Justus-Liebig-University Gießen

Faculty of Economics

Chair of Statistics and Econometrics

(VWL VII)

Prof. Dr. Peter Winker

Conventional and Newer Methods for Volatility Forecasting Compared: An Empirical Analysis

Master Thesis

presented by

Lenon Ferreira

Master of Science

Advisor:

Albina Latifi

Gießen, September 30, 2024

Table of Contents

List of Abbreviations

Adj. R.V.	Adjusted Realized Volatility
AIC	Akaike Information Criterion
AR	Autoregressive
ARCH	Autoregressive Conditional Heteroscedasticity
ARMA	Autoregressive Moving-Average
aSPA	Average Superior Predictive Ability
BCI	Bartlett confidence interval
BIC	Bayesian Informaiton Criterion
C.V.	cross-validation
DAX	Deutscher Aktienindex
DGP	data-generating process
EGARCH	Exponential Generalized Autoregressive Conditional Heteroscedasticity
EMH	Efficient Market Hypothesis
Fig.	Figure
GARCH	Generalized Autoregressive Conditional Heteroscedasticity
GED	Generalized Error Distribution
GJR-GARCH	Glosten-Jagannathn-Runkle Generalized Autoregressive Conditional Heteroscedasticity
HAR	Heterogeneous Autoregressive
i.i.d.	independently and identically distributed
IC	information criterion
J.B.	Jarque-Bera
LSTM	Long Short-Term Memory
MA	Moving Average
MAE	Mean Absolute Error

MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MSE	Mean Squared Error
MZ	Mincer-Zarnowitz
NYSE	New York Stock Exchange
RMSE	Root Mean Squared Error
S.R.	Squared Return
SP500	Standard and Poor's 500
SVM	Support Vector Machine
SVR	Support Vector Regression
Tab.	Table
uSPA	Uniform Superior Predictive Ability
VaR	Value at Risk
XGBoost	Extreme Gradient Boosting

List of Tables

List of Figures

1. Introduction

Volatility forecasting is a crucial task in finance, with significant implications for risk management, asset pricing, and portfolio optimization. Accurate forecasts of volatility enable investors, portfolio managers, and risk analysts to assess potential risks and optimize their asset allocations more reliably. Traditionally, econometric models like Generalized Autoregressive Conditional Heteroscedasticity (GARCH) and its extensions have dominated the field of volatility forecasting. These models are well-established in the literature, grounded in the stylized facts inherent to financial time series, and represent what we define as conventional methods.

Machine Learning (ML) models have become increasingly widespread in recent years. These data-driven techniques are flexible and capable of capturing complex non-linear relationships that traditional econometric models cannot, and they constitute our newer approach for volatility forecasting. Therefore, a key question arises: can machine learning methods outperform traditional econometric models in forecasting volatility, particularly over longer horizons?

This thesis aims to empirically investigate this question by comparing the conventional econometric models, such as GARCH, Exponential Generalized Autoregressive Conditional Heteroscedasticity (EGARCH) and Glosten-Jagannathn-Runkle Generalized Autoregressive Conditional Heteroscedasticity (GJR-GARCH), with ML models Support Vector Regression (SVR), Extreme Gradient Boosting (XGBoost) and Long Short-Term Memory (LSTM). We expect to contribute to the literature by evaluating the performance of these models on forecasting return volatility for a horizon of 5 days for two large equity indices, namely the Deutscher Aktienindex (DAX) and Standard and Poor's 500 (SP500). We use three different evaluation measures: Mean Squared Error (MSE), Mincer-Zarnowitz (MZ)-Regression R^2_{MZ}, and a financial application based on Value at Risk (VaR). Our approach is holistic, focusing on the whole forecasting horizon instead of only the h-step ahead forecast. Additionally, we use two types of volatility proxies—the Squared Return (S.R.) computed from a single daily observation and the Adjusted Realized Volatility (Adj. R.V.) based on high-frequency, intraday data. This approach enables us to investigate how the models perform under different volatility definitions, since true volatility cannot be directly observed. To avoid confusion, by volatility we mean the variance, if not otherwise stated.

Our study begins with a discussion of the most important stylized facts observed in financial time series, with a particular focus on volatility. These stylized facts provide the foundation for the models whose performance will be analyzed in the later sections. Section 3 presents a synthesis of the existing body of research on volatility forecasting, forming the literature review.

Data description, the computation of the volatility proxies and evaluation methods are presented throughout Section 4. Sections 5 and 6 present the econometric models and ML, respectively. Each one of these sections contains the estimation procedure and evaluation of the models in isolation, meaning that econometric models (GARCH-family) are compared only with econometric models, and ML models are compared only with ML models. Section 7 compares ML with econometric models and presents our financial application results. Section 8 provides the conclusion.

We find that the best method for volatility forecasting is based on the forecaster's goal. In some of our evaluation metrics, ML performed better than the standard econometric models, in particular at minimizing the average MSE of the Adj. R.V. forecasts. However, for all other purposes, the GARCH-family models tended to have an advantage over the data-driven methods.

2. Financial Time Series: Stylized Facts

The underlying section presents and discusses some of the key stylized facts about financial time series, with particular emphasis on the volatility of asset returns. Where applicable, we provide representations using data from the two indices included in our analysis: the DAX and SP500. Stylized facts are empirical regularities observed in the social sciences that are generally accepted as true, although they still require full theoretical explanation. While these regularities do not constitute complete explanatory theories, they help social scientists identify consistent trends and patterns in the data. Once established, these pattern lay the groundwork for the development of theories and models (Hirschman 2016: 605-606; Sewell 2011: 2).

A good volatility model must, first and foremost, be able to accurately forecast volatility. Since the statistical modeling of time series relies on repetitive patterns in the data generated by some data-generating process (DGP), any model attempting to predict volatility needs to consider and reproduce at least some, if not all, of these regularities

Fig. 1: Correlogram: Returns and Squared Returns

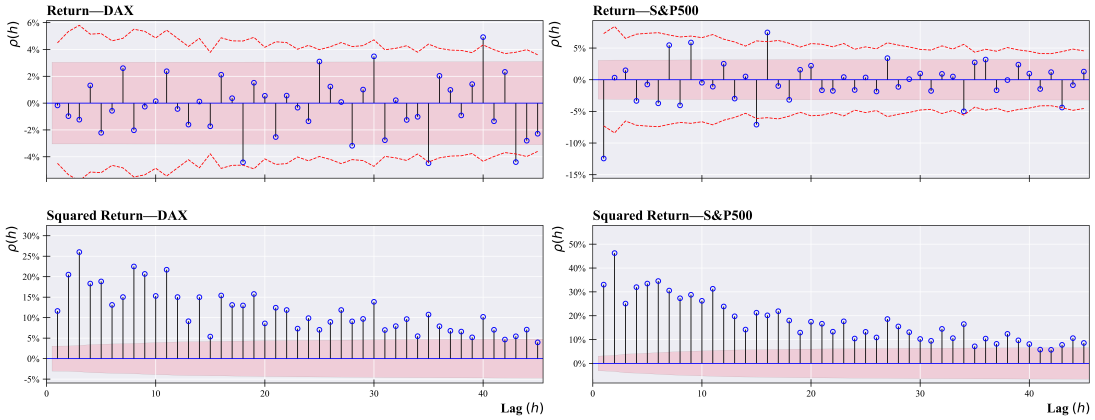


Figure illustrates the autocorrelation of the return (upper row) and squared return (bottom row) for the DAX (first column) and SP500 (second column) for a total of 45 lags.

(Engle and Patton 2001: 237-238).

Our first stylized fact about financial time series partially contradicts the Efficient Market Hypothesis (EMH). According to the EMH, security prices fully reflect all available information (Fama 1970: 383), implying that future price behavior cannot be predicted using historical data. Future prices should be independent of past prices, and the best estimate of a security's price at time $t+1$, given the information set ψ_t at time t, is simply the price at t. This behavior is modeled by a random walk:

$$\begin{aligned} p_t &= p_{t-1} + \varepsilon_t \\ \varepsilon_t &\overset{i.i.d.}{\sim} Q\left(0, \sigma^2\right) \end{aligned} \tag{2.1}$$

Taking expectations shows that the series has no memory, and if p_t represents the log-price of a security, and ε_t is a white noise error term, then the return on this security $r_t = \Delta p_t = \varepsilon_t$ is random itself. However, studies have documented some degree of serial correlation in asset returns (cf. Fama (1970), Jegadeesh (1990), Gaunt and Gray (2003), and Martin (2021)), leading to our first stylized fact: returns are weakly dependent. However, as Fama (1965: 56) states, the importance a researcher should give to this matter is a monotonic function of the amount of dependence found in the data.

Figure (Fig.) 1 presents a correlogram that illustrates the autocorrelation $\rho(h)$ between observations r_t^k and r_{t+h}^k for both the return ($k = 1$) and squared return ($k = 2$) series of the DAX and SP500 indices, over a range of 45 days (lags). The analysis indicates

that the return series generally exhibit low autocorrelation. Specifically, for the DAX, the highest autocorrelation is observed at the 40-th lag, with a value of 4.92%. For the S&P500, the highest autocorrelation occurs at the first lag, with an approximate value of 11% in absolute terms. The red-shaded area represents the Bartlett confidence interval (BCI), where autocorrelations within this range are not significantly different from zero. The dashed red lines represent an adjustment of the confidence interval to account for a weak white noise process, as described by Francq and Zakoian (2019: 127). Most observations for both the DAX and S&P500 fall within the corrected band, but many exceed the BCI, supporting the stylized fact that while returns exhibit weak (if any) correlation, they are not independent.

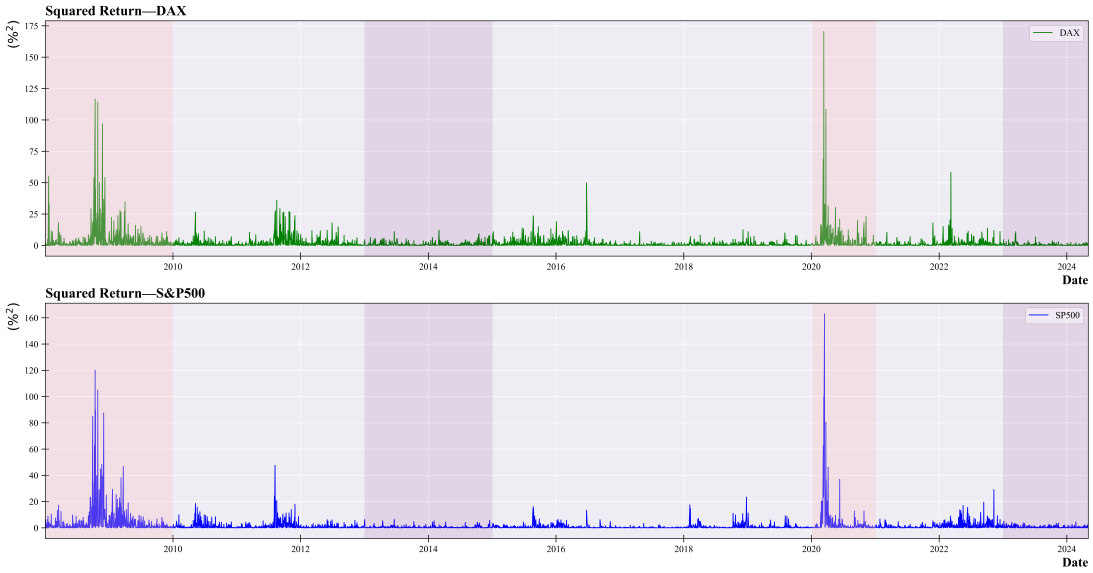
In this study, as is common in the literature, S.R. are used as one of the volatility proxies. From this point forward, our focus will be on the stylized facts concerning volatility, the primary subject of this study. The autocorrelation in S.R. is notably higher for both indices. For the DAX, the largest autocorrelation reaches approximately 26% at the third lag, more than five times the maximum value observed in the return series. Moreover, 43 out of the 45 autocorrelations exceed 4.92%, and all but one are statistically significant with 95% confidence according to the BCI.

Similarly, for the S&P500, the highest autocorrelation in S.R. is 46.27% at the first lag, with 42 out of 45 autocorrelations exceeding the maximum value observed in the return series, and only two of them lie inside the 95% confidence band, i.e., are not statistically significant different from zero.

The correlogram provides valuable information about the behavior of asset return volatility. The high autocorrelation in S.R. indicates that a large (or small) value of volatility, i.e., a large (or small) price fluctuation, is followed by another large (or small) value of volatility. This behavior, the second stylized fact, is known as *volatility clustering*, first documented by Mandelbrot (1963: 418) in his study of price behavior in speculative markets. Further, as the slow decay in the autocorrelation indicates, volatility has *long memory*, i.e., its value today have an impact on its value many periods in the future. In summary, volatility fluctuates between high and low levels over time, but in general, it will return to its long-run mean, property known as *mean-reverting* and is our third stylized fact (Engle and Patton 2001: 6-7).

Fig. 2 illustrates the concept of volatility clustering using the S.R. series. The pink-shaded areas highlight two periods of very high volatility in the financial markets: first,

Fig. 2: Volatility Clustering: Example



The figure illustrates the squared returns (in $\%^2$) for the DAX (top panel) and SP500 (bottom panel) over the period from 2 January 2008 to 3 May 2024. The shaded areas represent different volatility clustering regimes, with the pink areas indicating periods of high volatility, and the purple areas representing periods of low volatility.

following the financial crisis of 2008, and second, during the onset of the COVID-19 pandemic in 2020. The purple-shaded areas highlight periods of relative market stability, when volatility was low for both indices. Other instances of high volatility evident from the chart include the Euro-Area debt crisis 2010–2012 and the beginning of 2022, following the invasion of Ukraine by the Russian Federation.

The fourth systematic behavior observed in financial time series is the asymmetric response of stock market volatility to price changes of equal magnitude but opposite signs. Specifically, there is a tendency for volatility to increase more following a negative return than after a positive return of the same magnitude. This phenomenon is commonly referred to as the *leverage effect*, a term first introduced by Christie (1982). The leverage effect posits that when a company's stock price declines, the equity value of the firm decreases, leading to an increase (ceteris paribus) in the firm's debt-to-equity ratio. This higher leverage makes the company's security riskier for shareholders, which is reflected in the volatility of asset returns.

Finally, financial theory often assumes that stock (log) returns follow a normal distribution, as seen in the Black-Scholes method for option pricing Black and Scholes (1973: 640-645) and the standard parametric approach for calculating Value-at-Risk (Morgan et al. 1996: 6). However, the flaws in this assumption cannot be overstated. The dis-

tribution of asset returns is known to be *leptokurtic*, meaning it exhibits fat tails—tails fatter than those of a normal distribution. This implies that outliers, or price crashes in financial jargon, are much more likely than a normal distribution would suggest.

For instance, in a standard normal distribution, the probability of observing a value more than 3 standard deviations (or z-scores) from the mean is approximately 0.27%. However, for our data, about 1.52% of the DAX and 1.82% of the S&P500 standardized returns exceed this threshold[1]. While these percentages may seem small, they represent a much higher probability of extreme events. Specifically, using a normal distribution would underestimate the likelihood of extreme events by factors of approximately 5.6 for the DAX and 6.7 for the S&P500.

