Lisbon School of Economics & Management
Universidade de Lisboa

MASTER

APPLIED ECONOMETRICS AND FORECASTING

MASTER´S FINAL WORK

DISSERTATION

THE PERFORMANCE OF THE TABULAR PRIOR-DATA FITTED NETWORK IN VOLATILITY FORECASTING: AN EMPIRICAL COMPARISON WITH HAR AND MACHINE LEARNING MODELS

GERRIT WILLEM GIJS NIJLAND

SUPERVISION:

JOÃO AFONSO BASTOS

JULY - 2026

GLOSSARY

HAR - Heterogeneous Autoregressive

AR - Autoregressive

ML - Machine learning

RV - Realized variance

KAN - Kolmogorov-Arnold Network

TKAN - Temporal Kolmogorov-Arnold Network

RNN - Recurrent Neural Network

LSTM - Long Short-Term Memory

GRU - Gated Recurrent Unit

TCN - Temporal Convolutional Network

XGB - Extreme Gradient Boosting

LGBM - Light Gradient Boosting Machine

TabPFN - Tabular Prior-data Fitted Network

SCM - Structural Causal Model

MSE - Mean squared error

QLIKE - Quasi-likelihood

MCS - Model Confidence Set

DM - Diebold-Mariano

CW - Clark-West

EnbPI - Ensemble Batch Prediction Intervals

LOO - Leave-one-out

QS - Quantile score

WQS - Weighted quantile score

ADF - Augmented Dickey-Fuller

Keywords: Volatility forecasting, Realized variance, High frequency data, Heterogeneous Autoregressive model, Machine learning, Kolmogorov-Arnold Network, TabPFN, Conformal prediction, Quantile forecasting

ABSTRACT

This paper forecasts realized variance, a nonparametric proxy for volatility constructed from high-frequency data, for the stock of Emerson Electric. Intraday price data spanning approximately 18 years are obtained from the Trade and Quote database hosted by Wharton Research Data Services, and the resulting realized variance series is complemented by a set of exogenous predictors. Forecasts are generated using several variants of the Heterogeneous Autoregressive model alongside a range of machine learning approaches, including the Kolmogorov–Arnold Network and the Tabular Prior-data Fitted Network. For each model, we evaluate conditional mean forecasts, prediction intervals, and quantile forecasts across three horizons: one day ahead, the average week ahead, and the average month ahead. In this single-stock setting, TabPFN achieves the strongest performance across all forecasting tasks at the week- and month-ahead horizons, while no single model dominates at the day-ahead horizon.

Contents

1 INTRODUCTION

Volatility forecasting is a central topic in financial econometrics, due to its direct relevance for risk management, asset pricing and portfolio allocation. (Poon & Granger 2005, 2003, Andersen et al. 2005) While asset returns are notoriously difficult to predict due to low serial correlation in logarithmic returns, volatility exhibits a higher degree of predictability because it exhibits stylized facts like volatility clustering, long memory persistence, heavy tail return distributions and leverage effects. (Teräsvirta & Zhao 2014, Baillie 1996, Huang & Luo 2024, Bekaert & Wu 2000) This has led to the development of a wide range of econometric and statistical techniques looking to capture the complexity of volatility dynamics, (So et al. 2022) think of the GARCH model as developed by Bollerslev (1986), and the numerous extensions that followed after its original publication. (Teräsvirta 2009) Also among these volatility models is the Heterogeneous Autoregressive (HAR) model of Corsi (2009), which has become a benchmark model due to its ability to approximate long memory patterns using a parsimonious and interpretable linear structure. The model combines non-parametrically constructed realized variance regressors at multiple frequencies, embedded within a parametric AR(3) functional form. After the initial publication of the model, numerous extensions have been proposed to account for nonlinearities, structural breaks and jumps that are not uncommon in financial markets. (Khan 2015, Wen et al. 2016, Andersen et al. 2012)

Despite the development of econometric methods, the increased availability of rich information sets in today's world has stimulated interest in machine learning methods for volatility forecasting. Machine learning models are designed to capture nonlinear interactions and deal with large predictor spaces without requiring strong parametric assumptions. (Gunnarsson et al. 2024) Recent empirical work suggests that machine learning techniques can extract predictive signals beyond those captured by econometric models; traditional econometric models often suffer from overfitting once additional variables are included, whereas machine learning methods are generally better equipped to handle high-dimensional predictor spaces. (Christensen et al. 2023)

However, machine learning models are frequently criticized for being black boxes; their internal mechanisms are often opaque, making it hard to understand how inputs translate into predictions. Unlike traditional econometric models with interpretable parameters, many machine learning algorithms rely on complex nonlinear structures that offer little economic intuition. This lack of transparency can make them harder to explain and implement for practitioners. (Hoepner et al. 2021)

This paper seeks to build on and extend the existing literature on volatility forecasting in several important ways. Recent work has explored the use of machine learning algorithms for forecasting volatility. (Bucci 2020) More specifically, several studies have adopted this framework of comparing the forecasting performance of HAR models with various machine learning approaches. (Rahimikia & Poon 2026, Christensen et al. 2023, Brauneis & Sahiner 2024, Teller et al. 2022, Chun et al. 2025, Taneva-Angelova & Granchev 2025) This paper contributes to that literature by evaluating a broader range of machine learning models and by introducing the relatively new Kolmogorov–Arnold Network and TabPFN models, which, to the best of my knowledge, have not yet been applied in a realized variance forecasting setting.

Lastly, most studies comparing econometric and machine learning models focus mainly on point forecast accuracy. While informative, point forecasts provide limited insight into forecast uncertainty and tails of the forecasting distribution. To address this gap, we also evaluate prediction intervals and extreme outcomes. Specifically, we use Conformal Prediction to construct prediction and quantile regression to forecast tail events.

2 LITERATURE REVIEW

Christensen et al. (2023) compare several variants of the HAR model, including log-HAR, LevHAR, SHAR, and HARQ, with a range of machine-learning methods, including bootstrap aggregation, random forests, gradient boosting, and neural networks with varying depths and numbers of neurons. The analysis is conducted on a dataset of 29 individual stocks contained in the Dow Jones Industrial Average, spanning 17 years, using daily realized variances computed from intraday trading data. They find that machine learning models outperform HAR models in forecasting realized variance in a large regressor space-setting. Moreover, especially random forests and neural networks are preferred as they especially give superior prediction results relative to HAR models, particularly at longer horizons. A similar setup is used by Branco et al. (2024), who forecast realized volatility for ten global equity indices using intraday realized variance data over a period of 21 years, also including exogenous variables, amounting to 27 variables in total. Models include HAR, HAR-X, logHAR, LevHAR and SHAR, among others, as well as regularized versions. With respect to machine learning models, they include bootstrap aggregation, random forests, gradient boosting, fully connected neural networks with different topologies. They find that additional predictors improve forecasts for daily and weekly time horizons, where HAR-X and regularized linear models often have the lowest mean squared error. Also, machine learning models do not systematically outperform linear models. Next, Audrino & Chassot (2025) research the forecasting performance of the HAR model relative to a range of machine learning techniques using a dataset comprising 1,445 individual stocks over a time period of almost 9 years. They include realized volatility and the VIX in the predictor space. Despite extensive hyperparameter tuning for the ML methods, the machine learning models do not outperform the HAR benchmark when the HAR model is carefully fitted. The study finds that the HAR model's linear specification and interpretability as well a ease of computing make it a practical tool for realized volatility forecasting. The authors conclude that properly specified HAR models provide superior forecasting accuracy when the predictor space is small. Rahimikia & Poon (2026) evaluate the forecasting performance of a broad set of machine learning models for realized volatility using daily data from 2007 to 2022. The authors construct a rich feature set consisting of six HAR variables, 132 limit order book variables and nine news sentiment variables. Across more than seven million model variants, they find that ML models consistently outperform HAR models for roughly 90% of out-of-sample days, particularly when realized volatility is not extremely high. However, during periods of extreme volatility HAR models outperform ML methods, indicating that ML models perform worse in turbulent market regimes. Overall, the authors conclude that

neither ML nor HAR models dominate uniformly; performance depends on model design and volatility regimes. The authors emphasize that ML should not directly replace HAR in practice, and they highlight opportunities for future work, including multi-step forecasting, regime-switching methods and tree-based models. Li et al. (2024) use 5-minute high-frequency data from nine Chinese educational companies (2015–2022) to construct a weighted volatility index and compare the forecasting performance of the HAR model to a set of LSTM neural-network models across daily, weekly, and monthly horizons. Their results show that the HAR framework is effective for capturing short-, medium-, and long-memory components of volatility, but the LSTM models consistently outperform HAR models at all horizons, especially for weekly and monthly forecasts, where nonlinear dynamics are more dominant. The LSTM variants produce substantially lower RMSE, MAE, and MAPE values, with the largest performance improvement observed in the monthly volatility forecasts, indicating that LSTM's ability to model nonlinear and long-term dependencies gives it a clear advantage over the linear HAR structure. Overall, the study concludes that LSTM delivers superior predictive accuracy on daily, weekly, and monthly volatility, with the performance gap widening as the forecast horizon increases. Li et al. (2025) study realized volatility in the Shanghai Agricultural Stock Index using 5-minute high-frequency data from March 2017 to May 2021, evaluating the comparative forecasting performance of HAR, LSTM, and a hybrid PCA-LSTM model. Across all experiments, the results show that LSTM consistently outperforms the HAR model under identical forecasting conditions, delivering higher predictive accuracy. When the models are expanded to a larger regressor space, the advantage of LSTM becomes even clearer. Most notably, the study finds that the PCA-LSTM model achieves the best overall performance, producing the lowest forecasting errors among all models. The authors conclude that while HAR captures long memory structure effectively, machine learning architectures, especially PCA-LSTM, provide superior accuracy for realized volatility forecasting.

Kolmogorov–Arnold Networks have recently been applied to a range of time-series forecasting problems, among which in financial settings characterized by nonlinear dynamics and evolving regimes. Xu et al. (2024) investigate the use of KANs for volatility forecasting and introduce two variants: Temporal KAN for univariate series and Multivariate Temporal KAN for multivariate environments. Using financial time-series constructed from daily OHLCV data with realized volatility as the target, they show that T-KAN attains predictive performance comparable to or exceeding MLP, RNN, and LSTM benchmarks despite a substantially smaller parameter set, while MT-KAN further improves accuracy by explicitly exploiting cross-series interactions. The authors attribute these re-

sults to the architectural principle of KANs, which model complex temporal relationships through adaptive nonlinear mappings applied at the level of individual inputs, allowing the model to respond flexibly to persistence, regime changes, and nonlinear volatility feedback while remaining interpretable through post-training symbolic analysis. Complementing this evidence, Yamak et al. (2025) provide a comprehensive review of KAN-based models for time-series forecasting across univariate, multivariate, and high-frequency settings, covering benchmark and real-world datasets from electricity load, traffic, weather, financial markets, hydrology, and sensor networks. The review documents that KAN-based models and their extensions, including TKAN, SigKAN, and hybrid variants incorporating attention or convolutional components, frequently match or outperform LSTMs and Transformers while relying on fewer effective parameters and offering direct insight into feature contributions. At the same time, the authors highlight important trade-offs, including increased computational cost during training, scalability challenges in high-dimensional settings, sensitivity to architectural and regularization choices, and a gradual erosion of interpretability as KANs are embedded in more complex hybrid frameworks. Next, Genet & Inzirillo (2024) propose the Temporal Kolmogorov–Arnold Transformer, which replaces recurrent modules in the Temporal Fusion Transformer with Temporal KAN layers tailored to sequential data. The model is evaluated on hourly cryptocurrency trading data over 2020–2022, with the task of multi-step prediction of traded notional volumes using observed market signals and calendar features. The results show that TKAT outperforms GRU- and LSTM-based benchmarks at longer forecasting horizons and that substituting LSTM layers with KAN-based temporal components consistently improves predictive stability within a transformer framework. These gains are attributed to KAN-based temporal processing that captures nonlinear and regime-dependent dynamics more effectively than standard recurrent layers, although the authors note higher training costs and weaker single-step performance due to the overall architectural complexity. Zhang et al. (2025) study the use of Kolmogorov–Arnold Networks in electricity demand forecasting by integrating KAN components into three established time-series models: Temporal Convolutional Networks, Bidirectional LSTMs, and Transformers. Using monthly UK electricity demand data from 2015 to 2024, augmented with economic and meteorological variables such as temperature, precipitation, wind speed, and consumer prices, the authors evaluate the proposed hybrid models against their standard counterparts. The empirical results show that incorporating KAN layers consistently improves forecasting accuracy across all architectures, with notable reductions in RMSE and MAE and higher explanatory power, particularly for medium- to long-horizon forecasts. The performance gains are attributed to the ability of KAN components to refine feature representations and capture complex nonlinear relationships that are insufficiently modeled by standard

fully connected output layers. At the same time, the authors note trade-offs related to increased model complexity and computational cost, especially when KANs are combined with already resource-intensive architectures such as Transformers, highlighting the need to balance accuracy gains against practical deployment considerations. Lastly, Hoo et al. (2025) propose TabPFN-TS, which uses the tabular model TabPFN-v2 for time-series forecasting by converting each time step into a row of features and applying it as a regression model. TabPFN itself is a Transformer trained beforehand on many synthetic tabular datasets to approximate Bayesian predictions: at inference, it takes the observed data as input and directly outputs predictions for new points in a single forward pass, with no need for retraining or optimization on the dataset. (Hollmann et al. 2022) The authors evaluate performance using 28 forecasting tasks across domains like energy, retail, and finance. They find that TabPFN-TS performs best among the compared models in this setting, outperforming time-series foundation models such as TiRex, Chronos-Bolt, Moirai, Toto, and Sundial on the same tasks in terms of MASE and WQL errors.

3 DATA

3.1 Computation of realized variance

We use a dataset containing 18 years of intraday (millisecond timestamps) price data for Emerson Electric, listed on the New York Stock Exchange, covering the period January 3, 2006 to December 29, 2023. This period includes several major macroeconomic events that contributed to increased market-wide volatility, including the Global Financial Crisis (2007–2009), the COVID-19 market crash (2020), and the Russia–Ukraine war (2022–present). The data is retrieved from the NYSE Trade and Quote database, hosted by Wharton Research Data Services. Regarding data filtering, we retain only observations recorded during regular trading hours (09:30–16:00) and remove outliers using algorithms as implemented by Barndorff-Nielsen et al. (2009). Additionally, we exclude the data for December 12, 2006, because Emerson Electric underwent a 2-for-1 stock split on that day, which halved the share price and generated an extremely high daily realized variance that does not reflect any underlying economic event. After following this procedure, we end up with data for $T = 4,528$ trading days. As the daily realized variance will be constructed from the intraday 5-minute logarithmic returns of the stock price, it is useful to understand the assumptions on how the logarithmic price evolves over time. In definition 3.1, those dynamics are explained.

Definition 3.1 *Logarithmic price dynamics*
Consider the stochastic process $X = (\Omega, \mathcal{F}, (\mathcal{F})_{t\geq 0}, (X_t)_{t\geq 0}, \mathbb{P})$ *where* $(\Omega, \mathcal{F}, \mathbb{P})$ *is a probability space,* $(\mathcal{F})_{t\geq 0}$ *is a filtration, i.e. an increasing family of* σ*-algebras and* $(X_t)_{t\geq 0}$ *is the logarithmic price process. The logarithmic price follows the semi-martingale process as specified in 1:*

$$dX(t) = \mu(t)dt + \sigma(t)dW(t), \quad \forall t \in [0, T], \tag{1}$$

where $\mu(t)$ *is a drift process with finite variation,* $\sigma(t)$ *is a càdlàg stochastic volatility process and* $W(t)$ *is a standard Brownian Motion.*

To capture volatility, we consider the concept of integrated variance, which is the cumulative variance of a process over a specified time interval. Since the integrated variance over day t is defined as

$$IV_t = \int_{t-1}^{t} \sigma(s)^2 \, ds \tag{2}$$

and is unobserved, we require an estimator for it. Under the assumed log-price dynamics in equation 1, the process is a continuous Itô semimartingale without jumps. In

this setting, the quadratic variation of $X(t)$ over $[t-1, t]$ equals the integrated variance, i.e. $QV_t = IV_t$. Barndorff-Nielsen & Shephard (2002) show that realized variance constructed from high-frequency intraday returns satisfies $RV_t \xrightarrow[n\to\infty]{p} QV_t$. Because $QV_t = IV_t$ under continuity, realized variance provides a consistent estimator of integrated variance, so $RV_t \xrightarrow[n\to\infty]{p} IV_t$. Moreover, Barndorff-Nielsen & Shephard (2002) showed that as the intraday sampling frequency increases, the error distribution converges to that given in 3.

$$\sqrt{n}\,(RV_t - IV_t) \xrightarrow{d} \mathcal{N}\left(0,\ 2\int_{t-1}^{t} \sigma_s^4\, ds\right) \tag{3}$$

The daily realized variance is constructed from intraday 5-minute logarithmic returns, and the explicit construction of daily realized variance is displayed in 4:

$$RV_t = \sum_{i=1}^{n} r_{t,i}^2 \qquad where \quad r_{t,i} = \log\left(\frac{P_{t,i}}{P_{t,i-1}}\right), \tag{4}$$

where t is the day and $n = 1/\Delta$ is the number of intraday logarithmic returns, where Δ is the length of intraday sampling interval (5 minutes). Lastly, $r_{t,i}$ is the i-th intraday log-return of day t.

3.2 Exogenous variables

In addition to the regressors constructed from past realized variances, the paper incorporates a set of exogenous variables, namely: the U.S. 3-month Treasury Bill rate (TMTB), the Economic Policy Uncertainty index (EPU), the Chicago Board Options Exchange Volatility Index (VIX), crude oil prices and the Business Conditions index (BC). The data for the first four variables are sourced from the Federal Reserve Bank of St.Louis, and the data for the last variable is sourced from the Federal Reserve Bank Philadelphia.

3.3 Summary statistics

Let $\{RV_t\}_{t=1}^{T}$ denote the daily realized variance series. In our daily-frequency setting, the monthly HAR regressor $RV_{t-1|t-22}$ is constructed from the preceding 22 trading days. Therefore, the first 22 observations of the daily realized variance series cannot be used for model estimation, as they are required to compute the monthly HAR component. Consequently, this variable first becomes available on the 23rd day, which we set as $t = 1$ in our sample. For the exogenous variables, we handle the few missing values by replacing the missing value with observed value of the day before. In table 1, the summary statistics of all variables are presented.

Table 1: Summary Statistics

Acronym	Mean	Median	Minimum	Maximum	Standard deviation	Skewness	Kurtosis
RV_t	3.604	1.670	0.165	190.694	8.929	10.782	163.061
$RV_{t\|t-4}$	3.605	1.914	0.281	124.711	7.108	8.564	100.036
$RV_{t\|t-21}$	3.605	2.171	0.427	68.755	5.951	6.460	50.451
VIX	19.745	17.340	9.140	82.690	9.047	2.408	8.593
EPU	117.920	96.690	3.320	807.660	83.203	2.345	8.662
COP	72.491	71.450	-36.980	145.310	22.365	0.175	-0.384
TMTB	1.302	0.220	-0.050	5.360	1.733	1.227	0.038
BC	-0.287	-0.123	-26.578	9.647	2.220	-6.884	77.098

Because we want to report results for two different information sets, we define the corresponding data sets as $\mathcal{D}_{\text{HAR}}$ and $\mathcal{D}_{\text{HAR-X}}$. Let the lagged realized variance components entering the HAR model be collected in the vector

$$X_t = \left(RV_t,\, RV_{t|t-4},\, RV_{t|t-21}\right)^\top, \tag{5}$$

where RV_t denotes daily lag of realized variance, $RV_{t|t-4}$ average weekly lag of realized variance, and $RV_{t|t-21}$ average monthly lag of realized variance. In addition, let the set of exogenous variables be defined as

$$Z_t = \left(EPU_t,\, VIX_t,\, BC_t,\, COP_t,\, \Delta TMTB_t{}^{1}\right)^\top. \tag{6}$$

Using these components, the two data sets are formally defined as $\mathcal{D}_{\text{HAR}} = \left\{\, X_t \,\right\}_{t=1}^{T}$ and $\mathcal{D}_{\text{HAR-X}} = \left\{\, (X_t,\, Z_t) \,\right\}_{t=1}^{T}$, respectively.

