated preprocessing, model choice should be made per asset and per horizon using an explicit operational criterion.

On the present three-instrument universe (MT, DELL, and $\hat{\text{SPX}}$), all families achieve strong price-level fit, especially at $H = 1$, where differences in RMSE and SMAPE are marginal. This behavior is consistent with the well-known persistence of daily price levels. Even simple baselines can appear competitive under level-forecast metrics when the horizon is short and prices are strongly autocorrelated. Consequently, small metric gaps at $H = 1$ should not be over-read as strong evidence of structural superiority.

At the weekly horizon ($H = 5$), separation across families becomes more visible on the price scale, and the sequence models reported here (LSTM and CNN/TCN) are more frequently selected by the RMSE-on-price operational rule. In particular, LSTM is selected for MT at both horizons and for DELL at $H = 5$, while CNN/TCN is selected for the index ($\hat{\text{SPX}}$) at both horizons. However, these patterns should be interpreted cautiously because return-space performance remains weak and does not support a consistent conclusion of predictive superiority. In this sense, the present results are better understood as part of a leakage-controlled benchmarking exercise than as evidence of robust forecasting gains.

The horizon- and asset-dependent lookback patterns reinforce the need to treat both H and L as first-class configuration parameters. The operationally selected configurations favor short context ($L = 60$) for DELL and $\hat{\text{SPX}}$ in most cases, while MT benefits from longer memory at $H = 1$ ($L = 500$) and a moderate memory at $H = 5$ ($L = 180$). This supports deploying a horizon-specific selection policy rather than enforcing a single architecture and lookback across all tasks.

Several limitations qualify the scope of these conclusions. First, the empirical universe is intentionally small in the current implementation; broader claims require extending the evaluation to a larger and more diverse set of instruments. Second, the information set is restricted to price-derived signals; macroeconomic covariates, volatility proxies, order-flow information, and textual sentiment are not included. Third, evaluation focuses on statistical accuracy of price-level forecasts; transaction costs, bid–ask spreads, market impact, and risk-adjusted economic value are not assessed, so no claims are made about trading profitability.

Finally, although pairwise Diebold–Mariano tests against XGBoost are now reported on the full aligned forecast sample available for each asset–horizon–lookback configuration, broader uncertainty quantification, interval estimation, and more comprehensive multiple-model comparison procedures remain outside the present scope. In particular, block-bootstrap confidence intervals would be a natural next step to assess whether small metric differences are practically meaningful, especially at $H = 1$. Therefore, small metric differences should still be interpreted with caution, especially at the shortest horizon. However, these results are not robust across assets and horizons, and do not support a consistent conclusion of model superiority.

Future extensions should therefore (i) expand the asset universe and perform robust cross-asset aggregation; (ii) incorporate richer exogenous predictors while preserving the leakage-controlled design; (iii) extend uncertainty quantification and forecast comparison analysis beyond the current pairwise DM tests; and (iv) embed the forecasts into explicit

decision rules to evaluate economic value under realistic transaction costs and constraints. In practical deployment, the pipeline should also be coupled with scheduled retraining, rolling drift and performance monitoring, and explicit trigger rules for model replacement when degradation persists.

5. Conclusions

We presented a fully automated, reproducible, leakage-controlled framework for daily market forecasting with a horizon-specific model and lookback selection. The system integrates transparent baselines (zero- and mean-return), a boosted-tree learner (XGBoost), and deep temporal models (LSTM and CNN/TCN) under the following homogeneous protocol: (i) causal feature construction from OHLCV-derived information; (ii) strict chronological splitting with an explicit purging rule to avoid overlap between training windows and multi-day return labels; (iii) training-only robustification (winsorization and adaptive clipping); and (iv) auditable persistence of datasets, model objects, predictions, and metric tables.

Empirically, on the current three-instrument universe (MT, DELL, and the S&P 500 index), all model families achieve strong price-level fit at the next-day horizon ($H = 1$), but this behavior is largely explained by the strong persistence of daily price levels. In contrast, predictive performance in return space remains weak, with R^2 values close to zero or negative, and machine learning models do not consistently outperform naive benchmarks in a statistically robust manner.

At the weekly horizon ($H = 5$), separation across model families becomes more visible on the price scale, and the operational rule (minimizing external hold-out RMSE on the implied price scale, using SMAPE only as a tie-breaker) selects deep temporal models more frequently in this run. However, these selection patterns should be interpreted as operational and asset-specific rather than as evidence of consistent predictive superiority. Thus, the main value of the proposed framework lies in leakage-controlled benchmarking and horizon-wise comparison, not in demonstrating reliable predictive gains across assets.

The proposed pipeline is best interpreted as an engineering and evaluation template for calibrated level forecasting and systematic benchmarking, rather than as evidence of reliable short-horizon directional timing. The current study also has clear limitations: the asset universe is intentionally small; the information set is restricted to price-derived inputs; statistical forecast comparisons are limited to pairwise Diebold–Mariano tests on the full aligned forecast sample available for each asset–horizon–lookback configuration; and no uncertainty quantification or economic-value evaluation (e.g., transaction costs and risk-adjusted performance) is provided. Therefore, the numerical differences reported here should be viewed as illustrative of the framework’s capabilities under a controlled protocol, not as definitive claims of superiority for any architecture. Therefore, the primary contribution of this work is methodological, providing a rigorous and leakage-controlled benchmarking framework, rather than demonstrating consistent predictive gains in financial time series forecasting.

Future work should expand the universe of instruments, incorporate exogenous predictors (macroeconomic variables, volatility proxies, order-flow measures, and/or sentiment), and extend uncertainty quantification and statistical comparison beyond the current pairwise Diebold–Mariano tests to assess whether small metric differences are practically meaningful. An additional priority is to embed forecasts into explicit decision rules and evaluate out-of-sample economic value under realistic trading frictions. Within these extensions, the central contribution of this work remains, a transparent, auditable, leakage-controlled pipeline that enables fair, horizon-specific model selection and reproducible monitoring of forecasting performance over time.

Author Contributions: Conceptualization, F.A.N.P.; methodology, F.A.N.P., F.J.A.M. and J.M.B.; software, F.A.N.P.; validation, J.C.N.P.; formal analysis, F.A.N.P. and J.C.N.P.; investigation, F.A.N.P.; resources, F.A.N.P.; data curation, J.C.N.P.; writing—original draft preparation, F.J.A.M. and J.M.B.; writing—review and editing, F.A.N.P. and F.J.A.M.; visualization, A.R.C.; supervision, F.A.N.P.; project administration, F.A.N.P.; funding acquisition, F.A.N.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding. The article processing charge (APC) was funded by Universidad Politécnica de Lázaro Cárdenas (UPLC), Michoacán, Mexico.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: I sincerely appreciate the Universidad Politécnica de Lázaro Cárdenas for offering me the opportunity and the essential resources to carry out this project.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

OHLCV	Open, High, Low, Close, Volume
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
SMAPE	Symmetric Mean Absolute Percentage Error
R^2	Coefficient of Determination
LSTM	Long Short-Term Memory (network)
CNN	Convolutional Neural Network
TCN	Temporal Convolutional Network
XGBoost	Extreme Gradient Boosting
SMA	Simple Moving Average
EMA	Exponential Moving Average
APC	Article Processing Charge

Appendix A. Hyperparameter Configuration and Tuning Protocol

Table A1. Hyperparameter configuration and tuning protocol used in the pipeline. Baseline models are evaluated without hyperparameter optimization, whereas non-baseline models are calibrated within the pre-evaluation segment using Bayesian optimization under a leakage-controlled walk-forward scheme. The discrete lookback $L \in \{60, 180, 500\}$ is evaluated internally for each asset, horizon, and model family, and the final operational choice is based on walk-forward performance.

Model Family	Component	Configuration/Protocol
Common	Targets and splits	Return target $y_t^{(H)} = \log(P_{t+H}/P_t)$; chronological split with pre fraction 0.85 and external hold-out 0.15; explicit horizon purging at the pre/hold-out boundary.
Common	Robustification	Training-only winsorization quantile $q = 0.995$; adaptive clipping bound from training $ y $ quantile ($q = 0.995$) times multiplier (1.25) with floor 0.03 and cap 0.60.

Table A1. Cont.

Model Family	Component	Configuration/Protocol
Common	Scaling	Min–max scaling fitted on training rows only (up to last training anchor date); applied unchanged to validation, hold-out, and future windows.
Common	Hyperparameter tuning	Bayesian optimization with Optuna’s TPE sampler applied only to non-baseline models (XGBoost, LSTM, CNN/TCN) within the pre-evaluation segment; optimization objective: minimize validation RMSE _y across rolling folds; the external hold-out is never used.
Common	Lookback evaluation and selection	Discrete grid $L \in \{60, 180, 500\}$ evaluated per (asset, H , model family) under the internal walk-forward procedure. The final operational choice of L is based primarily on walk-forward mean LogRMSE in price space, with mean price-scale R^2 used as a secondary tie-breaker.
XGBoost	Search space/calibrated components	Tree booster regression with Bayesian calibration over model hyperparameters within the pre-evaluation segment; objective and evaluation remain defined on the return target.
XGBoost	Representative implementation settings	objective=reg:pseudohubererror, eval_metric=rmse, tree_method=hist, seed=42; early stopping and final number of boosting rounds determined within the time-respecting training procedure.
LSTM	Search space/calibrated components	Sequence architecture calibrated within the pre-evaluation segment, including recurrent width/depth and regularization-related choices, under Bayesian optimization and time-respecting validation.
LSTM	Representative implementation settings	Huber loss ($\delta = 1$), Adam optimizer, batch size 64, early stopping on internal validation, and final refit on the pre-evaluation segment under the leakage-controlled protocol.
CNN/TCN	Search space/calibrated components	Causal convolutional/temporal-convolutional architecture calibrated within the pre-evaluation segment, including filter, kernel, dilation, and regularization-related choices, under Bayesian optimization and time-respecting validation.
CNN/TCN	Representative implementation settings	Huber loss ($\delta = 1$), Adam optimizer, batch size 64, early stopping on internal validation, and final refit on the pre-evaluation segment under the leakage-controlled protocol.

Appendix B. Additional Figures (Complete External Hold-Out Plots)

This appendix reports the complete external hold-out plots for all model families and horizons ($H \in \{1, 5\}$) for each asset, using the selected lookback L stored in the output filenames.

Appendix B.1. ArcelorMittal (MT)

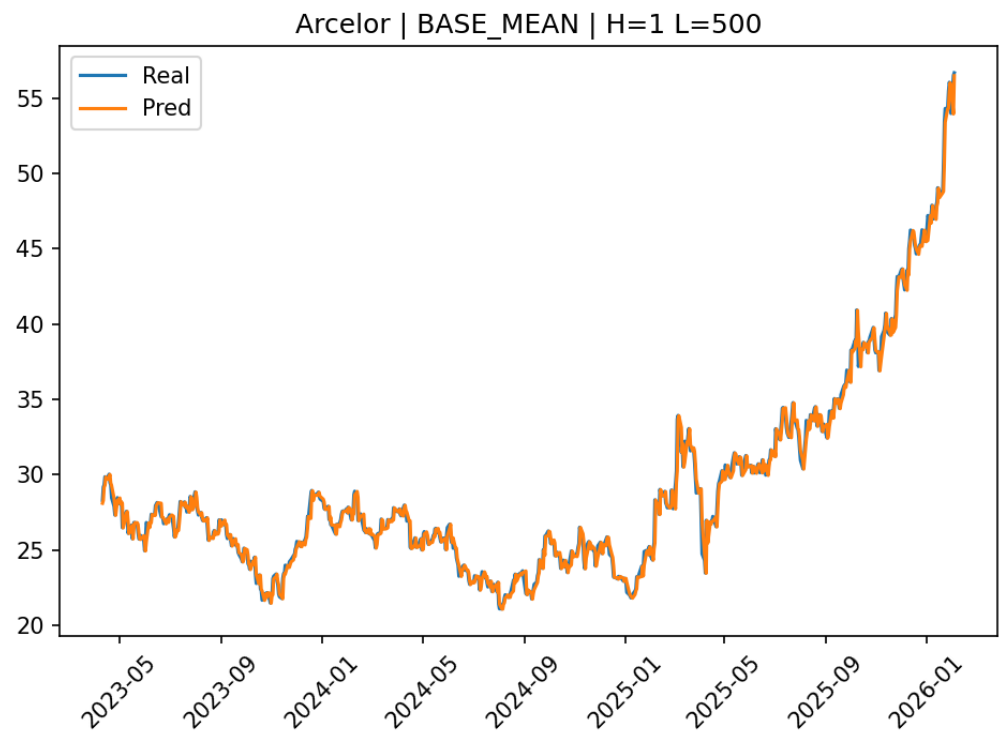


Figure A1. ArcelorMittal (MT)—Baseline (mean), horizon $H = 1$, lookback $L = 500$.