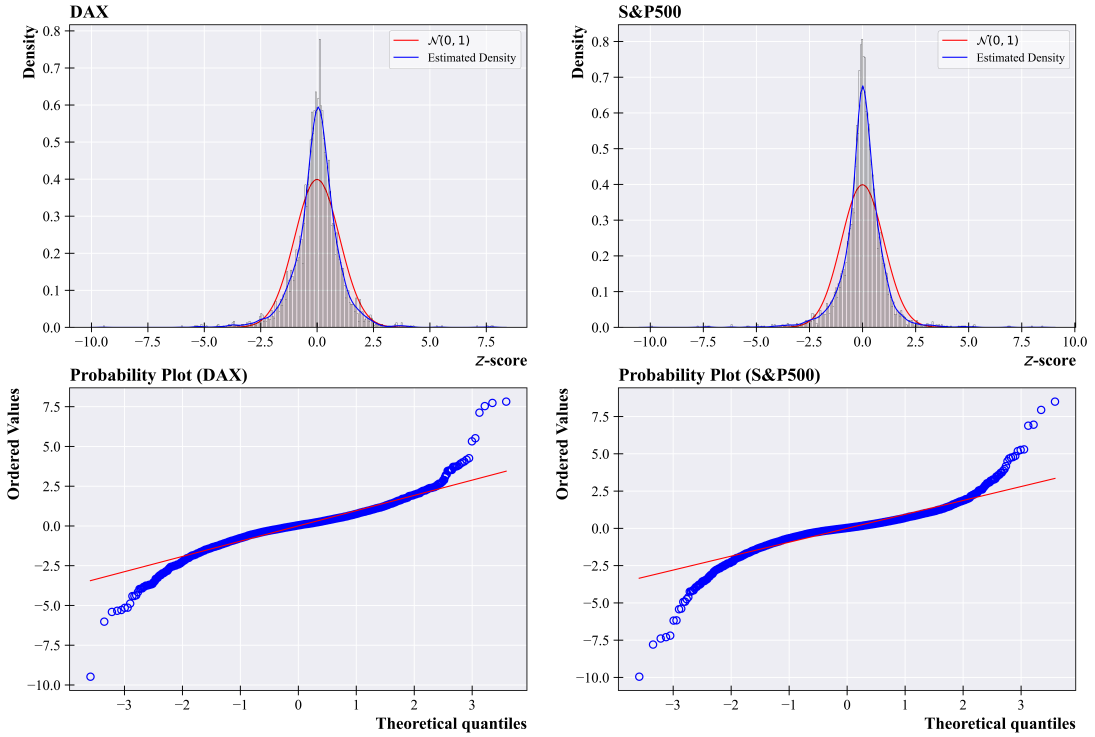
Fig. 3 (top row) presents a histogram of standardized daily returns for the DAX (left) and the S&P500 (right), along with their estimated kernel density (in blue) compared to the standard normal distribution (in red). As shown, the returns are more concentrated around zero, but the tails of the distribution carry substantially more weight than those of a standard normal distribution. The bottom row illustrate a Q-Q plot, with the quantiles of the empirical distribution (y–axis) being plotted against those of the standard normal distribution (x-axis). If the empirical distribution of our data perfectly matched the theoretical normal distribution, all the scatter points would lie on the 45-degree red line.

In the case of both indices, we observe significant deviations from the red line, particularly in the tails. While the points in the middle of the distribution aligns fairly well with the red line, indicating that the central point of the returns is approximately normal. However, the extreme quantiles at both ends deviate substantially, indicating fat tails of the distribution in both sides.

In summary, an optimal volatility model should capture the regularities described above. Its short-term forecasts should account for dependencies, particularly long memory, and heteroscedasticity in the variance (volatility clustering), while its long-term forecasts should converge to the long-run variance, reflecting the mean-reverting nature of volatility. Additionally, the model should consider the potential asymmetric response of volatility to innovations of different signs, as well as the leptokurtosis observed in return distributions, ensuring that extreme values are properly accounted for.

[1]The standardization was done using the sample daily variance: $\widehat{s} = \frac{1}{T-1}\sum_{t=1}^{T}[r_t - \bar{r}_t]^2$

Fig. 3: Density and Q-Q Plots of Standardized Returns



The figure illustrates the empirical distribution (top row) and Q-Q plot (bottom row) of standardized daily returns for the DAX (first column) and SP500 (second column). The top row compares the estimated density (blue) to the standard normal distribution (red), while the bottom row shows the Q-Q plot, where the empirical quantiles are plotted against the theoretical normal quantiles.

In the next section, we will review traditional statistical models specifically designed to address these regularities in volatility time series, as well as newer, data-driven approaches using machine learning and deep learning algorithms, which can be applied to more general settings. As the reader will gain a deeper understanding in Sections 5 and 6, no single model is universally applicable, but, at least from a theoretical perspective, some are more successful than others at replicating the stylized facts discussed so far.

3. Literature Review

Econometrics has traditionally prioritized interpretability and theory testing, focusing on models grounded in statistical theory. These models rely on the consistency and efficiency of estimators and their large-sample properties, such as those derived from the central limit theorem and law of large numbers, with confidence intervals helping to quantify estimator risk. The Ordinary Least Squares, the workhorse of econometrics,

is known to be best linear unbiased estimator when the Gauss-Markov assumptions are satisfied (Athey and Imbens 2019: 1-6). These assumptions have been adapted for time series analysis, particularly by replacing the assumption of independently and identically distributed (i.i.d.) data, with the one of stationarity. Strict stationarity requires that the statistical properties of the series remain unchanged under time shifts, and is rare in real-world problems; instead, second-order stationarity is more commonly assumed, requiring the random variable to have constant first and second moments, and auto-covariance depending only on the distance between the observations but not on time (Francq and Zakoian 2019: 2).

A challenge econometric models face is their high reliance on linearity. For example the Autoregressive Conditional Heteroscedasticity (ARCH) model for volatility, assumes a linear relationship between the current value of volatility and the past squared errors. This assumption of linearity can lead to poor performance if it is not satisfied. While econometric models are valuable for their tractability, they may be limited in capturing nonlinear relationships in the data (James et al. 2023: 23-25).

The former vice-president of the European Central Bank, Vítor Constâncio, noted in a 2014 speech that linear models serve as a good approximation of many phenomena in the economy as a whole. However, recent crises have shown that non-linear models are crucial for an accurate reflection of the interactions in the financial market. ML algorithms, such as regression trees, support vector machines and artificial neural networks, to cite a few, offer promising alternatives to econometric models due to their flexibility in capturing complex, nonlinear relationships in the data (cf. Masini, Medeiros, and Mendes (2023: 86-97)), since these models make few or even no assumptions about the DGP (Athey and Imbens 2019: 1-6).

Empirical evidence support the growing importance given to ML-algorithms in the financial domain. In a review of 57 studies for financial time series prediction, Henrique, Sobreiro, and Kimura (2019: 244) found that some kind of neural networks was used in 70% of the articles, followed by Support Vector Machine (and Regression) in approximately 37% of the studies. ML algorithms have demonstrated significant success in volatility forecasting. For example, Liu (2019) used a modified version of SVR and a Long Short-Term Memory (LSTM) network to forecast the variance of the S&P500 and Apple stock returns, finding that both models outperformed the conventional GARCH$(1, 1)$ model in out-of-sample performance. Similarly, Peng et

al. (2018) compared an SVR model with a GARCH structure against the traditional GARCH, EGARCH, and GJR-GARCH. In their study of one-step-ahead forecasts for cryptocurrency and exchange rate volatilities, the SVR outperformed the conventional models in 319 out of 420 cases. Additionally, Chen, Härdle, and Jeong (2010) compared Artificial Neural Networks and SVR with a GARCH structure to GARCH(1, 1), EGARCH(1, 1) and Moving Average models. Overall the SVR model was ranked the best model at predicting the volatility in the exchange rate considered by the authors, and was slightly worse than the EGARCH(1, 1) at predicting the New York Stock Exchange Index volatility.

Masini, Medeiros, and Mendes (2023) further extended the body of research by comparing machine learning algorithms with the Heterogeneous Autoregressive (HAR) model for Realized Volatility, proposed by Corsi (2009). Focusing on forecasting the one-step-ahead volatility of the constituents of the Dow Jones Industrial Average Index, they found that the Random Forest algorithm, combined with external covariates, led to a 9.4% reduction in out-of-sample MSE compared to the HAR model. However, such improvement was not observed with the Gradient Boosting algorithm, and in some instances, the out-of-sample MSE for tree based methods was higher than when forecasts were generated by the HAR model (cf. Masini, Medeiros, and Mendes (2023: 11)).

These studies suggest that incorporating ML algorithms into the forecaster's toolkit can yield improved predictions over standard econometric approaches. However, there is no guarantee that such improvements will always occur, and ML models often come with their own challenges, such as interpretability, overfitting and parameter-tuning. Moreover, the majority of the studies cited above have focused solely on one-step-ahead forecasts. Our study extends the literature by evaluating the models' ability to forecast volatility over a longer horizon, specifically a full trading week (i.e., 5 days), and by considering the XGBoost among our set of models, since literature implementing tree-based algorithm for financial market prediction is scarce (Henrique, Sobreiro, and Kimura 2019: 244). In the next section, we present the data used in our study, describe the construction of our volatility proxies, and outline our evaluation methods.

4. Data Description and Evaluation Methods

4.1. Data Source and Description

The data used in this study comprises intraday prices at 5-minute intervals and daily frequency for the S&P500 and DAX, acquired from First-Rate. Both indices represent the largest and most liquid equities in their respective markets—the United States of America (U.S.) and Germany, respectively. Together, the 500 constituents of the S&P500 account for approximately 80% of the market capitalization of all publicly listed companies in the U.S. According to the S&P Global, the index is often regarded as the best single measure of the largest U.S. equities. Similarly, the DAX, which was originally created in 1998 with 30 companies, and expanded to 40 in September 2021, represents around 80% of the market capitalization of publicly listed companies in Germany.

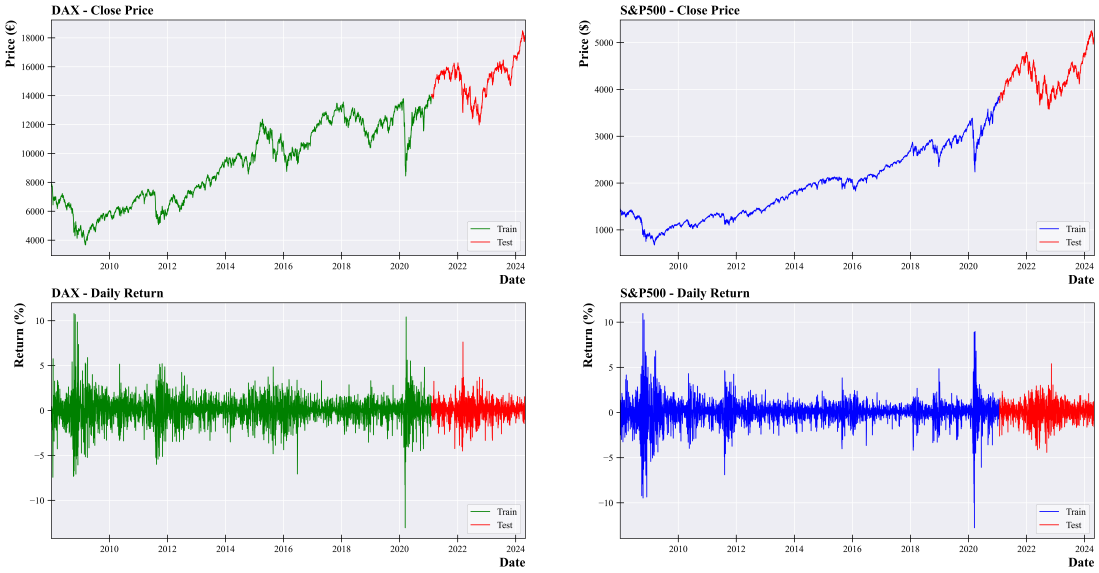
The analysis period spans from the first trading day of 2008 to May 3, 2024, covering over 17 years. This includes a total of 4,149 daily observations for the DAX and 4,113 for the S&P500. For model training and evaluation, the series are split into an 80% training and 20% test set. For the DAX, the test set begins on February 4, 2021, containing 830 observations, while for the S&P500, the test set begins on January 27, 2021, with 823 observations. However, as explained in the following sections, the last five days of the training set are used for model validation. Consequently, these days are included in the test set to evaluate the final performance. The returns used to compute volatility are based on continuously compounded returns. If an entire day t of price realizations is divided into m periods, the return for the i-th period is given by

$$r_{t,i,m} = \left[\log\left(p_{t-\frac{i-1}{m}}\right) - \log\left(p_{t-\frac{i}{m}}\right)\right] * 100. \tag{4.1}$$

We computed the daily return directly from the series at daily frequency, for reasons that will become clear in the next section. Fig. 4 illustrates the evolution of the price and return series. For both indices, the price series are clearly not stationary, but the return fluctuates around a mean of 0.02% for the DAX and 0.03% for the S&P500.

Table (Tab.) 1 presents descriptive statistics for the return series at daily frequency divided by set type as well as for the entire sample. Considering the entire sample, the returns for both indices exhibit slight left skewness, indicating that negative returns are

Fig. 4: Historical Price and Return Series for DAX and S&P500



The figure displays the historical closing prices (top) and daily returns (bottom) for the DAX (left) and SP500 (right) indices from 2 January 2009 until 3 May 2024. In each plot, the data is split into training and test sets (red), with the training data covering 20% of the observations.

more frequent than positive ones. The sample variance $s\,[r_t]$, and the mean value of the S.R. are practically the same, $1.90\%^2$ for the DAX and $1.65\%^2$ for the S&P500. The Adj. R.V., our second volatility proxy, has a mean of $1.78\%^2$ for the DAX and $1.25\%^2$ for the S&P500, both of which are lower than the variances computed from the S.R.. The Jarque-Bera (J.B.)-test for normality rejects the null hypothesis of normality with a p-value of practically zero for both indices. Additionally, a test for conditional heteroscedasticity applied directly to the S.R. rejects the hypothesis of constant variance for both series. Further, we observe a difference in the kurtosis between the test and training sets; in particular the S&P500 returns move from a period of leptokurtic distribution to a period of thin-tails. For both indices, the mean-value of the volatility proxies in the training set is larger than in the test set. These changes in the distribution can pose challenge to our models.

In the next section, we will turn our attention to how the volatility proxies used is this study are constructed, and address the reason why using the S.R. is not optimal as a proxy of volatility.

4.2. Creation of Volatility Proxies

Volatility, as a latent variable, cannot be directly observed, making the choice of a suitable proxy crucial for robust model evaluation and comparison. As Patton (2011: 246) emphasizes, an unbiased estimator does not necessarily "[...] lead to the same outcome as if the true latent variable were used." Using a noisy estimator, such as daily squared returns, alongside non-robust loss functions, can result in inconsistent model rankings, leading researchers to erroneously select suboptimal models due to the inefficiency of the volatility proxy.

Andersen and Bollerslev (1998: 886) note that the poor performance found in many of the studies in out-of-sample accuracy of standard econometric volatility models is not due to the models themselves, but rather the use of a volatility proxy based on only one observation per day. A more efficient proxy is the realized volatility, computed from high-frequency, intra-day, returns, given as

$$RV_t = \sum_{i=1}^{I} r_{t,i,m}^2 \tag{4.2}$$

One might assume that using the highest possible frequency, such as one-minute returns, would yield the best proxy for volatility. However, frequency below 5-minutes interval are known to introduce noise from microstructure effects, such as spurious correlation created by bid-ask spreads. On the other hand, the use of low frequency data risks omitting important price movements, leading to an incomplete reflection of the security's true price path (Hu et al. 2020: 3). Following the literature, we choose to use prices at 5-minutes frequency (i.e., $m = 288$).

A major distinction between our study and the work of Andersen and Bollerslev (1998) lies in the type of asset studied. While their research focuses on exchange rates, which are traded continuously across global markets, our study is centered on equities. Equity markets, such as the New York Stock Exchange (NYSE) for the S&P500, and XETRA for the DAX, have limited trading hours, restricting observations to specific time windows. To account for the price changes during non-trading hours, i.e., the overnight volatility, we follow the approach suggested by Martens (2002: 501) and Hol and Koopman (2002: 4-5), adjusting the *observed* realized volatility $\left(RV_t^D\right)$ by a

constant scaling factor:

$$_{adj.}RV_t = (1+c)\,RV_t^D, \tag{4.3}$$

$$\text{with } c = \frac{\sigma_{co}^2}{\sigma_{oc}^2} \tag{4.4}$$

where σ_{co}^2 represents the long-run close-to-open volatility, computed as the mean value of the overnight squared returns, and σ_{co}^2 is the long-run open-to-close volatility, i.e., the mean value of RV_t^D, both using only training-set observations.