[1]The variable is differenced to account for nonstationarity, as can be concluded from appendix A.1.

4 METHODOLOGY

4.1 Econometric models

The volatility modelling literature has been strongly influenced by the Heterogeneous Autoregressive model introduced by Corsi (2009), which has become a well-established benchmark in volatility forecasting due to its simplicity, interpretability, and strong empirical performance. By capturing the heterogeneous nature of volatility dynamics through daily, weekly, and monthly components, the HAR framework provides a flexible yet parsimonious way to model the long-memory volatility behavior in financial markets. Building on this foundation, numerous extensions have been proposed to incorporate exogenous predictors, nonlinear dynamics, leverage effects, asymmetries, and structural breaks, thereby broadening the model's applicability in modern volatility forecasting research. In this paper, we will consider a subset of them.

4.1.1 HAR model

The standard heterogeneous autoregressive model of Corsi (2009) captures volatility dynamics by combining daily, weekly, and monthly lagged aggregates, and is specified in 7:

$$RV_t = \beta_0 + \beta_1 RV_{t-1} + \beta_2 RV_{t-1|t-5} + \beta_3 RV_{t-1|t-22} + \varepsilon_t, \tag{7}$$

where the regressors are defined as lagged realized variances: the daily component is RV_{t-1}, the weekly component is $RV_{t-1|t-5} \equiv \frac{1}{5}\sum_{i=1}^{5} RV_{t-i}$, and the monthly component is $RV_{t-1|t-22} \equiv \frac{1}{22}\sum_{i=1}^{22} RV_{t-i}$.

4.1.2 HAR-X model

The idea of the HAR-X model, as used by Clements et al. (2024), is an extension of the HAR model in the sense that next to the endogenous realized variance regressors, the regressor space is augmented with the additional exogenous regressors. Hence, the dataset used for estimation and forecast evaluation is $\mathcal{D}_{\text{HAR-X}}$. The HAR-X model functional form is described in 8:

$$\begin{aligned} RV_t = \beta_0 &+ \beta_1 RV_{t-1} + \beta_2 RV_{t-1|t-5} + \beta_3 RV_{t-1|t-22} + \beta_4 EPU_{t-1} + \beta_5 VIX_{t-1} \\ &+ \beta_6 BC_{t-1} + \beta_7 COP_{t-1} + \beta_8 \Delta TMTB_{t-1} + \varepsilon_t. \end{aligned} \tag{8}$$

4.1.3 log-HAR model

The log-HAR model (Corsi 2009) models volatility in terms of the logarithm of realized variance rather than its level. Taking logs stabilizes the variance and mitigates the influence of extreme volatility observations that arise from the heavy-tailed nature of realized variance.

$$\log RV_t = \beta_0 + \beta_1 \log RV_{t-1} + \beta_2 \log RV_{t-1|t-5} + \beta_3 \log RV_{t-1|t-22} + \varepsilon_t \tag{9}$$

When converting the log-HAR forecast of $\log RV_t$ back to the level RV_t, it is important to note that one cannot simply apply the exponential function. The log-HAR model produces a prediction $\hat{f}(X_{t-1})$ (or $\hat{f}(X_{t-1}, Z_{t-1})$ in the HAR-X specification) of $\log RV_t$, where $\hat{f}$ denotes the estimated forecast function mapping past observations X_{t-1} to a forecast of $\log RV_t$. Because the exponential transformation is non-linear, the naive back-transformation $\exp(\hat{f})$ is biased by Jensen's inequality. To obtain an unbiased forecast of realized variance, we apply the standard log-normal bias correction, as also used by Christensen et al. (2023):

$$\mathbb{E}\Big[\exp\Big(\hat{f}(X_{t-1})\Big)\Big] = \exp\Big(\hat{f}(X_{t-1}) + \tfrac{1}{2}\,\widehat{\mathrm{Var}}\Big(\hat{f}(X_{t-1})\Big)\Big)\,, \tag{10}$$

and analogously for the HAR-X specification:

$$\mathbb{E}\Big[\exp\Big(\hat{f}(X_{t-1}, Z_{t-1})\Big)\Big] = \exp\Big(\hat{f}(X_{t-1}, Z_{t-1}) + \tfrac{1}{2}\,\widehat{\mathrm{Var}}\Big(\hat{f}(X_{t-1}, Z_{t-1})\Big)\Big)\,. \tag{11}$$

Here, $\widehat{\mathrm{Var}}(\hat{f})$ denotes the estimated variance of the log-HAR residuals computed on the training set. This correction is appropriate because the residuals of the log-HAR regression are approximately Gaussian, i.e., $\log RV_t$ is approximately conditionally normal given its predictors[2].

4.1.4 HAR-Q model

The HAR-Q model, as introduced by Bollerslev et al. (2016), incorporates realized quarticity as a proxy for intraday return variability, capturing the volatility-of-volatility. By incorporating realized quarticity as a measure of volatility-of-volatility, the HAR-Q model adapts to changes in intraday uncertainty, making it better suited for forecasting realized variance during periods of market turbulence. The HAR-Q model is defined as

[2]See Appendix A.3 for a visualization of the distribution of the training set residuals.

in 12, where $RQ_t = \frac{n}{3}\sum_{i=1}^{n} r_{t,i}^4$:

$$RV_t = \beta_0 + \left(\beta_1 + \beta_1^Q RQ_{t-1}^{1/2}\right) RV_{t-1} + \beta_2 RV_{t-1|t-5} + \beta_3 RV_{t-1|t-22} + \varepsilon_t. \tag{12}$$

4.1.5 LevHAR model

The LevHAR (Corsi & Renò 2012) model incorporates leverage terms to capture the asymmetric response of volatility to negative returns. This modification incorporates the well-known leverage effect and might improve forecasting accuracy when downside risk dominates market movements. The LevHAR model is defined as in 13:

$$RV_t = \beta_0 + \beta_1 RV_{t-1} + \beta_2 RV_{t-1|t-5} + \beta_3 RV_{t-1|t-22} + \gamma_1 r_{t-1}^- + \gamma_2 r_{t-1|t-5}^- + \gamma_3 r_{t-1|t-22}^- + \varepsilon_t, \tag{13}$$

where $r_{t-1|t-h} = \frac{1}{h}\sum_{i=1}^{h} r_{t-i}$ and $r_{t-1|t-h}^- = \min\left(0,\, r_{t-1|t-h}\right)$

4.1.6 SHAR model

Another way to model asymmetry is by considering the SHAR (Patton & Sheppard 2015) model, which decomposes realized variance into positive and negative semivariances to explicitly model volatility asymmetry. Since downside and upside returns might have different predictive content for future volatility, including semivariances might yield a more informative HAR structure, as described in 14;

$$RV_t = \beta_0 + \beta_1^- RV_{t-1}^- + \beta_1^+ RV_{t-1}^+ + \beta_2 RV_{t-1|t-5} + \beta_3 RV_{t-1|t-22} + \varepsilon_t, \tag{14}$$

where the positive and negative semivariances are defined as in 15:

$$RV_t^+ = \sum_{i=1}^{n} r_{t,i}^2 \, \mathbf{1}_{\{r_{t,i}>0\}} \qquad \text{and} \qquad RV_t^- = \sum_{i=1}^{n} r_{t,i}^2 \, \mathbf{1}_{\{r_{t,i}<0\}}. \tag{15}$$

4.2 Machine learning models

We now turn to several machine learning models with network- and tree-based architectures. While our primary focus lies on the (Temporal) Kolmogorov-Arnold Network and TabPFN, and on comparing their forecasting performance to the HAR models introduced earlier, it is also good practice to benchmark KANs against other neural network[3]

[3] All machine learning models (except TabPFN and TKAN) are estimated by minimizing a squared-error loss, combined with the architecture-specific regularization term.

and tree-based architectures. In addition, comparing these alternative network models to the HAR benchmarks provides a more complete picture of their relative strengths and weaknesses, particularly because we evaluate forecasting performance across multiple time horizons, where different model structures may perform better or worse relative to one another on different time horizons.

4.2.1 Kolmogorov-Arnold Network

Let $\mathbf{x}_t \in \mathbb{R}^d$ collect the predictors observed at time t, that is the regressors in $\mathcal{D}_{\text{HAR}}$ and, for the extended specification, additionally those in $\mathcal{D}_{\text{HAR-X}}$, and let y_t denote the target. At each forecast origin the estimation window contains the N observations $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$. Bold lowercase symbols denote vectors and bold uppercase symbols matrices or operators. The Kolmogorov-Arnold network of Liu et al. (2024) is derived from the representation theorem in Definition 4.2.

Definition 4.2 *Kolmogorov-Arnold representation theorem*
Let $f : [0,1]^d \to \mathbb{R}$ be a multivariate continuous function on a compact domain. Then f can be written as a finite composition of univariate continuous functions and addition: there exist continuous functions $\phi_{q,p} : [0,1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$ such that for all $\mathbf{x} = (x_1, \ldots, x_d) \in [0,1]^d$,

$$f(\mathbf{x}) = \sum_{q=1}^{2d+1} \Phi_q \left(\sum_{p=1}^{d} \phi_{q,p}(x_p) \right). \tag{16}$$

Theorem 4.2 guarantees a representation with one inner layer of width $2d + 1$. The architecture of Liu et al. (2024) generalizes it to arbitrary depth and width, which is the form used here: layers are indexed by $l = 0, \ldots, L$, written as a superscript, with n_l the width of layer l, $n_0 = d$ and $n_L = 1$. Instead of applying a fixed activation function at each node, KANs attach a learnable univariate function to each edge between nodes, parameterized by a B-spline basis.

Definition 4.3 *B-splines*
Let $\{\iota_v\}_{v \in \mathbb{Z}}$ be a non-decreasing knot sequence. The B-spline basis functions $B_v^{(\kappa)} : \mathbb{R} \to \mathbb{R}$ of degree κ follow the Cox–de Boor recursion

$$B_v^{(0)}(x) = \mathbf{1}_{[\iota_v, \iota_{v+1})}(x), \qquad B_v^{(\kappa)}(x) = \frac{x - \iota_v}{\iota_{v+\kappa} - \iota_v} B_v^{(\kappa-1)}(x) + \frac{\iota_{v+\kappa+1} - x}{\iota_{v+\kappa+1} - \iota_{v+1}} B_{v+1}^{(\kappa-1)}(x).$$

Each $B_v^{(\kappa)}$ is a nonnegative piecewise polynomial with local support on $[\iota_v, \iota_{v+\kappa+1}]$ and $\kappa - 1$ continuous derivatives (De Boor 1972).

Let $\phi_{j,p}^{(l)}$ denote the edge function connecting node p in layer l to node j in layer $l+1$. It is a linear combination of m B-spline basis functions on a bounded interval $[\underline{x}, \overline{x}]$ that covers the range of the standardized inputs,

$$\phi_{j,p}^{(l)}(x) = \sum_{v=1}^{m} c_{j,p,v}^{(l)} \, B_v^{(\kappa)}(x), \qquad x \in [\underline{x}, \overline{x}], \tag{17}$$

where the coefficients $c_{j,p,v}^{(l)} \in \mathbb{R}$ are learnable and determine the shape of $\phi_{j,p}^{(l)}$. We set $\kappa = 3$, so the edge functions are twice continuously differentiable cubic splines. A node sums the incoming transformed signals, so that a KAN layer computes

$$x_j^{(l+1)} = \sum_{p=1}^{n_l} \phi_{j,p}^{(l)}\big(x_p^{(l)}\big), \qquad j = 1, \ldots, n_{l+1}, \tag{18}$$

with $\mathbf{x}^{(l)} \in \mathbb{R}^{n_l}$ the vector of activations in layer l (Liu et al. 2024). Collecting the edge functions of a layer in the array

$$\mathbf{\Phi}^{(l)} = \begin{pmatrix} \phi_{1,1}^{(l)} & \cdots & \phi_{1,n_l}^{(l)} \\ \vdots & \ddots & \vdots \\ \phi_{n_{l+1},1}^{(l)} & \cdots & \phi_{n_{l+1},n_l}^{(l)} \end{pmatrix} \qquad \text{gives} \qquad \mathbf{x}^{(l+1)} = \mathbf{\Phi}^{(l)}\big(\mathbf{x}^{(l)}\big), \tag{19}$$

where $\mathbf{\Phi}^{(l)} : \mathbb{R}^{n_l} \to \mathbb{R}^{n_{l+1}}$ is a nonlinear operator whose entries are functions rather than scalars (Somvanshi et al. 2025). Applying it means evaluating each $\phi_{j,p}^{(l)}$ at $x_p^{(l)}$ and summing over p, as in (18); it is therefore not a matrix product, and $\mathbf{\Phi}^{(l)}$ is to be distinguished from the scalar outer functions Φ_q in (16). With $\mathbf{x}^{(0)} = \mathbf{x}_t$ and $n_L = 1$, the forecast is the composition of the layers,

$$\hat{y}_t = \big(\mathbf{\Phi}^{(L-1)} \circ \mathbf{\Phi}^{(L-2)} \circ \cdots \circ \mathbf{\Phi}^{(0)}\big)\,(\mathbf{x}_t). \tag{20}$$

The objective minimized over the estimation window is

$$\mathcal{J}_{\mathrm{KAN}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda_\phi \sum_{l,j,p} \int_{\underline{x}}^{\overline{x}} \left(\frac{\partial^2}{\partial x^2} \phi_{j,p}^{(l)}(x) \right)^2 dx, \qquad \lambda_\phi > 0, \tag{21}$$

following Yamak et al. (2025). The second derivative measures how sharply an edge function bends; penalizing its square discourages rapid changes in slope and so limits

how irregular the fitted functions can become.

4.2.2 Temporal Kolmogorov-Arnold Network

The model above maps a single vector $\mathbf{x}_t$ to a forecast. The remaining network models take an ordered input sequence $(\mathbf{x}_{t-S+1}, \ldots, \mathbf{x}_t)$ of length S, with sequence positions indexed by $u = t-S+1, \ldots, t$. The building block of a Temporal Kolmogorov-Arnold Network (Genet & Inzirillo 2024, Yamak et al. 2025) is the Recurrent Kolmogorov-Arnold Network layer, which augments (18) with an explicit memory state. Let $x_p^{(l)}(u)$ be the activation of node p in layer l at position u, and $s_p^{(l)}(u)$ the associated memory state. An RKAN layer computes

$$x_j^{(l+1)}(u) = \sum_{p=1}^{n_l} \psi_{j,p}^{(l)}\big(x_p^{(l)}(u),\, s_p^{(l)}(u)\big)\,, \qquad j = 1, \ldots, n_{l+1}, \tag{22}$$

where $\psi_{j,p}^{(l)} : \mathbb{R}^2 \to \mathbb{R}$ is a learnable edge function, parameterized as in (17) but taking the memory state as a second argument, so that its behaviour depends on the recent past (Genet & Inzirillo 2024). The memory state of each node evolves as

$$s_p^{(l)}(u) = w_{ss}^{(l)}\, s_p^{(l)}(u-1) + w_{sx}^{(l)}\, x_p^{(l)}(u), \qquad s_p^{(l)}(t-S) = 0, \tag{23}$$

with learnable scalars $w_{ss}^{(l)}, w_{sx}^{(l)} \in \mathbb{R}$ controlling temporal persistence (Genet & Inzirillo 2024). Equations (22) and (23) define a layer operator

$$\mathbf{x}^{(l+1)}(u) = \mathbf{\Psi}^{(l)}\big(\mathbf{x}^{(l)}(u),\, \mathbf{s}^{(l)}(u)\big)\,, \qquad \mathbf{\Psi}^{(l)} : \mathbb{R}^{n_l} \times \mathbb{R}^{n_l} \to \mathbb{R}^{n_{l+1}}, \tag{24}$$

with $\mathbf{s}^{(l)}(u) = \big(s_1^{(l)}(u), \ldots, s_{n_l}^{(l)}(u)\big)$. Following Genet & Inzirillo (2024), these layers are still composed across depth as in (20), but the composition is now time-indexed: at each position u the operators are applied in sequence from $\mathbf{x}^{(0)}(u) = \mathbf{x}_u$, with each memory state $\mathbf{s}^{(l)}(u)$ generated from $\mathbf{x}^{(l)}(u)$ through (23) before (24) is applied, while the memory states are propagated recursively along $u = t - S + 1, \ldots, t$. In the full TKAN each RKAN transformation is embedded in an LSTM-style cell (Section 4.2.4), whose gates regulate how much of the accumulated cell-state memory is carried forward across positions (Genet & Inzirillo 2024). With $n_L = 1$, the forecast is the activation of the single output node at the final sequence position,

$$\hat{y}_t = x_1^{(L)}(t). \tag{25}$$

4.2.3 Recurrent Neural Network

A recurrent neural network maps the input sequence to a forecast through a recursively updated hidden state (Lipton et al. 2015). Let $\mathbf{h}_u \in \mathbb{R}^{d_h}$ denote this hidden state at position u, a vector of d_h units that carries information forward along the sequence, and let $\tanh(\cdot)$ be the activation applied elementwise. The same weights act at every position, and the state recursion is

$$\mathbf{h}_u = \tanh(\mathbf{W}_x \mathbf{x}_u + \mathbf{W}_h \mathbf{h}_{u-1} + \mathbf{b}_h)\,, \qquad u = t - S + 1, \ldots, t, \tag{26}$$

where $\mathbf{W}_x \in \mathbb{R}^{d_h \times d}$ is the input weight matrix mapping the d-dimensional input $\mathbf{x}_u$ into the hidden space, $\mathbf{W}_h \in \mathbb{R}^{d_h \times d_h}$ is the recurrent weight matrix acting on the previous state, and $\mathbf{b}_h \in \mathbb{R}^{d_h}$ is a bias vector; all three are learnable. The state is initialised at $\mathbf{h}_{t-S} = \mathbf{0}$, so each sequence starts from a blank memory. By (26), $\mathbf{h}_u$ depends on $\mathbf{x}_u$ and $\mathbf{h}_{u-1}$, and therefore recursively on $(\mathbf{x}_{t-S+1}, \ldots, \mathbf{x}_u)$. Since a single value is forecast, the state at the final position is mapped to the target by a linear layer,

$$\hat{y}_t = \mathbf{w}^\top \mathbf{h}_t + b, \tag{27}$$

where $\mathbf{w} \in \mathbb{R}^{d_h}$ is a learnable weight vector, $b \in \mathbb{R}$ a learnable scalar bias, and $\hat{y}_t$ the forecast for position t.