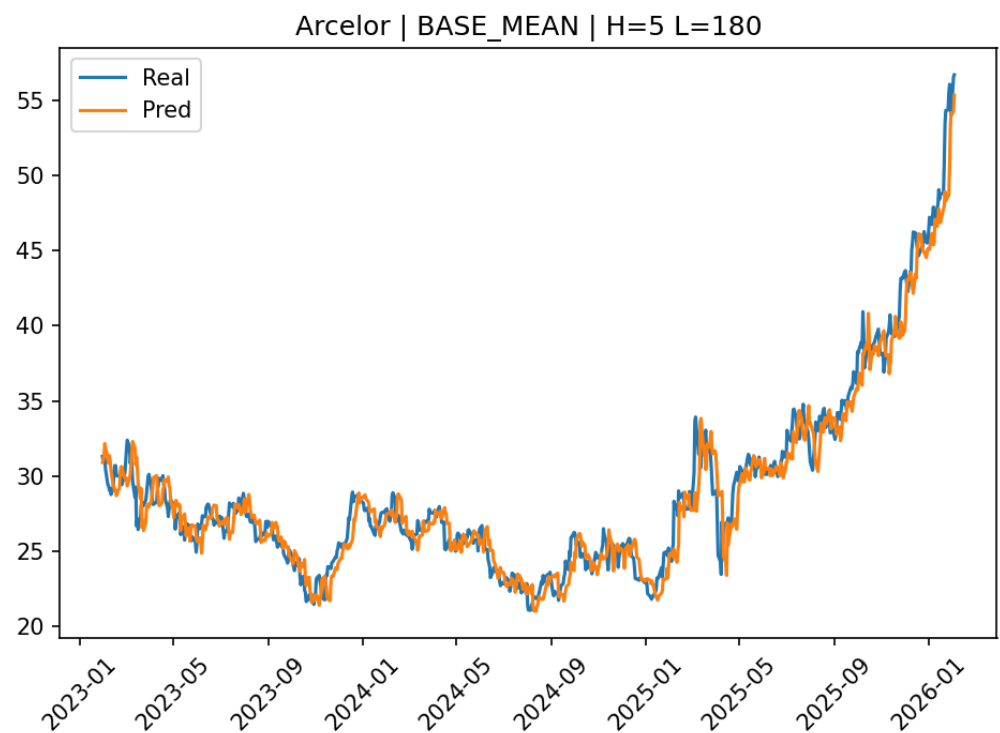


Figure A2. ArcelorMittal (MT)—Baseline (mean), horizon $H = 5$, lookback $L = 180$.

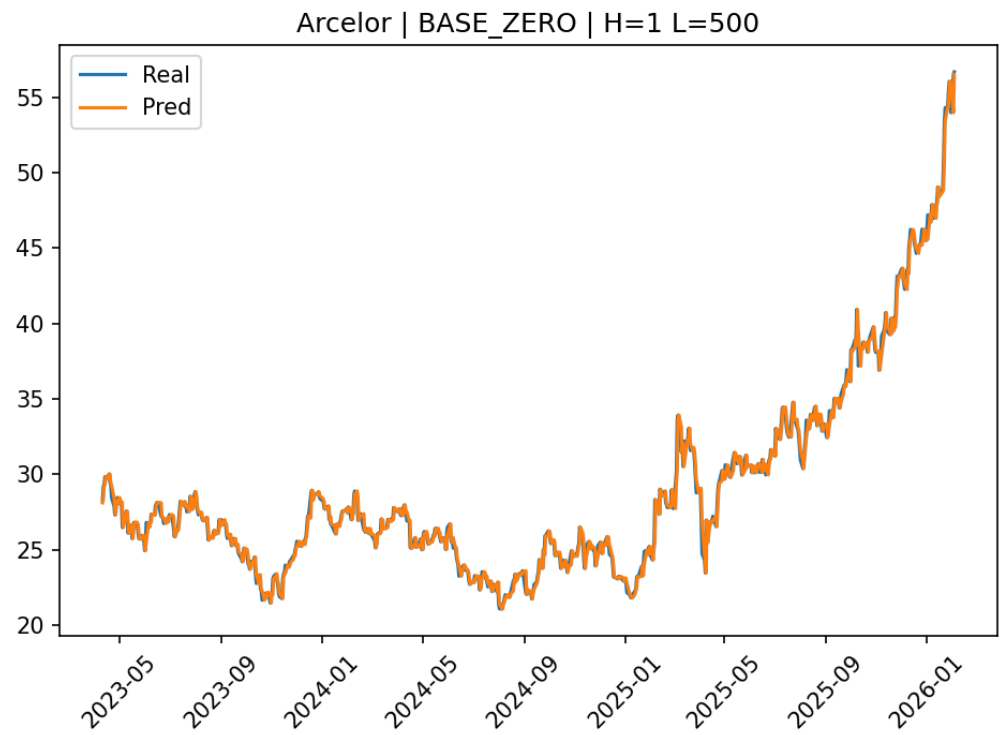


Figure A3. ArcelorMittal (MT)—Baseline (0), horizon $H = 1$, lookback $L = 500$.

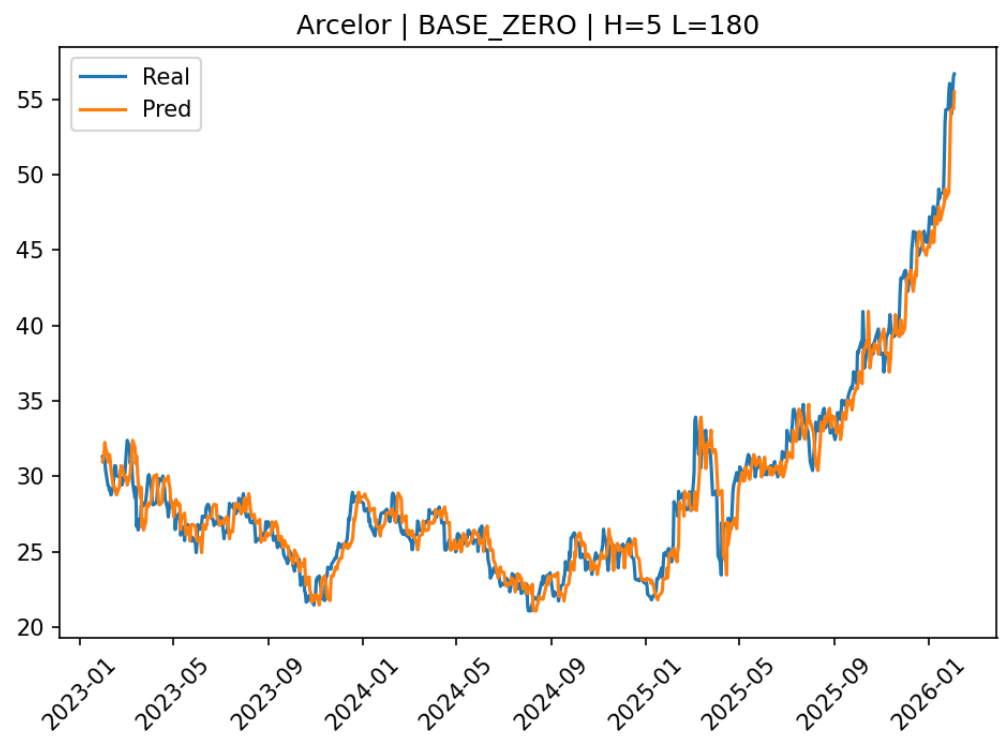


Figure A4. ArcelorMittal (MT)—Baseline (0), horizon $H = 5$, lookback $L = 180$.

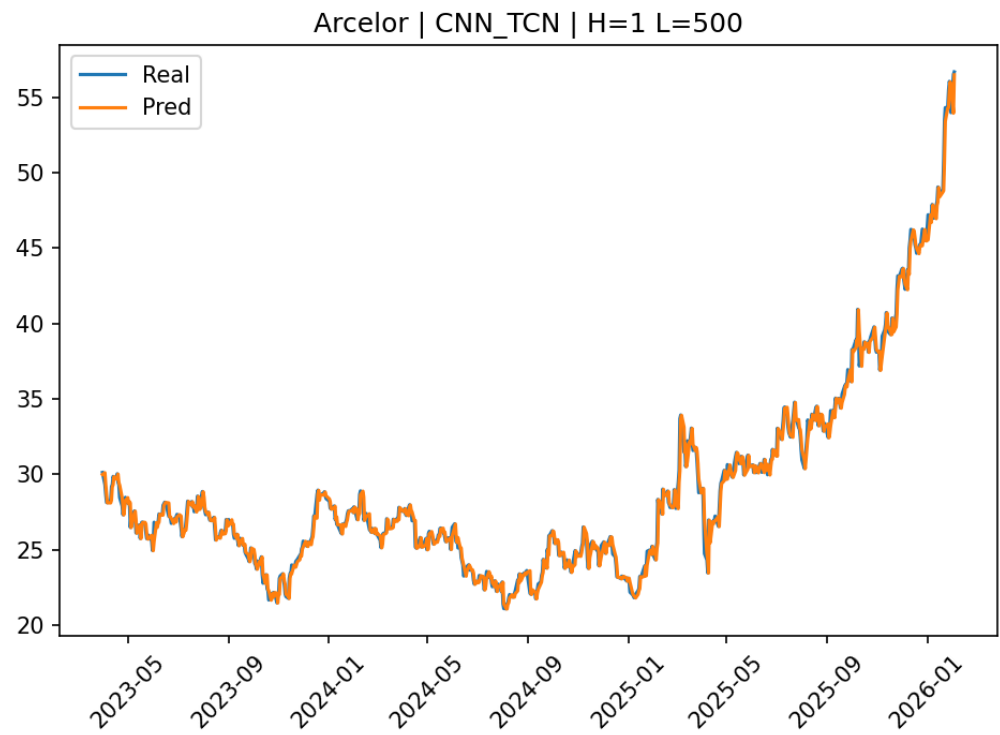


Figure A5. ArcelorMittal (MT)—CNN/TCN, horizon $H = 1$, lookback $L = 500$.

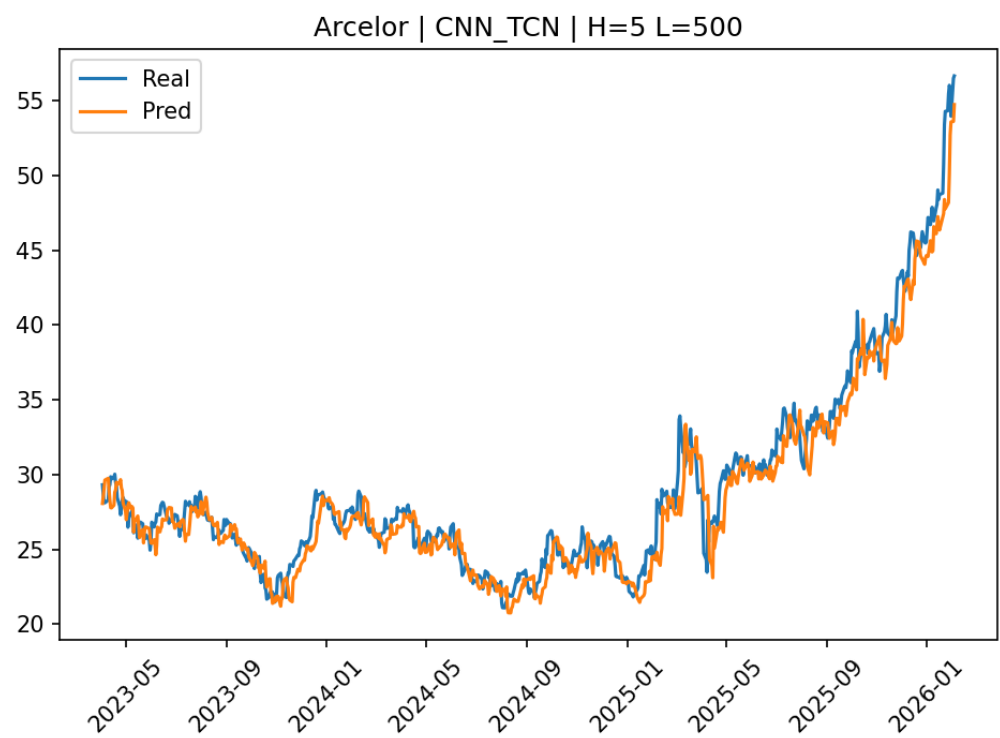


Figure A6. ArcelorMittal (MT)—CNN/TCN, horizon $H = 5$, lookback $L = 500$.