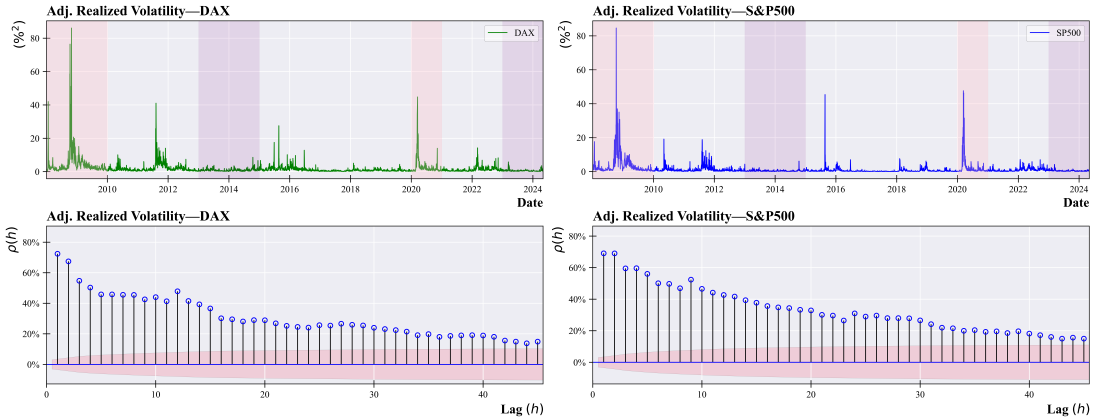
Under regular market conditions, the NYSE operates Monday through Friday from 9:30 a.m. to 4:00 p.m. providing 78 intra-day price realizations, while XETRA operates from 9:00 a.m. to 5 p.m., offering 102 intra-day price realizations.

To calculate the return for the first 5-min interval of the day, when no previous closing price is available, we compute the difference between the log-opening and -closing prices of the first interval. As detailed in Tab. 2, the total number of intraday observations in the First-Rate dataset exceed the official number of price realizations due to recorded timestamps outside the trading hours. To ensure consistency, we constrained the dataset to only include observations that occurred within the official trading hours, so that the total number of intraday returns for the DAX equals 102 and 78 for the S&P500. In cases where observations were missing before (or after) the official trading hours—such as early or late data–we forward- or back-filled the nearest available realization.

Our estimates for the close-to-open variance are $\sigma_{co,DAX}^2 = 0.6141\%^2$ and $\sigma_{co,S\&P}^2 = 0.1486\%^2$, respectively. The estimates for the open-to-close volatilities are $\sigma_{oc,DAX}^2 = 1.381\%^2$ and $\sigma_{oc,S\&P500}^2 = 1.215\%^2$. Consequently the scaling factor c is 1.445 for the DAX and 1.122 for the S&P500. This adjusted version of the realized volatility will be referred to as the Adj. Realized Volatility $\left(_{adj.}RV\right)$, and is along with the daily S.R., our second daily volatility proxy. Due to the inconsistency in the total number of intraday observations, we chose to calculate the S.R. using the daily series, which also serve as input to the econometric models explained through section 5.

Fig. 5 illustrates the Adj. R.V. series (top row) for the DAX (left) and S&P500 (right) along with their respective correlogram (bottom row). Similar to the S.R.s, the Adj. R.V.s exhibit a high degree of autocorrelation, with an even slower rate of decay. The shaded areas in the volatility series correspond to those shown for the S.R. in Fig. 2. Interest-

Fig. 5: Adjusted Realized Volatility and Correlogram



The figure presents the adjusted realized volatility (top panel) and its respective correlogram (bottom panel) for the DAX (left column) and the SP500 (right column).

ingly, as shown in Tab. 3 , the correlation between the S.R. and Adj. R.V. is not that strong for any of the indices. For the DAX, the correlation is 51.77%, while for the S&P500, it is 59.75%. If the Adj. R.V.s are indeed capturing more information about the true path of volatility, then the amount of noise in the S.R.s is substantial. For both indices, we also observe a negative correlation between returns and volatility proxies, supporting the leverage effect discussed in Section 2.

Having established the key features of our data and the construction of our volatility proxies, we now turn to the valuation methods used in our analysis. We begin by explaining the concept of cross-validation, detailing its purpose and how it is implemented in this study.

4.3. Evaluation Methods

4.3.1. Time Series Cross Validation

A wide range of algorithms is available to researchers aiming to forecast or explain a variable, including methods like Least Squares, Maximum Likelihood, and Empirical Risk minimization. The process of choosing the optimal model among a set of candidates is known as model selection, which should be done with careful consideration of the research goal, as it determines the entire modeling process (Arlot and Celisse 2018: 3). Shmueli (2010: 291-293) identifies three main goals for modeling: prediction, explanation, and description (not covered in this study). In an explanatory analysis, the model is based on theory, and the goal is to obtain an unbiased estimate

of the functional form $y_i = f(\mathbf{x}_i) + \upsilon_i$, relating covariates $\mathbf{x}_i$ to the target variable y_i in a causal sense, such that theoretical soundness and interpretability are prioritized. In contrast, as is the case in our study, predictive modeling focuses on estimating y as accurately as possible, where the model serves as a "means to an end".

To evaluate the performance of a model, several loss functions are available, which quantify the discrepancy between predictions and actual values. For a quadratic loss function, the expected prediction error can be decomposed as:

$$\mathbb{E}\left[\left\{y_i - \widehat{f}(\mathbf{x}_i)\right\}^2\right] = \mathbb{V}\left[\widehat{f}(\mathbf{x}_i)\right] + \mathsf{bias}^2 + \mathbb{V}\left[y_i\right], \tag{4.5}$$

where $\mathbb{V}[y_i]$ is the irreducible error due to random noise in the data, and $\mathbb{V}\left[\widehat{f}(\mathbf{x}_i)\right]$, represents the variance of the estimator. This equation illustrates the well-known bias-variance tradeoff. A model that is too simple may underfit the data, failing to capture the true relationship between the $\mathbf{x}_i$ and y_i. On the other hand, a model that is too complex may overfit, capturing patterns specific to the training samples but failing to generalize to new, unseen data, resulting in high variance (Strang 2019: 371; Arlot and Celisse 2018: 6-7). While in explanatory analysis the goal is to minimize the bias, in predictive tasks the objective is to achieve the best balance between the bias and variance.

Cross-validation is a widely used method to achieve the optimal bias-variance trade-off. It involves partitioning the data into k mutually exclusive folds, with one fold used for validation and the remaining $k - 1$ used for model training. The performance metrics from each of the k iterations are averaged to provide an overall estimate of the model's performance. However, the k-fold cross-validation assumes that the data is i.i.d. (Arlot and Celisse 2018: 1), which does not hold for times series, where past values influence the future. To account for this temporal dependency, a more suitable approach is cross-validation with a rolling forecast origin. In this method (cf. Fig. 6), a model is iteratively re-estimated on a expanding training set. The process starts with a reasonable number of observations to estimate the model and generate reliable forecasts. In each iteration, the first data point from the validation set of the previous iteration is added to the training data, the model is re-estimated, and a new set of forecasts is produced and evaluated. The model's overall performance is calculated as the average of all individual performances, ensuring that it is assessed on its ability to

Fig. 6: Cross-Validation with a Rolling Forecasting Origin

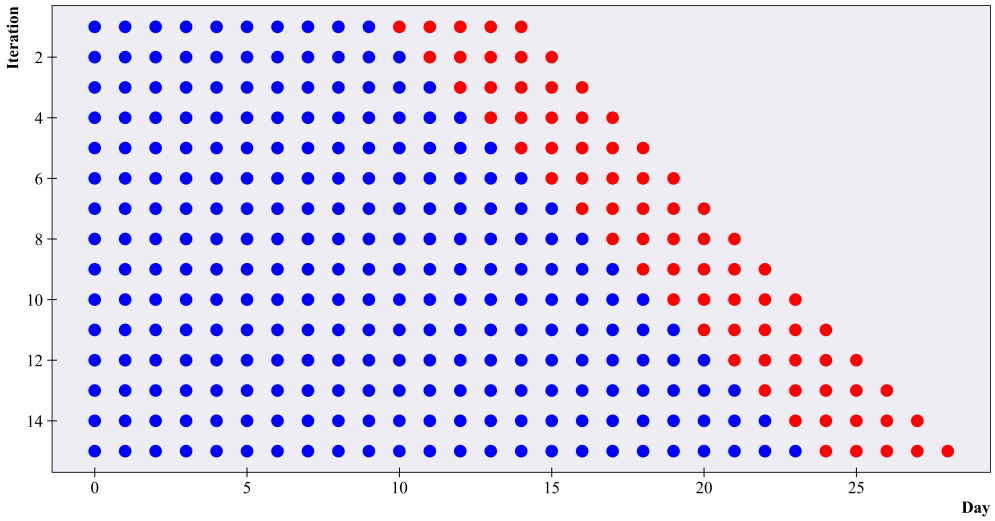


Figure based on representation of Hyndman and Athanasopoulos (2021: Chapter 5.10). The figure illustrates the concept of cross-validation with a rolling forecast origin for time series data. The blue dots represent the training set, and the red dots denote the target values at each iteration. As time progresses (x-axis), the training window expands, while the target values move forward. Figure illustrates the target values for a total of five days.

predict the future and that the origin of the data does not affect the model valuation (Tashman 2000: 440-441).

Common loss-functions used with cross-validation for regression problems include the Mean Absolute Error (MAE), which measures the average absolute difference between the target and forecast; the Mean Absolute Percentage Error (MAPE), a scale-free accuracy measure; and the MSE, which calculates the mean squared deviation between the true value and its estimate. Each loss function has distinctive advantages and limitations. As Hyndman and Koehler (2006: 682) and Hyndman and Athanasopoulos (2021: Chapter 5.8) note, minimizing the (Root) MSE yields forecasts of the mean of the target variable, and this loss function is quite sensitive to outliers due to its quadratic nature. Mincer and Zarnowitz (1969: 8) argue that such property is desirable for economic forecasting; and supporting the view of the authors, we argue that the MSE is particular appropriate when the target variable is the volatility, since volatility is used as a measure of risk. Consequently, in our study, we choose the MSE as the primary, though not exclusive, metric for model comparison.

For all models, but the LSTM, we implement cross-validation with a rolling forecast origin, using an initial training set of 250 daily observations (approximately one trading year). For each subsequent training sample, we forecast volatility over a 5-day

horizon. Our evaluation approach is holistic, meaning that while we may comment on the performance of a model at predicting the h-step-ahead volatility, we select models based on the lowest avg. MSE ($\overline{MSE}$) across all 5 forecasts.

Each forecast error is equally weighted (i.e., $1/5$), allowing a poor forecast at one step to be offset by better performance at another step. For the LSTM model, as will be discussed in Section 6, we use a train-validation split of the training set to train the models. Mathematically, our loss function is expressed as:

$$L_t = \frac{1}{5}\sum_{h=1}^{5}\left[(y_t - \widehat{y}_{t+h})^2\right] \tag{4.6}$$

$$\overline{MSE} = \frac{1}{(T^* - h)}\sum_{t=250}^{T^*-h} L_t, \tag{4.7}$$

where T^* represents the total number of observations in the training set.

In the following section, we introduce two additional performance evaluation metrics. First, we present the Average Superior Predictive Ability (aSPA) test, which allows us to statistically assess whether a model with a lower $\overline{MSE}$ is with a certain confidence superior to a competing model or whether the observed difference in accuracy is due to random chance. Second, we discuss the MZ regression, which provides a means to evaluate both the bias and forecastability of our models.

4.3.2. The aSPA-Test and the MZ-Regression

In the previous section, we introduced the concept of cross-validation and presented our loss function. While losses like the MSE, MAE, and MAPE are widely used for comparing forecast accuracy, they are all based on point forecasts and do not account for sampling uncertainty. It is possible, for example, that one model achieves a lower $\overline{MSE}$ than another only due to random variation.

To account for sampling uncertainty, Diebold and Mariano (2002) introduced a test statistic that allows researchers to compare the predictive accuracy of models under very general conditions. Specifically, consider the task of comparing the ability of models i and j to forecast the h-step ahead volatility, denoted σ^2_{t+h}. The forecast error for each model is given by:

$$\upsilon_{k,t+h} = \sigma^2_{t+h} - \widehat{\sigma}^2_{k,t+h}, \tag{4.8}$$

where $k = i, j$, and $\widehat{\sigma}_{k,t+h}$ is the volatility forecast produced by model k. Let $g(\upsilon_{k,t+h})$ be a loss function applied to the forecast of model k. The loss differential between the two models is then defined as

$$d^h_{ij,t} \equiv g(\upsilon_{i,t+h}) - g\left(\upsilon_{j,t+h}\right). \tag{4.9}$$

To test whether the losses of models i and j belong to the same population, we can test the null hypothesis that $\mathbb{E}\left[d^h_{ij,t}\right] = 0$ by means of a t-test:

$$t_{DM} = \sqrt{T}\frac{\bar{d}^h_{ij}}{\widehat{\sigma}\left(\bar{d}^h_{ij}\right)}, \tag{4.10}$$

where $\bar{d}^h_{ij}$ is the sample mean of loss differential, T is the sample size, and $\widehat{\sigma}\left(\bar{d}^h_{ij}\right)$ is the heteroscedasticity and autocorrelation consistent (HAC) estimate of the standard deviation of the loss differential. For large samples, the t_{DM} statistic converges to the standard normal distribution.

The DM test was originally developed for evaluating forecasting accuracy for a series of single h-step ahead predictions. Quaedvlieg (2021) extended this idea to multi-horizon forecasts. Specifically, the author proposed two tests: one for Uniform Superior Predictive Ability (uSPA) and the other for aSPA. The former tests the hypothesis that the forecast ability of a model is superior over the entire forecast horizon, while the latter evaluates the average superiority of the model over the multiple steps. In this work, we apply the aSPA test, giving equally weights to the loss differentials over the horizon, allowing poor performance in one step to be offset by better performance in other steps. This aligns with our calculation of the $\overline{MSE}$. The choice of weights is at the discretion of the forecaster and depends on their specific utility function.

In essence, the aSPA test is a Diebold-Mariano test applied to the weighted loss differential. However, in recognition of Quaedvlieg (2021) contribution, we refer to it as the aSPA. Following the author's methodology, we perform a right-tailed hypothesis test; as the weighting scheme is already embedded in our loss function L_t (cf. Eq. 4.6), we

test the null hypothesis:

$$H_0 : \overline{MSE}_i - \overline{MSE}_j \leq 0 \quad \text{vs.} \quad H_1 : \overline{MSE}_i - \overline{MSE}_j > 0 \tag{4.11}$$

If we fail to reject the null hypothesis, the forecasts produced by model i are at least as good as those produced by model j. If we reject the null, model j is considered superior in terms of average predictive performance accuracy at the chosen confidence level. For small samples, the author proposes a bootstrap approach to approximate the distribution of the test statistic. However, given our large sample size (over $3,000$ observations in the training set and 800 in the test set), we rely on the large-sample properties of the test. To compute the sample variance $\widehat{\sigma}^2\left(\overline{MSE}_i - \overline{MSE}_j\right)$, we use the Newey-West HAC estimator with quadratic spectral weights.

Besides the $\overline{MSE}$ we also use a second evaluation metric: The coefficient of determination of the so-called MZ-Regression. The MZ-Regression was first proposed by Mincer and Zarnowitz (1969) to evaluate the adequacy (unbiasedness and efficiency) of the forecasting method by running an OLS regression of the actual value of the target variable on its forecast. The R^2_{MZ} of this regression tells us how much of the variation in future volatility can be explained by the forecast produced by our models and is a common measure for out-of-sample evaluation, as noted by Andersen and Bollerslev (1998: 890), Hansen and Lunde (2005: 877), and Kambouroudis, McMillan, and Tsakou (2016: 9). Following the same approach for the $\overline{MSE}$, we compute an average R^2_{MZ} over the forecast horizon:

$$\overline{R^2}_{MZ} \quad = \quad \frac{1}{5}\sum_{h=1}^{5} R^2_{MZ,t+h}. \tag{4.12}$$

Both the aSPA test and the $\overline{R^2}_{MZ}$ are used in our evaluation scheme to rank the models solely in the test set. For model selection based on the training set, we rely only on the $\overline{MSE}$. In the next section, we will introduce our final evaluation—a financial application based on the VaR. Since the VaR is directly related to volatility, it provides economic significance to our forecasts.

4.3.3. Financial Application: Value at Risk

Understanding the dynamics and evolution of an asset's variance is theoretically valid, but its practical usefulness is limited. However, financial theory provides a broad range of applications for variance, such as performance metrics like the Sharpe Ratio, assessments of systematic risk, derivative pricing, and, most relevant for our study, Value at Risk (VaR).