4.2.4 Long Short-Term Memory

A Long Short-Term Memory network (Hochreiter & Schmidhuber 1997) is a recurrent architecture designed to mitigate the vanishing-gradient problem through gated additive state updates. Alongside the hidden state $\mathbf{h}_u \in \mathbb{R}^{d_h}$ it maintains a cell state $\mathbf{c}_u \in \mathbb{R}^{d_h}$, which carries information across positions additively. Let $\sigma(\cdot)$ denote the logistic sigmoid. At each position u, given $\mathbf{x}_u$ and $\mathbf{h}_{u-1}$, the LSTM computes an input, forget and output gate and a candidate cell update,

$$\begin{aligned} \mathbf{i}_u &= \sigma(\mathbf{W}_i \mathbf{x}_u + \mathbf{U}_i \mathbf{h}_{u-1} + \mathbf{b}_i), & \mathbf{f}_u &= \sigma(\mathbf{W}_f \mathbf{x}_u + \mathbf{U}_f \mathbf{h}_{u-1} + \mathbf{b}_f), \\ \mathbf{o}_u &= \sigma(\mathbf{W}_o \mathbf{x}_u + \mathbf{U}_o \mathbf{h}_{u-1} + \mathbf{b}_o), & \tilde{\mathbf{c}}_u &= \tanh(\mathbf{W}_c \mathbf{x}_u + \mathbf{U}_c \mathbf{h}_{u-1} + \mathbf{b}_c), \end{aligned} \tag{28}$$

for $u = t - S + 1, \ldots, t$, where the subscripts i, f, o and c label the gate rather than index a sum, $\mathbf{W}_\bullet \in \mathbb{R}^{d_h \times d}$ act on the input, $\mathbf{U}_\bullet \in \mathbb{R}^{d_h \times d_h}$ on the previous hidden state, and $\mathbf{b}_\bullet \in \mathbb{R}^{d_h}$ are bias vectors, all learnable. The two states are updated as

$$\mathbf{c}_u = \mathbf{f}_u \odot \mathbf{c}_{u-1} + \mathbf{i}_u \odot \tilde{\mathbf{c}}_u, \qquad \mathbf{h}_u = \mathbf{o}_u \odot \tanh(\mathbf{c}_u), \qquad \mathbf{c}_{t-S} = \mathbf{h}_{t-S} = \mathbf{0}, \tag{29}$$

where $\odot$ denotes elementwise multiplication. The forget gate $\mathbf{f}_u$ controls the retention and decay of past information, the input gate $\mathbf{i}_u$ determines how new information is incorporated, and the output gate $\mathbf{o}_u$ governs the exposure of the cell state. Because (29) updates $\mathbf{c}_u$ additively, information can be preserved over long horizons, while the hidden state remains bounded by the $\tanh$ transformation. The forecast is obtained by mapping the final hidden state $\mathbf{h}_t$ to the target through a linear layer, $\hat{y}_t = \mathbf{w}^\top \mathbf{h}_t + b$, with learnable $\mathbf{w} \in \mathbb{R}^{d_h}$ and $b \in \mathbb{R}$.

4.2.5 Gated Recurrent Unit

The Gated Recurrent Unit (Chung et al. 2014) simplifies the LSTM by not maintaining a separate cell state: it keeps only a single hidden state $\mathbf{h}_u \in \mathbb{R}^{d_h}$, on which two gates act directly,

$$\mathbf{z}_u = \sigma(\mathbf{W}_z \mathbf{x}_u + \mathbf{U}_z \mathbf{h}_{u-1} + \mathbf{b}_z), \qquad \mathbf{r}_u = \sigma(\mathbf{W}_r \mathbf{x}_u + \mathbf{U}_r \mathbf{h}_{u-1} + \mathbf{b}_r), \tag{30}$$

for $u = t - S + 1, \ldots, t$, where z and r label the update and reset gate (not summation indices), $\sigma(\cdot)$ is the logistic sigmoid, and the learnable parameters $\mathbf{W}_\bullet \in \mathbb{R}^{d_h \times d}$, $\mathbf{U}_\bullet \in \mathbb{R}^{d_h \times d_h}$ and $\mathbf{b}_\bullet \in \mathbb{R}^{d_h}$ act on the input, the previous hidden state and as bias respectively. The reset gate determines how much of the previous state enters the candidate state,

$$\tilde{\mathbf{h}}_u = \tanh(\mathbf{W}_h \mathbf{x}_u + \mathbf{U}_h(\mathbf{r}_u \odot \mathbf{h}_{u-1}) + \mathbf{b}_h)\,, \tag{31}$$

while the update gate controls how much of that candidate replaces the previous state,

$$\mathbf{h}_u = (\mathbf{1} - \mathbf{z}_u) \odot \mathbf{h}_{u-1} + \mathbf{z}_u \odot \tilde{\mathbf{h}}_u, \qquad \mathbf{h}_{t-S} = \mathbf{0}, \tag{32}$$

where $\odot$ is elementwise multiplication and $\mathbf{1}$ a vector of ones. Because the candidate is added to the retained state rather than overwriting it, information can be carried across long horizons, and unlike the LSTM the state is exposed in full at every position. The forecast is obtained by mapping the final hidden state to the target through a linear layer, $\hat{y}_t = \mathbf{w}^\top \mathbf{h}_t + b$, with learnable $\mathbf{w} \in \mathbb{R}^{d_h}$ and $b \in \mathbb{R}$.

4.2.6 Temporal Convolutional Network

A Temporal Convolutional Network (Lea et al. 2016) models sequential data using one-dimensional convolutions along the temporal dimension rather than recurrent state updates. Time dependence is captured by dilated causal convolutions, which ensure that

the representation at a given position depends only on the current and preceding inputs, so that the time order is preserved. Let $\mathbf{h}_u^{(l)} \in \mathbb{R}^{n_l}$ denote the representation at position u in layer l, with $\mathbf{h}_u^{(0)} = \mathbf{x}_u$ and $\mathbf{h}_u^{(l)} = \mathbf{0}$ for $u < t - S + 1$, so that positions preceding the input sequence contribute nothing to the convolution. Each layer computes

$$\mathbf{h}_u^{(l)} = a\left(\sum_{k=0}^{K-1} \mathbf{W}_k^{(l)}\, \mathbf{h}_{u-\delta_l k}^{(l-1)} + \mathbf{b}^{(l)}\right), \qquad l = 1, \ldots, L, \quad u = t - S + 1, \ldots, t, \tag{33}$$

where $\mathbf{W}_k^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ are the filter weights, $\mathbf{b}^{(l)} \in \mathbb{R}^{n_l}$ is a bias vector, K is the filter size, k indexes the taps within the filter and δ_l is the dilation factor of layer l, which sets the spacing between the input elements entering the convolution. The elementwise activation $a(\cdot)$ is essential: without it the composition of (33) across layers would collapse to a single linear filter. Because the sum runs only over indices $u - \delta_l k \leq u$, the representation at u is a function of current and past inputs only, and increasing δ_l with depth widens the span of history covered without enlarging K. The forecast is obtained by mapping the final-layer representation at the last position to the target through a linear layer, $\hat{y}_t = \mathbf{w}^\top \mathbf{h}_t^{(L)} + b$, with learnable $\mathbf{w} \in \mathbb{R}^{n_L}$ and $b \in \mathbb{R}$.

4.2.7 Extreme Gradient Boosting

Extreme Gradient Boosting models the prediction as an additive ensemble of regression trees (Chen & Guestrin 2016). Using the estimation window $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, the prediction after r boosting rounds is

$$\hat{y}_i^{(r)} = \sum_{k=1}^{r} f_k(\mathbf{x}_i), \qquad f_k \in \mathcal{F}, \tag{34}$$

where $\mathcal{F}$ is the space of regression trees and f_k assigns a score to each of its terminal leaves. The final prediction is $\hat{y}_i = \hat{y}_i^{(R)}$, with R the total number of trees. With the squared-error loss, the training objective is

$$\mathcal{J}_{\mathrm{XGB}} = \frac{1}{2}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \sum_{k=1}^{R}\left(\gamma J_k + \frac{1}{2}\lambda \sum_{j=1}^{J_k} w_{j,k}^2\right), \tag{35}$$

where J_k is the number of leaves of tree k, $w_{j,k}$ the prediction assigned to leaf j of tree k, and λ and γ penalize large leaf weights and tree complexity respectively. Trees are added one at a time, each fitted to a second-order Taylor expansion of the loss around the current prediction. Within this subsection let g_i and h_i denote the first- and second-order

derivatives of the loss with respect to $\hat{y}_i^{(r-1)}$, which for the squared-error loss in (35) are

$$g_i = \hat{y}_i^{(r-1)} - y_i, \qquad h_i = 1. \tag{36}$$

For a fixed tree structure, the optimal prediction in leaf j is

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}, \tag{37}$$

with I_j the set of observations falling into that leaf. Splits are selected by the implied reduction in the objective: for a candidate split of a leaf into children I_L and I_R,

$$\text{Gain} = \frac{1}{2}\left(\frac{\left(\sum_{i \in I_L} g_i\right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I_L \cup I_R} g_i\right)^2}{\sum_{i \in I_L \cup I_R} h_i + \lambda}\right) - \gamma, \tag{38}$$

and the split is retained only if this quantity is positive (Chen & Guestrin 2016).

4.2.8 Light Gradient Boosting Machine

LightGBM (Ke et al. 2017) is a gradient boosting algorithm designed to improve computational efficiency on large and high-dimensional datasets. Predictions are an additive ensemble of trees as in (34), fitted sequentially to the negative gradients $-g_i$ from (36) using the same second-order approximation and split criterion (38). The differences lie in tree construction and data handling. LightGBM grows trees leaf-wise, splitting at each step the leaf with the largest gain, which produces deeper and asymmetric trees and is therefore usually combined with a cap on the number of leaves J_k. For efficiency it discretizes continuous features into histogram bins and accumulates gradient statistics at the bin level, and it applies Gradient-Based One-Side Sampling, which retains observations with large $|g_i|$ while subsampling those with small gradients, and Exclusive Feature Bundling, which reduces the feature dimension by combining mutually exclusive sparse features (Ke et al. 2017).

4.2.9 Tabular Prior-data Fitted Network

TabPFN[4] is a Prior-Data Fitted Network, abbreviated here as PFN. A prior defines a space $\mathcal{H}$ of data-generating mechanisms relating an input $\mathbf{x}$ to an output y; a mechanism $\eta \in \mathcal{H}$ has prior probability $p(\eta)$ and induces a distribution $p(D \mid \eta)$ over datasets. Given observed pairs $D_{\text{train}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)\}$, the posterior predictive distribution for

[4]In this paper, we use the TabPFN-3 model released by Prior Labs.

a query $\mathbf{x}$ averages over mechanisms, weighting each by how well it explains the data,

$$p(y \mid \mathbf{x}, D_{\text{train}}) \;\propto\; \int_{\mathcal{H}} p(y \mid \mathbf{x}, \eta)\, p(D_{\text{train}} \mid \eta)\, p(\eta)\, d\eta. \tag{39}$$

The integral is intractable, so a PFN replaces it by a transformer q_θ fitted once, in advance, to synthetic datasets from the prior: a mechanism is drawn $\eta \sim p(\eta)$, a dataset $D \sim p(D \mid \eta)$ is split into D_{train} and a held-out pair $(\mathbf{x}_{\text{test}}, y_{\text{test}})$, and θ minimizes

$$\mathcal{J}_{\text{PFN}} = \mathbb{E}[-\log q_\theta(y_{\text{test}} \mid \mathbf{x}_{\text{test}}, D_{\text{train}})]\,, \tag{40}$$

the expectation taken over these draws. Minimizing (40) makes q_θ approximate (39) (Hollmann et al. 2022), so no mechanism is ever identified: the network maps observed pairs directly to the predictive distribution Bayesian inference would return. Consequently θ is fixed before the empirical data are seen, and a forecast is a single forward pass over D_{train} and the query point, with no estimation, tuning or model selection at the forecast origin. All structure comes from the prior $p(\eta)$, which in TabPFN mixes structural causal models and Bayesian neural networks drawn with a preference for few nodes and parameters (Hollmann et al. 2022). That prior describes dependence between the columns of a table and encodes nothing about time; moreover D_{train} is passed as a permutation-invariant set, so row order carries no information. Any dependence on the past must therefore enter through the features.

To apply TabPFN to realized-variance forecasting, we follow the TabPFN-TS (Hoo et al. 2025) idea of representing a time series as a table with one row per time step. Because each forecast in our setting is made at origin t using information available at t, we can include lagged realized-variance terms directly as features. For each horizon h, we construct a small conditioning set consisting of the most recent M observations,

$$D_t^{(h)} = \left\{ \left(\mathbf{x}_u,\, y_u^{(h)}\right) \right\}_{u=t-M}^{t-1},$$

where $\mathbf{x}_u$ contains the daily, weekly, and monthly realized-variance averages, and, for the HAR-X specification, the exogenous covariates. TabPFN then performs in-context learning on this set and produces a predictive distribution for the next value. Our forecast is the conditional mean,

$$\hat{y}_t^{(h)} = \mathbb{E}_{q_\theta}\left[\, y \mid \mathbf{x}_t,\, D_t^{(h)} \,\right],$$

so that TabPFN conditions on exactly the same information as the HAR and HAR-

X models, differing only in the functional form used to map past observations to future values.

4.3 Accuracy quantification

4.3.1 Loss functions

Volatility forecast accuracy evaluation is complicated by the fact that the conditional variance is latent and must be proxied. Let r_t denote an asset return with conditional mean zero and conditional variance as in 41:

$$\sigma_t^2 \equiv \mathbb{E}\left[r_t^2 \mid \mathcal{F}_{t-1}\right]. \tag{41}$$

Following Patton (2011), realized variance is treated as a conditionally unbiased (but noisy) proxy for the latent conditional variance,

$$RV_t = \sigma_t^2 + \eta_t, \qquad \mathbb{E}[\eta_t \mid \mathcal{F}_{t-1}] = 0, \tag{42}$$

where the noise term arises from discrete intraday sampling. Under the diffusion framework described in Definition 3.1, realized variance satisfies:

$$\mathbb{E}[RV_t \mid \mathcal{F}_{t-1}] = \sigma_t^2, \qquad \mathrm{MSE}_{t-1}(RV_t) = \mathbb{E}\left[(RV_t - \sigma_t^2)^2 \mid \mathcal{F}_{t-1}\right] = \frac{2\sigma_t^4}{n}, \tag{43}$$

Thus, increasing the intraday sampling frequency improves the precision of the volatility proxy at rate n^{-1}. In the empirical analysis, realized variance is constructed using 5-minute returns, corresponding to $n = 78$. Because forecast evaluation minimizes expected loss using realized variance rather than the latent conditional variance, the choice of loss function is crucial. As shown by Patton (2011), a loss function is appropriate for volatility forecast comparison if its minimizer remains the true conditional variance despite measurement error in the proxy. Let $\widehat{RV}_{t|t-1}$ denote a volatility forecast formed at time $t-1$ for day t. An appropriate loss function must satisfy

$$\widehat{RV}_{t|t-1} = \sigma_t^2, \tag{44}$$

which ensures that forecast rankings are not distorted by noise in realized variance. Both the mean squared error and the quasi-likelihood loss satisfy this condition and therefore yield consistent forecast comparisons even at finite intraday sampling frequencies. In

contrast, alternative loss functions such as the mean absolute error generally imply

$$\widehat{RV}_{t|t-1} = \sigma_t^2 + \mathcal{O}(n^{-1}), \tag{45}$$

so that exact consistency is achieved only asymptotically as the sampling frequency increases. Restricting attention to MSE and QLIKE therefore ensures that volatility forecast evaluation in this study reflects genuine differences in predictive accuracy rather than artefacts of volatility measurement error. Accordingly, forecast accuracy is evaluated using the mean squared error and the quasi-likelihood loss. The MSE is defined as

$$\mathcal{L}^{MSE} = \left(RV_{t+h} - \hat{RV}_{t+h|t}\right)^2. \tag{46}$$

The QLIKE loss, derived from the Gaussian quasi-likelihood, is defined as

$$\mathcal{L}^{QLIKE} = \frac{RV_{t+h}}{\hat{RV}_{t+h|t}} - \log\left(\frac{RV_{t+h}}{\hat{RV}_{t+h|t}}\right). \tag{47}$$

4.3.2 Model Confidence Set

To compare a potentially large collection of competing forecasting models while accounting for statistical uncertainty, we employ the Model Confidence Set procedure of Hansen et al. (2011). The framework aims to identify the subset of models that cannot be statistically distinguished from the best performing model(s) in terms of expected loss at a given confidence level α. Let $\mathcal{M}_0 = \{1, \ldots, m_0\}$ denote the initial set of models. For each model $i \in \mathcal{M}_0$, forecast performance is summarized by a sequence of losses $\mathcal{L}_{i,t} = \mathcal{L}(y_t, \hat{y}_{i,t})$. Relative performance between models i and j is measured by the loss differential:

$$d_{ij,t} = \mathcal{L}_{i,t} - \mathcal{L}_{j,t}, \tag{48}$$

with finite and time-invariant expectation $\mu_{ij} = \mathbb{E}(d_{ij,t})$. Models are ranked according to expected loss, and the set of superior models is defined as

$$\mathcal{M}^* = \left\{i \in \mathcal{M}_0 : \mu_{ij} \leq 0 \ \text{ for all } j \in \mathcal{M}_0\right\}. \tag{49}$$

The MCS procedure constructs a data-dependent confidence set $\widehat{\mathcal{M}}^*_{1-\alpha}$ for $\mathcal{M}^*$ via a sequential testing algorithm. Starting from $\mathcal{M}_0$, the null hypothesis of equal predictive ability

$$H_0^{\mathcal{M}} : \mu_{ij} = 0 \quad \text{for all } i, j \in \mathcal{M} \tag{50}$$

is tested for a candidate subset $\mathcal{M} \subseteq \mathcal{M}_0$. Inference is based on a bootstrap-based range statistic (Hansen et al. 2003) in 51:

$$T_R = \max_{i,j \in \mathcal{M}} \frac{|\bar{d}_{ij}|}{\sqrt{\widehat{\mathrm{Var}}(\bar{d}_{ij})}}, \tag{51}$$

where $\bar{d}_{ij} = T^{-1} \sum_{t=1}^{T} d_{ij,t}$ and $\widehat{\mathrm{Var}}(\bar{d}_{ij})$ is estimated using a circular moving-block bootstrap to account for potential dependence in the loss differentials. The block length is set data-dependently to the largest BIC-selected autoregressive order across all loss-differential series, capped at 20 lags, so that the blocks preserve the serial dependence of the most persistent pair. [5] If the EPA null cannot be rejected, the procedure terminates and the remaining models are retained. If the null hypothesis is rejected, model elimination is based on relative average performance. Define

$$\bar{d}_{i\cdot} = \frac{1}{|\mathcal{M}| - 1} \sum_{j \in \mathcal{M},\, j \neq i} \bar{d}_{ij}, \tag{52}$$

the model with the largest $\bar{d}_{i\cdot}$ is eliminated according to $e_M = \arg\max_{i \in \mathcal{M}} \bar{d}_i$ (Zammarchi & Kawasaki 2026) and the equal predictive ability hypothesis is re-tested on the reduced model set. This process is repeated until the null can no longer be rejected, yielding the final Model Confidence Set.

4.3.3 Diebold-Mariano test

To assess whether model i and model j differ in predictive accuracy, we employ the Diebold & Mariano (1995) test based on squared-error loss. For each forecast origin $t \in \mathcal{T}_h$, define the loss differential

$$d_{ij,t} = \mathcal{L}_{i,t} - \mathcal{L}_{j,t}, \tag{53}$$

where $\mathcal{T}_h$ denotes the set of all time points at which h-step-ahead forecasts are evaluated. The sample mean of the loss differentials is

$$\bar{d} = \frac{1}{N} \sum_{t \in \mathcal{T}_h} d_{ij,t}, \tag{54}$$

where N is the number of out-of-sample forecasts. Because h-step-ahead forecast errors overlap, the sequence $\{d_{ij,t}\}$ is serially correlated up to lag $h - 1$. A heteroskedasticity-

[5] We assume that $\mathbb{E}[|d_{ij,t}|] < \infty$ and that $\{d_{ij,t}\}$ is stationary and ergodic.

and autocorrelation-consistent estimator of the long-run variance of the loss differentials is therefore required. Following Diebold & Mariano (1995), we estimate

$$\widehat{\mathrm{Var}}(d) = \hat{\gamma}_d(0) + 2\sum_{l=1}^{h-1}\left(1 - \frac{l}{h}\right)\hat{\gamma}_d(l), \tag{55}$$

where $\hat{\gamma}_d(l)$ denotes the sample autocovariance of $\{d_{ij,t}\}$ at lag l. The weights $1 - l/h$ correspond to the Bartlett kernel, whose bandwidth we set equal to the forecast horizon, so that the truncation lag is $h - 1$ (0, 4 and 21 lags for $h = 1$, 5 and 22 respectively) and the variance estimate is guaranteed non-negative. The Diebold–Mariano test statistic is then defined as

$$DM = \frac{\bar{d}}{\sqrt{\widehat{\mathrm{Var}}(d)/N}}, \tag{56}$$

and is asymptotically standard normal under the null hypothesis of equal predictive accuracy, $\mathbb{E}[d_{ij,t}] = 0$.