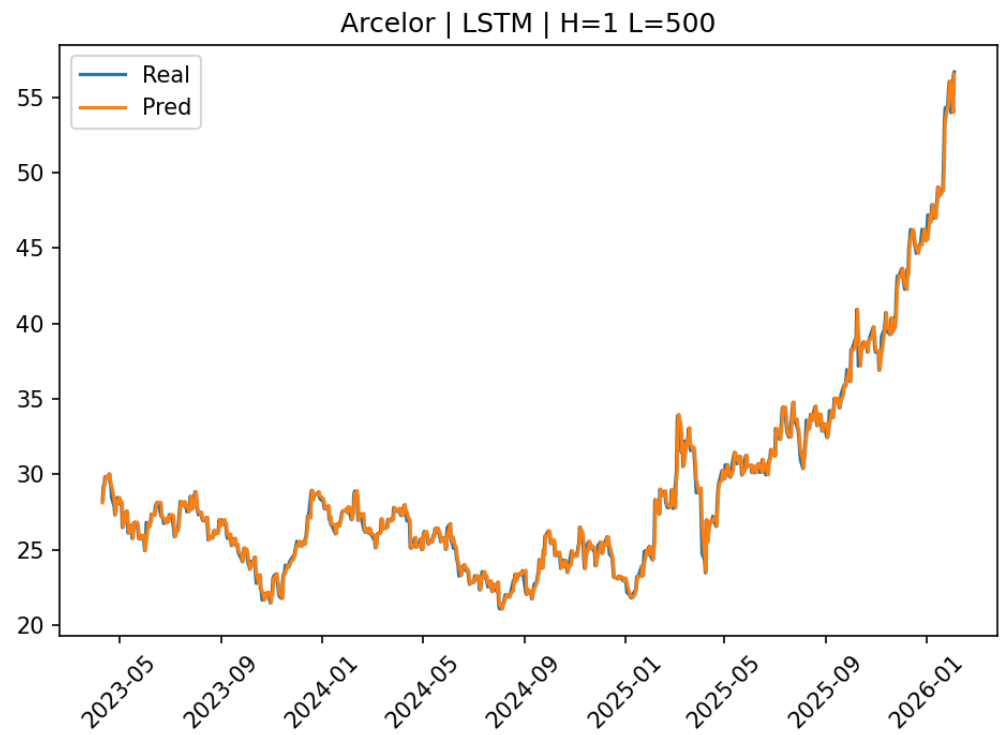


Figure A7. ArcelorMittal (MT)—LSTM, horizon $H = 1$, lookback $L = 500$.

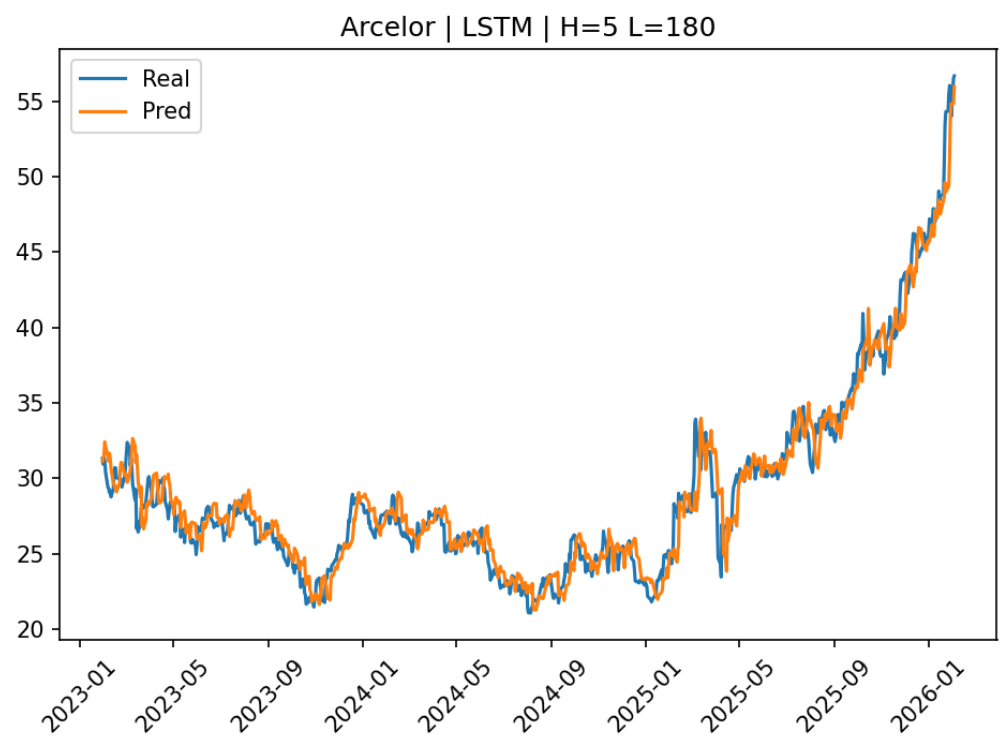


Figure A8. ArcelorMittal (MT)—LSTM, horizon $H = 5$, lookback $L = 180$.

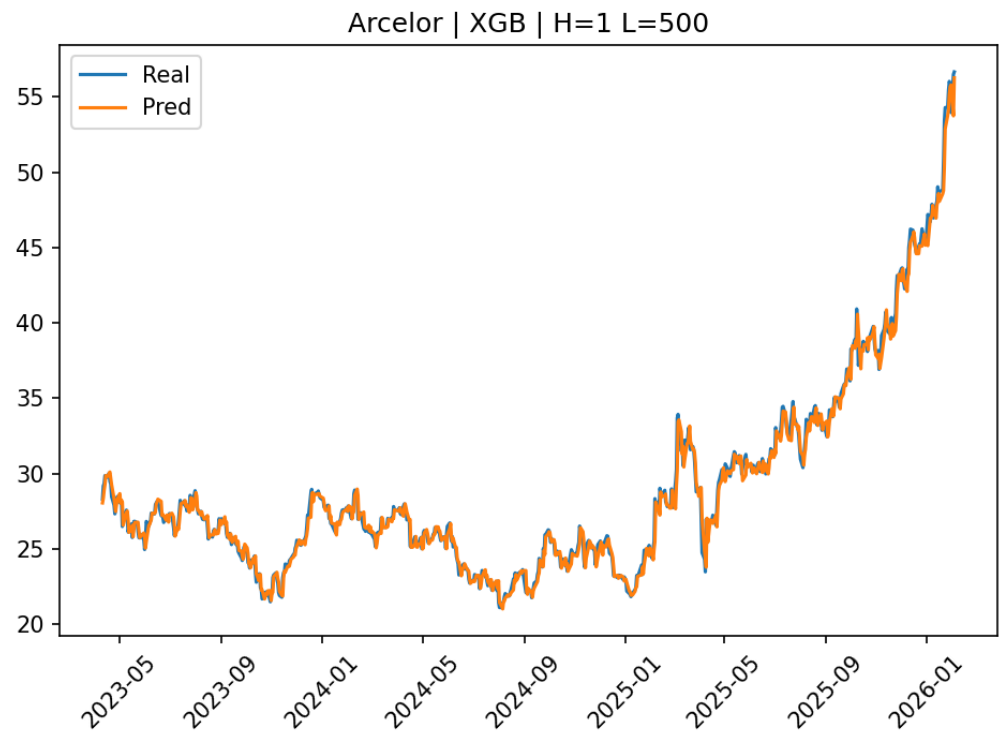


Figure A9. ArcelorMittal (MT)—XGBoost, horizon $H = 1$, lookback $L = 500$.

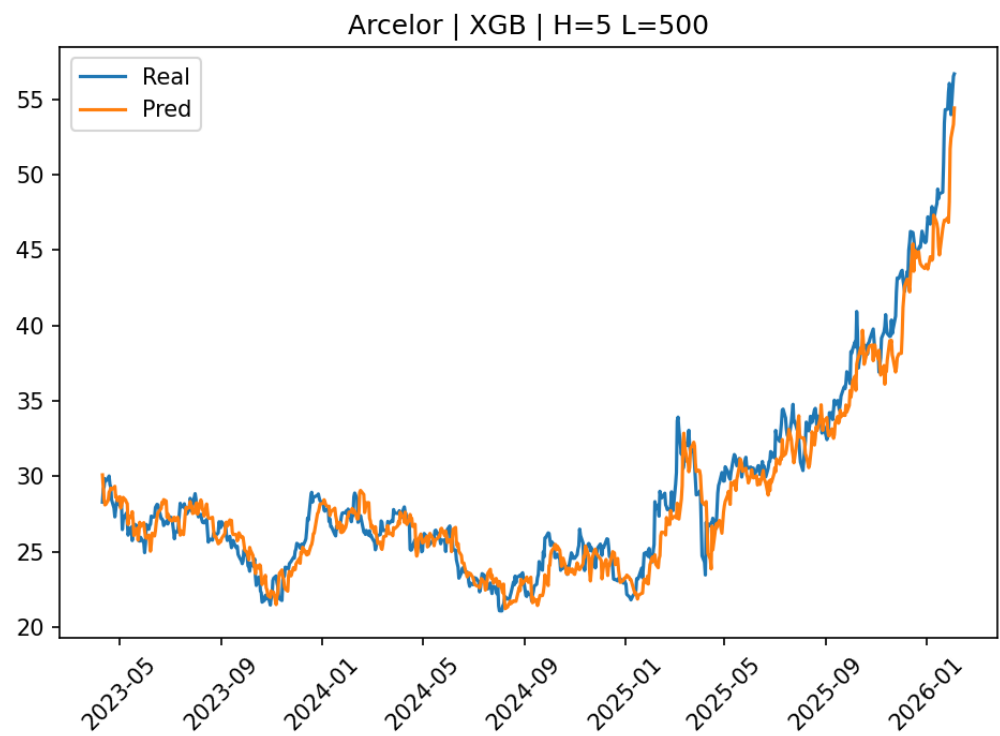


Figure A10. ArcelorMittal (MT)—XGBoost, horizon $H = 5$, lookback $L = 500$.

Appendix B.2. Dell Technologies (DELL)

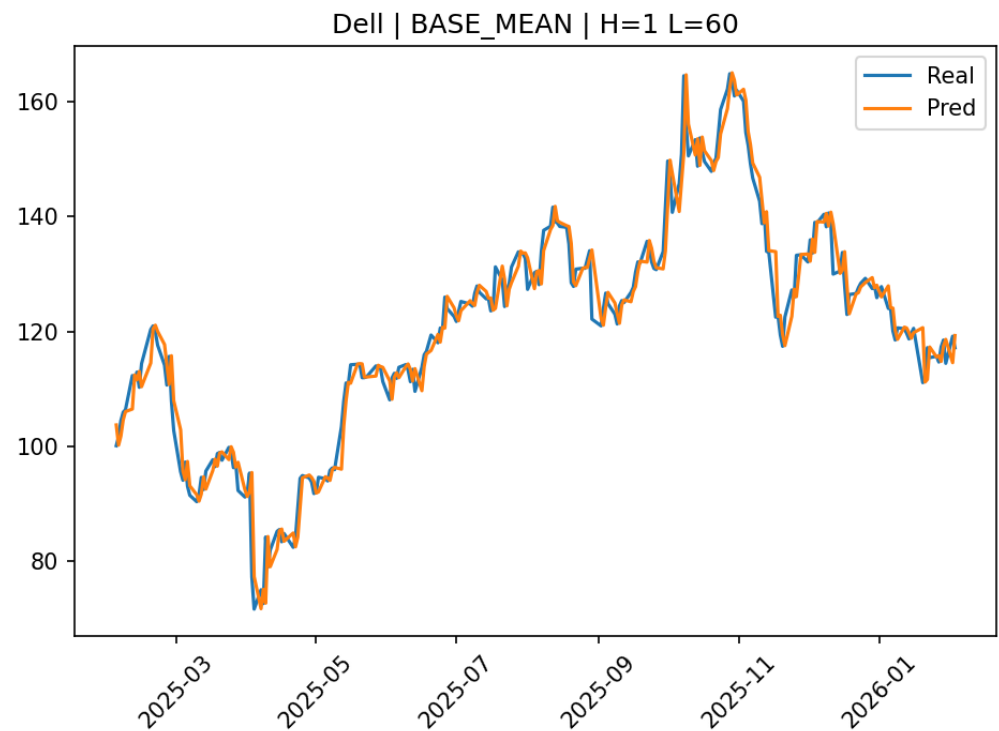


Figure A11. Dell Technologies (DELL)—Baseline (mean), horizon $H = 1$, lookback $L = 60$.