VaR is a measure of risk that quantifies the potential losses in a portfolio resulting from typical market price fluctuations. More specifically, it estimates the maximum expected loss over a specified time period T with a given probability α, under the assumption that the portfolio remains constant. The method is theoretically simple and consolidates all portfolio risk into a single figure (Linsmeier and Pearson 1996: 3). According to Abad, Benito, and López (2014: 15-16), VaR has traditionally been estimated using one of the following methods: non-parametric simulations using the α-th quantile of the histogram of returns as its value, semi-parametric approaches like Monte Carlo simulations, or parametric methods such as the variance-covariance approach, on which we will focus in this study.

The parametric VaR formula is given by:

$$VaR = \mu_t + \Phi^{-1}(1-\alpha)\sqrt{T}\sigma_t \tag{4.13}$$

where μ_t is the expected return of the asset during the holding period T, σ_t is the return volatility expressed as daily standard deviation, $\Phi^{-1}(1-\alpha)$ is the inverse cumulative distribution function at the confidence level $1-\alpha$. Note that Eq. (4.13) represents a one-sided confidence interval and assumes that the one-day variance scales proportionally with the holding period.

To make full use of the volatilities estimated by our models, we calculate the VaR for a 5-day holding period at a 5% significance level. Rather than using $\sqrt{T}\sigma_t$, we replace it with the square root of the sum of the forecasted volatilities over the 5-day forecast horizon, as follows:

$$\widehat{\sigma}_{5\text{day}} = \sqrt{\widehat{\sigma}^2_{t+1} + \cdots + \widehat{\sigma}^2_{t+5}}. \tag{4.14}$$

Since we do not have a model to estimate the mean, we will assume it to be zero. This assumption is reasonable given that the 5-day return over our entire sample is only 0.1% for the S&P500 and 0.06% for the DAX. Following the standard approach proposed by Morgan et al. (1996: 7), we will use the normal distribution for $\Phi(\cdot)$. Although the normal distribution may not be ideal due to the fat tails in the return distribution (as noted in our discussion of stylized facts), the VaR is being used for relative comparison. If the normal distribution is inappropriate, it will affect all models equally.

In our simulation, \$100 (€100) is invested daily in the S&P500 (DAX), and each investment is held for 5 trading days. The portfolio is assumed to be fully hedged against price fluctuations within the 95% VaR boundaries, both positive and negative. This means any loss within the VaR limit is completely offset by the hedge. However, if the return exceeds the VaR threshold, whether as a gain or a loss, the portfolio experiences the full impact of these returns. Importantly, we treat each investment as an independent portfolio, meaning the VaR is calculated relative to the initial investment, without accounting for overlaps across trading periods.

As is the case with the $\overline{R^2}_{MZ}$, we only report the values using our VaR application to evaluate models in the test set. With 830 (823) VaR calculations for the DAX (S&P500), we would expect no more than 41 violations in each direction with 95% confidence. To account for the total number of violations, we scale the portfolio final value. Let *F.V.* represent the total portfolio (final) value, N_{pos} be the total number of positive VaR violations, and N_{neg} the total number of negative violations. The final value of our fictitious portfolio will be given as:

$$F.V. \quad = \quad 100 * \left[1 + \left(\frac{41}{N_{pos}} \right) \sum_{t=1}^{T} V_{pos}(t) - \left(\frac{N_{neg}}{41} \right) \sum_{t=1}^{T} \left| V_{neg}(t) \right| \right]. \tag{4.15}$$

Where $V_i(t)$, is the positive (pos) and negative (neg) return above the VaR calculated at day t, and T is the total number of test-samples. If a model produced zero violation, we would have a *F.V.* of $83,000$ for the DAX and $82,300$ for the S&P500. Consequently, the model producing the highest *F.V.* will be considered the best.

It is crucial to account for positive VaR violations; otherwise, models that systematically overestimate volatility, resulting in fewer violations, may appear to perform better simply because the VaR is rarely breached. By scaling gains and losses based on the

total number of violations we promote the selection of models that provide a narrower and more accurate forecast of VaR, seeking a balance between prediction accuracy and the economic implications of the forecasts.

5. Econometric Models for Volatility: (G)Arch-Models

5.1. Overview of the Models and Their Properties

5.1.1. ARCH and GARCH

The first volatility model to consider the temporal dynamics and heteroscedasticity in the data was the Autoregressive Conditional Heteroscedasticity (ARCH(q)) model, created by Engle (1982). This model was later generalized by Bollerslev (1986), and became known as the GARCH(p, q) model. Before the development of these models, time series analysis primarily focused on models for the conditional mean of a stochastic process, such as Autoregressive (AR)(p) and moving Moving Average (MA)(q). Although these models are relevant, they fail to capture the dynamics of many economic time series, especially the changing variance over time. Engle and Bollerslev (1986: 4-5) show that in an AR process, the future variance depends only on the forecast horizon, not on the available information set ψ_t. This limitation arises because these mean models assume homoscedasticity of the error variance. Before the appearance of the ARCH model, researchers attempted either to capture heteroscedasticity by modeling it as a function of external factors or disregarded the non-constancy in the variance completely (Engle 1982: 997-998). The ARCH model was created to address and relax the homoscedasticity assumption, by modeling volatility as a linear function of the past squared error.

A random variable ε_t is said to follow an ARCH(q) process, if it can be described as:

$$r_t = \mu_t + \varepsilon_t \tag{5.1}$$

$$\varepsilon_t = \sigma_t \eta_t, \text{ with } \eta_t \overset{i.i.d.}{\sim} \mathcal{N}(0,1) \tag{5.2}$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2. \tag{5.3}$$

where r_t is a random variable, μ_t is the mean equation, ε_t is deviation of the random variable from its mean, σ_t^2 is the conditional volatility, and η_t is a white noise error

term.

The GARCH(p,q) model extends the variance equation by p lagged values of the conditional volatility. Defining $\mathcal{A}(\mathbb{L}) \equiv \alpha_1\mathbb{L} + \cdots + \mathbb{L}^q\alpha_q$ and $\mathcal{B}(\mathbb{L}) \equiv \beta_1\mathbb{L} + \cdots + \beta_p\mathbb{L}^p$ to be polynomials of order p and q, and letting $\mathbb{L}$ be the so-called lag-operator, so that $\mathbb{L}^i z_t = \mathbb{L}z_{t-i}$, the GARCH$(p,q)$ volatility equation can be concisely written as:

$$\sigma_t^2 = \omega + \mathcal{A}(\mathbb{L})\,\varepsilon_t^2 + \mathcal{B}(\mathbb{L})\,\sigma_t^2. \tag{5.4}$$

where the coefficients of the model are restricted to be positive to ensure that the estimated volatility is non negative with probability 1. By including lagged-values of the squared return and conditional volatility, the model is capable of accounting for the clustering behavior shown in Section 2.

The relationship between an ARCH(q) and a GARCH(p,q) model is analogous to that between the AR(p) and MA(q) models. Just as a MA(q) can be shown to be equivalent to an AR(∞), a GARCH(p,q) model can be shown to have an ARCH(∞) representation. Following Bollerslev (1986: 309), the infinite order ARCH representation is derived by solving Eq. (5.4) to σ_t^2 and stating the model as

$$\sigma_t^2 = \omega\,[1 - \mathcal{B}(\mathbb{L})]^{-1} + [1 - \mathcal{B}(\mathbb{L})]^{-1}\,\mathcal{A}(\mathbb{L})\,\varepsilon_t. \tag{5.5}$$

This representation is relevant because it shows that a GARCH process can be well approximated by an ARCH(q) model, for sufficiently large q; this implies that the GARCH(p,q) model need to estimate substantially fewer coefficients than the ARCH(q). Further, it is also possible to show that a GARCH(p,q) model has an Autoregressive Moving-Average (ARMA)(p,r) representation by stating the model in (5.4) in terms of the measurement error $\varepsilon_t^2 - \sigma_t^2 = \upsilon_t$, where υ_t is assumed to be independently (but not identically) distributed:

$$\varepsilon_t^2 = \omega + \sum_{i=1}^{r=\max\{p,q\}} (\beta_i + \alpha_i)\,\varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \upsilon_{t-i} + \upsilon_t \tag{5.6}$$

where $\beta_i = 0$ for $i > p$. From Eq. (5.6) we can derive the covariance stationarity condition for the GARCH(p,q) process to be $\sum_{i=1}^{r}(\beta_i + \alpha_i) < \left|1\right|$ and $\sum_{i=1}^{p}\beta_i < \left|1\right|$. The ARMA$(p,r)$ representation has been proven useful in model selection; Bollerslev (1988) showed that the autocorrelation function of ε_t^2 mimics those of the standard

ARMA model, so that the use of correlograms for identifying the orders of the model can be an alternative to more computationally expensive methods.

Forecasting with the model is best illustrated by means of a GARCH(1, 1), without any loss of generalization. By taking conditional expectation on both sides of the volatility equation and forwarding the forecasts into the future, Tsay (2010: 133) shows that the h-step ahead conditional volatility (for $h > 1$) can be computed using the following recursive relation:

$$\begin{aligned} \mathbb{E}\left[\sigma^2_{t+h}\middle|\psi_t\right] &= \sigma^2 + \phi^{h-1}\left[\sigma^2_{t+h-1} - \sigma^2\right] \\ \text{with } \phi &= \alpha + \beta. \end{aligned} \tag{5.7}$$

where $\sigma^2 = \frac{\omega}{1-[\alpha+\beta]}$ is the long-run variance of the process. Hence, as the forecast horizon expands, the forecasted volatility converges to its long-run mean, capturing the mean-reverting property of volatility. Finally, it is straight-forward to show that the distribution of a random variable following any of the models belonging to the GARCH-family with normally distributed innovations $\{\eta_t\}$ will have a kurtosis that is at least as large as the normal, i.e., larger or equal to 3. Inserting Eq. (5.2) into the kurtosis formula yields:

$$\frac{\mathbb{E}\left[\varepsilon^4\right]}{\left(\mathbb{E}\left[\varepsilon^2\right]\right)^2} = \frac{\mathbb{E}\left(\mathbb{E}\left[\varepsilon^4_t\middle|\psi_{t-1}\right]\right)}{\left[\mathbb{E}\left(\mathbb{E}\left[\varepsilon^2_t\middle|\psi_{t-1}\right]\right)^2\right]} = \mathbb{E}\left[\eta^4\right]\frac{\mathbb{E}\left[\sigma^4\right]}{\mathbb{E}\left(\sigma^2\right)^2} = 3 * \frac{\mathbb{E}\left[\sigma^4\right]}{\mathbb{E}\left(\sigma^2\right)^2} \quad \hat{=} \quad \kappa_\varepsilon. \tag{5.8}$$

By Jensen's inequality, $\mathbb{E}\left[\sigma^2_t\right]^2 \leq \mathbb{E}\left[\sigma^4_t\right]$, and consequently $\kappa_\varepsilon \geq 3$, showing that the model is capable of accounting for the fat tails seen in the distribution of returns. An analytical derivation using the variance equation for a GARCH(1, 1) model is found in the work of Tsay (2010: 118, 132).

One of the major drawbacks in the GARCH(p, q) model is its inability to account for the leverage effect, because it assumes that only the magnitude of the lagged values matters. The Exponential GARCH (EGARCH) and the Glosten-Jagannathan-Runkle GARCH (GJR-GARCH) models extend the GARCH framework to allow for asymmetric effects in volatility. In the following sections, we will explore these models in detail, highlighting their improvement upon the standard GARCH model by incorpo-

rating the directional impact of past shocks.

5.1.2. EGARCH

In an effort to address the limitations of the GARCH(p, q) model, Nelson (1991) proposed the EGARCH model, where volatility evolves multiplicatively, as opposed to the linear structure of the standard GARCH(p, q), and is modeled in logarithmic form, relaxing the positivity constraint on the coefficients. Additionally, the volatility equation is augmented with a component capable of accounting for the asymmetric effect of positive and negative innovations on volatility.
The author specifies the model as follows:

$$\ln\left(\sigma_t^2\right) = \omega + \sum_{i=1}^{\infty} \alpha_i g\left(\eta_{t-1}\right), \alpha_1 \equiv 1 \tag{5.9}$$

$$\text{where } g\left(\eta_t\right) = \theta\eta_t + \gamma\left(\left|\eta_t\right| - \mathbb{E}\left[\left|\eta_t\right|\right]\right). \tag{5.10}$$

Here, $g\left(\eta_t\right)$ is defined to be an i.i.d. random variable with mean zero, and θ and γ are real-valued constants. Note that when the innovation η_{t-1} is positive, the impact of $g\left(\eta_{t-1}\right)$ on log-volatility amounts to $\left[\left(\theta+\gamma\right)\eta_{t-1} - \gamma\mathbb{E}\left[\left|\eta_{t-1}\right|\right]\right]$. Conversely, when the innovation η_{t-1} is negative, (i.e., $\left|\eta_{t-1}\right| = -\eta_{t-1}$), the effect of $g\left(\eta_{t-1}\right)$ on log-volatility is $\left[\left(\theta-\gamma\right)\eta_{t-1} - \gamma\mathbb{E}\left[\left|\eta_{t-1}\right|\right]\right]$. For a standard normally distributed η_t,the log-volatility equation becomes:

$$\ln\left(\sigma_t^2\right) = \omega + \sum_{i=1}^{\infty} \alpha_i \left(\theta\eta_{t-i} + \gamma\left[\left|\eta_{t-i}\right| - \sqrt{\frac{2}{\pi}}\right]\right) \tag{5.11}$$

An ARMA(p, q) specification allows for a more parsimonious representation of the model by removing the infinite summation, defined as:

$$\ln\left(\sigma_t^2\right) = \omega + \frac{1 + \mathcal{A}\left(\mathbb{L}\right)}{1 - \mathcal{B}\left(\mathbb{L}\right)} g\left(\eta_{t-i}\right), \tag{5.12}$$

$$\text{where } \mathcal{A}\left(\mathbb{L}\right) = \sum_{i=1}^{q} \alpha_i \mathbb{L}^i, \text{ and } \mathcal{B}\left(\mathbb{L}\right) = \sum_{i=1}^{p} \beta_i \mathbb{L}^i \tag{5.13}$$

Covariance stationarity demands that both the numerator and denominator converge, i.e., $\mathcal{A}\left(1\right) < 1$ and $\mathcal{B}\left(1\right) < 1$, under the assumption that $1 + \mathcal{A}\left(\mathbb{L}\right)$ and $1 - \mathcal{B}\left(\mathbb{L}\right)$ have no common roots (Nelson 1991: 354; Tsay 2010: 143).

Both the theoretical future volatility and the kurtosis of an EGARCH(p, q) process are more complex to derive than those of the GARCH(p, q) model, as they involve nonlinear terms. Tsay (2010: 148) gives an analytic derivation of the h-step ahead volatility and He, Teräsvirta, and Malmsten (2002: 871-872) of the kurtosis for an EGARCH$(1, 1)$. Instead of presenting the analytic derivation for the forecast, we present the method used in our work, which follows the standard implementation of the arch-package in python.

According to the official ARCH package documentation (Sheppard 2024: 21), multi-step ahead forecasts for models that are linear in the squares, such as the (G)ARCH(p, q), and GJR-GARCH(p, q), are generated analytically. However, for the EGARCH model, the package relies on simulation for all forecasts beyond the first step. For $h = 1$, the model inputs are available in-sample, allowing for an analytical forecast. For any $h > 1$, B paths of future innovations $\{\eta_{t+1,b}, \ldots, \eta_{t+h,b}\}$ are independently drawn from the distribution used to estimate the model, and these innovations are used as inputs. A single simulated path for the EGARCH$(1, 1)$ model would be the following:

$$\sigma^2_{t+h,b} = \sigma^{2\beta}_{t+h-1} \exp\left\{\omega + \theta\left(\frac{\varepsilon_{t+h-1}}{\sigma_{t+h-1}}\right) + \gamma\left(\left|\frac{\varepsilon_{t+h-1}}{\sigma_{t+h-1}}\right| - \mathbb{E}\left[\left|\frac{\varepsilon_{t+h-1}}{\sigma_{t+h-1}}\right|\right]\right)\right\}. \quad (5.14)$$

The simulated residual for each path is computed as:

$$\varepsilon_{t+h,b} = \eta_{t+h}\sqrt{\sigma^2_{t+h,b}}, \quad (5.15)$$

and the final forecast for the variance is the average of the variances over the B simulated paths:

$$\mathbb{E}\left[\varepsilon^2_{t+h}\right] = \sigma^2_t = B^{-1}\sum_{b=1}^{B}\sigma^2_{t+h,b}. \quad (5.16)$$

To conclude, the EGARCH(p, q) model represents an improvement of the GARCH(p, q) due to its ability of capturing a further stylized fact, namely the leverage effect, that the GARCH(p, q) implementation is unable to account for. However, the model's complexity can pose challenges in implementation and interpretation. Alternative models have been proposed to capture asymmetric volatility responses in a more straightforward manner. One such model is the GJR-GARCH(p, q), which does not involve

exponential terms and is much more similar to the standard GARCH(p, q) than the EGARCH(p, q) model, and will be the subject of study in the next section.