4.3.4 Clark-West test

When comparing nested models (HAR nested within LevHAR, HARQ, or SHAR), the classical Diebold-Mariano test tends to favour the more parsimonious model in finite samples. To correct for this upward bias, we apply the Clark & West (2007) test. In what follows, model i denotes the nested HAR specification, while model j represents an enlarged model that nests HAR. For each forecast origin $t \in \mathcal{T}_h$, where $\mathcal{T}_h$ denotes the set of time points at which h-step-ahead forecasts are evaluated, the Clark-West adjusted loss differential is defined as

$$f_{ij,t} = \mathcal{L}_{i,t} - \mathcal{L}_{j,t} + \left(\widehat{RV}^{(i)}_{t+h|t} - \widehat{RV}^{(j)}_{t+h|t}\right)^2. \tag{57}$$

The sample mean of the adjusted loss differentials is

$$\bar{f} = \frac{1}{N}\sum_{t\in\mathcal{T}_h} f_{ij,t}, \tag{58}$$

where N denotes the number of out-of-sample forecasts. Because h-step-ahead forecast errors overlap, the sequence $\{f_{ij,t}\}$ is serially correlated up to lag $h - 1$. Analogously to the Diebold–Mariano test, we therefore employ a heteroskedasticity- and autocorrelation-

consistent estimator of the long-run variance,

$$\widehat{\mathrm{Var}}(f) = \hat{\gamma}_f(0) + 2\sum_{l=1}^{h-1}\left(1 - \frac{l}{h}\right)\hat{\gamma}_f(l), \tag{59}$$

where $\hat{\gamma}_f(l)$ is the sample autocovariance of $\{f_{ij,t}\}$ at lag l, and the Bartlett weights and horizon-dependent truncation are the same as in (55).The Clark-West test statistic is then defined as

$$CW = \frac{\bar{f}}{\sqrt{\widehat{\mathrm{Var}}(f)/N}}, \tag{60}$$

and is asymptotically standard normal under the null hypothesis that the larger model provides no improvement in predictive accuracy.

4.4 Uncertainty quantification

Conformal prediction provides a distribution-free framework for constructing prediction sets with finite-sample validity. (Shafer & Vovk 2008) The key idea is to assign conformity scores that measure how well observations agree with the data and to construct prediction sets by comparing new samples to these scores. Under exchangeability, the resulting prediction interval $C_\alpha(X_{n+1})$ satisfies 61, independently of the underlying data distribution.

$$\mathbb{P}\big(Y_{n+1} \in C_\alpha(X_{n+1})\big) \geq 1 - \alpha \tag{61}$$

In time series settings, exchangeability is violated, motivating extensions such as Ensemble Batch Prediction Intervals (EnbPI) as proposed by Xu & Xie (2021). Consider the model

$$y_t = f(x_t) + \varepsilon_t, \tag{62}$$

where the error process $\{\varepsilon_t\}$ may exhibit temporal dependence. EnbPI constructs residual-based conformity scores in a manner analogous to conformal prediction using leave-one-out (LOO) ensemble predictors. Given bootstrap estimators $\{\hat{f}_b\}_{b=1}^{B}$, the LOO predictor is

$$\hat{f}_{-i}(x_i) = \mathcal{G}\big(\{\hat{f}_b(x_i) : i \notin S_b\}\big), \tag{63}$$

where S_b denotes the set of indices sampled (with replacement) to form the b-th bootstrap dataset used to fit $\hat{f}_b$. The corresponding residuals $\hat{\varepsilon}_i = |y_i - \hat{f}_{-i}(x_i)|$ serve as conformity scores. Prediction intervals at time t are constructed as

$$C_t^\alpha = \Big[\, \hat{f}_{-t}(x_t) \pm q_{1-\alpha}\big(\{\hat{\varepsilon}_i\}_{i=t-T}^{t-1}\big) \,\Big], \tag{64}$$

where $q_{1-\alpha}(\cdot)$ denotes the empirical $(1-\alpha)$-quantile of the most recent residuals. Hence, intervals are centered at the LOO prediction and their width is determined by past residuals, updated in a rolling window to adapt to temporal changes. EnbPI does not guarantee exact finite-sample marginal coverage. Instead, under mild assumptions on the error process and the approximation quality of the predictors, it achieves approximately valid marginal coverage, as is displayed in 65 (Xu & Xie 2021):

$$.\mathbb{P}(Y_t \in C_t^{\alpha}) \geq 1 - \alpha - e_T, \quad \text{with } e_T \to 0. \tag{65}$$

4.5 Quantile-based forecast evaluation

Point forecasts summarize the conditional mean of realized variance but provide no information about forecast accuracy in different regions of the conditional distribution. Since practitioners are particularly concerned with increased uncertainty in their risk management, we complement point forecast evaluation with quantile-based methods that explicitly assess predictive performance in the right tail of the distribution.

Quantile forecasts are obtained by minimizing the loss as shown in 66 (Koenker & Bassett Jr 1978), which penalizes under-prediction more heavily than over-prediction for $\tau > 0.5$.

$$\rho_\tau = \big(RV_{t+h} - \widehat{Q}_{t+h|t}(\tau)\big)\big(\tau - \mathbf{1}_{\{RV_{t+h} < \widehat{Q}_{t+h|t}(\tau)\}}\big) \tag{66}$$

Quantile forecast accuracy is evaluated using the quantile score in 67, as introduced by Gneiting & Raftery (2007),

$$QS_{t+h|t}(\tau) = \big(RV_{t+h} - \widehat{Q}_{t+h|t}(\tau)\big)\big(\tau - \mathbf{1}_{\{RV_{t+h} < \widehat{Q}_{t+h|t}(\tau)\}}\big), \tag{67}$$

where lower values indicate better predictive accuracy and the score provides a local evaluation at a fixed quantile level. To test equal quantile forecast accuracy relative to a benchmark model between model i and j, we employ the following test statistic, as mentioned by Meligkotsidou et al. (2019):

$$t_{i,j} = \frac{\overline{QS}_i(\tau) - \overline{QS}_j(\tau)}{\sqrt{\widehat{\mathrm{Var}}(QS_i(\tau) - QS_j(\tau))\,/N}}, \tag{68}$$

where $\overline{QS}_i(\tau)$ denotes the average quantile score at level τ, and $\widehat{\mathrm{Var}}(QS_i(\tau) - QS_j(\tau))$ is a HAC-consistent estimator of the long-run variance of the quantile score differential. Under the null of equal expected quantile accuracy, $t(\tau)$ is asymptotically standard normal. To augment our single quantile assessment with insight into forecasting performance

over a specific region of the distribution, we also compute the weighted quantile score as in 69:

$$WQS_{t+h|t} = \int_0^1 QS_{t+h|t}(\tau)\,\omega(\tau)\,d\tau, \tag{69}$$

which aggregates quantile scores using a weighting function $\omega(\tau)$ that corresponds to a specific region of interest of the distribution. As said, we focus on the right tail of the distribution and therefore set $\omega(\tau) = \tau^2$, as suggested by Meligkotsidou et al. (2019). The differences in WQS are evaluated by using WQS in place of QS in 68.

5 EMPIRICAL RESULTS

In reporting our results, we make a distinction between the results generated using the HAR dataset $\mathcal{D}_{\text{HAR}}$ and results generated using the HAR-X dataset $\mathcal{D}_{\text{HAR-X}}$, to see if certain models perform better or worse when additional predictors are included. Additionally, we want to assess forecasting power for different time horizons, namely one day-ahead, average-week ahead and average-month ahead. We do so by reporting Model Confidence Sets and Diebold-Mariano test results; the two are complementary. Because our model set implies a large number of pairwise comparisons, we rely on the MCS algorithm to limit the risk of us drawing flawed conclusions from repeated pairwise testing; it delivers the subset of models that cannot be rejected as inferior to the best-performing model(s) at confidence level $1 - \alpha$. The Diebold-Mariano test is then used to gain further insight into the relative performance of the reduced number of models retained in the MCS. We evaluate forecast performance using both We evaluate forecast performance using both the mean squared error and the quasi-likelihood loss function. MSE provides a symmetric measure of absolute forecast accuracy, while QLIKE is an asymmetric loss that penalizes volatility underestimation more strongly. This asymmetry is particularly relevant for practitioners, as underestimating volatility is typically more costly than overestimation for risk management. Reporting results under both loss functions therefore allows us to assess accuracy from an absolute level perspective as well as from a risk-sensitive perspective that places greater emphasis on under prediction. Next to accuracy, uncertainty will be evaluated by reporting prediction intervals.

As part of the forecasting procedure, the data are first split into a training set containing the initial 80% of the sample and a test set containing the remaining 20%. For both the econometric models and the machine learning models, forecasts are generated using a rolling window approach with a fixed window length of $W = 3{,}622$ observations, so that each forecast origin re-estimates the model on the most recent W observations, discarding the oldest observation as one new observation enters the window. After examining the stationarity[6] of the variables, we find that, at the 5% significance level, the 3-month Treasury Bill rate is non-stationary. To address this issue the variable is transformed by taking its first difference before being included in the HAR-type models. For the machine learning models, all input variables are standardized before training; while such scaling is not required for tree-based algorithms, it is nevertheless applied as it does not impair their performance. For the HAR-type models, when exogenous variables are included, all input variables are standardized prior to model estimation.

[6]The results of the Augmented Dickey-Fuller tests are provided in Appendix A.1.

For the machine learning models, we use rolling window cross-validation on the training set to tune hyperparameters[7]. Hyperparameter tuning is performed separately for each forecasting horizon, to account for the fact that different horizons may benefit from different model architectures and degrees of regularization.

Finally, it is important to clarify the interpretation of the forecasting horizons. The one-day-ahead forecast is denoted by RV_{t+1}. The week-ahead forecast refers to the average realized variance over the subsequent trading week and is denoted by $RV_{t+5|t+1}$. Similarly, the month-ahead forecast corresponds to the average realized variance over the following trading month and is denoted by $RV_{t+22|t+1}$. Here, $RV_{t+h|t+1} \equiv \frac{1}{h}\sum_{i=1}^{h} RV_{t+i}$.

5.1 Model Confidence Set results

In our application, we use (part of) the procedure as described by Hansen et al. (2003), specifically as how it is outlined in 'Appendix B' in their paper. One can interpret the results as follows: the MCS is a set-valued confidence region that contains the best forecasting model with probability at least $1 - \alpha$. Models included in the MCS cannot be statistically rejected as inferior given the data, but inclusion does not imply equal performance or dominance. (Hansen et al. 2003) We report results for $\mathcal{L}^{QLIKE}$ and $\mathcal{L}^{MSE}$, which are displayed in table 2 and table 3, respectively.

Table 2: Models included in the Model Confidence Set across estimation sets and forecast horizons, $\alpha = 0.05$, $B = 10{,}000$, loss function: $\mathcal{L}^{QLIKE}$

Estimated on $\mathcal{D}_{HAR}$			**Estimated on $\mathcal{D}_{HAR\text{-}X}$**		
RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$	RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$
HAR	TabPFN	TabPFN	Log-HARX	TabPFNX	TabPFNX
Log-HAR			TKANX		
HAR-Q			LGBMX		
SHAR			XGBX		
LevHAR			RNNX		
TKAN					
LGBM					
XGB					
RNN					
TabPFN					

For the day-ahead, we observe that all the HAR models cannot be ruled out as statistically inferior when models are estimated on $\mathcal{D}_{HAR}$, under both loss functions. Under $\mathcal{L}^{MSE}$, only the LSTM gets ruled out. Under $\mathcal{L}^{QLIKE}$, the MCS eliminates the KAN, LSTM, GRU and TCN. When we visually inspecting the forecasts in figures 1 and 2, it can be observed that none of the models is really able to capture the magnitude of the

[7]The hyperparameter tuning procedure is available in Appendix A.4.

Table 3: Models included in the Model Confidence Set across estimation sets and forecast horizons, $\alpha = 0.05$, $B = 10,000$, loss function: $\mathcal{L}^{MSE}$

Estimated on $\mathcal{D}_{HAR}$			**Estimated on $\mathcal{D}_{HAR\text{-}X}$**		
RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$	RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$
HAR	TabPFN	TabPFN	Log-HARX	TabPFNX	TabPFNX
Log-HAR			TKANX		
HAR-Q			LGBMX		
SHAR			RNNX		
LevHAR			GRUX		
KAN			TabPFNX		
TKAN					
LGBM					
XGB					
RNN					
GRU					
TCN					
TabPFN					

largest variance spikes. Also, quite a lot of models forecast with some lagged response, so they forecast a spike (although of too small magnitude), when the actual surge in volatility already happened.

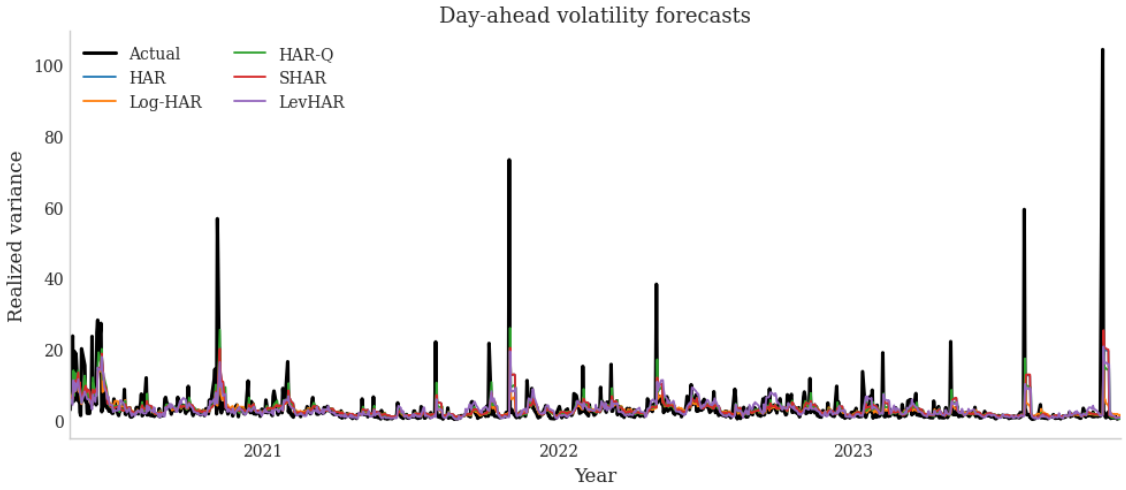


Figure 1: Forecasts of HAR-type models estimated on $\mathcal{D}_{HAR}$

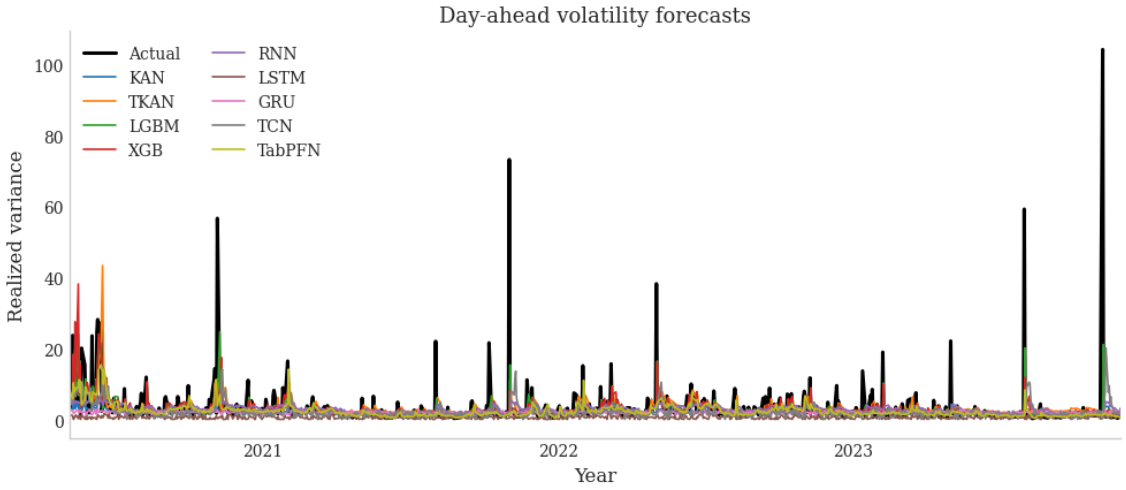


Figure 2: Forecasts of machine learning models estimated on $\mathcal{D}_{HAR}$

Next, when models are estimated on $\mathcal{D}_{HAR-X}$, the only HAR-type model retained in the MCS for the day-ahead horizon under both loss functions is Log-HARX. Moreover, substantially fewer machine learning models are included compared to the case where models are estimated on $\mathcal{D}HAR$, suggesting that the excluded HAR and ML models may not be sufficiently capable of handling the larger predictor set. When we consider the visualizations of the forecasts of models estimated on $\mathcal{D}_{HAR-X}$ in figures 3 and 4, again it is observed neither the HAR nor the machine learning models are able to capture the magnitude of the largest realized variance spikes. This suggests that incorporating additional macroeconomic variables does not substantially improve the models' forecasting accuracy.

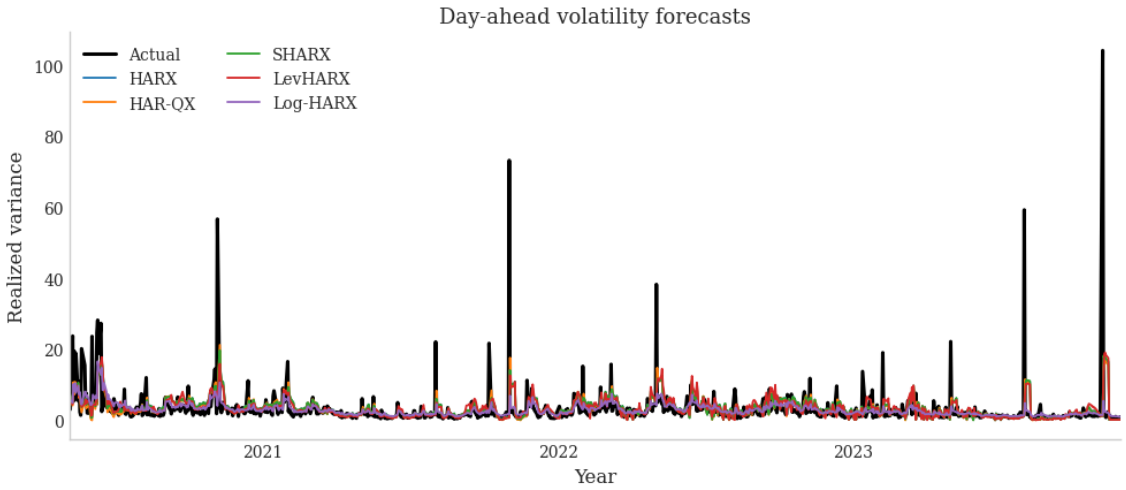


Figure 3: Forecasts of HAR-type models estimated on $\mathcal{D}_{HAR-X}$

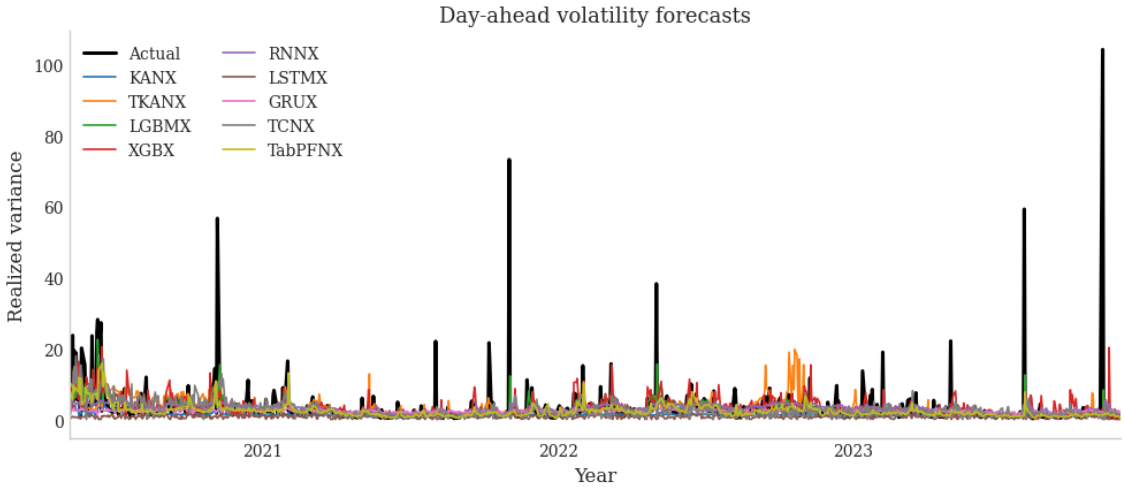


Figure 4: Forecasts of machine learning models estimated on $\mathcal{D}_{HAR-X}$

The results for the week-ahead and month-ahead forecasting horizons strongly favor TabPFN. As shown in Tables 2 and 3, TabPFN is the only model that remains in the Model Confidence Set across both loss functions and both information sets. This indicates that all competing models are statistically rejected as inferior at the 5% significance level, while TabPFN cannot be rejected as belonging to the set of best-performing models. These findings provide strong evidence that TabPFN achieves highly competitive predictive performance relative to all other models at medium and longer forecasting horizons. This conclusion is further supported by the full pairwise Diebold–Mariano test results reported in Appendix A.6. In particular, for models estimated on $\mathcal{D}_{HAR}$, TabPFN consistently exhibits superior predictive ability in pairwise comparisons against every alternative model, under both loss functions and for both the average week-ahead and average month-ahead horizons. The superiority of TabPFN is especially conveniently observable when considering the mean squared error, where it delivers substantially more accurate forecasts than all competing models. The results for the models estimated on $\mathcal{D}_{HAR}$ are largely consistent with those obtained for $\mathcal{D}_{HAR-X}$, with one exception: the comparison between SHARX and TabPFNX is statistically insignificant. A visual inspection suggests that this is driven by instability in the SHARX forecasts. In particular, SHARX exhibits severe overprediction toward the end of the test sample, leading to large spikes in the QLIKE loss. This inflates the variance of the loss differentials, reducing the power of the Diebold–Mariano test and resulting in insignificance. Despite this, the relative mean squared error remains low, indicating that the lack of significance is primarily due to irregular behavior of the SHARX forecasts rather than comparable predictive performance.