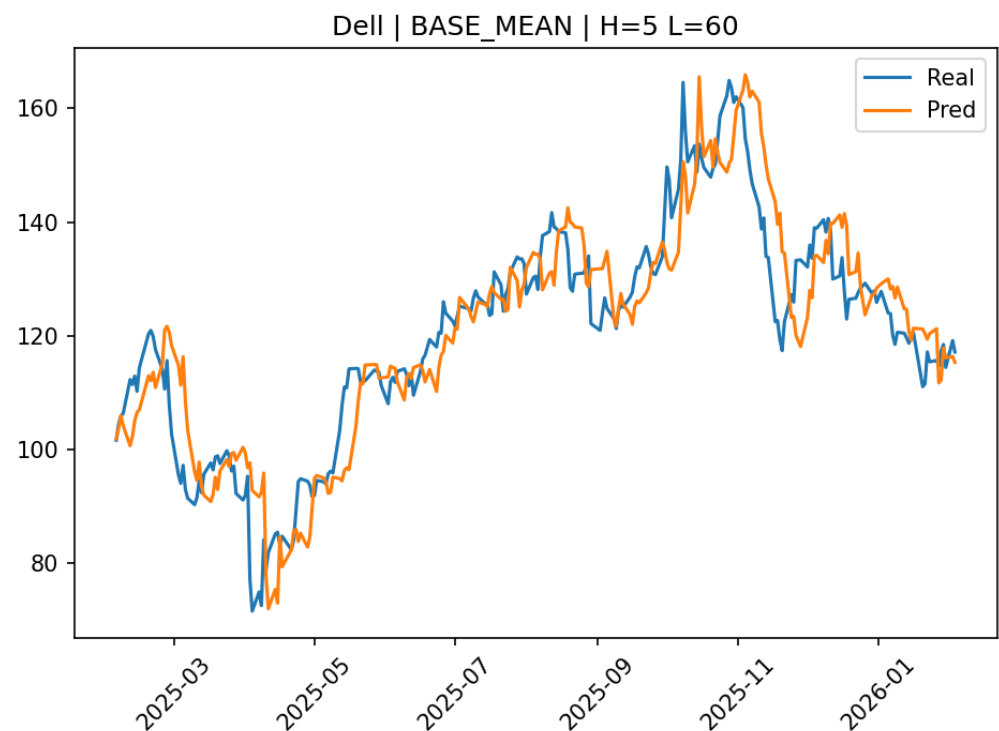


Figure A12. Dell Technologies (DELL)—Baseline (mean), horizon $H = 5$, lookback $L = 60$.

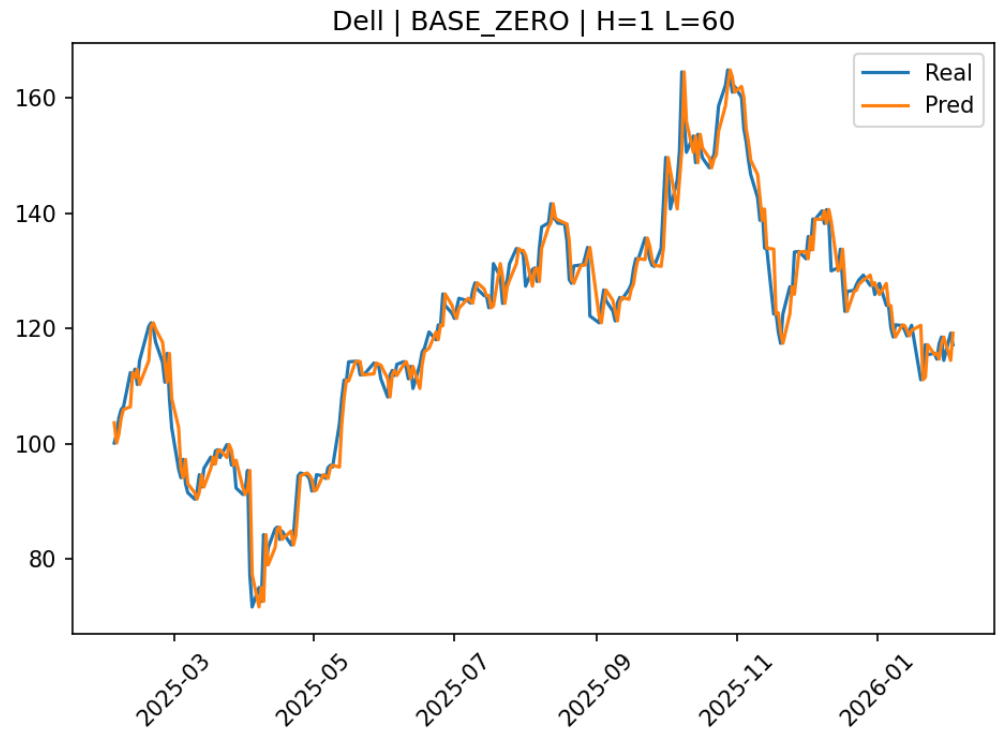


Figure A13. Dell Technologies (DELL)—Baseline (0), horizon $H = 1$, lookback $L = 60$.

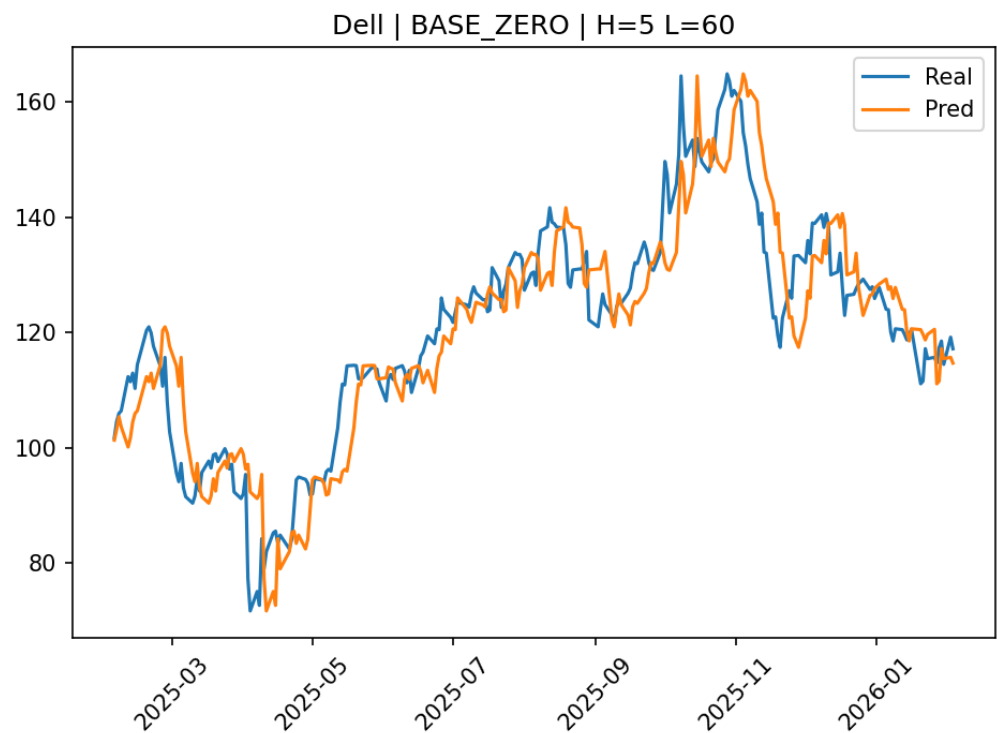


Figure A14. Dell Technologies (DELL)—Baseline (0), horizon $H = 5$, lookback $L = 60$.

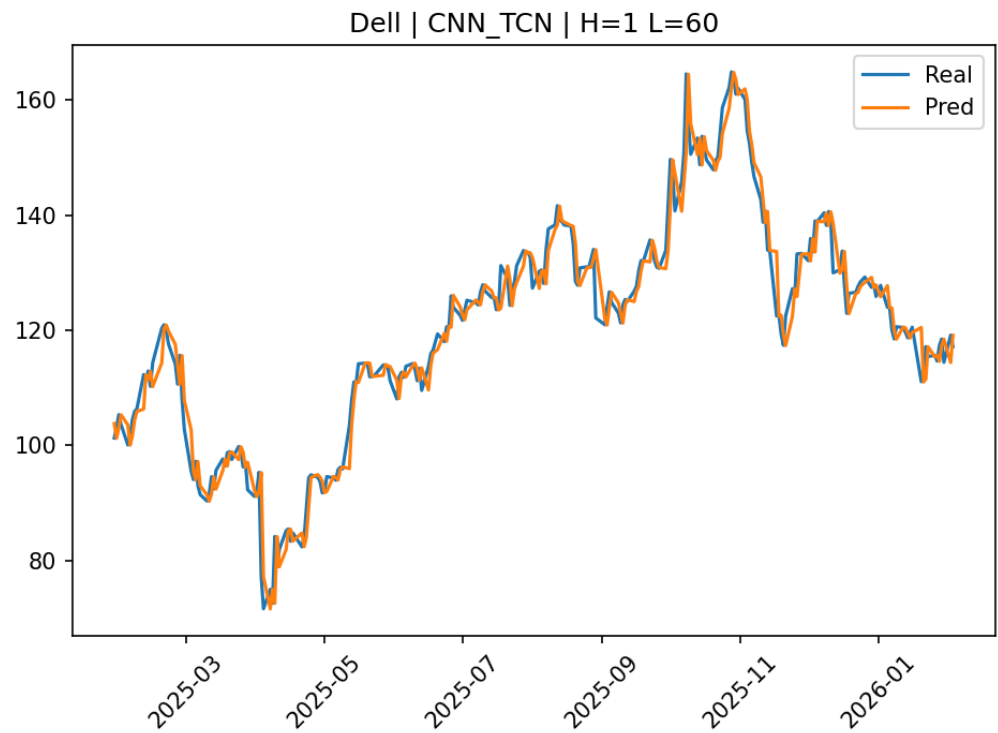


Figure A15. Dell Technologies (DELL)—CNN/TCN, horizon $H = 1$, lookback $L = 60$.

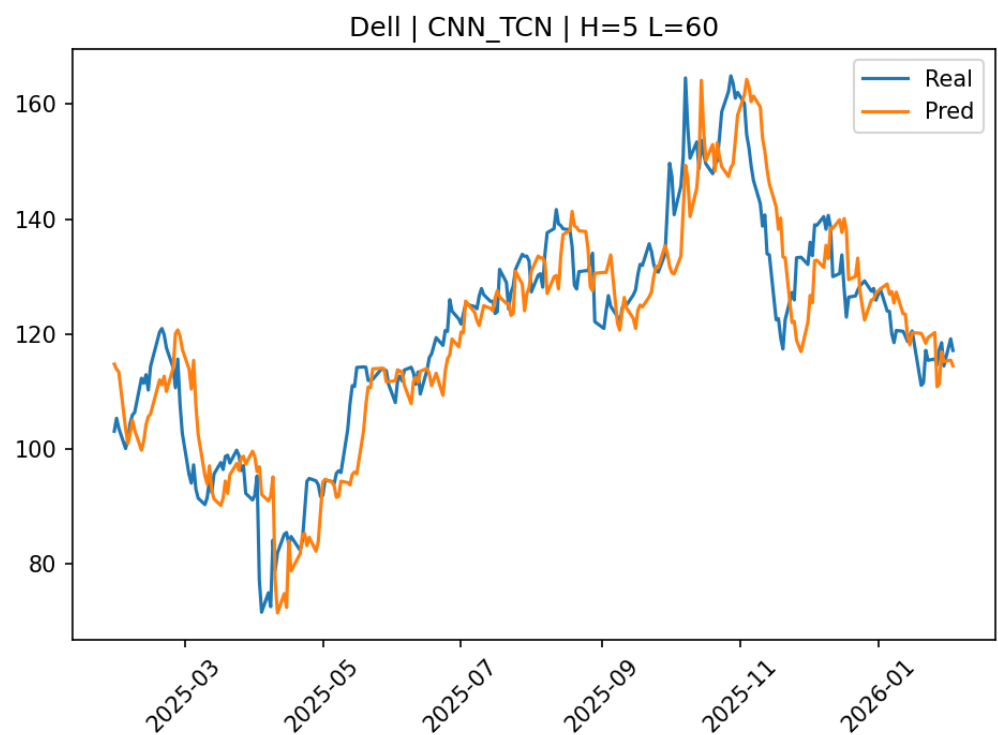


Figure A16. Dell Technologies (DELL)—CNN/TCN, horizon $H = 5$, lookback $L = 60$.