5.1.3. GJR-GARCH

The GJR-GARCH(p, q) model was introduced by Glosten, Jagannathan, and Runkle (1993: 1787), to address inconsistencies in the literature regarding the correlation between the risk premium and market volatility. While introductory financial theory posits a positive correlation between risk premium and volatility, empirical results have been mixed. Among the GARCH-in-Mean specifications used by the authors was one accounting for different reactions of volatility according to sign of the shock. The proposed model, which would later be known as the GJR-GARCH(p, q) (referring to the first letters of the authors' names) was specified as follows:

$$\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \gamma\varepsilon_{t-1}^2\mathbf{I} + \beta\sigma_{t-1}^2 \tag{5.17}$$

where $\mathbf{I}$ is an indicator that equals 1 if the past innovation is negative and 0 otherwise, implying that following a shock of 1 unit in terms of past innovation, the market reaction leads to α effect on current volatility, and $(\alpha + \gamma)$ when the innovation is negative. The requirement of positivity of coefficients still holds, so that the reaction to a negative shock will be at least as high as the reaction to a positive one. The authors demonstrated that different specifications can lead to very different conclusions. While the standard GARCH-M model indicated a positive but statistically insignificant relationship between the expected risk premium and market volatility, the model accounting for the difference in the sign of the innovation found a significant negative coefficient, supporting a negative correlation between the expected risk premium and volatility, and suggesting that investors lower their expectations during difficult market conditions.

The GJR-GARCH(p, q) model has an advantage over the EGARCH(p, q) model in that it retains a more tractable variance process, similar to the original GARCH specification. This makes the GJR-GARCH model easier to interpret and implement while still accounting for asymmetries in volatility.

Unfortunately, the article presenting the GJR-GARCH(p, q) model lacks the technical

details about its statistical properties; nevertheless, an ARMA(r, p) representation is promptly achieved by following the same arguments presented in the GARCH-case. Defining $\varepsilon_t^2 - \sigma_t^2 = \upsilon_t$ to be the measurement error, we can derive the ARMA(r, p) representation for the general form of the model in Eq. (5.17) by rewriting it solely in terms of $\left\{\varepsilon_t^2\right\}$ and the error:

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \left(\alpha_i + \gamma_i \mathbf{I}_{t-i}\right) \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 \tag{5.18}$$

$$\varepsilon_t^2 = \omega + \sum_{i=1}^{r=\max(p,q)} \left(\alpha_i + \gamma_i \mathbf{I}_{t-i} + \beta_i\right) \varepsilon_{t-i}^2 - \sum_{i=1}^{p} \beta_i \upsilon_{t-i} + \upsilon_t, \tag{5.19}$$

and the wide sense stationarity condition is promptly seen from its unconditional variance. Assuming randomness of the innovation, $\mathbf{I}_{t-i}$, is Bernoulli distributed with $\mathbb{E}\left[\mathbf{I}_{t-i}\right] = \frac{1}{2}$, it follows that

$$\mathbb{E}\left[\sigma_t^2\right] = \omega + \sum_{i=1}^{q} \left(\alpha_i \mathbb{E}\left[\varepsilon_{t-i}^2\right] + \gamma_i \mathbb{E}\left[\mathbf{I}_{t-i} \varepsilon_{t-i}^2\right]\right) + \sum_{i=1}^{p} \beta_i \mathbb{E}\left[\sigma_t^2\right] \tag{5.20}$$

$$\sigma^2 = \frac{\omega}{1 - \left(\sum_{i=1}^{q} \left[\alpha_i + \frac{1}{2}\gamma_i\right] + \sum_{i=1}^{q} \beta_i\right)} = \frac{\omega}{1 - \left(\sum_{i=1}^{q} \phi_i + \sum_{i=1}^{p} \beta_i\right)} \tag{5.21}$$

leading to the requirement $\left(\sum_{i=1}^{q} \phi_i + \sum_{i=1}^{p} \beta_i\right) < 1$ for the existence of the second moment, and thus, stationarity. In order to ensure positivity of the variance, the parameters ϕ_i and β_i are constraint to be positive.

As illustrated by Poon and Poon (2005: 42), the GJR-GARCH(p, q) model allows the h–step ahead forecast to be computed analytically. For the GJR-GARCH$(1, 1)$, without loss of generality, the recursive relation is given as

$$\mathbb{E}\left[\widehat{\sigma}_{t+h}^2 \middle| \psi_t\right] = \omega + \left(\frac{1}{2}\left[\alpha + \gamma\right] + \beta\right) \widehat{\sigma}_{t+h-1}^2. \tag{5.22}$$

Having established the key properties of the GJR-GARCH(p, q) model in capturing asymmetric volatility effects, we now turn our attention to the estimation and evaluation of these models.

5.2. Choosing The Baseline(s)

5.2.1. Model Specification Selection

The analysis and results presented in this section are based solely on our training sample and have two main objectives: First, to present our method for selecting the optimal model specification from a set of candidates; second, to evaluate and identify the GARCH-family models that will be compared against the ML models discussed in Section 6.

One of the most commonly used methods for selecting the lag structure and specification of a time series model is to minimize an information criterion (IC), such as the Akaike Information Criterion (AIC) or the Bayesian Informaiton Criterion (BIC). These criteria balance the goodness of fit with model complexity by penalizing additional parameters to avoid overfitting, making them useful tools for comparing the quality of models. However, in their standard form, these criteria assume homoscedastic residual variance, among other distributional assumptions (Rossi et al. 2020: 247). This can be problematic for volatility models, where heteroscedasticity is one of the phenomenon being modeled. As a result, the standard ICs may be inappropriate for such models. To address this limitation, Brooks and Burke (2003: 561-562) proposed heteroscedasticity-adjusted versions of these criteria:

$$HAIC = \sum_{t=1}^{T} \ln\left(\widehat{\sigma}_t^2\right) + 2k \tag{5.23}$$

$$HBIC = \sum_{t=1}^{T} \ln\left(\widehat{\sigma}_t^2\right) + k \log\left(T\right), \tag{5.24}$$

where $\widehat{\sigma}_t$ is the in-sample conditional volatility produced by the volatility model, T is the total number of observations, and k the number of estimated parameters.

To assess the reliability of these criteria in selecting the optimal models for out-of-sample forecasts, we compared the models chosen by the $HAIC$ and $HBIC$ with those selected by our data-driven approach: cross-validation. Specifically, our goal was to identify the model specification that minimizes the out-of-sample $\overline{MSE}$ for each class of GARCH-family models and to evaluate whether the models with the lowest $HAIC$ and $HBIC$ yield comparable performance in terms of $\overline{MSE}$.

Throughout Section 5, we presented the GARCH-family models under the assumption

of normally distributed innovations $\{\eta_t\}$; however, this assumption is not necessary. Other distributions may better capture the empirical distribution (and other stylized facts) of returns. For example, Nelson (1991: 352-355) studied the EGARCH model using a Generalized Error Distribution (GED), which includes a shape parameter ν that controls the thickness of the tails. When $\nu = 2$, the GED matches the normal distribution; when $\nu < 2$, the distribution exhibits fatter tails. Engle and Bollerslev (1986: 33-35) used the standardized Student's t-distribution to estimate GARCH and IGARCH models, where the degrees of freedom, like the shape parameter of the GED, are estimated from the data.

An extensive study of potential distributions is beyond the scope of this paper, but interested readers may refer to Ampadu et al. (2024), who thoroughly examined the performance of GARCH models under different distributions. In our study, we focus on the two of them: the normal and the GED.

GARCH models can be estimated by OLS using a two-step approach, where residuals from a mean model are first obtained, and used in the estimation of the volatility equations in a second step (Francq and Zakoian 2019: Chapter 6). Alternatively, they can be estimated via (Quasi) Maximum Likelihood Estimation (MLE). We opt for MLE, which is standard in the literature, to jointly estimate the parameters most likely to have generated the data.

The GED distribution adds an additional parameter, ν, to the estimation. We include this parameter in k in the IC-formulas. For consistency, we estimate the same number of models for each volatility proxy and security. Given the low serial-correlation in the return series for both indices, we entertained only two types of mean equations: one with a non zero and one with a zero constant mean.

For a given security, volatility proxy, mean equation, and distribution, we estimated 30 ARCH models with lags ranging from 1 to 30. For the GARCH, EGARCH, and GJR-GARCH, we estimated 30 models each, using all combinations of $p \in [0, 5]$ and $q \in [1, 5]$, where p and q are integer values. Since 5 of the GARCH models overlap with the ARCH specifications, we actually estimated 25 unique GARCH models. In total, we fitted $1,840$ models—460 for each combination of volatility proxy and security. Out of these, 98 models were removed due to convergence problems, all of them with either a GJR-GARCH or EGARCH specification.

Tabs. 4 and 5 present the optimal models selected by *HAIC*, *HBIC*, and cross-

validation (C.V.) for the DAX and S&P500, respectively. The last two columns show the $\overline{MSE}$ and the ranking of the models within their classes. For example, the optimal model selected by *HAIC* to forecast the DAX volatility in the ARCH class has a constant mean, 22 lags, and normally distributed innovations. Its $\overline{MSE}$ is 25.173%[4], and it ranks 52nd among 120 ARCH specifications for forecasting $S.R.$; for the target variable ${}_{adj.}R.V.$, it achieved an $\overline{MSE}$ of 5.263%[4], ranking 67th. The top-ranked ARCH models has a constant mean, 11 lags, and GED innovations when forecasting $S.R.$, and a zero mean, 10 lags, and normally distribution innovations when forecasting the ${}_{adj.}R.V.$, with an $\overline{MSE}$ of 5.434%[4].

Rather than listing each model's specification, we refer to their classes, volatility proxy, and security. Their specifications can be found in the cited tables. The tuples in the $\overline{MSE}$ and Rank columns correspond to the volatility proxies ($S.R.$; ${}_{adj.}R.V.$). As the input for GARCH models are the residual series, the same models are evaluated for both volatility proxies, resulting in a single row for each model.

From the tables, it is evident that models selected by *HAIC* and *HBIC* generally do not rank highly in terms of $\overline{MSE}$. The exceptions are the GARCH model selected by the *HAIC* for forecasting the $S.R._{DAX}$, which ranked second, and the EGARCH model for forecasting the $S.R._{S\&P500}$, which ranked fourth. Thus, using ICs for model selection has been shown to be suboptimal when the goal is out-of-sample performance based on $\overline{MSE}$. The reason for the poor performance of the ICs may include: 1) ICs are based on in-sample information, and 2) ICs are based on one-step-ahead volatility, whereas $\overline{MSE}$ averages the MSE's from one to five step ahead forecasts.

Visual inspection of partial and auto-correlograms is another common method for selecting time-series model order. However, the patterns in these plots are less clear-cut for models with an ARMA specification compared to pure AR and MA processes, and selecting the order based on the method does not guarantee optimal out-of-sample performance. Therefore, we did not pursue such procedure.

Next, we focus on the models selected by C.V.. For the DAX, the ranking of the model classes is consistent across both volatility proxies. The EGARH model had the lowest $\overline{MSE}$, followed by the GJR-GARCH, GARCH, and ARCH. Both EGARCH models were very similar in specifications: a parsimonious order $(P, Q) = (1, 1)$, and GED innovations; and differed only in the mean specifications.

The consistency in ranking did not hold for the S&P500. When forecasting $S.R.$, the

ranking was identical to the DAX: EGARCH performed best, followed by the GJR-GARCH, GARCH and ARCH. However, when the volatility proxy was the ${}_{adj.}R.V.$, the GJR-GARCH performed worse than the GARCH and ARCH specifications, but the EGARCH held the top position.

Overall, EGARCH specifications dominated the top position in terms of $\overline{MSE}$: the 98 smallest $\overline{MSE}_{DAX}^{adj.R.V.}$ and the 72 smallest $\overline{MSE}_{S\&P500}^{adj.R.V.}$ were achieved by some kind of EGARCH model[2]. Similarly, the 62 lowest $\overline{MSE}_{DAX}^{S.R.}$ and the 69 lowest $\overline{MSE}_{S\&P500}^{S.R.}$ also belonged to the EGARCH class. This emphasizes the importance of accounting for the leverage effect to produce accurate forecasts.

Another notable finding is the difference in $\overline{MSE}$ between models forecasting the ${}_{adj.}R.V.$ and $S.R.$. For example, the mean $\overline{MSE}$ across all top-ranked models for the $S.R._{DAX}$ is $24.397\%^4$, while for the ${}_{adj.}R.V._{DAX}$, it is $4.357\%^4$. In terms of Root Mean Squared Error (RMSE), these values are $4.939\%^2$ and $2.085\%^2$, being the former 2.36 times higher than the training $\overline{S.R.}_{DAX}$ and the later only 1.05 higher than the training ${}_{adj.}\overline{R.V.}$. A similar pattern holds for the S&P500, though less pronounced, where the average $RMSE_{S\&P500}^{S.R.}$ is 2.66 times higher than the $\overline{S.R.}_{S\&P500}$ and the average $RMSE_{S\&P500}^{adj.R.V.}$ is 1.79 times higher than the ${}_{adj.}\overline{R.V.}$. These results suggest that GARCH-family models are much more accurate at forecasting the ${}_{adj.}R.V.$ than the $S.R.$, consistent with Andersen and Bollerslev (1998), who emphasized the importance of selecting appropriate volatility proxies.

We conclude this section by selecting the top-ranked models presented in Tabs. 4 and 5 as our baselines. These models will be compared against the ML models in terms of forecasting accuracy and VaR in Section 7. In the next section, we will fit these models to the entire training set and evaluate their performance on the test set based on $\overline{MSE}$, aSPA, and $\overline{R}_{MZ}$.

5.2.2. Parameter Estimation and Model Evaluation

The models that performed best in terms of $\overline{MSE}$ on the training-set is already known to the reader. Although cross-validation is often regarded as an "out-of-sample" evaluation, re-estimating parameters for each new observation is both impractical and time-consuming, especially for ML models. Re-estimation can create a significant compu-

[2]Where appropriate, we follow the convention of writing the security used in the calculation in the subscript and the volatility proxy in the superscript of the variables.

tational burden, often taking hours. For the average investor, who requires forecasts for timely decisions, such re-estimation is not feasible. Therefore, a more practical approach involves estimating the model once, fixing the parameters, and using it for future predictions.

Following this approach, we re-estimated the best-performing models using the entire training set. The results, including coefficients, p-values, and standard errors, are detailed in Section A.6 of the appendix. A substantial amount of models' coefficients are not statistically significant. For example, in the ARCH models for the S&P500 (cf. Tabs. 13 and 17), only 9 out of the 29 lags in the $S.R.$ model and 11 out of the 30 lags in the ${}_{adj.}R.V.$ model are significant at the 90% confidence level. Another example is the EGARCH(3, 1) model for forecasting the $S.R._{S\&P500}$ (Tab. 15), where the last two coefficients for the lagged log-volatilities are exactly zero. In practice, this makes the model equivalent to an EGARCH(1, 1). Investigating whether a more parsimonious model with fewer parameters would improve test-set performance is beyond the scope of this study and left for future research.

For consistency, the models were used as specified in the regression outputs to generate forecasts with a horizon of 5 days. Alongside the $\overline{MSE}$, we calculated the $\overline{R^2}_{MZ}$ as a second measure of comparison, and evaluated the predictive superiority of the models using the aSPA test.

Tab. 6 presents the test-set performance of the models, with values reported in tuples. The first element refers to model performance in forecasting the $S.R.$, and the second refers to forecasting the ${}_{adj.}R.V.$. As previously noted, the model specifications are detailed in Tabs. 4 and 5. Before analyzing specific models, it is important to note the average $\overline{R^2}_{MZ}$ across all models: $\overline{\overline{R^2}}^{S.R.}_{MZ,DAX} = \overline{\overline{R^2}}^{S.R.}_{MZ,S\&P500} = 10.06\%$, while for the ${}_{adj.}R.V.$ proxy, the averages are $\overline{\overline{R^2}}^{adj.R.V.}_{MZ,DAX} = 33.54\%$ and $\overline{\overline{R^2}}^{adj.R.V.}_{MZ,S\&P500} = 35.34\%$. This indicates that GARCH-family models explain ${}_{adj.}R.V.$ much better than $S.R.$.