Increasing the significance level from $\alpha = 0.05$ to $\alpha = 0.10$ makes the Model Confidence Set procedure less conservative and thus more aggressive in eliminating models;

the corresponding results are reported in Appendix A.5. Notably, the LevHAR model is then eliminated under $\mathcal{L}^{QLIKE}$, which, together with Figure 5, suggests that it may underpredict relative to other models, as the quasi-likelihood loss penalizes this strongly; an effect that becomes statistically significant at the 10% level. In addition, TabPFN is also eliminated when α increases, while under $\mathcal{L}^{MSE}$, TCN is further excluded compared to the 5% MCS. For models estimated on $\mathcal{D}_{HAR-X}$, Log-HARX is the only one that survives under $\mathcal{L}^{MSE}$, indicating that a simple log transform of realized variance in combination with its parsimonious structure dominates other more complicated linear and machine learning models, as well as the simpler HAR model. The conclusions for the average week-ahead and month-ahead horizons remain unchanged.

5.2 Pairwise Diebold-Mariano test results

While the Model Confidence Set identifies models that cannot be jointly rejected as inferior, it does not imply equality of predictive accuracy for the models in the MCS. To further characterize relative performance within the confidence set, we report pairwise Diebold–Mariano (and Clark–West) tests. [8] We only display the loss matrices for the day-ahead forecasts here, as the average week-ahead and average month-ahead sets only contain TabPFN. In figures 5 and 6, the results are displayed. Each cell reports the relative loss between two models, defined as in 70:

$$\frac{\text{MSE}_j}{\text{MSE}_i} \quad or \quad \frac{\text{QLIKE}_j}{\text{QLIKE}_i}, \tag{70}$$

where i corresponds to row the model in row i and j corresponds to the model in column j. Alongside the relative losses, statistical significance is reported for 10%, 5%, and 1% significance levels. The DM test evaluates

$$H_0 : \mathbb{E}[d_{ij,t}] = 0 \quad \text{vs.} \quad H_1 : \mathbb{E}[d_{ij,t}] \neq 0, \tag{71}$$

where a relative loss smaller (larger) than 1, when accompanied by statistically significant test results, indicates that the model listed in column j has superior (inferior) forecasting performance compared to the model listed in row i, and we indicate this by colouring the cell green (red).

Because several HAR–type models are nested versions of one another, we also report the Clark–West test alongside the Diebold–Mariano test. In nested model settings, the

[8] Full pairwise relative MSE and test results (for both predictor spaces and loss functions) for all evaluated models are provided in appendix A.6.

standard Diebold–Mariano test is biased toward the smaller model, while the Clark–West corrects for this bias. The one sided Clark-West test compares the forecasting accuracy of the HAR(-X) to the HARQ(-X), SHAR(-X) and LevHAR(-X), where the following hypothesis are formulated:

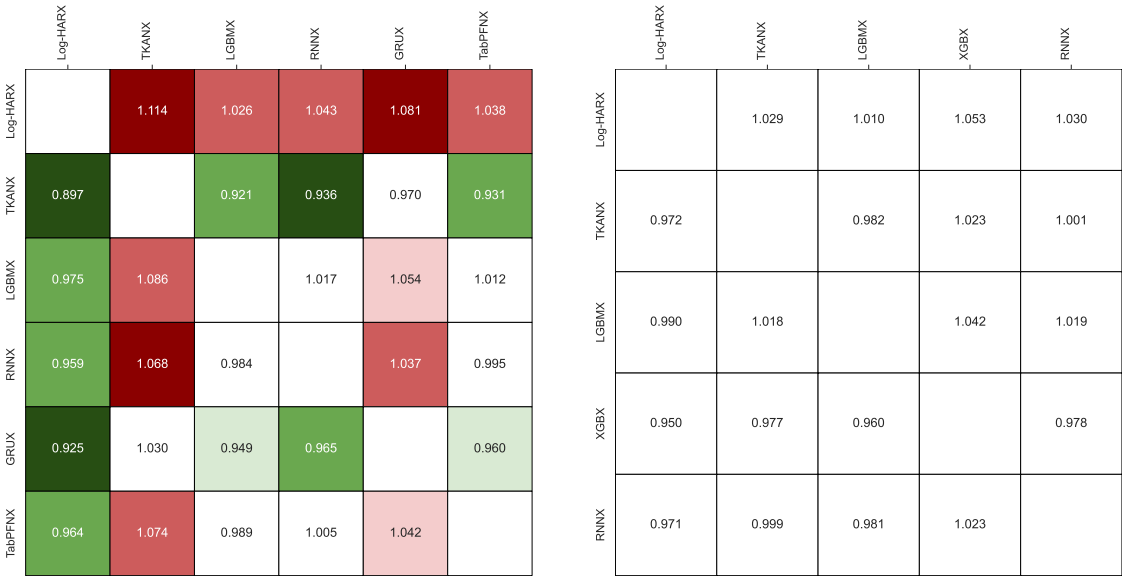
$$H_0 : \mathbb{E}[f_{ij,t}] \leq 0 \quad \text{vs.} \quad H_1 : \mathbb{E}[f_{ij,t}] > 0. \tag{72}$$

Rejection of the null hypothesis in favor of the alternative indicates that the enlarged model delivers a statistically significant improvement in forecasting accuracy relative to the nested HAR benchmark, after correcting for estimation bias arising from model nesting. When interpreting figures 5 and 6, it is important to note that, for nested model comparisons, significance is assigned as follows: the Clark–West test result is reported first, and if this test is insignificant, the Diebold–Mariano test result is used instead. Also note that the models not mentioned in any of the figures are already considered inferior to the models that are mentioned under that specific loss function and predictor set.

When focusing on the models estimated on $\mathcal{D}_{HAR}$ in Figure 5, Log-HAR and TabPFN emerge as the strongest performers under $\mathcal{L}^{MSE}$, outperforming most alternatives in pairwise tests. This suggests that, under a symmetric loss function, these models most frequently deliver the most accurate point forecasts in direct comparisons. In contrast, under $\mathcal{L}^{QLIKE}$, XGB dominates the majority of pairwise comparisons, followed by Log-HAR, indicating they performs better most often when underprediction is penalized more heavily and are therefore particularly effective when avoiding underestimation is crucial. The models with the most inferior pairwise performances under $\mathcal{L}^{MSE}$ seem to be HAR, followed by HAR-Q, GRU and TCN, supporting the argument that even when one only wants to consider HAR-type models, it is still worthwhile to consider alternative specifications from the standard HAR for short horizon forecasting. Under $\mathcal{L}^{QLIKE}$, LevHAR appears to perform worst in most pairwise comparisons among the models in the MCS, followed by TabPFN.

(a) Loss function: $\mathcal{L}^{MSE}$ (b) Loss function: $\mathcal{L}^{QLIKE}$

Figure 5: Relative loss matrices of models estimated on $\mathcal{D}_{HAR}$ and retained in MCS at the day-ahead horizon; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).



(a) Loss function: $\mathcal{L}^{MSE}$ (b) Loss function: $\mathcal{L}^{QLIKE}$

Figure 6: Relative loss matrices of models estimated on $\mathcal{D}_{HAR-X}$ and retained in MCS at the day-ahead horizon; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).

Next, when considering the models retained in the MCS when estimated on the extended predictor set in figure 6, it can be observed that Log-HARX is statistically superior to all models retained in the MCS in the respective pairwise comparisons under $\mathcal{L}^{MSE}$, which is noteworthy given its relatively simple specification compared to extensively hyperparameter-tuned machine learning models. Under $\mathcal{L}^{QLIKE}$, all pairwise comparisons are insignificant.

5.3 Prediction Intervals

In this implementation, we use what is denoted as 'Algorithm 1' in Xu & Xie (2021). In our application, we use the mean as the aggregation function $\mathcal{G}$ and we use online updating by setting $s = 1$. Prediction intervals are generated using Ensemble Batch Prediction Intervals for all models except TabPFN. This is because TabPFN directly outputs an approximation of the posterior predictive distribution, from which prediction intervals can be obtained by extracting the corresponding quantiles.

To assess each model's ability to quantify uncertainty, we consider two metrics. Equation 73 defines the empirical coverage (EC), which measures the proportion of observations for which the true value y_t lies within the prediction interval $[L_t, U_t]$:

$$\text{EC} = \frac{1}{N} \sum_{t=1}^{N} \mathbf{1}\{L_t \leq y_t \leq U_t\} \tag{73}$$

In addition, we evaluate the sharpness of the prediction intervals using the median relative interval width, defined as in 74:

$$\text{MRIW} = \underset{t=1,\ldots,N}{\text{median}} \left(\frac{U_t - L_t}{y_t} \right) \tag{74}$$

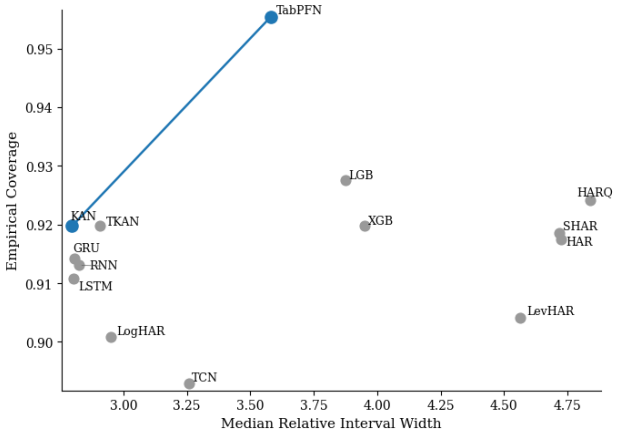
which captures the median width of the prediction intervals relative to the actual values.

Although EnbPI does not guarantee exact marginal coverage in finite samples, it provides approximate coverage guarantees under mild assumptions. In practice, these guarantees may not hold exactly, particularly in finite samples or under model misspecification. Therefore, empirical coverage serves as a practical measure of calibration, indicating how well prediction intervals capture the actual values. When combined with the median relative interval width, which measures the sharpness of the intervals, this allows for a comprehensive assessment of uncertainty quantification by evaluating the trade-off between reliability and informativeness. In Figures 7 and 8, Pareto plots based on 95%

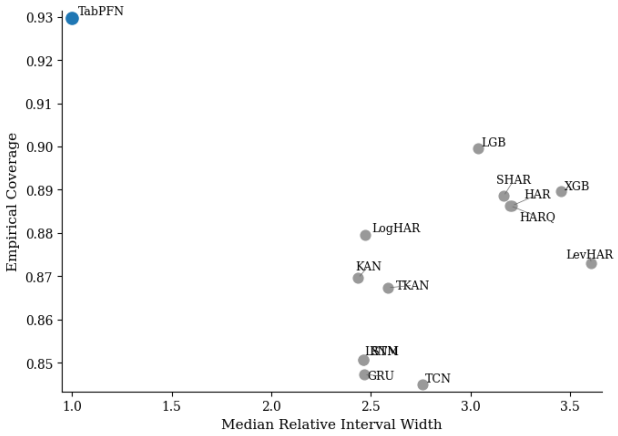
prediction intervals are presented with empirical coverage and median relative interval width on the axes, enabling the identification of Pareto-efficient models.

For the day-ahead horizon, both KAN and TabPFN lie on the Pareto frontier for models estimated on $\mathcal{D}_{HAR}$ and $\mathcal{D}_{HAR-X}$, indicating that neither model is strictly dominated in terms of the trade-off between coverage and interval width. Specifically, KAN achieves the lowest median relative interval width under both information sets, producing the narrowest prediction intervals, whereas TabPFN attains the highest empirical coverage. Another noteworthy results is that the HAR models (except for Log-HAR) seem to produce larger median relative interval widths than the machine learning models, under both information sets. A possible explanation for this finding is the limited flexibility of HAR-type models, which rely on a linear structure. This restricts their ability to capture nonlinearities and more complex dynamics (e.g. regime shifts) in the data, leading to larger residuals. Since EnbPI constructs prediction intervals based on the empirical distribution of these residuals, this translates directly into wider intervals. In contrast, machine learning models can often better approximate the underlying data-generating process, leading to smaller residual dispersion and, consequently, sharper prediction intervals.

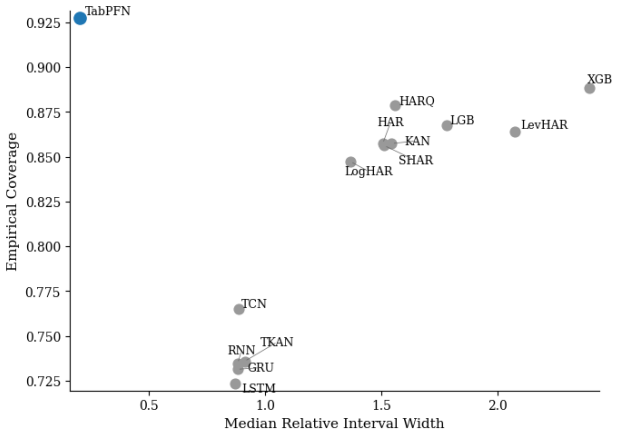
For the average week-ahead and average month-ahead horizons, this trade-off disappears, as TabPFN clearly dominates the competing models. In these settings, it simultaneously achieves the highest empirical coverage and the smallest median relative interval width, and does so with a substantial margin, indicating both highly reliable and informative prediction intervals.



(a) Day-ahead horizon

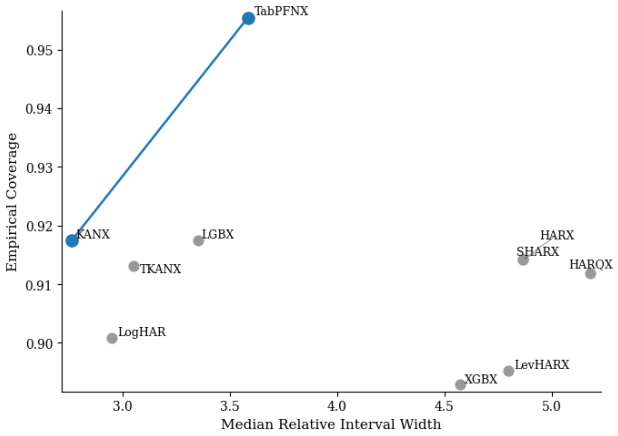


(b) Average week-ahead horizon

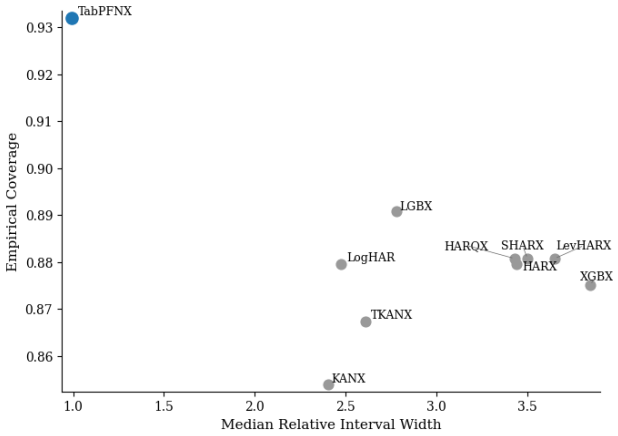


(c) Average month-ahead horizon

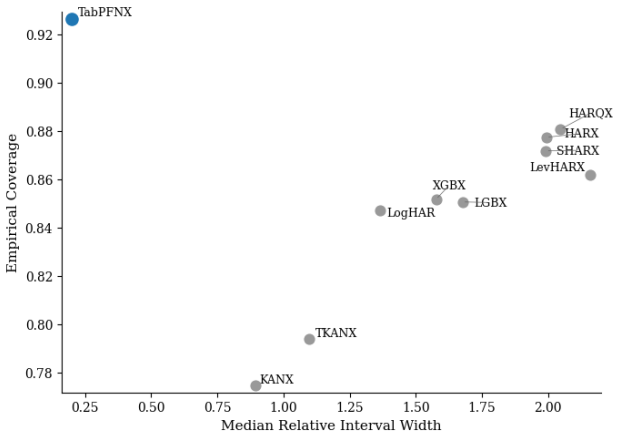
Figure 7: Pareto plot of interval width and empirical coverage, for models estimated on $\mathcal{D}_{HAR}$



(a) Day-ahead horizon



(b) Average week-ahead horizon



(c) Average month-ahead horizon

Figure 8: Pareto plot of interval width and empirical coverage, for models estimated on $\mathcal{D}_{HAR\text{-}X}$

5.4 Upper-tail quantile forecasting accuracy

Now, we consider quantile forecast[9] accuracy, specifically the right tail. This is done by report relative Weighted Quantile Scores[10], calculated as in 75:

$$\frac{WQS_j}{WQS_i}, \tag{75}$$

where i corresponds to row the model in row i and j corresponds to the model in column j. Alongside the relative losses, statistical significance is indicated by light, normal and dark colour codes, corresponding to the 10%, 5%, and 1% significance levels. These colours reflect p-values from pairwise equal quantile forecast accuracy tests, the hypotheses for these tests are shown in 76:

$$H_0 : \mathbb{E}[WQS_i - WQS_j] = 0 \quad \text{vs.} \quad H_1 : \mathbb{E}[WQS_i - WQS_j] \neq 0, \tag{76}$$

where a relative loss smaller (larger) than 1, when accompanied by statistically significant test results, indicates that the model listed in column j has superior (inferior) quantile forecasting performance compared to the model listed in row i, and we indicate this by colouring the cell green (red).