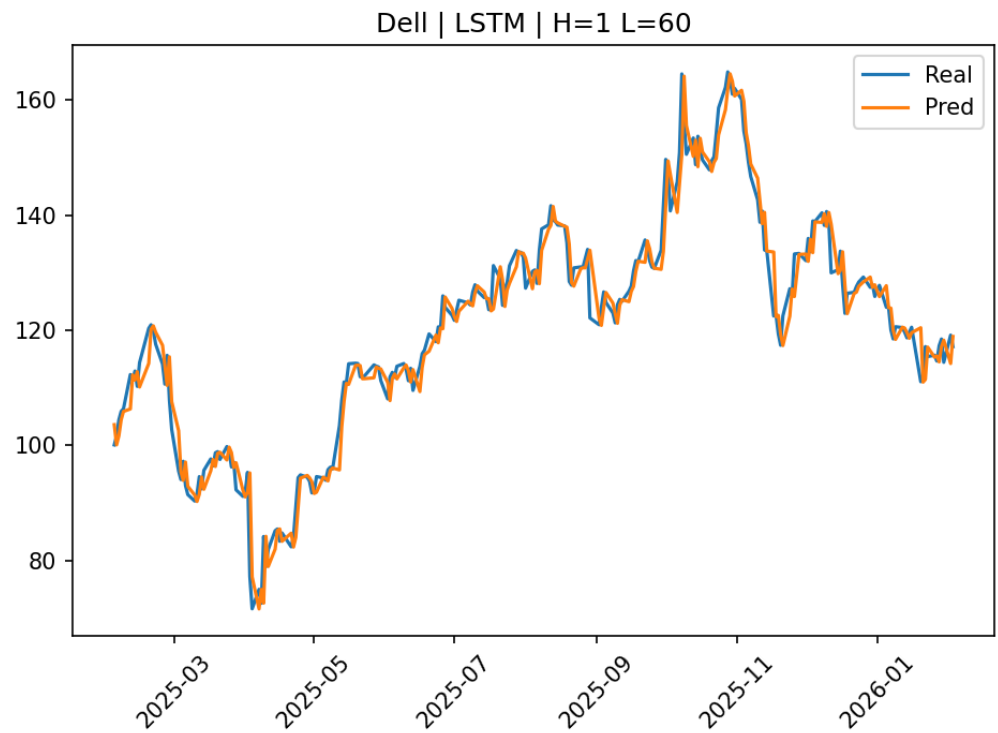


Figure A17. Dell Technologies (DELL)—LSTM, horizon $H = 1$, lookback $L = 60$.

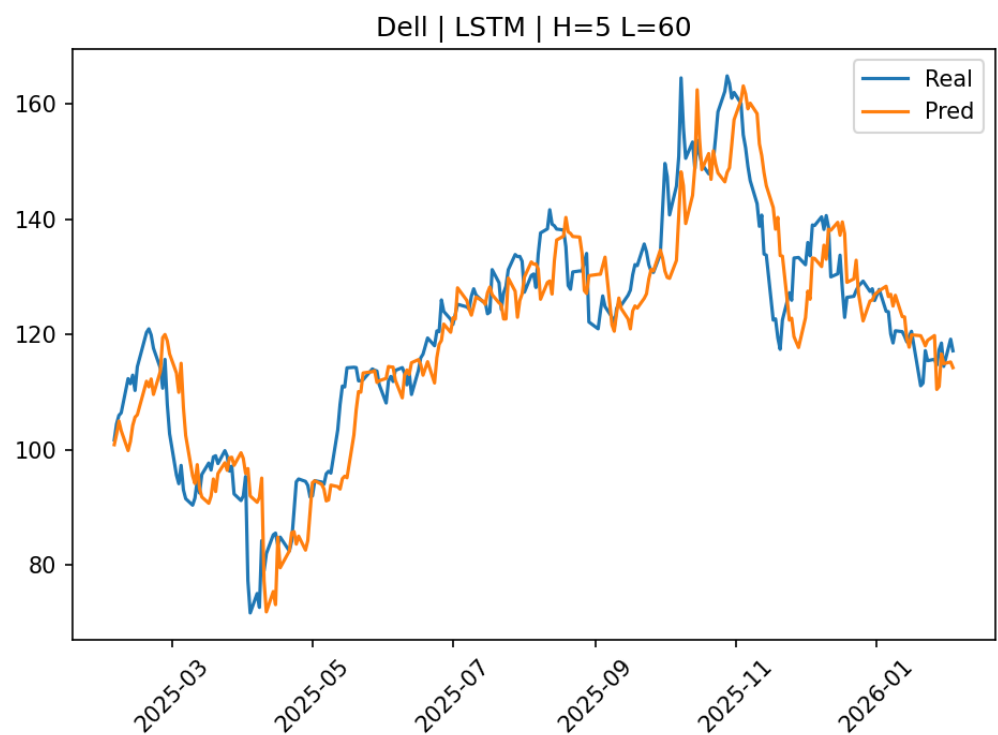


Figure A18. Dell Technologies (DELL)—LSTM, horizon $H = 5$, lookback $L = 60$.

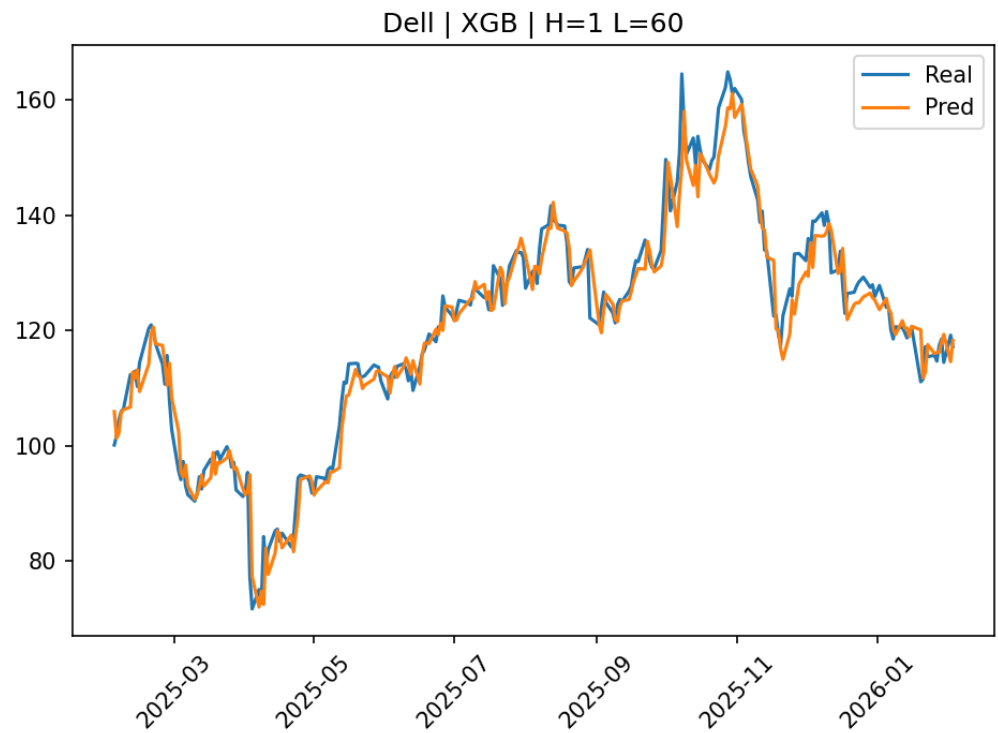


Figure A19. Dell Technologies (DELL)—XGBoost, horizon $H = 1$, lookback $L = 60$.

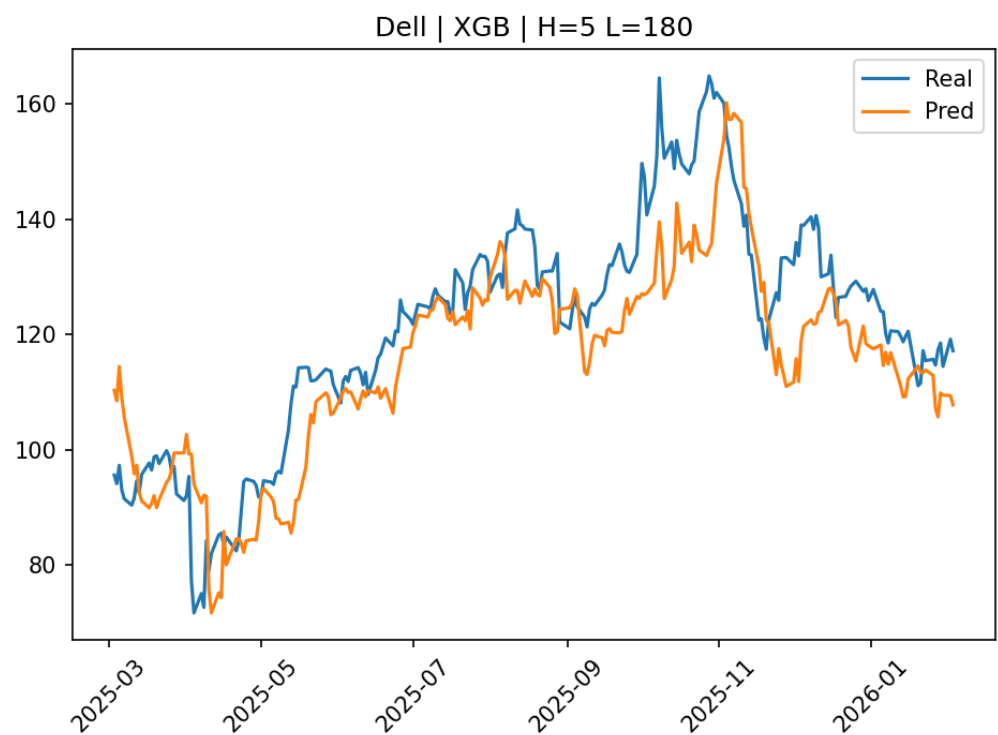


Figure A20. Dell Technologies (DELL)—XGBoost, horizon $H = 5$, lookback $L = 180$.

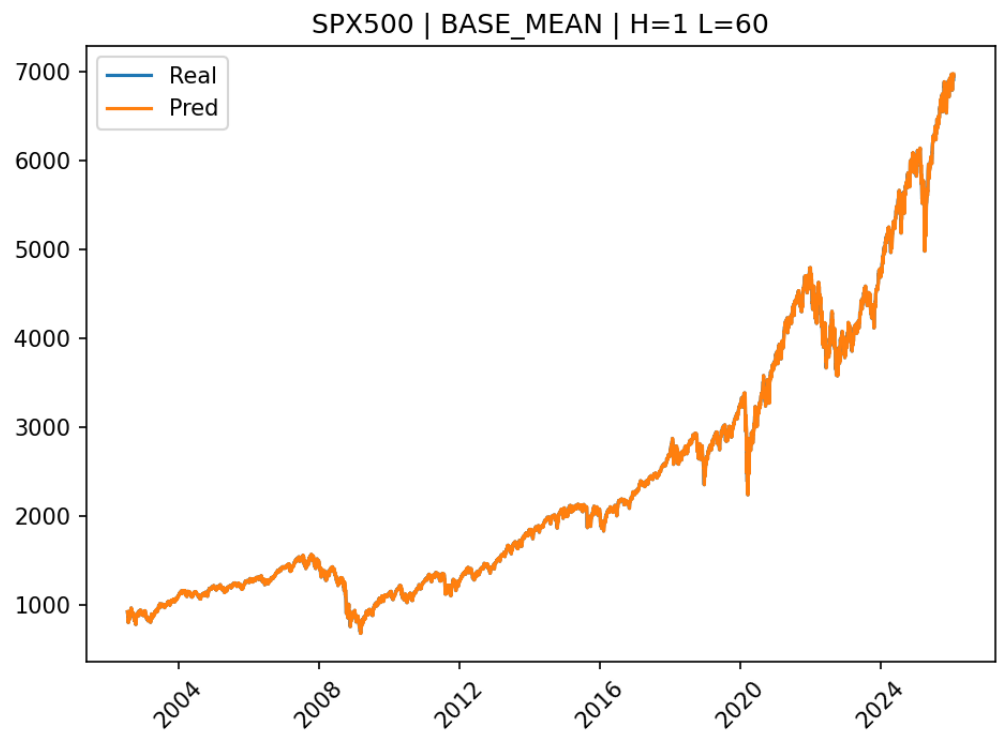
Appendix B.3. S&P 500 Index ($\hat{SPX}$)

Figure A21. S&P 500 Index ($\hat{SPX}$)—Baseline (mean), horizon $H = 1$, lookback $L = 60$.