The best model in terms of $\overline{MSE}^{S.R.}_{DAX}$ was the GJR-GARCH specification, achieving 7.645%[4], closely followed by the EGARCH with a value of 7.655%[4]. The aSPA test validates the superior performance of both models over the remaining ones (c.f. Tab. 9). However, the test shows no significant difference between the EGARCH and GJR-GARCH. Fig. 7 provides a visual representation of the total MSE across the forecast horizon. Each square in the figure represents the MSE computed from the h-step forecast, with $h = 1, \ldots, 5$. The first element in the tuples inside the squares shows

the percentage contribution of each h-step MSE to the total, while the second element presents the model's absolute performance compare to other classes. For example, the GJR-GARCH model had the lowest total $MSE^{S.R.}_{DAX}$ across all steps except the fourth, where it ranked second to the EGARCH model, which consistently placed second overall. ARCH, by contrast, had the highest MSE across all models, though the aSPA test suggests that the performance differences may be due to random variation.

In terms of $\overline{R^2}^{S.R.}_{DAX}$, the ranking shifts: ARCH forecasts explain, on average, 0.34% more of the variation in the $S.R.$ than GARCH. Meanwhile, the EGARCH forecasts explanatory power outperforms the GJR-GARCH by 0.05%.

For the S&P500, the lowest $\overline{MSE}^{S.R.}_{S\&P500}$ was achieved by the EGARCH model ($4.61\%^4$), followed by GJR-GARCH, GARCH, and ARCH. This ranking aligns with the training set results, but the aSPA test (see Tab. 10) finds that the EGARCH specification only outperforms the GJR-GARCH at the 10% significance level. In terms of $\overline{R^2}^{S.R.}_{MZ,S\&P500}$, the EGARCH forecasts also lead the ranking, explaining 11.96% of the variation in the target variable, and the ARCH model performs worst, with an $\overline{R^2}_{MZ}$ of only 8.85%.

Turning to the models whose target variable was the ${}_{adj.}R.V.$, the GJR-GARCH had the smallest error for the DAX, with a $\overline{MSE}$ of $1.052\%^4$, followed closely by the EGARCH ($1.078\%^4$). The aSPA test confirms that the GJR-GARCH predictive performance is statistically superior to all models except EGARCH, and all models outperformed the ARCH. In terms of $\overline{R^2}_{MZ}$, the EGARCH again outperformed the GJR-GARCH specification (41.61% vs. 38.04%). Interestingly, the γ_1 coefficient in both models, which captures asymmetry in volatility's response to negative shocks, differs: GJR-GARCH shows a positive and highly significant value (0.1862), while the EGARCH estimates a negative value (-0.1371). This contrast may explain why EGARCH forecasts exhibit higher explanatory power.

As shown in Fig. 7, for the S&P500, the GARCH model had the lowest $\overline{MSE}$ ($0.897\%^4$) for forecasting the ${}_{adj.}R.V.$[3], diverging from the training set results where EGARCH was the best model. The EGARCH ranked third in the test set, behind the ARCH specification. However, in terms of $\overline{R^2}_{MZ}$, the EGARCH and GJR-GARCH ranked first (40.02%) and second (38.01%) respectively, with the GARCH coming in third. This suggests that while the GARCH model generates smaller forecast errors, it struggles

[3]The figure actually shows the lowest total MSE, which is equivalent to having the lowest average over all steps.

to capture the swings and trends in the volatility series.

In summary, choosing the best model depends on the forecasting goal. If minimizing error is the objective, the models that ranked highest in terms of $\overline{MSE}$ should be selected. However, if the goal is to capture future volatility trends and movements, the models with highest $\overline{R^2}_{MZ}$ are preferable.

In the next section, we shift our focus to ML data-driven models. We begin by providing an overview of the models, along with their technical specifications. As with the standard econometric models for volatility, we first assess the performance of the machine learning models in isolation. In Section 7, we then compare the performance of all models and present the results for VaR.

Fig. 7: Cumulative h-step Ahead MSE (GARCH-Family)

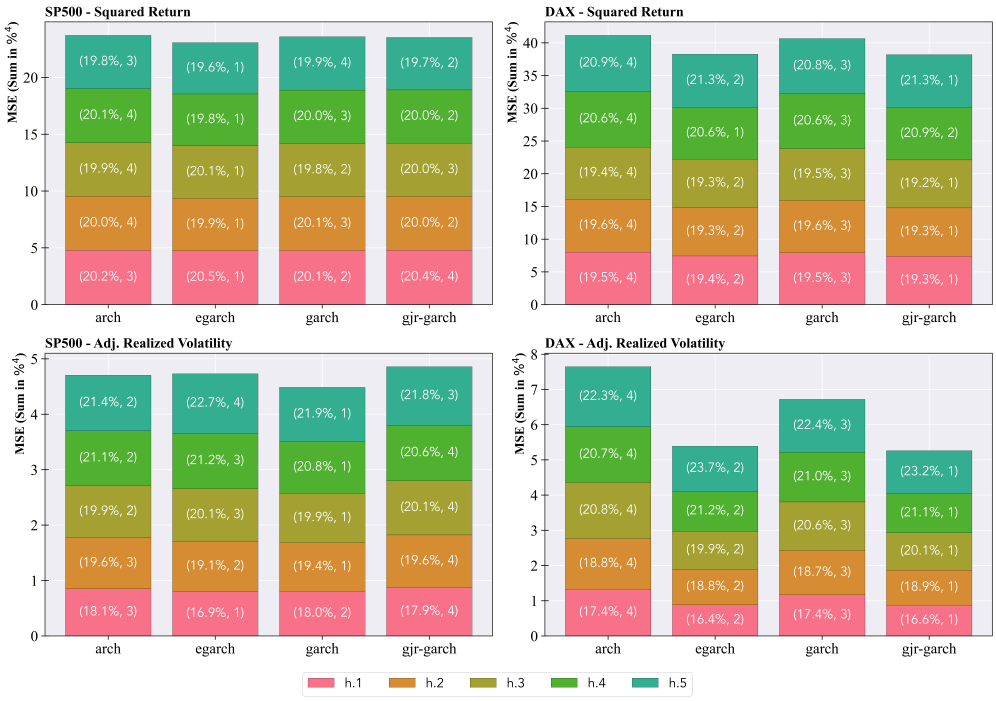


Figure illustrates the cumulative MSE of 1- to 5-step ahead forecasts from GARCH-family models for the SP500 (left column) and the DAX (right column). The upper row is related to the metrics computed using the $S.R.$, and the lower row the $_{adj.}R.V.$ as target variable. Each color represents a forecast horizon (h.1 to h.5), with the percentages indicating the contribution of each horizon to the total MSE. The second element in the tuple indicates the rank of the model at forecasting the h-th step volatility.

6. Newer Methods: Machine and Deep Learning Approach

6.1. Overview of the Models

6.1.1. Support Vector Regression

SVR is a generalization of the Support Vector Machine (SVM) to regression problems introduced by Drucker et al. (1996). The key idea behind SVM is to separate the feature space using a hyperplane. This hyperplane is positioned exactly in the middle of the distance between the closest vectors from each class, known as the support vectors. The objective is to maximize the margin, i.e., the distance between the hyperplane and the classes' boundaries, ensuring the best separation between the classes (James et al. 2023: 370-372). The idea of "margin" is translated to an epsilon insensitive tube in the SVR. In order to estimate a function $F(x, \mathbf{w})$ that best approximates what Drucker et al. (1996) called "the ground truth" $G(x)$ by optimizing $\mathbf{w}$, the authors suggest the following loss function:

$$\mathcal{L}(y_i, F(\mathbf{w}, \mathbf{x})) = \begin{cases} 0 & \text{if } \left|y_i - F(\mathbf{w}, \mathbf{x})\right| < \varepsilon \\ \left|y_i - F(\mathbf{w}, \mathbf{x})\right| - \varepsilon & \text{otherwise} \end{cases} \tag{6.1}$$

The loss function forms a tube of radius ε around the predicted values, where prediction error falling inside the tube is disregarded by the loss function, and the only errors considered are those above or below the tube, whose magnitude is captured by the variables ξ_i, and ξ_i^* respectively (see Fig. 8). As written by Awad and Khanna (2015: 67) the optimization problem of SVR "[...] attempts to find the narrowest tube centered around the surface, while minimizing the prediction error", and can be stated as follows:

$$\min_{\mathbf{w}, \xi_i^*, \xi_i, b} \frac{1}{2}\mathbf{w}^t\mathbf{w} + C\left(\sum_{i=1}^{n} \xi_i^* + \xi_i\right) \tag{6.2}$$

$$\text{s.t.} y_i - f(\mathbf{w}, \mathbf{x}) - b \leq \varepsilon + \xi_i \tag{6.3}$$

$$f(\mathbf{w}, \mathbf{x}) + b - y_i \leq \varepsilon + \xi_i^* \tag{6.4}$$

$$\xi_i^*, \xi_i \geq 0 \tag{6.5}$$

Fig. 8: SVR with Epsilon-Insensitive Tube

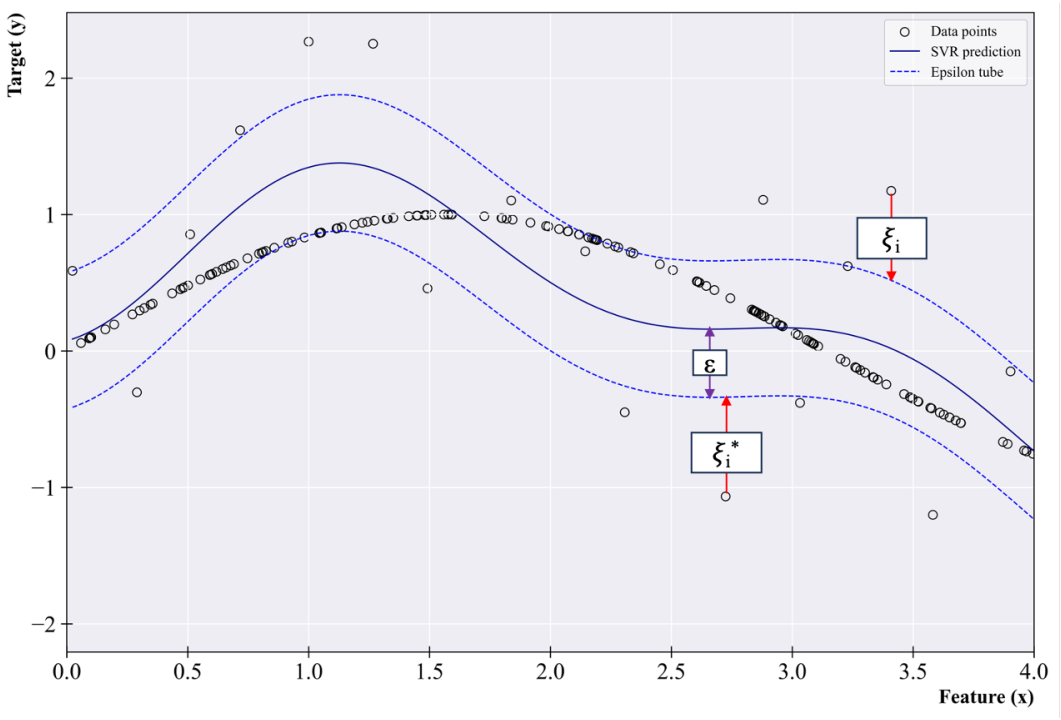


Figure illustrates the concept of Support Vector Regression (SVR) using a sinusoidal function as the underlying data generating process. The solid line represents the SVR prediction, and the dashed lines form the ε-insensitive tube, within which no penalty is applied. Slack variables ξ_i and ξ_i^* are shown for points outside the tube, where errors are penalized.

where b is a bias-term and C is a regularization parameter. Assuming that $f(\mathbf{w}, \mathbf{x})$ is a linear function and letting $x_i = [1, x_1, \ldots, x_m]$ be the input vector, the problem can be stated in its dual form as:

$$\begin{aligned} \max_{\alpha_i, \alpha_i^*} \mathcal{L} &= -\varepsilon \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{N} y_i (\alpha_i^* - \alpha_i) \\ &\quad -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i^* - \alpha_i)\left(\alpha_j - \alpha_j\right)\left(\mathbf{x}_i^T \mathbf{x}_j\right) \qquad (6.6) \\ \text{s.t.} 0 \leq \alpha_i &\leq C \qquad (6.7) \\ 0 \leq \alpha_i^* &\leq 0 \qquad (6.8) \\ \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) &= 0 \qquad (6.9) \end{aligned}$$

where α_i and α_i are Lagrange multipliers from the primal problem. Non-linear relationships can be captured by the so-called Kernel trick, where the input vector is first transformed from the non-linear input spaced to a linear feature space by using a kernel function $K(\mathbf{x})$. In which case, the $\left(x_i^T x_j\right)$ term is equation (6.6) is replaced by $K\left(x_i, x_j\right)$.

6.1.2. XGBoost

Extreme Gradient Boosting (XGBoost) is an advanced implementation of gradient boosting techniques, and it can be viewed as an extension or evolution of the Classification and Regression Trees (CART) algorithm. Before delving into the specifics of XGBoost, it is essential to understand the foundational concept of decision trees.
A decision tree algorithm recursively splits the feature space based on specific rules to make predictions. The process begins at the root node, representing the entire dataset. This node is then divided into two or more child nodes based on splitting rules that aim to minimize the loss. Each split generates internal nodes that further partition the data. The terminal nodes, which no longer split, are known as leaf nodes, where final predictions are made (Quinlan 1986: 86-87).
One limitation of decision trees is their tendency to become overly complex, leading to overfitting and poor generalization to new data. To address this, several solutions have been proposed. Two of them are:

- Tree pruning: This technique reduces the number of splits in a decision tree, thereby simplifying its structure (James et al. 2023: 335-336).

- Ensemble methods: This technique involves combining multiple models to make predictions. The idea is that by aggregating the outputs of several models, the ensemble can produce more accurate and robust predictions than any individual model could on its own. As explained by (James et al. 2023: 343), given a set of n i.i.d. random variables z, the variance of the sample mean is $\mathbb{V}[\bar{z}] = \frac{\sigma_z^2}{n}$, hence, increasing n will reduce the variance. Methods like Bagging (Breiman 1996) and Random Forest introduced (Breiman 2001) are two well known algorithms that fall inside this category. The former involves creating multiple subsets of the data using bootstrap sampling, fitting numerous regression trees, and averaging their predictions. Random forest enhance this approach by randomly selecting subsets of features for each tree, reducing correlation among the trees and improving model accuracy in not yet seen data.

XGBoost, like Random Forest and Bagging, is an ensemble method but employs gradient boosting technique to construct the trees. This method relies on multiple small models, which individually contribute minimally to the learning process, and are, thus, known as weak learners. However, when combined, they produce a powerful model.

The core idea of boosting is the sequential application of weak learners to modified versions of the data, each time aiming to correct errors from preceding models (Hastie 2009: 337). The "gradient" component indicates the use of gradient descent to minimize the loss function, iteratively adjusting the latest weak learner's predictions in the direction that reduces the loss.

Adopting the same notation from and based on the explanation of Chen and Guestrin (2016: 2), the tree ensemble method is mathematically expressed as:

$$y_i = \phi(\mathbf{x}_i) = \sum_{k=1}^{K} f_k(\mathbf{x}_i), \; f_k \in \mathcal{F} \tag{6.10}$$

where $\mathcal{F}$ denotes the space of all possible decision trees, y_i is a real valued target variable (i.e., $y_i \in \mathbb{R}$), and $\mathbf{x}_i$ is a vector with m covariates (i.e., $\mathbf{x}_i \in \mathbb{R}^m$). The goal is to select the set of f_k that minimizes the loss function:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} \ell(\widehat{y_i}, y_i) + \sum_{k=1}^{K} \Omega(f_k), \tag{6.11}$$

$$\text{where } \Omega(f) = \gamma T + \frac{1}{2}\lambda \parallel \boldsymbol{\omega} \parallel^2 \tag{6.12}$$

where T represents the number of leaves in the tree, λ is a regularization parameter controlling the weights of the leaves ($\boldsymbol{\omega}$) (also known as Ridge Regularization), and $\ell(*)$ is a convex and twice-differentiable function measuring the discrepancy between the target y_i and its prediction $\widehat{y_i}$.