The results are shown in Figures 9 and 10. For day-ahead quantile forecasts based on models estimated with $\mathcal{D}_{HAR}$, the LSTM performs worst, being inferior in all pairwise comparisons. GRU, KAN, RNN, TCN, and TKAN are also outperformed in most cases, indicating relatively weak performance. In contrast, LGBM, LevHAR, Log-HAR, SHAR, TabPFN, and XGB exhibit either statistically insignificant differences or significant superiority over competitors. A similar pattern holds for $\mathcal{D}_{HAR-X}$, with one key difference: TabPFNX is now outperformed by LGBMX and Log-HARX, which achieve stronger predictive accuracy. Overall, no single model consistently dominates at the day-ahead horizon.

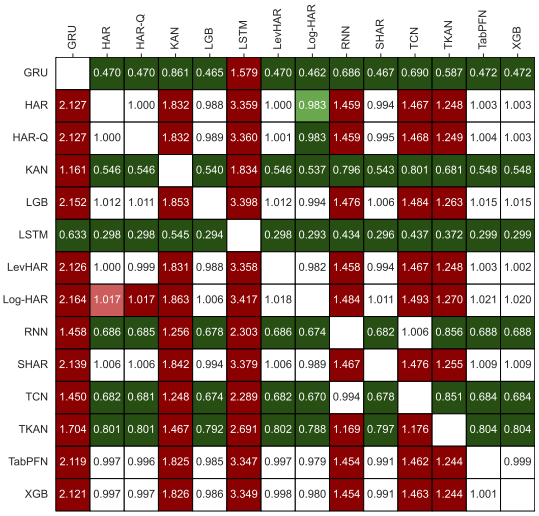
In contrast, for the average week-ahead and month-ahead horizons, TabPFN uniformly dominates all other models across both predictor sets. This indicates that TabPFN exhibits superior quantile forecasting performance relative to all other models.

To confirm the conclusions drawn from the Weighted Quantile Score pairwise tests, the

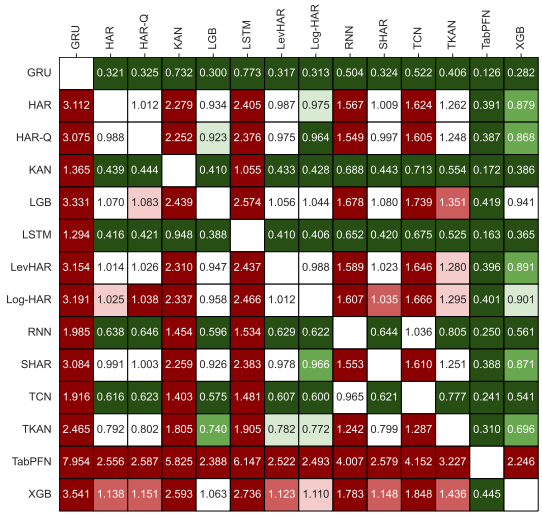
[9] Quantile forecasts for TabPFN are obtained by extracting them from its posterior predictive distribution rather than by directly optimizing the quantile loss function in 66.

[10] In this application, the integral is approximated discretely as $WQS_{t+h|t} = \sum_{\tau \in \{0.90, 0.95, 0.99\}} \omega(\tau)\, QS_{t+h|t}(\tau)$.

Quantile Score pairwise tests at $\tau = 0.95$ in Appendices 15 and 16 and at $\tau = 0.99$ in Appendices 17 and 18 yield largely consistent results, aside from a limited number of individual pairwise differences in significance. The overall pattern remains unchanged: at the day-ahead horizon, LSTM is dominated in all pairwise comparisons under both information sets, while GRU, KAN, RNN, TCN, and TKAN are outperformed in most comparisons. The conclusions for the remaining models are consistent with those based on the Weighted Quantile Score, while at the average week-ahead and month-ahead horizons, TabPFN continues to exhibit superior quantile forecasting accuracy, despite some pairwise comparisons now being significant at $\alpha = 0.05$ or $\alpha = 0.10$ when considering models estimated on $\mathcal{D}_{HAR-X}$ instead of at $\alpha = 0.01$ everywhere.



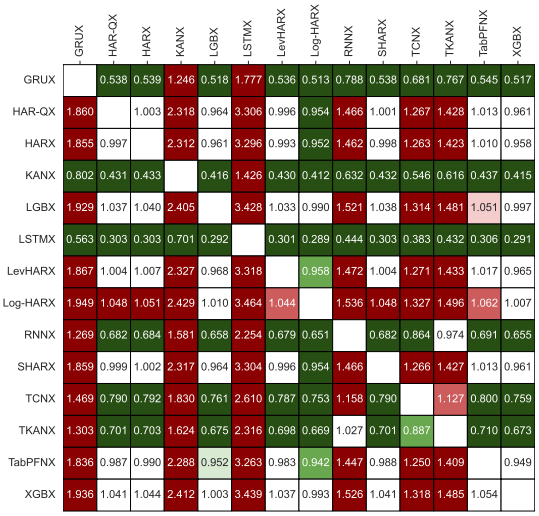
(a) Day-ahead horizon



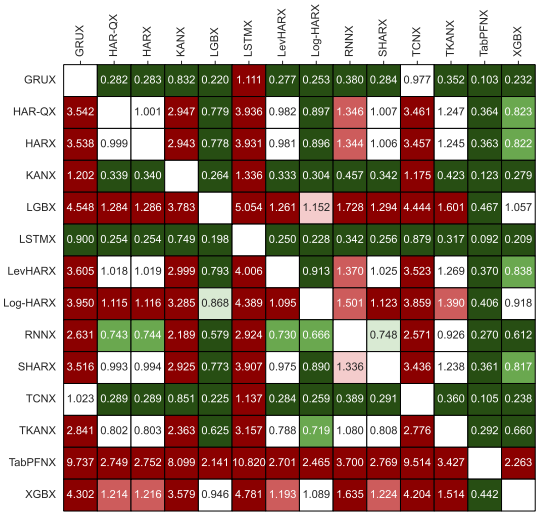
(b) Average week-ahead horizon

(c) Average month-ahead horizon

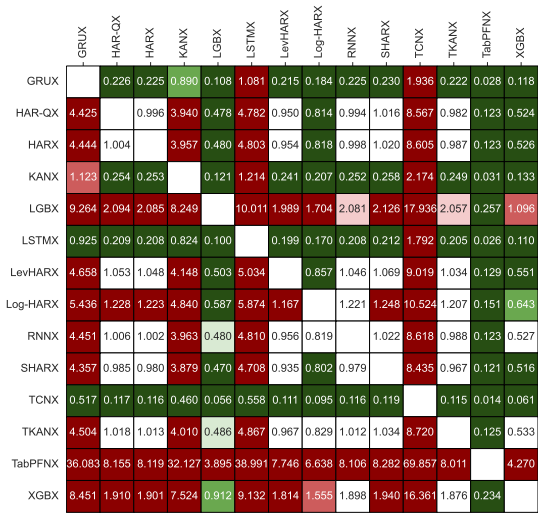
Figure 9: Pairwise equal predictive accuracy test results with Weighted Quantile Scores, estimated on $\mathcal{D}_{HAR}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).



(a) Day-ahead horizon



(b) Average week-ahead horizon



(c) Average month-ahead horizon

Figure 10: Pairwise equal predictive accuracy test results with Weighted Quantile Scores, estimated on $\mathcal{D}_{HAR-X}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).

5.5 Interpreting TabPFN's performance across horizons

Two components of the synthetic prior used in TabPFN-3 are especially relevant for realized variance forecasting. First, the prior includes a temporal component: SCMs are extended into discrete-time Dynamic Structural Causal Models, allowing the model to learn from synthetic tasks where rows arrive in time order and where future values depend on past ones (Grinsztajn et al. 2026). This mirrors the structure of realized variance, which is itself a time-indexed sequence with persistent dynamics. Because TabPFN is pretrained on synthetic temporal tasks, it enters our forecasting problem with an inductive bias toward reasoning over ordered observations rather than treating them as i.i.d. samples. This aligns with our rolling-window setup, where each forecast uses only past realized variance to predict a future value.

Second, the prior includes explicit out-of-distribution prediction tasks. These tasks expose the model to synthetic situations where the test data differ from the training data, and where correct predictions require extrapolation rather than interpolation. Grinsztajn et al. (2026) highlight that this out-of-distribution prior enables TabPFN to extrapolate beyond the range of recently observed values. Realized variance frequently exhibits sudden spikes or regime shifts that push the series outside the range covered by the current estimation window. Models trained only on historical data must estimate parameters from these limited windows, whereas TabPFN can rely on inference strategies learned during pretraining to remain stable under such shifts.

These two prior components matter most at the average week- and month-ahead horizons. At these horizons, the forecast target is a moving average of several future days, so consecutive targets overlap heavily and contain relatively little independent information. HAR and other ML models must infer parameters from these weak signals. TabPFN, by contrast, can fall back on its pretrained temporal reasoning and out-of-distribution-aware inference, which are designed to generalize robustly from small, overlapping sets of observations and to handle situations where the target lies outside the range of recent values.

At the one-day-ahead horizon, realized variance is close to linearly persistent, and simple autoregressive or HAR-type models already capture most of the predictive structure. In this regime, the advantages of TabPFN's temporal and out-of-distribution priors are less pronounced, which aligns with the absence of a single clearly dominant model at the shortest horizon.

6 CONCLUSION

For the one-day-ahead horizon, the results show no clear dominance of one single model across all evaluation criteria. Instead, forecasting performance depends on the chosen loss function and predictor set. When models are estimated on $\mathcal{D}_{HAR}$, the Model Confidence Set results indicate that multiple HAR-type specifications cannot be statistically rejected as inferior under both MSE and QLIKE. However, pairwise comparisons reveal that specific models do stand out conditionally: Log-HAR and TabPFN perform best under MSE in the sense that they are superior in most pairwise comparisons, while XGB and Log-HAR are most effective under QLIKE, indicating stronger robustness against volatility underestimation. For models estimated on the extended information set $\mathcal{D}_{HAR-X}$, performance becomes more selective. Log-HARX emerges as the only HAR-type model consistently retained in the MCS, and it is statistically superior to all competing models under MSE in pairwise comparisons. This highlights that a parsimonious specification with a log transformation can outperform more complex models when additional predictors are included. At the same time, many machine learning models are excluded, suggesting difficulties in effectively exploiting the larger information set. Across both predictor sets, an important limitation persists: none of the models accurately captures the magnitude of extreme volatility spikes. From an uncertainty perspective, no single model dominates the trade-off between calibration and sharpness. TabPFN achieves the highest empirical coverage, while KAN produces the narrowest intervals. For quantile forecasting, no model consistently dominates across all pairwise comparisons; instead, performance should be interpreted in terms of the frequency with which models dominate or are dominated. Under this interpretation, linear and tree-based methods, together with TabPFN, perform relatively well, whereas network-based architectures are dominated in a substantial share of comparisons.

In contrast to the short horizon, the results for the average week-ahead and month-ahead forecasts are remarkably clear and consistent across all evaluation dimensions. The Model Confidence Set results show that TabPFN is the only model retained across both loss functions and both predictor sets. This implies that all competing models are statistically rejected as inferior at the 5% level. These findings are strongly reinforced by the pairwise Diebold–Mariano tests, which demonstrate that TabPFN consistently outperforms all alternative models in terms of predictive accuracy. This dominance extends beyond point forecasts. In terms of uncertainty quantification, TabPFN clearly outperforms all other models by simultaneously achieving the highest empirical coverage and the lowest median relative interval width. Unlike the day-ahead case, there is no trade-off between

reliability and sharpness: TabPFN provides both highly accurate and highly informative prediction intervals. Similarly, for upper-tail forecasting, TabPFN uniformly dominates all competing models across both predictor sets and horizons. The Weighted Quantile Score results indicate that it delivers superior quantile forecasts for the upper tail. Importantly, these findings hold regardless of whether additional macroeconomic predictors are included. This suggests that TabPFN is able to effectively extract relevant information from both smaller and larger predictor sets, in contrast to some models whose performance deteriorates when the dimensionality increases. Taken together, the results provide consistent evidence that, for this series, TabPFN produced the most accurate and best-calibrated week- and month-ahead forecasts under every criterion considered. Whether this ranking extends to other assets, sample periods, or market conditions cannot be established from a single-asset design and remains a topic for further research. The consistency of the result across both loss functions, both predictor sets, and both medium-term horizons nevertheless makes a broader cross-sectional evaluation a worthwhile next step for further research. Although we do not explicitly evaluate computational cost or implementation complexity, it is worth noting that TabPFN is relatively convenient to use compared to many alternative machine learning models. In particular, it does not require explicit hyperparameter tuning by the user, which substantially simplifies the modeling pipeline relative to methods such as gradient boosting or neural networks. In addition, TabPFN produces an approximation to the posterior predictive distribution, which allows prediction intervals to be obtained directly by extracting the relevant quantiles. This avoids the need for auxiliary procedures to construct prediction intervals.

7 DISCUSSION

Several limitations of the empirical analysis should be taken into account when interpreting the results. First, the modeling framework does not explicitly account for the presence of jumps in the log-price process. While realized variance inherently captures both the continuous and jump components of price variation, ignoring this decomposition may affect how volatility dynamics are modeled. In particular, jump-driven movements tend to be abrupt and non-persistent, whereas most of the models considered are designed to capture smoother, persistent dynamics. As a result, the inability of the models to accurately capture the magnitude of extreme volatility spikes may, at least in part, be attributed to this omission rather than purely to model inadequacy. Second, the construction of the test set may have influenced the evaluation of forecasting performance, especially with respect to extreme events. The test period begins immediately after the realized variance spike associated with the COVID-19 market crash, meaning that the most extreme volatility realization is not included in the out-of-sample evaluation. At the same time, the empirical results show that none of the models is able to accurately predict large volatility spikes, regardless of the predictor set or modeling approach. While this suggests a general limitation in forecasting extreme realized variance, the exclusion of such an extreme event from the test set may have affected the relative performance rankings, particularly for models that are better suited to capturing nonlinearities or tail behavior. Third, in the quantile forecasting analysis, hyperparameter configurations were not optimized specifically for quantile loss functions, but instead taken from the point forecasting setup to keep computational cost manageable. This may have disadvantaged machine learning models in particular, as their performance is typically sensitive to the choice of objective function. Therefore, the relatively weaker performance of several machine learning models in the right-tail accuracy evaluation may partly reflect suboptimal tuning rather than solely limitations of the models themselves. Fourth, we did not explicitly model regimes of high and low volatility. Financial markets often exhibit regime-dependent behavior, where volatility dynamics differ between calm and turbulent periods, which was not taken into account in this research. Fifth, the way the models arrive at their forecasts differs fundamentally, which complicates a direct comparison. The HAR-type and machine learning models are re-estimated on the rolling training window at each forecast origin, with the machine learning models additionally relying on hyperparameters selected through a grid search over a pre-specified set of candidate values, so that an unfavourable or insufficiently wide grid may have led to configurations that understate their true forecasting ability. TabPFN, by contrast, is never fitted to the training data at all: its parameters are fixed from synthetic pre-training and it produces forecasts purely by conditioning on the observations supplied

in context, meaning that the relative performance we report partly reflects a difference in how the models learn rather than in forecasting ability alone.

REFERENCES

Andersen, T. G., Bollerslev, T., Christoffersen, P. & Diebold, F. X. (2005), 'Volatility forecasting'.

Andersen, T. G., Dobrev, D. & Schaumburg, E. (2012), 'Jump-robust volatility estimation using nearest neighbor truncation', *Journal of Econometrics* **169**(1), 75–93.

Audrino, F. & Chassot, J. (2025), 'Hard to beat: The overlooked impact of rolling windows in the era of machine learning', *International Journal of Forecasting* .

Baillie, R. T. (1996), 'Long memory processes and fractional integration in econometrics', *Journal of Econometrics* **73**(1), 5–59.

Barndorff-Nielsen, O. E., Hansen, P. R., Lunde, A. & Shephard, N. (2009), 'Realized kernels in practice: Trades and quotes'.

Barndorff-Nielsen, O. E. & Shephard, N. (2002), 'Econometric analysis of realized volatility and its use in estimating stochastic volatility models', *Journal of the Royal Statistical Society Series B: Statistical Methodology* **64**(2), 253–280.

Bekaert, G. & Wu, G. (2000), 'Asymmetric volatility and risk in equity markets', *The Review of Financial Studies* **13**(1), 1–42.

Bollerslev, T. (1986), 'Generalized autoregressive conditional heteroskedasticity', *Journal of Econometrics* **31**(3), 307–327.

Bollerslev, T., Patton, A. J. & Quaedvlieg, R. (2016), 'Exploiting the errors: A simple approach for improved volatility forecasting', *Journal of Econometrics* **192**(1), 1–18.

Branco, R. R., Rubesam, A. & Zevallos, M. (2024), 'Forecasting realized volatility: Does anything beat linear models?', *Journal of Empirical Finance* **78**, 101524.

Brauneis, A. & Sahiner, M. (2024), 'Crypto volatility forecasting: Mounting a har, sentiment, and machine learning horserace', *Asia-Pacific Financial Markets* pp. 1–33.

Bucci, A. (2020), 'Realized volatility forecasting with neural networks', *Journal of Financial Econometrics* **18**(3), 502–531.

Chen, T. & Guestrin, C. (2016), Xgboost: A scalable tree boosting system, *in* 'Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining', pp. 785–794.

Christensen, K., Siggaard, M. & Veliyev, B. (2023), 'A machine learning approach to volatility forecasting', *Journal of Financial Econometrics* **21**(5), 1680–1727.

Chun, D., Cho, H. & Ryu, D. (2025), 'Volatility forecasting and volatility-timing strategies: A machine learning approach', *Research in International Business and Finance* **75**, 102723.

Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. (2014), 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *arXiv preprint arXiv:1412.3555* .

Clark, T. E. & West, K. D. (2007), 'Approximately normal tests for equal predictive accuracy in nested models', *Journal of Econometrics* **138**(1), 291–311.

Clements, A., Preve, D. & Tee, C. (2024), 'Harvesting the har-x volatility model', *Available at SSRN 4733597* .

Corsi, F. (2009), 'A simple approximate long-memory model of realized volatility', *Journal of Financial Econometrics* **7**(2), 174–196.

Corsi, F. & Renò, R. (2012), 'Discrete-time volatility forecasting with persistent leverage effect and the link with continuous-time volatility modeling', *Journal of Business & Economic Statistics* **30**(3), 368–380.

De Boor, C. (1972), 'On calculating with b-splines', *Journal of Approximation Theory* **6**(1), 50–62.

Diebold, F. X. & Mariano, R. S. (1995), 'Comparing predictive accuracy', *Journal of Business and Economic Statistics* **13**(3), 253–263.

Genet, R. & Inzirillo, H. (2024), 'A temporal kolmogorov-arnold transformer for time series forecasting', *arXiv preprint arXiv:2406.02486* .

Gneiting, T. & Raftery, A. E. (2007), 'Strictly proper scoring rules, prediction, and estimation', *Journal of the American Statistical Association* **102**(477), 359–378.

Grinsztajn, L., Flöge, K., Key, O., Birkel, F., Jund, P., Roof, B., Manium, M., Hoo, S. B., Bühler, M., Garg, A. et al. (2026), 'Tabpfn-3: Technical report', *arXiv preprint arXiv:2605.13986* .

Gunnarsson, E. S., Isern, H. R., Kaloudis, A., Risstad, M., Vigdel, B. & Westgaard, S. (2024), 'Prediction of realized volatility and implied volatility indices using ai and machine learning: A review', *International Review of Financial Analysis* **93**, 103221.

Hansen, P. R., Lunde, A. & Nason, J. M. (2003), 'Choosing the best volatility models: the model confidence set approach', *Oxford Bulletin of Economics and Statistics* **65**, 839–861.

Hansen, P. R., Lunde, A. & Nason, J. M. (2011), 'The model confidence set', *Econometrica* **79**(2), 453–497.

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural Computation* **9**(8), 1735–1780.

Hoepner, A. G., McMillan, D., Vivian, A. & Wese Simen, C. (2021), 'Significance, relevance and explainability in the machine learning age: an econometrics and financial data science perspective', *The European Journal of Finance* **27**(1-2), 1–7.