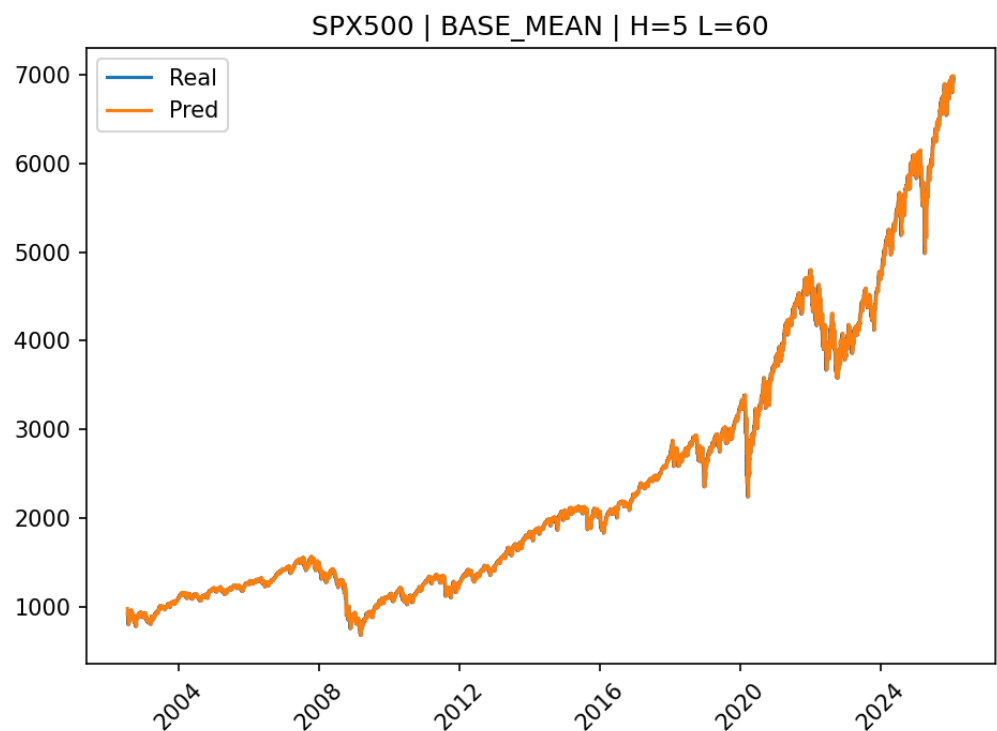


Figure A22. S&P 500 Index ($\hat{SPX}$)—Baseline (mean), horizon $H = 5$, lookback $L = 60$.

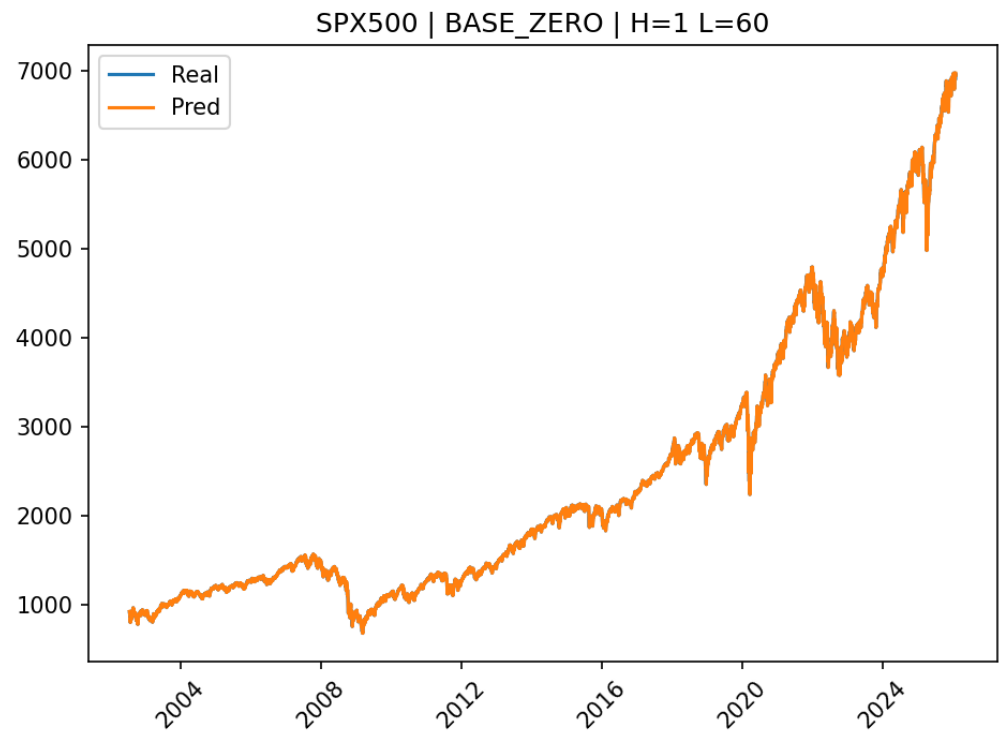


Figure A23. S&P 500 Index ($\hat{\text{SPX}}$)—Baseline (0), horizon $H = 1$, lookback $L = 60$.

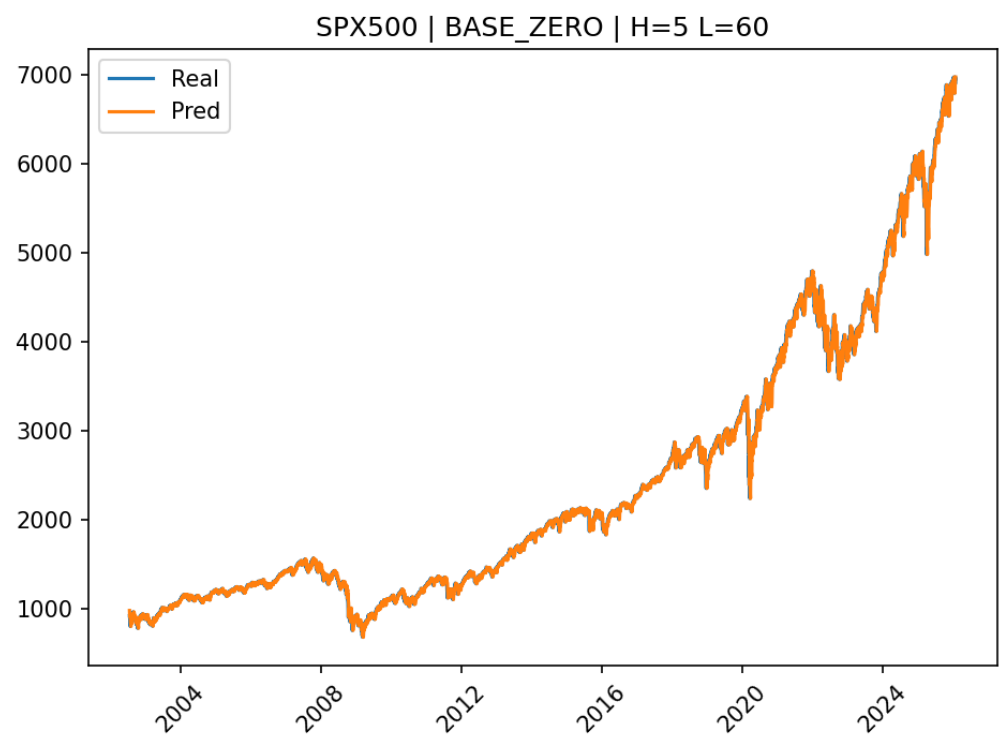


Figure A24. S&P 500 Index ($\hat{\text{SPX}}$)—Baseline (0), horizon $H = 5$, lookback $L = 60$.

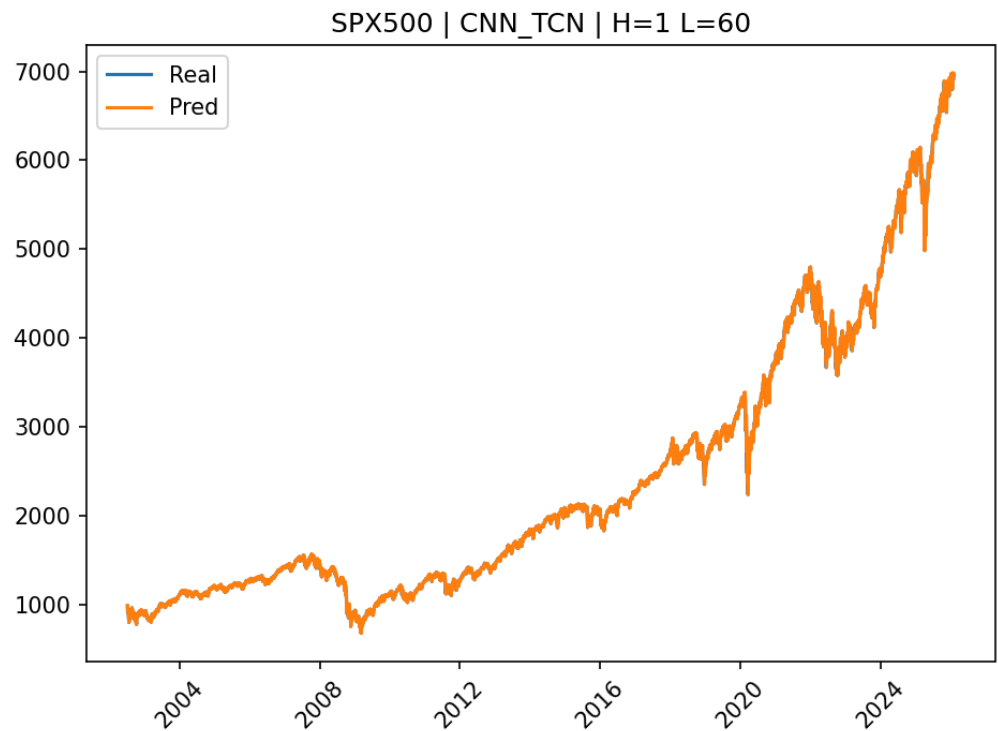


Figure A25. S&P 500 Index ($\hat{\text{SPX}}$)—CNN/TCN, horizon $H = 1$, lookback $L = 60$.

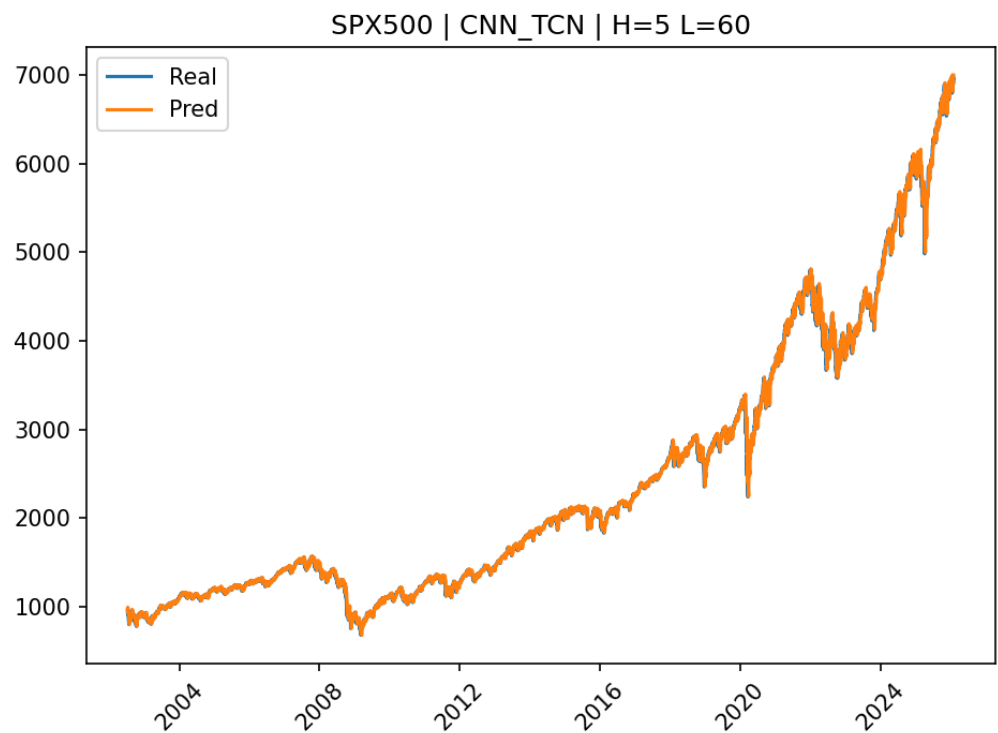


Figure A26. S&P 500 Index ($\hat{\text{SPX}}$)—CNN/TCN, horizon $H = 5$, lookback $L = 60$.