As highlighted by the authors, directly minimizing the loss is not feasible in Euclidean Space since functions themselves are parameters. Consequently, the model is trained in an additive manner. In each iteration t, the output of the function that leads to most improvement is added to the model in the following manner:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \ell\left(y_i, \widehat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)\right) + \Omega(f_t). \tag{6.13}$$

To simplify the optimization, a second-order Taylor approximation is employed:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[\ell\left(y_i, \widehat{y}^{(t-1)}\right) + \overbrace{\frac{\partial \ell\left(y_i, \widehat{y}^{(t-1)}\right)}{\partial \widehat{y}^{(t-1)}}}^{g_i} f_t\left(\mathbf{x}_i\right) + \frac{1}{2} \overbrace{\frac{\partial^2 \ell\left(y_i, \widehat{y}^{(t)}\right)}{\partial^2 \widehat{y}^{(t-1)}}}^{h_i} f_t^2\left(\mathbf{x}_i\right) \right] + \Omega f\left(f_t\right). \tag{6.14}$$

This formulation makes it clear that the optimization is contingent on past predictions $\widehat{y}^{(t-1)}$, and the usage of first (g_i) and second order gradients (h_i) enables the algorithm to discern the optimal direction for minimizing the loss. This iterative approach of "gliding through the values of the loss function" to identify its minimum is termed *gradient descent.* The extreme part is due to the addition of the (Ridge) regularization to the model[4].

In summary, XGBoost is a state-of-the-art method based on tree algorithms. It is flexible and applicable to both regression and classification problems by simply changing the loss function employed. XGBoost achieves higher accuracy by making the model more generalizable through its regularization parameters and the use of weak learners. Additionally, it incorporates the main concept of random forests by allowing for the random selection of features for each tree Chen and Guestrin (2016: 2-3).

XGBoost may be particularly well-suited for the problem of predicting asset volatility. Its non-parametric approach enables it to capture non-linear relationships and interactions between features, making a good option for modeling complex financial data, and its enhanced generalization makes it valuable for out-of-sample predictions. In the next section, we will introduce and explain the final model among the "newer approaches" for volatility forecasting: the Long Short-Term Memory (LSTM) network, a deep learning model created to deal specifically with time series.

6.1.3. Long Short Term Memory (LSTM)

LSTM (Long Short-Term Memory) is a special type of Recurrent Neural Network (RNN). In a standard RNN, each layer consists of "cells" that take two inputs: the input x_t at time t and the previous hidden state h_{t-1}, which carries information from prior steps. Sherstinsky (2020: 3-9) shows that the standard system RNN-system evolves

[4]Interested readers who wish to understand how the choice of λ influences the optimal weight are encouraged to resort to the work of Chen and Guestrin (2016: 3)

according to a discrete-time non-linear ordinary delay differential equation:

$$\begin{aligned} h_t &= \mathbf{W}_s h_{t-1} + \mathbf{W}_x x_t + b \\ y_t &= \sigma(h_t) \end{aligned}$$

where $\mathbf{W}_s$ and $\mathbf{W}_x$ are weight matrices, and b is a bias term. The output y_t is a non-linear transformation of h_t using a non-linear activation function σ (typically the sigmoid).

In theory, RNNs should be capable of capturing all time dependencies within a series. However, as noted by Hochreiter and Schmidhuber (1997: 1) and extensively studied by Sherstinsky (2020: 16-17), a major challenge in standard RNNs is the vanishing or exploding gradient problem, which poses difficulties in the optimization of the weights, limiting the amount of information that the model can carry through the system. To address this limitation, Hochreiter and Schmidhuber (1997: 6-8) proposed the LSTM architecture, which introduces input (i_t) and output gates (o_t) to regulate information flow within the network, and was later extended by Gers, Schmidhuber, and Cummins (1999) to contain a forget gate (f_t).

The input and forget gate control how the current cell state c_t is updated by selectively retaining important information from the input variable (x_t) and the previous hidden state (h_{t-1}). The forget gate acts as a mask to determine how much of the previous cell state c_{t-1} should be discarded, while the input gate i_t controls how much new information ($\check{c}_t$) is added to the cell state. Finally, the output gate decides which part of the updated cell state should be passed to the next hidden state (h_t) (Yu et al. 2019: 1237-1239). Mathematically, the system is described as:

Fig. 9: LSTM-Cell Representation

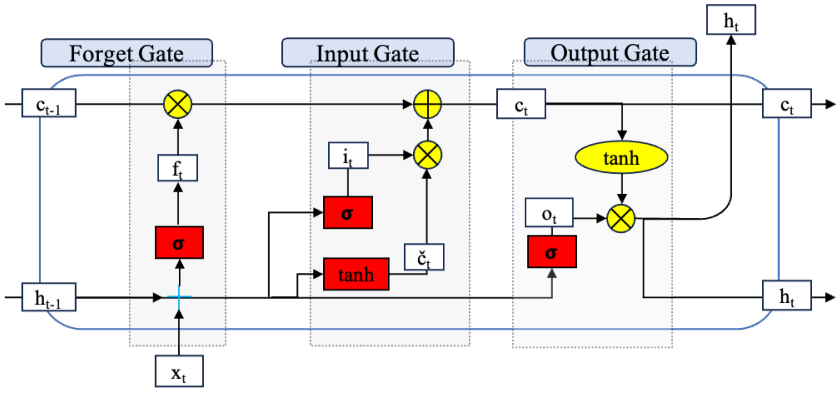


Figure based on the representation of Yu et al. (2019: 1240). The plus symbol in blue represents a concatenation of the input vectors and the state vector . The red boxes represents the neural nodes, with the respective standard operations, where the input vectors are multiplied by their respective set of weights, a bias term added and an activation function applied to the results. The σ represents the sigmoid activation, and the yellowed shape represents a point-wise operation. in this case represents a point-wise multiplication and a point-wise addition.

$$f_t = \sigma\left(\mathbf{W}_{fh}h_{t-1} + \mathbf{W}_{fx}x_t + b_f\right) \tag{6.15}$$

$$i_t = \sigma\left(\mathbf{W}_{ih}h_{t-1} + \mathbf{W}_{ix}x_t + b_i\right), \tag{6.16}$$

$$\check{c}_t = \tanh\left(\mathbf{W}_{\check{c}}h_{t-1} + \mathbf{W}_{\check{c}x}x_t + b_{\check{c}}\right), \tag{6.17}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \check{c}_t, \tag{6.18}$$

$$o_t = \sigma\left(\mathbf{W}_{oh}h_{t-1} + \mathbf{W}_{ox}x_t + b_o\right), \tag{6.19}$$

$$h_t = o_t \tanh\left(c_t\right) \tag{6.20}$$

$$y_t = h_t \tag{6.21}$$

Fig. 9 illustrates the system inside a LSTM cell. The red box illustrates the neural-nodes operations, the yellow shapes illustrate point wise operations, with $\oplus$ being additions and $\otimes$ multiplication. Like in other forms of neural networks, LSTM-cells can be staked to obtain multi-layers (cf. Yu et al. (2019: 1244-1249)).

In summary, LSTMs are well-suited for forecasting volatility due to their ability to capture long-term dependencies, such as the long-memory property in volatility discussed in Section 2. Additionally, the use of non-linear activation functions enables LSTMs to model complex, non-linear dynamics in the data. The gating mechanism allows the model to retain relevant historical information while filtering out noise.

In the next section, we will implement the models discussed in Section 6, provide details on the training process, and evaluate their performance. We will first assess the models on the training data to select the best set of parameters, fit them to the test set, and present the final evaluation.

6.2. Model Implementation and Evaluation

As we did with the econometric models, we begin this section by presenting the performance of the ML models evaluated on the training set. In the pursuit of obtaining the optimal set of hyperparameters for each of the ML models, we conducted grid-search, exhaustively trying all combinations of selected hyper-parameters. We cross-validated both the SVR and the XGBoost models in the same manner as with the GARCH-family models. In contrast, the LSTM models were were trained on the first 80% of the training set, with the remaining 20% used as the validation set, since the training procedure for deep learning models takes significantly longer than for the other two. To ensure positivity of the estimated variance, we trained the models on the logarithmic version of our volatility proxies and standardized the data using a normal-standard scaler to aid algorithm convergence. Each model was tested with up to five lags of the log-volatility proxy as covariate.

For a given volatility proxy, lag order, and security, we estimated 32 SVR, 216 XGBoost, and 108 LSTM models. For the SVR models, we tried two kernel functions: the linear kernel and the radial basis function kernel, the latter yielding better results for both volatility proxies. For the LSTM, we experimented with both single layer and two layers configurations. The single layer was only better at forecasting the $S.R._{.}^{DAX}$. To avoid overwhelming the reader, we refrain from listing the specific hyper-parameter configurations for each model. The grid-search parameters and the chosen set of specifications is presented to the reader in Appendix under Section A.2.2..

The standard implementation of SVR and XGBoost in Python only allows for one-step ahead forecasts. To produce multi-step forecasts, we used a recursive approach, feeding the forecasts generated at each step back into the model. The LSTM, however, was trained to forecast the entire 5-day horizon at once, using covariates available up to the forecast origin.

Tab. 7 presents the $\overline{MSE}$, their rankings, as well as the optimal lag order of the covariates achieved in our grid search. If the reader compares these results with the training

set evaluation table for the econometric models, they will see that the $\overline{MSE}$ of those models in forecasting the $S.R.$ is, on average, lower than that achieved by the ML models. In contrast, when the volatility proxy is the ${}_{adj.}R.V.$, the ML models performed better, on average, in the training set.

However, we should be cautious when comparing these models based on the training set, as, for the reasons previously explained, the LSTM specifications were not evaluated under the same conditions as the remaining models. Instead, we will focus on the test set evaluation of the ML models, whose performance metrics are listed in Tab. 8 and visually presented in terms of total MSE over the 5 forecasted days in Fig. 10.

We being our evaluation by discussing the $\overline{R^2}^{S.R.}_{MZ,DAX}$ of the LSTM model, which ranked second best among the compared models. Its superior average performance over the XGBoost is primarily due to higher R^2_{MZ} in the forecasts of the three last steps. However, it performed extreme poorly for the 1- and 2-step ahead forecasts, producing constant predictions for all observations. This lack of variation resulted in an R^2_{MZ} of exactly zero. One potential reason for this poor performance in the earlier steps is that the model was trained based on the $\overline{MSE}$ directly, and it may have learned that forecasting a constant value minimizes the overall error. Another possible explanation is that the model was not provided with sufficient covariate information to generate accurate and meaningful forecasts for a whole trading week.

Inspecting Fig. 10, we can also observe that this strategy of predicting constant values for the 1- and 2-step ahead $S.R.$, while focusing on later steps, allows the LSTM to achieve the lowest $\overline{MSE}$, with a value of 9.178%[4]. The SVR model, which achieved the highest $\overline{R^2}_{MZ,DAX}$, had the worst performance concerning the forecast error. However, based on the aSPA p-values presented in Tab. 9, the difference in predictive performance is only significant in favor of the LSTM against the XGBoost, with no significant difference between the SVR and XGBoost—an outcome that seems counterintuitive.

For the models used to forecast the $S.R._{S\&P500}$, the optimal LSTM model showed more variation in the forecasts than the optimal model for the $S.R._{DAX}$. It also achieved the best $\overline{R^2}_{MZ,S\&P500}$, though only slightly better than the SVR model (5.67% vs. 5.60%). In contrast, the LSTM had the worst $\overline{MSE}$, with the SVR model securing first place. The superior average predictive performance of the SVR over both the LSTM and XGBoost models was confirmed by the aSPA test, even at the 99% confidence level.

XGBoost, which ranked second, also outperformed the LSTM at the 10% significance level.

Turning to the evaluation of the models forecasting the ${}_{adj.}R.V.$, we observe the same pattern as with the econometric models: the explanatory power of the forecasts was generally higher, and the average forecast error lower, than for the models forecasting the $S.R.$. On average, across all models, $\overline{R^2}^{S.R.}_{MZ,DAX}$ was 6.06%, while $\overline{R^2}^{adj.R.V.}_{MZ,DAX}$ was 34.46%. For the S&P500, $\overline{R^2}^{S.R.}_{MZ,S\&P500}$ was 4.99%, compared to 31.15% average reached by the models forecasting the ${}_{adj.}R.V._{S\&P500}$. In terms of forecast error, the average $\overline{MSE}^{S.R.}_{DAX}$ was 9.57%[4], compared to $\overline{MSE}^{adj.R.V.}_{DAX}$ of 1.054%[4], while for the S&P500, the average $\overline{MSE}^{S.R.}_{S\&P500}$ was 5.73%[4], while the average $\overline{MSE}^{adj.R.V.}_{S\&P500}$ amounted to only 0.59%[4]. These differences suggest that the importance of the volatility proxy, as discussed for the econometric models, also applies to the ML models.

The evaluation of single model performances reveals that the lowest $\overline{MSE}^{adj.R.V.}_{DAX}$ was achieved by the SVR model, with a value of 0.969%[4]. However, despite the strong overall performance, Fig. 10 shows that the SVR did not perform best in forecasting one-step ahead ${}_{adj.}R.V.$, where the XGBoost took the lead. According to the aSPA p-values presented in Tab. (9), the superior performance in terms of $\overline{MSE}$ of the SVR was statistically significant at the 90% confidence level over the performance of the XGBoost and at the 95% confidence level over that of the LSTM. In contrast, the performance gap between the XGBoost, which achieved the second lowest $\overline{MSE}$ of 1.052%[4], and the LSTM, with a $\overline{MSE}$ of 1.1142%[4], was not statistically significant.

When evaluating the models using the $\overline{R^2}^{adj.R.V.}_{MZ,DAX}$ metric, the ranking mirrors the results based on the $\overline{MSE}$. The SVR leads with an $\overline{R^2}_{MZ,DAX}$ of 36.74%, followed by the XGBoost with 33.59% and LSTM with 33.07%.

For the S&P500, the ranking of models in forecasting the ${}_{adj.}R.V.$ follows the same pattern as for the DAX. The SVR outperforms both XGBoost and LSTM in both $\overline{R^2}_{MZ}$ and $\overline{MSE}$, achieving an $\overline{R^2}_{MZ}$ of 33.68% and an $\overline{MSE}$ of 0.573%[4]. The aSPA test further confirms the superior predictive ability of the SVR over the LSTM at the 99% level, whose $\overline{MSE}^{adj.R.V.}_{S\&P500}$ amounts to 0.602%[4].

In conclusion, as with the econometric models, determining the best machine learning model depends on the evaluation criterion. If the primary objective is minimizing forecast error, the SVR is the optimal model for forecasting the $S.R._{S\&P500}$. However, for the DAX, the aSPA test indicates no statistically significant difference in

performance between the models. Recommending the LSTM in its current form is dangerous, as its stronger performance stems from forecasting the first two periods as constants. Nonetheless, if the sole criterion is overall $\overline{MSE}$, the LSTM could still be considered a viable option.

When forecasting the ${}_{adj.}R.V.$ of the S&P500, there is no significant difference between XGBoost and SVR in terms of performance. However, since the SVR also achieves a higher $\overline{R^2}_{MZ}$, it would be our model of choice. Similarly, for the DAX, the SVR demonstrates superior predictive performance and the highest explanatory power among the models evaluated.

With the evaluation of both econometric and ML models now complete, we will compare the two approaches. In order for us to be able to provide a ranking for the models using monetary values, we will also use our financial application based on the value at risk as an evaluation criterion. Details about and results of this evaluation will be discussed in the following section.