Hollmann, N., Müller, S., Eggensperger, K. & Hutter, F. (2022), 'Tabpfn: A transformer that solves small tabular classification problems in a second', *arXiv preprint arXiv:2207.01848* .

Hoo, S. B., Müller, S., Salinas, D. & Hutter, F. (2025), 'From tables to time: Extending tabpfn-v2 to time series forecasting', *arXiv preprint arXiv:2501.02945* .

Huang, Y. & Luo, Y. (2024), 'Forecasting conditional volatility based on hybrid garch-type models with long memory, regime switching, leverage effect and heavy-tail: Further evidence from equity market', *The North American Journal of Economics and Finance* **72**, 102148.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. & Liu, T.-Y. (2017), 'Lightgbm: A highly efficient gradient boosting decision tree', *Advances in Neural Information Processing Systems* **30**.

Khan, M. Y. (2015), Advances in applied nonlinear time series modeling, PhD thesis, lmu.

Koenker, R. & Bassett Jr, G. (1978), 'Regression quantiles', *Econometrica* pp. 33–50.

Lea, C., Vidal, R., Reiter, A. & Hager, G. D. (2016), Temporal convolutional networks: A unified approach to action segmentation, *in* 'European Conference on Computer Vision', Springer, pp. 47–54.

Li, H., Huang, X., Luo, F., Zhou, D., Cao, A. & Guo, L. (2025), 'Revolutionizing agricultural stock volatility forecasting: a comparative study of machine learning and har-rv models', *Journal of Applied Economics* **28**(1), 2454081.

Li, X., Li, D., Cheng, Y. & Li, W. (2024), 'Forecasting the volatility of educational firms based on har model and lstm models considering sentiment and educational policy', *Heliyon* **10**(19).

Lipton, Z. C., Berkowitz, J. & Elkan, C. (2015), 'A critical review of recurrent neural networks for sequence learning', *arXiv preprint arXiv:1506.00019* .

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y. & Tegmark, M. (2024), 'Kan: Kolmogorov-arnold networks', *arXiv preprint arXiv:2404.19756* .

Meligkotsidou, L., Panopoulou, E., Vrontos, I. D. & Vrontos, S. D. (2019), 'Quantile forecast combinations in realised volatility prediction', *Journal of the Operational Research Society* **70**(10), 1720–1733.

Patton, A. J. (2011), 'Volatility forecast comparison using imperfect volatility proxies', *Journal of Econometrics* **160**(1), 246–256.

Patton, A. J. & Sheppard, K. (2015), 'Good volatility, bad volatility: Signed jumps and the persistence of volatility', *Review of Economics and Statistics* **97**(3), 683–697.

Poon, S.-H. & Granger, C. (2005), 'Practical issues in forecasting volatility', *Financial Analysts Journal* **61**(1), 45–56.

Poon, S.-H. & Granger, C. W. J. (2003), 'Forecasting volatility in financial markets: A review', *Journal of Economic Literature* **41**(2), 478–539.

Rahimikia, E. & Poon, S.-H. (2026), 'Machine learning for realised volatility forecasting', *Journal of Empirical Finance* p. 101739.

Shafer, G. & Vovk, V. (2008), 'A tutorial on conformal prediction.', *Journal of Machine Learning Research* **9**(3).

So, M. K., Chu, A. M., Lo, C. C. & Ip, C. Y. (2022), 'Volatility and dynamic dependence modeling: Review, applications, and financial risk management', *Wiley Interdisciplinary Reviews: Computational Statistics* **14**(5), e1567.

Somvanshi, S., Javed, S. A., Islam, M. M., Pandit, D. & Das, S. (2025), 'A survey on kolmogorov-arnold network', *ACM Computing Surveys* **58**(2), 1–35.

Taneva-Angelova, G. & Granchev, D. (2025), 'Deep learning and transformer architectures for volatility forecasting: Evidence from us equity indices', *Journal of Risk and Financial Management* **18**(12), 685.

Teller, A., Pigorsch, U. & Pigorsch, C. (2022), 'Short-to long-term realized volatility forecasting using extreme gradient boosting', *Available at SSRN 4267541* .

Teräsvirta, T. (2009), An introduction to univariate garch models, *in* 'Handbook of Financial Time Series', Springer, pp. 17–42.

Teräsvirta, T. & Zhao, Z. (2014), Stylized facts of return series, robust estimates and three popular models of volatility, *in* 'Perspectives on Econometrics and Applied Economics', Routledge, pp. 67–94.

Wen, F., Gong, X. & Cai, S. (2016), 'Forecasting the volatility of crude oil futures using har-type models with structural breaks', *Energy Economics* **59**, 400–413.

Xu, C. & Xie, Y. (2021), Conformal prediction interval for dynamic time-series, *in* 'International Conference on Machine Learning', PMLR, pp. 11559–11569.

Xu, K., Chen, L. & Wang, S. (2024), 'Kolmogorov-arnold networks for time series: Bridging predictive power and interpretability', *arXiv preprint arXiv:2406.02496* .

Yamak, P. T., Li, Y., Zhang, T. & Pathan, M. S. (2025), 'Kolmogorov-arnold networks for time series forecasting: a comprehensive review', *Cluster Computing* **28**(14), 929.

Zammarchi, G. & Kawasaki, Y. (2026), 'Assessing the quality of forecasting quantitatively using model confidence set: from naive models to neural networks', *Quality & Quantity* pp. 1–25.

Zhang, Y., Cui, L. & Yan, W. (2025), 'Integrating kolmogorov–arnold networks with time series prediction framework in electricity demand forecasting', *Energies* **18**(6), 1365.

A APPENDIX

A.1 Stationarity test results

The Augmented Dickey-Fuller test results are reported in table 4. The regression to derive the test statistic is

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^{p} \delta_i \Delta y_{t-i} + \varepsilon_t$$

where γ is the parameter of interest. Then, the ADF test statistic is

$$ADF = \frac{\hat{\gamma}}{\text{SE}(\hat{\gamma})}.$$

The hypotheses that are being considered in this test are: $H_0 : \gamma = 0 \quad vs. \quad H_1 : \gamma < 0$. Here, if the null hypothesis indicates the presence of a unit root (hence nonstationarity), whereas the alternative hypothesis indicates absence of a unit root. As a consequence, the 3-Month Treasury Bill rate variable appears nonstationary if we consider a 5% significance level. It can be observed that the first difference, ΔTMTB, is stationary.

Table 4: Augmented Dickey-Fuller test results

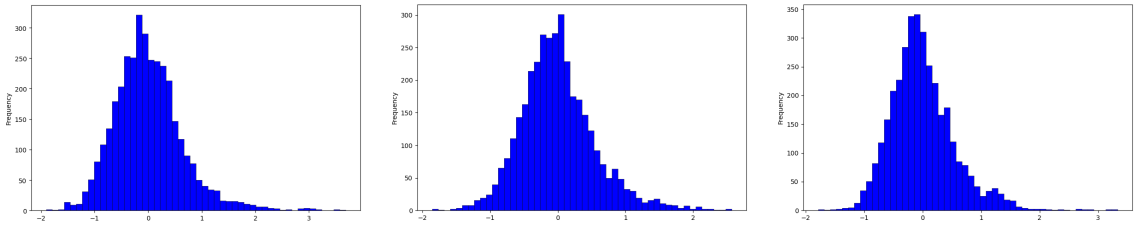
variable	test statistic	p-value
RV	-7.913	0.000
RV_{t-1}	-7.915	0.000
$RV_{t-1\|t-5}$	-7.139	0.000
$RV_{t-1\|t-22}$	-7.227	0.000
VIX	-5.249	0.000
EPU	-5.412	0.000
COP	-2.997	0.035
TB3M	-0.901	0.788
BC	-8.419	0.000
ΔTMTB	-9.967	0.000

A.2 Multicollinearity

Table 5: Correlation Matrix of regressors in $\mathcal{D}_{HAR-X}$

	RV_{t-1}	$RV_{t\|t-4}$	$RV_{t\|t-21}$	VIX_t	EPU_t	COP_t	$TMTB_t$	BC
RV_{t-1}	1.000	0.787	0.582	0.572	0.312	-0.132	-0.066	-0.367
$RV_{t\|t-4}$	0.787	1.000	0.786	0.711	0.401	-0.174	-0.085	-0.499
$RV_{t\|t-21}$	0.582	0.786	1.000	0.749	0.485	-0.234	-0.107	-0.669
VIX_t	0.572	0.711	0.749	1.000	0.497	-0.084	-0.210	-0.361
EPU_t	0.312	0.401	0.485	0.497	1.000	-0.126	-0.207	-0.281
COP_t	-0.132	-0.174	-0.234	-0.084	-0.126	1.000	-0.006	0.126
$TMTB_t$	-0.066	-0.085	-0.107	-0.210	-0.207	-0.006	1.000	0.019
BC_t	-0.367	-0.499	-0.669	-0.361	-0.281	0.126	0.019	1.000

A.3 Distribution of residuals of log-realized variance



(a) Day-ahead (b) Average week-ahead (c) Average month-ahead

Figure 11: Histograms of the training set residuals of the log-HAR model at all forecast horizons.

A.4 Hyperparameter tuning

Table 6: Hyperparameter tuning for network-based models

Shared hyperparameters	**KAN/TKAN**	**RNN**	**LSTM**	**GRU**	**TCN**
B $\in \{32, 64, 128, 256, 350\}$	W $\in \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$	H $\in \{5, 8, 13\}$	H $\in \{2, 3, 5\}$	H $\in \{3, 4, 6\}$	H $\in \{3, 6\}$
S $\in \{60, 120, 200, 300\}$	SG $\in \{3, 5, 7, 10\}$	L $\in \{1, 2\}$	L $\in \{1, 2\}$	L $\in \{1, 2\}$	L $= 1$
$\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$ (except TKAN)					
LR $= 10^{-3}$ (default Adam)					
D $= 0.2$					

Notes. Abbreviations: W = width; SG = spline grid; H = hidden units; L = number of layers; B = batch size; S = training steps; LR = learning rate; D = dropout rate. During structural hyperparameter selection (prior to regularization tuning), the regularization parameter λ was fixed at 10^{-4}. Also, batch size $B = 128$ and training steps $S = 50$ were used prior to optimization and regularization tuning.

Table 7: Hyperparameter tuning for tree-based models

LightGBM	**XGBoost**
Leaves $\in \{15, 30, 60, 100\}$	Max depth $\in \{3, 5, 7\}$
Depth $\in \{3, 5, 7\}$	Depth $\in \{3, 5, 7\}$
$\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$	$\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$
Learning rate = default	Learning rate = default
Min. leaf samples = 50	Min. child weight = 5
Number of trees = 1000	Number of trees = 1000

Notes. LR = learning rate. During structural hyperparameter selection (prior to regularization tuning), the regularization parameter λ was fixed at 10^{-4}.

A.5 Model Confidence Set under alternative significance level

Table 8: Models included in the Model Confidence Set across estimation sets and forecast horizons, $\alpha = 0.10$, $B = 10,000$, loss function: $\mathcal{L}^{QLIKE}$

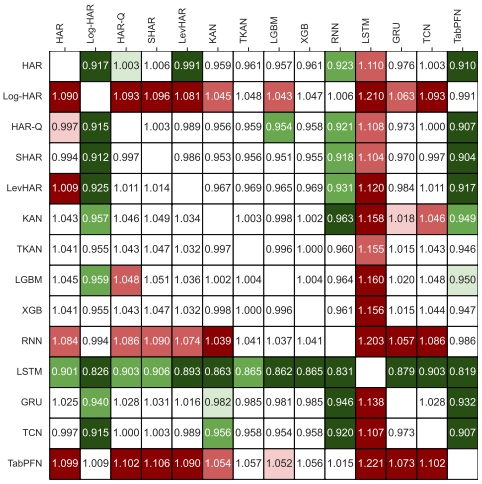
Estimated on $\mathcal{D}_{HAR}$			**Estimated on $\mathcal{D}_{HAR\text{-}X}$**		
RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$	RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$
HAR	TabPFN	TabPFN	Log-HARX	TabPFNX	TabPFNX
Log-HAR			TKANX		
HAR-Q			LGBMX		
SHAR			XGBX		
TKAN			RNNX		
LGBM					
XGB					
RNN					

Table 9: Models included in the Model Confidence Set across estimation sets and forecast horizons, $\alpha = 0.10$, $B = 10,000$, loss function: $\mathcal{L}^{MSE}$

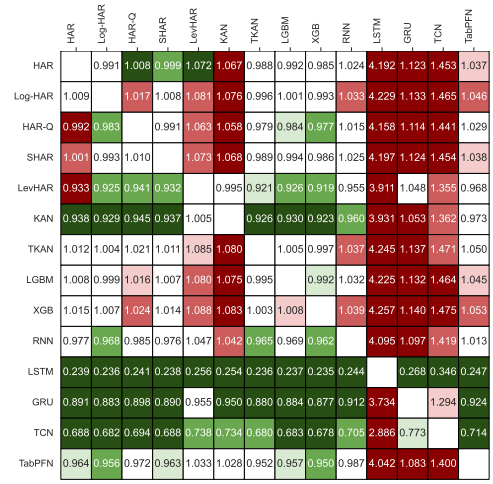
Estimated on $\mathcal{D}_{HAR}$			**Estimated on $\mathcal{D}_{HAR\text{-}X}$**		
RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$	RV_{t+1}	$RV_{t+5\|t+1}$	$RV_{t+22\|t+1}$
HAR	TabPFN	TabPFN	Log-HARX	TabPFNX	TabPFNX
Log-HAR					
HAR-Q					
SHAR					
LevHAR					
KAN					
TKAN					
LGBM					
XGB					
RNN					
GRU					
TabPFN					

A.6 Full pairwise Diebold-Mariano test results

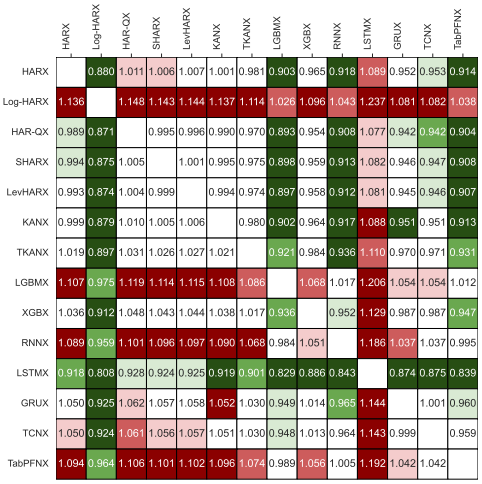
A.6.1 Day-ahead forecasts



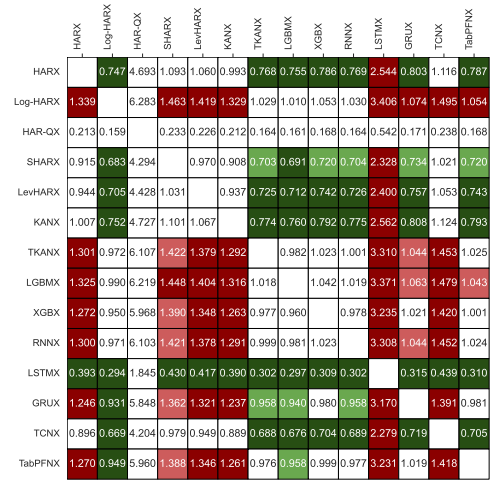
(a) $\mathcal{D}_{HAR}$, $\mathcal{L}^{MSE}$



(b) $\mathcal{D}_{HAR}$, $\mathcal{L}^{QLIKE}$



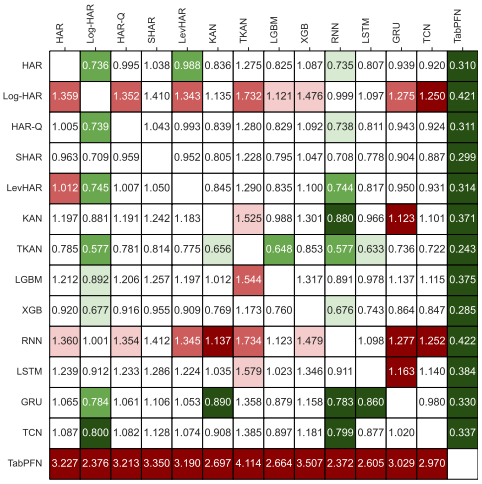
(c) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{MSE}$



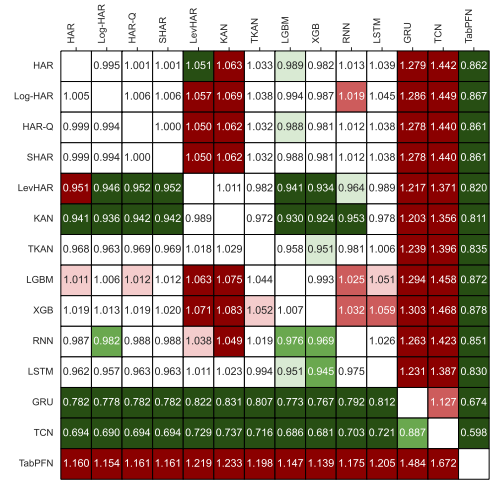
(d) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{QLIKE}$

Figure 12: Day-ahead relative loss matrices; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, dark = 10%, 5%, 1%; green/red = column better/worse than row).

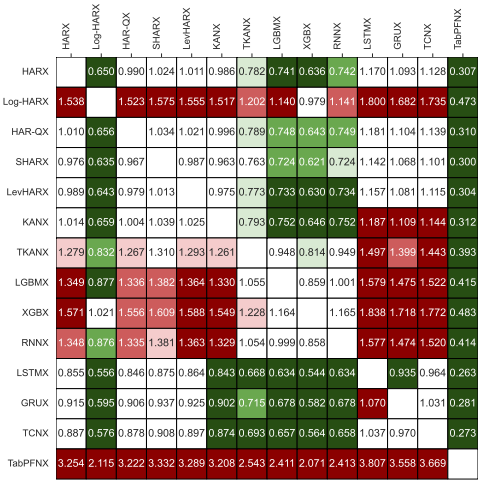
A.6.2 Week-ahead forecasts



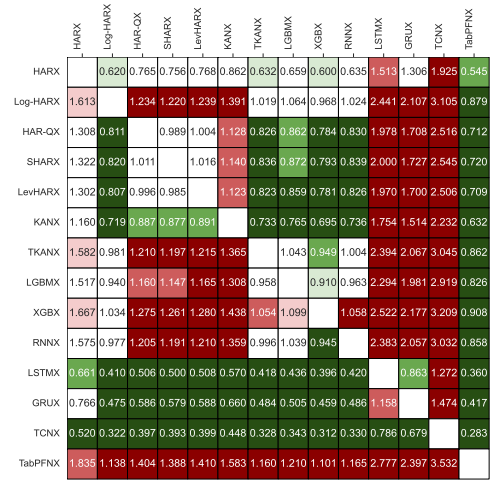
(a) $\mathcal{D}_{HAR}$, $\mathcal{L}^{MSE}$



(b) $\mathcal{D}_{HAR}$, $\mathcal{L}^{QLIKE}$



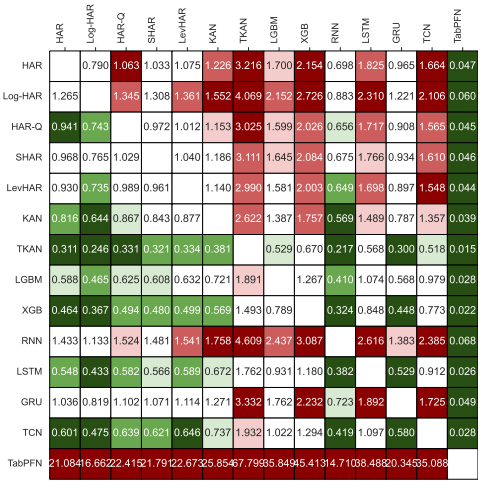
(c) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{MSE}$



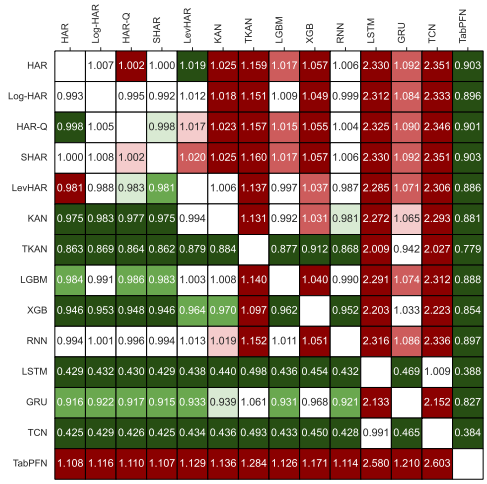
(d) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{QLIKE}$

Figure 13: Week-ahead relative loss matrices; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, dark = 10%, 5%, 1%; green/red = column better/worse than row).