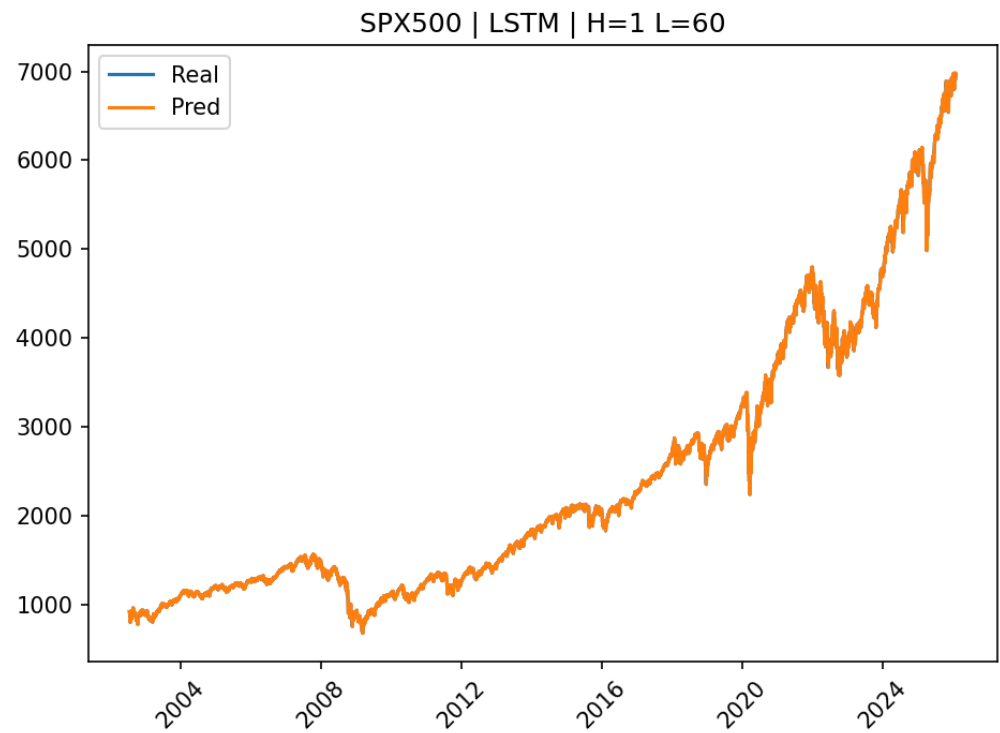


Figure A27. S&P 500 Index ($\hat{\text{SPX}}$)—LSTM, horizon $H = 1$, lookback $L = 60$.

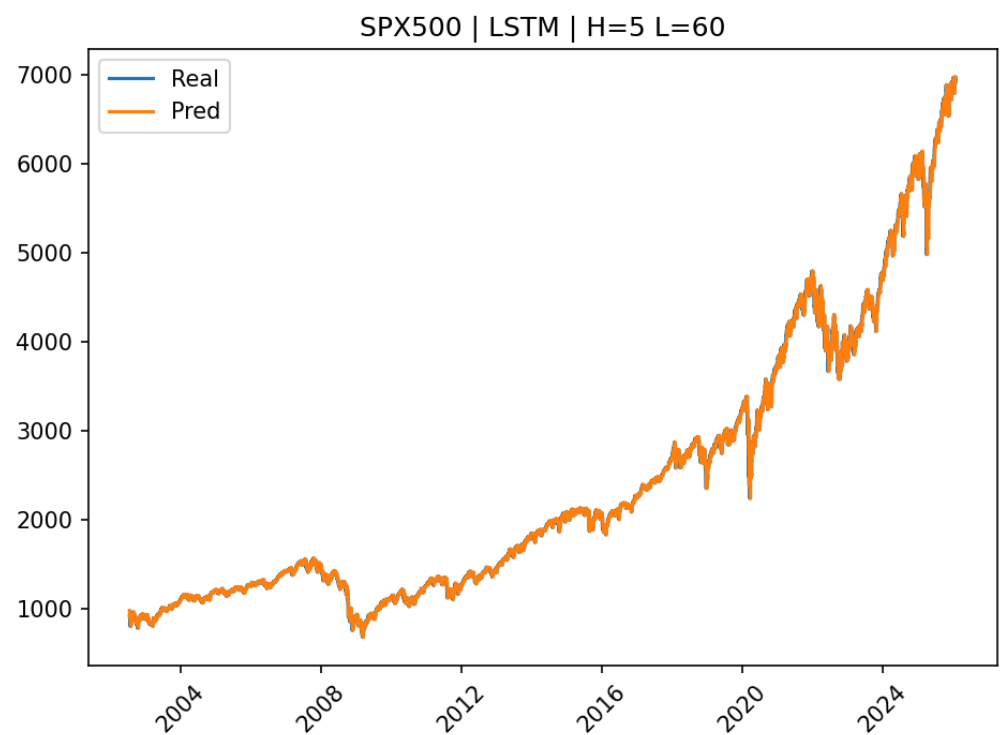


Figure A28. S&P 500 Index ($\hat{\text{SPX}}$)—LSTM, horizon $H = 5$, lookback $L = 60$.

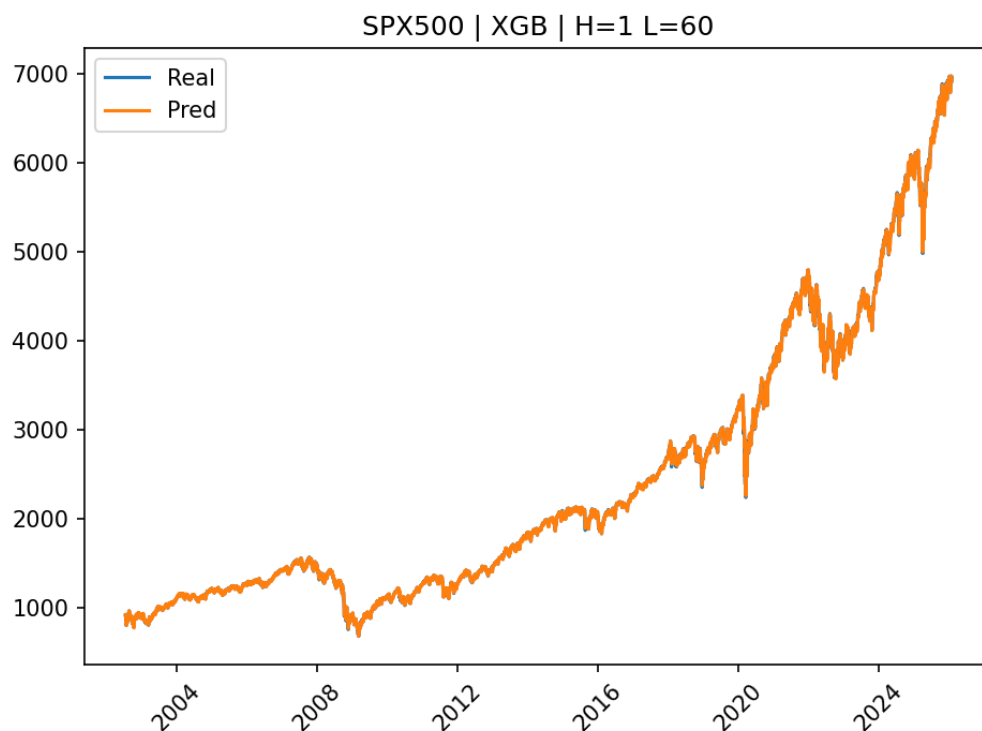


Figure A29. S&P 500 Index ($\hat{\text{SPX}}$)—XGBoost, horizon $H = 1$, lookback $L = 60$.

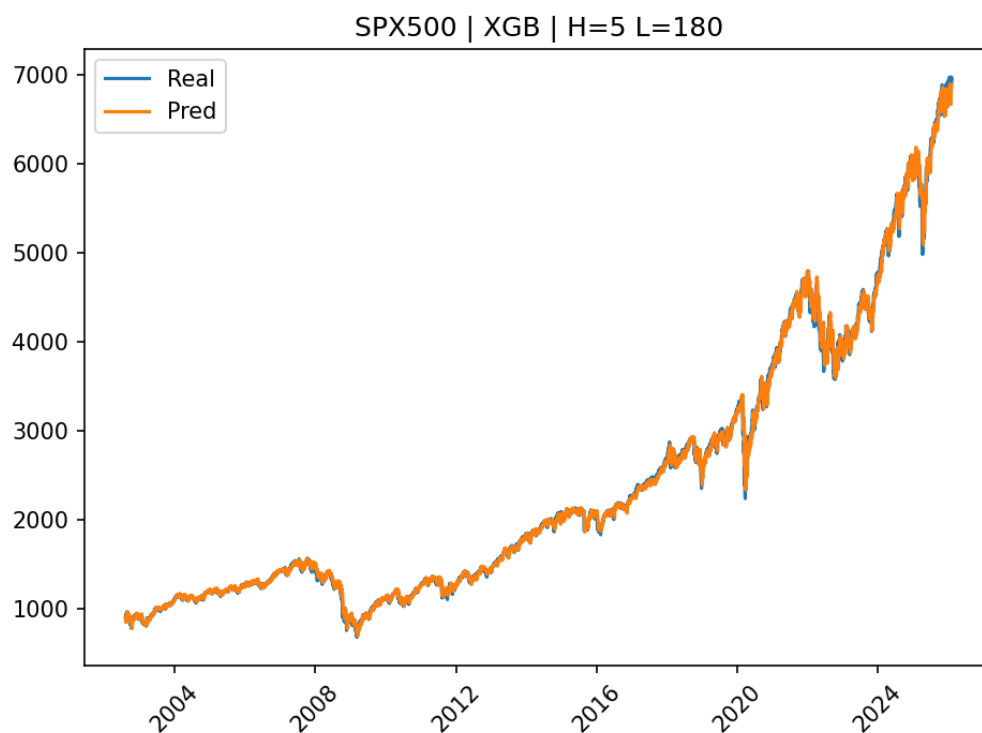


Figure A30. S&P 500 Index ($\hat{\text{SPX}}$) —XGBoost, horizon $H = 5$, lookback $L = 180$.

Appendix C. Additional Asset-Level Return- Versus Price-Space Contrasts

This appendix reports two illustrative asset-level contrasts between evaluation in return space and evaluation in price space. These examples are not part of the main operational selection rule of the paper; rather, they are included to clarify how model

ranking can depend on the evaluation domain and on asset class. Apple is used as an example where a deep temporal model substantially improves price-level tracking despite weaker return-space errors, whereas Bitcoin illustrates a case in which the gains of a machine-learning model over trivial baselines are present but much more modest.

Appendix C.1. Apple: Return-Space Versus Price-Space Contrast

Table A2. Apple: return-space evaluation and Diebold–Mariano comparisons against the historical-mean baseline.

Horizon	Best Baseline	L	$RMSE_y$	R_y^2	Best ML Model	L	$RMSE_y$	R_y^2	DM p -Value
$H = 1$	BASE_MEAN	500	0.011852	−0.000388	CNN/TCN	60	0.019318	−0.000865	0.926
$H = 5$	BASE_MEAN	500	0.023919	−0.002321	CNN/TCN	60	0.046013	−0.001810	0.899

Table A3. Apple: contrast between return-space and price-space evaluation for the best baseline and the best non-baseline model.

Horizon	Model	L	$RMSE_y$	$RMSE_P$
$H = 1$	BASE_MEAN	500	0.011852	29.993
$H = 1$	CNN/TCN	60	0.019318	4.169
$H = 5$	BASE_MEAN	500	0.023919	62.614
$H = 5$	CNN/TCN	60	0.046013	10.117

The Apple results reveal a sharp contrast between predictive accuracy in return space (y) and predictive accuracy in price space (P). For both $H = 1$ and $H = 5$, the historical-mean baseline (BASE_MEAN) achieves the lowest RMSE in return space, with $RMSE_y = 0.011852$ at $H = 1$ and $RMSE_y = 0.023919$ at $H = 5$. By comparison, the best non-baseline model, CNN/TCN with $L = 60$, produces larger return-space errors (0.019318 and 0.046013, respectively).

However, this ranking reverses when evaluation is performed in price space. At $H = 1$, the baseline yields $RMSE_P = 29.993$, whereas CNN/TCN reduces this error to 4.169. At $H = 5$, the contrast becomes even stronger: the baseline reaches $RMSE_P = 62.614$, while CNN/TCN remains at 10.117. Thus, although the deep model is weaker in pointwise return prediction, it is far more effective at tracking the realized price trajectory.

The Diebold–Mariano comparisons against the historical-mean baseline yield p -values of 0.926 for $H = 1$ and 0.899 for $H = 5$, indicating no statistically significant superiority of CNN/TCN in return space. This reinforces an important methodological point: improvements in reconstructed price paths do not necessarily imply improved return predictability.

The Apple case therefore illustrates that price-space accuracy can substantially overstate the practical forecasting advantage of a model if return-space performance is not reported simultaneously. In highly persistent financial series, low errors in price levels may partly reflect trajectory anchoring rather than genuinely stronger forecasting skill in the economically relevant return target. For this reason, return-space metrics and formal statistical tests should remain central in comparative forecast evaluation, while price-space metrics can be interpreted as complementary indicators of operational path-tracking quality.

Appendix C.2. Bitcoin: Return-Space Versus Price-Space Contrast

Table A4. Bitcoin: return-space evaluation and Diebold–Mariano comparisons against the historical-mean baseline.

Horizon	Best Baseline	L	$RMSE_y$	R_y^2	Best ML Model	L	$RMSE_y$	R_y^2	DM p -Value
$H = 1$	BASE_ZERO	500	0.024072	−0.000322	XGBoost	180	0.023801	−0.000118	0.041
$H = 5$	BASE_ZERO	500	0.051024	−0.000014	XGBoost	60	0.050611	−0.000009	0.048

Table A5. Bitcoin: contrast between return-space and price-space evaluation for the best baseline and the best non-baseline model.

Horizon	Model	L	$RMSE_y$	$RMSE_p$
$H = 1$	BASE_ZERO	500	0.024072	2362.904
$H = 1$	XGBoost	180	0.023801	2318.117
$H = 5$	BASE_ZERO	500	0.051024	5131.843
$H = 5$	XGBoost	60	0.050611	5022.390

Bitcoin exhibits a substantially different pattern from Apple. Here, trivial benchmarks remain highly competitive, and the best machine-learning improvements are present but modest. For both horizons, the best baseline is the zero-return predictor (BASE_ZERO) with $L = 500$, while the best non-baseline model is XGBoost.

In return space, XGBoost yields only marginal improvements over the baseline. At $H = 1$, $RMSE_y$ decreases from 0.024072 to 0.023801, and at $H = 5$ it decreases from 0.051024 to 0.050611. The corresponding R_y^2 values remain very close to zero and slightly negative for all models, which is consistent with the weak signal-to-noise ratio typically observed in highly volatile assets such as Bitcoin.

In price space, XGBoost also improves upon the baseline, but the gains remain incremental rather than dramatic. At $H = 1$, $RMSE_p$ decreases from 2362.904 to 2318.117, and at $H = 5$ from 5131.843 to 5022.390. Unlike the Apple case, there is no large reversal in model ranking between return space and price space. Instead, both evaluation domains tell a broadly consistent story: machine learning captures some weak exploitable structure, but the advantage over trivial baselines remains limited.

The Diebold–Mariano p -values are 0.041 for $H = 1$ and 0.048 for $H = 5$, indicating statistical significance at the 5% level. Even so, the economic magnitude of the improvement is small. Therefore, the Bitcoin case suggests a more cautious interpretation: although machine-learning models can outperform trivial baselines, the incremental gain may or may not justify the added model complexity depending on the intended application, tolerance to forecast error, and downstream economic constraints such as transaction costs and turnover.

Taken together, the Apple and Bitcoin examples show that model superiority is asset-dependent and evaluation-domain-dependent. In some assets, a model may provide substantial gains in price-path tracking while remaining weak in return prediction; in others, the same model class may only produce small but statistically detectable improvements over strong trivial benchmarks. These contrasts support the broader argument of the paper that comparative forecasting studies should report results in both return and price spaces, while keeping return-space evaluation central to statistical interpretation.

Appendix C.3. Bayesian Optimization and Hyperparameter Selection

In addition to model comparison, hyperparameter tuning was performed using Bayesian optimization within the pre-evaluation training block. The optimization procedure was strictly confined to rolling validation splits (walk-forward), ensuring that the

final external hold-out set remained completely untouched during model selection. This design preserves the integrity of the out-of-sample evaluation and avoids any form of information leakage.

Bayesian optimization was applied to all non-baseline models, including XGBoost, LSTM, and CNN/TCN, across both assets (Apple and Bitcoin), prediction horizons ($H = 1, 5$), and lookback windows ($L \in \{60, 180, 500\}$). The optimization objective was defined in terms of minimizing the return-space error (RMSE_y) over the validation folds.

The results indicate that optimal hyperparameter configurations are highly dependent on both the asset and the prediction horizon. For Apple, deeper temporal architectures (CNN/TCN) with shorter lookback windows ($L = 60$) were consistently selected, reflecting the presence of stronger short-term temporal structure. In contrast, for Bitcoin, tree-based models such as XGBoost exhibited greater robustness, with optimal configurations favoring moderate lookback windows ($L = 180$ for $H = 1$ and $L = 60$ for $H = 5$).

These findings reinforce the importance of horizon-specific and asset-specific hyperparameter tuning. Using a single fixed configuration across all horizons and assets would fail to capture the heterogeneity observed in financial time series. Instead, Bayesian optimization provides a principled mechanism for adapting model complexity and regularization to the underlying data-generating process.

Overall, the inclusion of Bayesian optimization within a leakage-controlled framework strengthens the reliability of the reported results and supports the broader methodological contribution of this work: combining strict temporal validation with adaptive model calibration leads to more robust and interpretable forecasting comparisons.

References

1. Reschenhofer, E.; Mangat, M.K.; Zwatz, C.; Guzmics, S. Evaluation of current research on stock return predictability. *J. Forecast.* **2020**, *39*, 334–351. [\[CrossRef\]](#)
2. Bhowmik, R.; Wang, S. Stock Market Volatility and Return Analysis: A Systematic Literature Review. *Entropy* **2020**, *22*, 522. [\[CrossRef\]](#)
3. Lv, W.; Qi, J. Stock Market Return Predictability: A Combination Forecast Perspective. *Int. Rev. Financ. Anal.* **2022**, *84*, 102376. [\[CrossRef\]](#)
4. Chhajer, P.; Shah, M.; Kshirsagar, A. The Applications of Artificial Neural Networks, Support Vector Machines, and Long–Short Term Memory for Stock Market Prediction. *Decis. Anal. J.* **2022**, *2*, 100015. [\[CrossRef\]](#)
5. Masini, R.P.; Medeiros, M.C.; Mendes, E.F. Machine Learning Advances for Time Series Forecasting. *J. Econ. Surv.* **2023**, *37*, 76–111. [\[CrossRef\]](#)
6. Lu, W.; Li, J.; Wang, J.; Qin, L. A CNN–BiLSTM–AM Method for Stock Price Prediction. *Neural Comput. Appl.* **2021**, *33*, 4741–4753. [\[CrossRef\]](#)
7. Chen, Y.; Fang, R.; Liang, T.; Sha, Z.; Li, S.; Yi, Y.; Zhou, W.; Song, H. Stock Price Forecast Based on CNN–BiLSTM–ECA Model. *Sci. Program.* **2021**, *2021*, 2446543. [\[CrossRef\]](#)
8. Livieris, I.E.; Pintelas, E.; Pintelas, P. A CNN–LSTM Model for Gold Price Time-Series Forecasting. *Neural Comput. Appl.* **2020**, *32*, 17351–17360. [\[CrossRef\]](#)
9. Mndawe, S.T.; Paul, B.S.; Doorsamy, W. Development of a Stock Price Prediction Framework for Intelligent Media and Technical Analysis. *Appl. Sci.* **2022**, *12*, 719. [\[CrossRef\]](#)
10. Ayala, J.; García-Torres, M.; Vázquez Noguera, J.L.; Gómez-Vela, F.; Divina, F. Technical analysis strategy optimization using a machine learning approach in stock market indices. *Knowl.-Based Syst.* **2021**, *228*, 107119. [\[CrossRef\]](#)
11. Su, Y.; Cui, C.; Qu, H. Self-Attentive Moving Average for Time Series Prediction. *Appl. Sci.* **2022**, *12*, 3602. [\[CrossRef\]](#)
12. Vancsura, L.; Tatay, T.; Bareith, T. Navigating AI-Driven Financial Forecasting: A Systematic Review of Current Status and Critical Research Gaps. *Forecasting* **2025**, *7*, 36. [\[CrossRef\]](#)
13. Quaedvlieg, R. Multi-Horizon Forecast Comparison. *J. Bus. Econ. Stat.* **2019**, *39*, 40–53. [\[CrossRef\]](#)
14. Yilmaz, F.M.; Yildiztepe, E. Statistical Evaluation of Deep Learning Models for Stock Return Forecasting. *Comput. Econ.* **2024**, *63*, 221–244. [\[CrossRef\]](#)
15. Kumari, D.A.T. The Impact of Financial Literacy on Investment Decisions: With Special Reference to Undergraduates in Western Province, Sri Lanka. *Asian J. Contemp. Educ.* **2020**, *4*, 110–126. [\[CrossRef\]](#)
16. Bürgi, C.R.S. Assessing the Accuracy of Directional Forecasts. *Appl. Econ.* **2025**, *57*, 7909–7920. [\[CrossRef\]](#)

17. Lazarus, E. Horizon-Dependent Risk Pricing: Evidence from Short-Dated Options. *SSRN Electron. J.* **2022**. [[CrossRef](#)]
18. Vayanos, D.; Vila, J.-L. A Preferred-Habitat Model of the Term Structure of Interest Rates. *Econometrica* **2021**, *89*, 77–112. [[CrossRef](#)]
19. Erdemlioglu, D.; Gradojevic, N. Heterogeneous Investment Horizons, Risk Regimes, and Realized Jumps. *Int. J. Financ. Econ.* **2021**, *26*, 617–643. [[CrossRef](#)]
20. Nishiwaki, T. Impact of Different Investment Horizons in Heterogeneous Agent Models: Do Long-Term Traders Bring Market Stability? *J. Econ. Behav. Organ.* **2022**, *196*, 393–401. [[CrossRef](#)]
21. Abolghasemi, M.; Hyndman, R.J.; Spiliotis, E.; Bergmeir, C. Model Selection in Reconciling Hierarchical Time Series. *Mach. Learn.* **2022**, *111*, 739–789. [[CrossRef](#)]
22. Hahn, Y.; Langer, T.; Meyes, R.; Meisen, T. Time Series Dataset Survey for Forecasting with Deep Learning. *Forecasting* **2023**, *5*, 315–335. [[CrossRef](#)]
23. Nagula, P.K.; Alexakis, C. A Novel Machine Learning Approach for Predicting the NIFTY50 Index in India. *Int. Adv. Econ. Res.* **2022**, *28*, 155–170. [[CrossRef](#)]
24. Xu, J.; He, J.; Gu, J.; Wu, H.; Wang, L.; Zhu, Y.; Wang, T.; He, X.; Zhou, Z. Financial Time Series Prediction Based on XGBoost and Generative Adversarial Networks. *Int. J. Circuits Syst. Signal Process.* **2022**, *16*, 79. [[CrossRef](#)]
25. Jiang, M.; Liu, J.; Zhang, L.; Liu, C. An Improved Stacking Framework for Stock Index Prediction by Leveraging Tree-Based Ensemble Models and Deep Learning Algorithms. *Phys. A* **2020**, *541*, 122272. [[CrossRef](#)]
26. Yu, C.; Liu, F.; Zhu, J.; Guo, S.; Gao, Y.; Yang, Z.; Liu, M.; Xing, Q. Gradient Boosting Decision Tree with LSTM for Investment Prediction. *arXiv* **2025**, arXiv:2505.23084. [[CrossRef](#)]
27. Mehtab, S.; Sen, J.; Dutta, A. Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models. In *Machine Learning and Metaheuristics Algorithms, and Applications; Communications in Computer and Information Science*; Springer: Singapore, 2021; pp. 88–106. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.