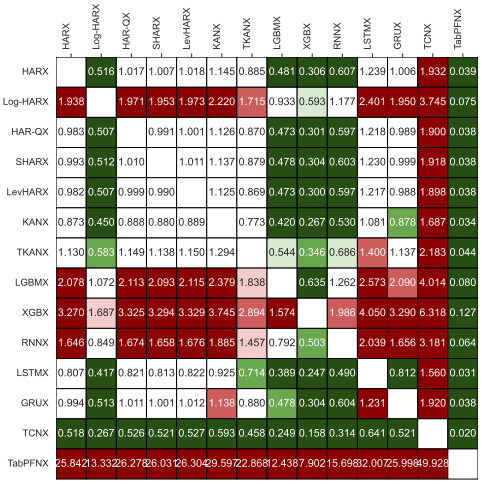
A.6.3 Month-ahead forecasts



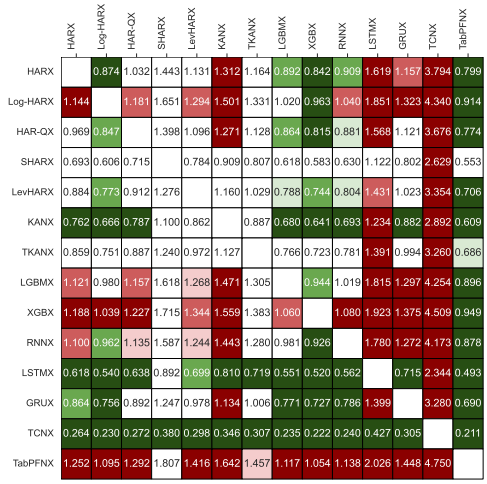
(a) $\mathcal{D}_{HAR}$, $\mathcal{L}^{MSE}$



(b) $\mathcal{D}_{HAR}$, $\mathcal{L}^{QLIKE}$



(c) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{MSE}$

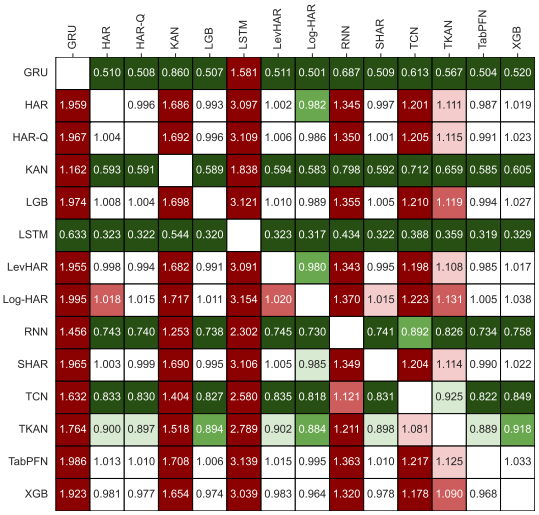


(d) $\mathcal{D}_{HAR\text{-}X}$, $\mathcal{L}^{QLIKE}$

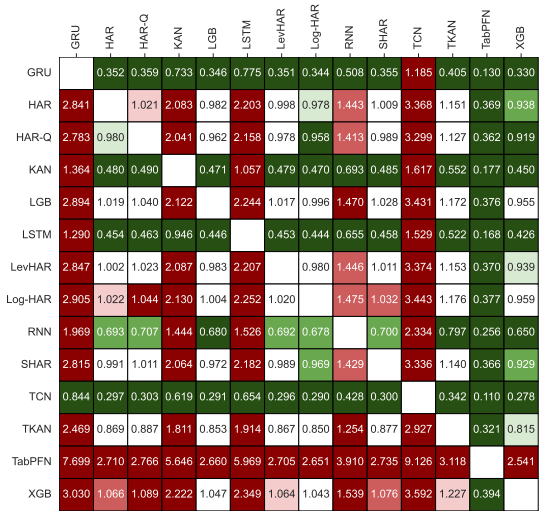
Figure 14: Month-ahead relative loss matrices; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, dark = 10%, 5%, 1%; green/red = column better/worse than row).

A.7 Quantile Score pairwise tests

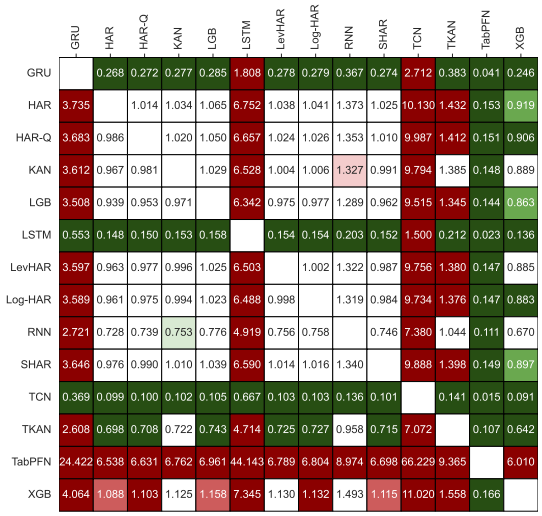
A.7.1 $\tau = 0.95$



(a) Day-ahead horizon

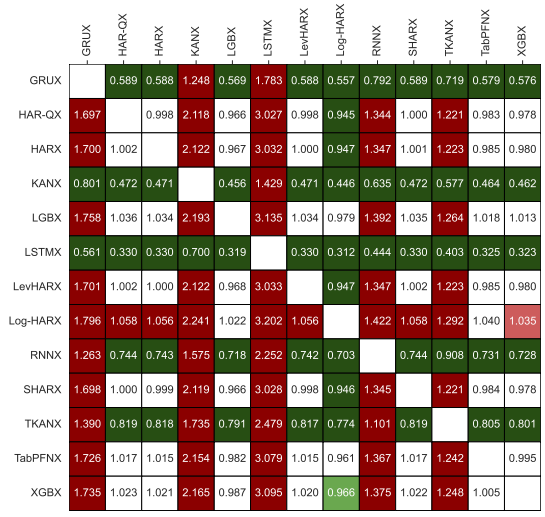


(b) Average week-ahead horizon



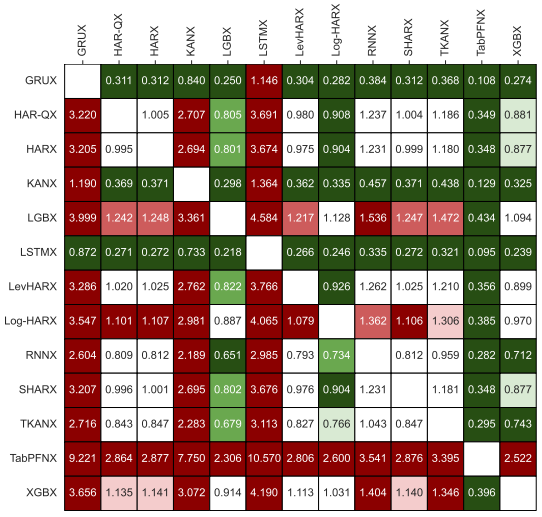
(c) Average month-ahead horizon

Figure 15: Pairwise equal predictive accuracy test results with Quantile Scores ($\tau = 0.95$), estimated on $\mathcal{D}_{HAR}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).



	GRUX	HAR-QX	HARX	KANX	LGBX	LSTMX	LevHARX	Log-HARX	RNNX	SHARX	TKANX	TabPFNX	XGBX
GRUX		0.589	0.588	1.248	0.569	1.783	0.588	0.557	0.792	0.589	0.719	0.579	0.576
HAR-QX	1.697		0.998	2.118	0.966	3.027	0.998	0.945	1.344	1.000	1.221	0.983	0.978
HARX	1.700	1.002		2.122	0.967	3.032	1.000	0.947	1.347	1.001	1.223	0.985	0.980
KANX	0.801	0.472	0.471		0.456	1.429	0.471	0.446	0.635	0.472	0.577	0.464	0.462
LGBX	1.758	1.036	1.034	2.193		3.135	1.034	0.979	1.392	1.035	1.264	1.018	1.013
LSTMX	0.561	0.330	0.330	0.700	0.319		0.330	0.312	0.444	0.330	0.403	0.325	0.323
LevHARX	1.701	1.002	1.000	2.122	0.968	3.033		0.947	1.347	1.002	1.223	0.985	0.980
Log-HARX	1.796	1.058	1.056	2.241	1.022	3.202	1.056		1.422	1.058	1.292	1.040	1.035
RNNX	1.263	0.744	0.743	1.575	0.718	2.252	0.742	0.703		0.744	0.908	0.731	0.728
SHARX	1.698	1.000	0.999	2.119	0.966	3.028	0.998	0.946	1.345		1.221	0.984	0.978
TKANX	1.390	0.819	0.818	1.735	0.791	2.479	0.817	0.774	1.101	0.819		0.805	0.801
TabPFNX	1.726	1.017	1.015	2.154	0.982	3.079	1.015	0.961	1.367	1.017	1.242		0.995
XGBX	1.735	1.023	1.021	2.165	0.987	3.095	1.020	0.966	1.375	1.022	1.248	1.005	

(a) Day-ahead horizon



	GRUX	HAR-QX	HARX	KANX	LGBX	LSTMX	LevHARX	Log-HARX	RNNX	SHARX	TKANX	TabPFNX	XGBX
GRUX		0.311	0.312	0.840	0.250	1.146	0.304	0.282	0.384	0.312	0.368	0.108	0.274
HAR-QX	3.220		1.005	2.707	0.805	3.691	0.980	0.908	1.237	1.004	1.186	0.349	0.881
HARX	3.205	0.995		2.694	0.801	3.674	0.975	0.904	1.231	0.999	1.180	0.348	0.877
KANX	1.190	0.369	0.371		0.298	1.364	0.362	0.335	0.457	0.371	0.438	0.129	0.325
LGBX	3.999	1.242	1.248	3.361		4.584	1.217	1.128	1.536	1.247	1.472	0.434	1.094
LSTMX	0.872	0.271	0.272	0.733	0.218		0.266	0.246	0.335	0.272	0.321	0.095	0.239
LevHARX	3.286	1.020	1.025	2.762	0.822	3.766		0.926	1.262	1.025	1.210	0.356	0.899
Log-HARX	3.547	1.101	1.107	2.981	0.887	4.065	1.079		1.362	1.106	1.306	0.385	0.970
RNNX	2.604	0.809	0.812	2.189	0.651	2.985	0.793	0.734		0.812	0.959	0.282	0.712
SHARX	3.207	0.996	1.001	2.695	0.802	3.676	0.976	0.904	1.231		1.181	0.348	0.877
TKANX	2.716	0.843	0.847	2.283	0.679	3.113	0.827	0.766	1.043	0.847		0.295	0.743
TabPFNX	9.221	2.864	2.877	7.750	2.306	10.570	2.806	2.600	3.541	2.876	3.395		2.522
XGBX	3.656	1.135	1.141	3.072	0.914	4.190	1.113	1.031	1.404	1.140	1.346	0.396	

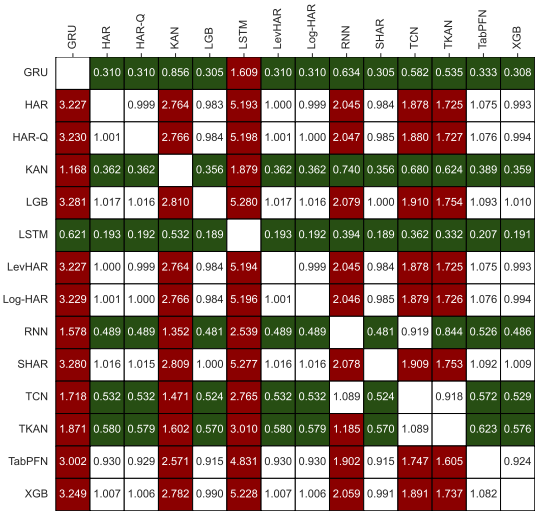
(b) Average week-ahead horizon

	GRUX	HAR-QX	HARX	KANX	LGBX	LSTMX	LevHARX	Log-HARX	RNNX	SHARX	TKANX	TabPFNX	XGBX
GRUX		0.256	0.253	0.890	0.110	1.088	0.238	0.212	0.227	0.256	0.264	0.029	0.131
HAR-QX	3.900		0.986	3.471	0.430	4.243	0.930	0.826	0.886	1.000	1.029	0.114	0.509
HARX	3.956	1.014		3.521	0.436	4.303	0.943	0.837	0.898	1.014	1.044	0.116	0.517
KANX	1.124	0.288	0.284		0.124	1.222	0.268	0.238	0.255	0.288	0.296	0.033	0.147
LGBX	9.073	2.327	2.294	8.076		9.871	2.164	1.921	2.061	2.326	2.394	0.266	1.185
LSTMX	0.919	0.236	0.232	0.818	0.101		0.219	0.195	0.209	0.236	0.243	0.027	0.120
LevHARX	4.194	1.075	1.060	3.732	0.462	4.562		0.888	0.952	1.075	1.107	0.123	0.548
Log-HARX	4.723	1.211	1.194	4.204	0.521	5.139	1.126		1.073	1.211	1.246	0.138	0.617
RNNX	4.403	1.129	1.113	3.919	0.485	4.791	1.050	0.932		1.129	1.162	0.129	0.575
SHARX	3.900	1.000	0.986	3.471	0.430	4.243	0.930	0.826	0.886		1.029	0.114	0.509
TKANX	3.790	0.972	0.958	3.373	0.418	4.123	0.904	0.802	0.861	0.972		0.111	0.495
TabPFNX	34.142	8.755	8.631	30.387	3.763	37.144	8.141	7.228	7.753	8.754	9.009		4.460
XGBX	7.656	1.963	1.935	6.814	0.844	8.329	1.826	1.621	1.739	1.963	2.020	0.224	

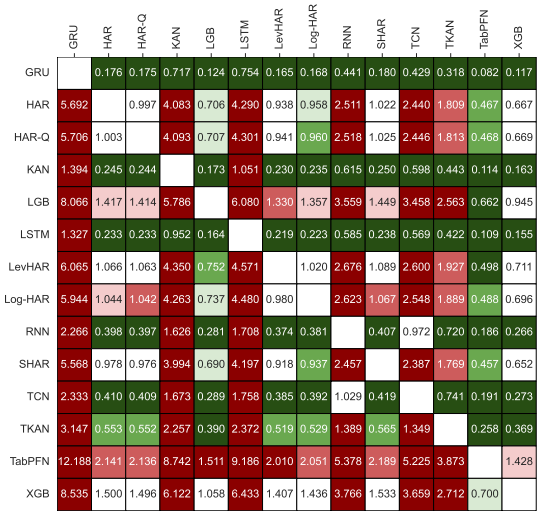
(c) Average month-ahead horizon

Figure 16: Pairwise equal predictive accuracy test results with Quantile Scores ($\tau = 0.95$), estimated on $\mathcal{D}_{HAR-X}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).

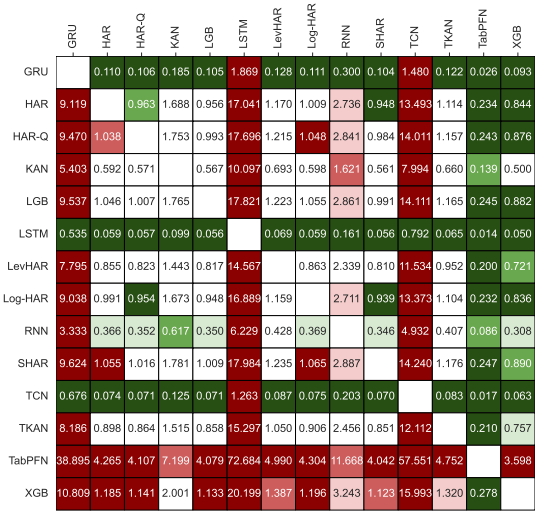
A.7.2 $\tau = 0.99$



(a) Day-ahead horizon

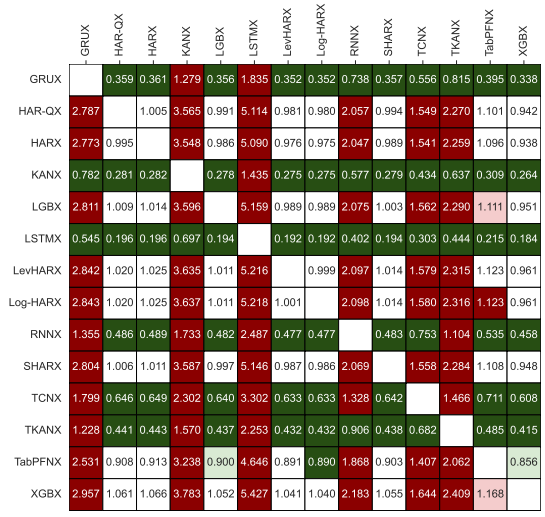


(b) Average week-ahead horizon

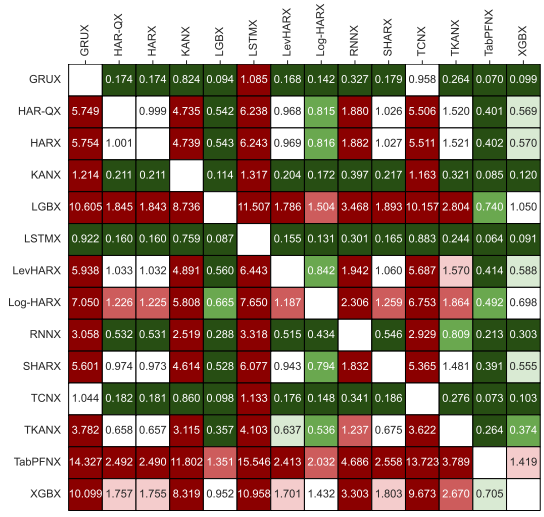


(c) Average month-ahead horizon

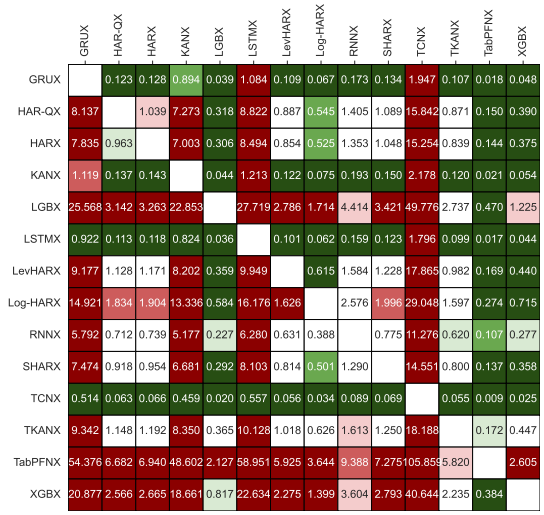
Figure 17: Pairwise equal predictive accuracy test results with Quantile Scores ($\tau = 0.99$), estimated on $\mathcal{D}_{HAR}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).



(a) Day-ahead horizon



(b) Average week-ahead horizon



(c) Average month-ahead horizon

Figure 18: Pairwise equal predictive accuracy test results with Quantile Scores ($\tau = 0.99$), estimated on $\mathcal{D}_{HAR-X}$; cell colours indicate statistical significance of pairwise forecast comparisons (light, medium, and dark shading corresponding to the 10%, 5%, and 1% levels, respectively, with green (red) denoting superior (inferior) performance of the column model relative to the row model).