Fig. 10: Cumulative h-step Ahead MSE (ML-Models)

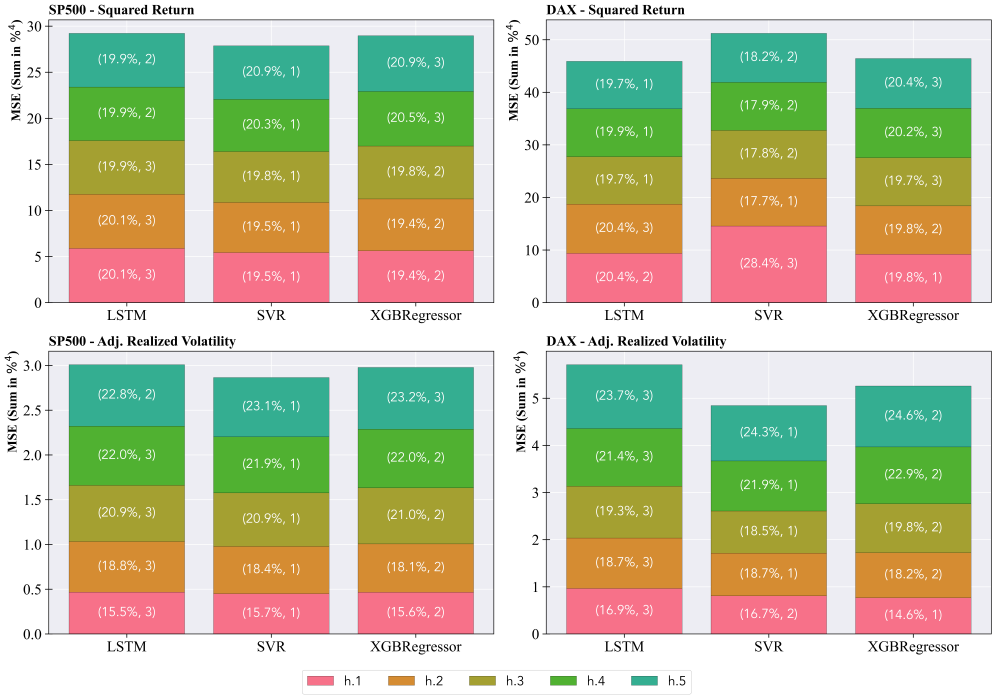


Figure illustrates the cumulative MSE of 1- to 5-step ahead forecasts from ML models for the SP500 (left column) and the DAX (right column). The upper row is related to the metrics computed using the $S.R.$, and the lower row the ${}_{adj.}R.V.$ as target variable. Each color represents a forecast horizon (h.1 to h.5), with the percentages indicating the contribution of each horizon to the total MSE. The second element in the tuple indicates the rank of the model at forecasting the h-th step volatility.

7. Results and Discussion

By now, we have evaluated the econometric and the ML models in isolation. This section compares and conclude our evaluation considering all models, with a special focus on the comparison between ML and the GARCH-family models.

As discussed in Section 6, the mean $\overline{MSE}^{S.R.}$ across all ML models is 9.57%[4] for the DAX and 5.73%[4] for the S&P500. In comparison, the average $\overline{MSE}^{S.R.}$ for the econometric models is 7.91%[4] for the DAX and 4.63%[4] for the S&P500, indicating that, on average, econometric models perform better when minimizing the squared errors using the $S.R.$ as the volatility proxy.

When using the $_{adj.}R.V.$ proxy, the mean $\overline{MSE}^{adj.R.V.}$ across all ML models is 1.054%[4] for the DAX and 0.59%[4] for the S&P500. Thus, a meta-comparison reveals that econometric models generally outperform ML models when $S.R.$ is the volatility proxy, whereas ML models perform better when the $_{adj.}R.V.$ is used.

These findings align with the $\overline{MSE}$ rankings shown in Tab.12. The four models with the lowest $\overline{MSE}$ all belong to the GARCH-family when using the $S.R.$. However, when using $_{adj.}R.V.$, the two best models for the DAX and the three best models for the S&P500 belong to a class of the ML models. The p-values in Tab. 9 for the DAX and 10 for the S&P500 support our results. We observe in the first table that the EGARCH model excelled in having the lowest $\overline{MSE}^{S.R.}_{DAX}$ compared against all other models, its average superior predictive ability can only be rejected against the GJR-GARCH, where no difference can be stated. Similar results are seen for the GJR-GARCH model, which obtained the second lowest $\overline{MSE}^{S.R.}_{DAX}$. In the case of the S&P500, the aSPA test shows no significant difference in the predictive performance between the GARCH-models when forecasting the $S.R.$, only the EGARCH model achieved better performance than the GJR-GARCH. In contrast, when the comparison is against the ML models, the p-values of the tests are almost zero, meaning that GARCH-family models outperformed all other ML models with 99% confidence.

An examination of the p-values of the aSPA test for models forecasting the $_{adj.}R.V.$ reveals a different pattern. The SVR achieved the lowest $\overline{MSE}^{adj.R.V.}_{DAX}$, followed closely by the XGBoost. The GJR-GARCH model ranked third, while ARCH performed the worst. However, the aSPA test fails to reject the null hypothesis that the predictive accuracy of the GJR-GARCH is at least as good as that of SVR or any other ML

model. A similar conclusion holds when comparing the GJR-GARCH with EGARCH. Nevertheless, both SVR and XGBoost show superior performance over the ARCH and GARCH, as indicated by the aSPA results. The difference in predictive performance is even more pronounced for the $\overline{MSE}^{adj.R.V.}_{S\&P500}$, with the SVR outperforming all other models, with 99% confidence.

Shifting our focus to the average $\overline{R^2}_{MZ}$ across all models, Section 5 revealed that the average $\overline{R^2}^{S.R.}_{MZ}$ for the DAX and the S&P500 was around 10.06% using the GARCH-family models. In contrast, ML models showed substantially lower values, averaging 6.06% for the DAX and only 4.99% for the S&P500. This suggests that, for both indices, forecasts from econometric models were, on average, more effective in explaining future variations in $S.R.$ than those from ML models. This is further supported by the rankings in Tab. 12, where the top four models for explanatory power of the S&P500, and three of the top four of the DAX volatility, belong to the GARCH-family. Using the ${}_{adj.}R.V.$ as the volatility proxy, the econometric models explained, on average, 33.54% of future DAX volatility and 35.34% of the S&P500. Meanwhile, ML models explained 34.45% of the DAX and 31.15% of the S&P500 future volatility. A closer look in the second column of Tab. 12 (second element of the tuples) shows that the ranking of $\overline{R^2}_{MZ}$ is more mixed than that of the $\overline{MSE}$: while three of the top four models for $\overline{R^2}^{adj.R.V.}_{MZ,S\&P500}$ belong to the GARCH-family, two of the best models for $\overline{R^2}^{adj.R.V.}_{MZ,DAX}$ are ML models. However, for both indices, the two best-performing models—EGARCH and GJR-GARCH—belong to the econometric class.

The (scaled) final value ($F.V.$) of the portfolios, computed as explained in Section 4.3.3 (see Eq. (4.15)), is presented in Tab. 11. Regardless of the volatility proxy used or the index examined, our weighting scheme generally led to portfolio final values that are less than the total amount invested. For the DAX, the model yielding the highest portfolio value when the volatility proxy was the $S.R.$ was the ARCH (€82,991.11), followed by GJR-GARCH (€82,986.91)[5]. The first fourth best positions were all occupied by an econometric model; the best position achieved by a ML model was the fifth, occupied by the SVR with a $F.V.$ of €82,555.11. When the ${}_{adj.}R.V._{DAX}$ was used, LSTM took the first place (€82,991.99), followed by the ARCH, GJR-GARCH, EGARCH and GARCH, SVR and XGBoost. This suggests that, on average, the econo-

[5] we refer to the models, but meant is the VaR application computed with the volatilities forecasted by these models

metric models performed better in our VaR financial application, with the LSTM being the exception.

For the S&P500, when using the $S.R._{S\&P500}$, the EGARCH model achieved the best performance, with a final portfolio value of $\$82,285.91$, followed by the ARCH. The next three best performers were the GARCH, GJR-GARCH, and SVR models, while the XGBoost produced the lowest *F.V.* ($\$81,384.72$). When the VaR boundaries were calculated based on the ${}_{adj.}R.V._{S\&P500}$ forecasts, the EGARCH excelled with a final value of $\$82,282.63$, followed by the GARCH ($\$82,304.84$), ARCH ($\$82,277.08$), and GJR-GARCH($\$82,265.86$); a ML model, namely the LSTM, appears only in the fifth position, with a value of $\$82,168.50$, followed by the XGBoost and SVR.

In summary, it is difficult to state conclusively that econometric models performed better than ML models. The superiority of one model over another depends on the forecasting objective and which volatility proxy is available to the forecaster. However, when we assign equal weights to the three evaluation dimensions—$\overline{MSE}$, $\overline{R^2}_{MZ}$, and the *F.V.* from our VaR application— the overall ranking of the models are provided in the Total-Rank column of Tab. 12.

For the DAX, when forecasting the $S.R.$, the GJR-GARCH$(5,4)$ model with a constant mean and GED-distributed innovations performed best overall. The second-best model was the EGARCH$(1,1)$ with a constant mean and GED-distributed innovations, followed by the ARCH$(1,1)$ with a constant mean and GED-distributed innovations. The best performing ML model(s) were the LSTM and SVR, which shared the fifth overall best performance.

When the ${}_{adj.}R.V._{DAX}$ is the proxy used, the GJR-GARCH$(5,1)$ model with constant mean and normally distributed innovations ranked first, followed by the EGARCH$(1,1)$ with a zero mean and GED-distributed innovations. The next three positions were taken by the SVR, XGBoost, and LSTM models.

For the S&P500, using the $S.R._{S\&P500}$ as volatility proxy, the first best overall performance was achieved by the EGARCH$(2,1)$ with a zero mean and GED-distributed innovations [6], followed by the GJR-GARCH$(5,2)$ with constant mean and normally distributed innovations. The best-performing ML model was the SVR, which ranked fifth, while the XGBoost and LSTM models occupied the two last positions. When the volatility proxy used was the ${}_{adj.}R.V._{S\&P500}$, the EGARCH$(3,1)$ with a zero mean and

[6] After re-estimation on the entire training set, the EGARCH was found to have an order of $(1,1)$.

normally distributed innovations led the rank. The GARCH(3, 1) with zero mean and normally distributed innovations ranked second, and the SVR model ranked third. The lowest overall performance was seen with the LSTM model.

In the next section, we synthesize the main points from our comprehensive evaluation and provide concluding remarks on the implications for volatility modeling and future research directions.

8. Conclusion

Our empirical analysis revealed that both conventional econometric models and newer machine learning approaches are valuable for volatility forecasting, though their performance varies depending on the context—specifically, the volatility proxy used and the application chosen by the forecasters. In particular, ML models outperformed the econometric models in minimizing the squared error when forecasting the ${}_{adj.}R.V.$ (computed from high-frequency, intra-day returns) for both indices analyzed. The SVR and XGBoost models achieved the top two performances in these cases. However, when data was limited to daily frequency and the volatility proxy was the $S.R.$, econometric models outperformed the ML models for both indices, particularly models that account for the leverage effect.

If the researcher's goal is to explain the movements and trends in future volatility, econometric models performed, on average, better than ML models when using $S.R.$ as volatility proxy. When ${}_{adj.}R.V.$ was the target, the results were more mixed, though econometric models like EGARCH and GJR-GARCH occupied the first places irregardless of the volatility proxy used.

In our financial application, where we created a fictive portfolio value based on VaR intervals and the number of VaR violations, econometric models consistently secured the top four positions. The only exception was the LSTM model, which yielded the best performance for a portfolio based on the DAX, using the ${}_{adj.}R.V.$ forecasts to compute the VaR boundaries.

When evaluating performance from a broader perspective, giving equal weight to each metric considered, we conclude that econometric models performed better than ML models when $S.R.$ was the volatility proxy. When ${}_{adj.}R.V.$ was used, the ranking was more mixed, with the top two positions held by econometric models, and the next two

by ML models. However, if only first-place performance is considered, econometric models outperformed the ML. These results are consistent for the DAX and the S&P500.

Our study focused on univariate modeling, but ML models allow for the inclusion of other covariates, which may lead to better results. For the ML models, we faced a trade-off between hyper-parameter tuning and the number of lagged values of volatility to include as regressors. While the econometric models had up to ten covariates (excluding the ARCH), with five lagged values of the squared errors and five lagged values of conditional volatility , the ML models had at most five covariates, accounting only for past values of the volatilities. For future research, we recommend comparing models that include other covariates that could potentially influence volatility.

Appendix

A. Tables

A.1. Descriptive Statistics

Section presents three tables, the first being the "Descriptive Statistics of DAX and S&P500 Daily Returns", the second "Daily Price Realizations – Counts", and the third is the "Correlation Coefficients".

Tab. 1: Descriptive Statistics of DAX and S&P500 Daily Returns

Security:	DAX			S&P500		
Sample:	Total	Train	Test	Total	Train	Test
$\overline{r_t}$ (%)	0.019	0.017	0.031	0.030	0.029	0.035
$s[r_t]$ $\left(\%^2\right)$	1.902	2.094	1.135	1.653	1.774	1.1733
$\overline{S.R.}$ $\left(\%^2\right)$	1.902	2.094	1.131	1.654	1.774	1.730
${}_{adj.}\overline{R.V.}$ $\left(\%^2\right)$	1.795	1.995	0.995	1.25	1.3630	0.8150
Kurtosis (r)	8.133	7.913	4.784	12.246	13.048	1.739
Skewness (r)	−0.197	−0.213	0.026	−0.520	−0.557	−0.207
JB p-value	0.00	0.00	0.00	0.00	0.00	0.00
Arch-test p-val	0.00	0.00	0.00	0.00	0.00	0.00
Obs.	4149	3319	830	4113	3290	823

Table presents descriptive statistics for the DAX and SP500 returns (r_t), distinguished by set type. Bars above the variables represent their mean values. We present three different measures of variance: the sample variance, calculated as the sum of the squared mean deviations $s[r_t]$, the adjusted realized volatility adj. (${}_{adj.}R.V.$), and the squared returns ($S.R.$). Additionally, the table displays the p-values for the Jarque-Bera test for normality of the returns, as well as for the test for heteroscedasticity in the squared returns (ARCH test).

Tab. 2: Intra-Day Price Realizations – Descriptive Statistics

Security	Days	Mean	Std.	Min.	25%	50%	75%	Max.
DAX	4149	106.04	2.91	64	106	106	106	113
S&P500	4113	79.73	2.77	47	80	80	80	90

The table presents descriptive statistics on the number of intraday observations for both the DAX and S&P500. The column "Days" indicates the total number of trading days considered in the dataset. A regular trading day for the S&P500 typically includes 78 close prices at 5-minute intervals between 9:30 a.m. and 4:00 p.m., while a typical trading day for the DAX contains 102 close prices between 9:00 a.m. and 5:30 p.m.. As shown in the table, the average number of intraday observations recorded in the dataset reflects these trading hours. The average number of intraday observations for the DAX and S&P500 are 106.04 and 79.73, respectively, amounting to a total of 439,957 intraday observations for the DAX and 327,919 for the S&P500 between 02 January 2008 and 03 May 2024.

Tab. 3: Correlation Coefficients

Correlation %	r_{DAX}	$S.R._{DAX}$	${}_{adj.}R.V._{DAX}$	$r_{S\&P500}$	$S.R._{S\&P500}$	${}_{adj.}R.V._{S\&P500}$
r_i	1	—	—	1	—	—
$S.R._i$	−5.31	1	—	−12.55	1	—
${}_{adj.}R.V._i$	−12.8	51.77	1	−13.22	59.75	1

Table presents the correlation coefficients between returns (r), Squared Returns (S.R.), and Adjusted Realized Volatility (R.V.) for the DAX and SP500 indices. The subscript i refers to the respective index accompanying the variables in the columns.

A.2. Train-Test Results

A.2.1. GARCH-Family

Tab. 4: Cross Validation Results—Train Set (DAX)

—	Mean	(P, Q)	Dist.	Vol-Proxy	$\overline{MSE}\left(\%^4\right)$	Rank
ARCH						
HAIC	Constant	$(0, 22)$	Normal	(S.R.; Adj. R.V.)	(25.173; 5.263)	(52; 67)
HBIC	Constant	$(0, 14)$	Normal	(S.R.; Adj. R.V.)	(25.162; 5.061)	(49; 32)
C.V.	Constant	$(0, 11)$	GED	S.R.	24.817	1
C.V.	Zero	$(0, 10)$	Normal	Adj. R.V.	4.634	1
GARCH						
HAIC	Constant	$(5, 3)$	Normal	(S.R.; Adj. R.V.)	(24.678; 4.731)	(2; 15)
HBIC	Constant	$(1, 1)$	Normal	(S.R.; Adj. R.V.)	(24.829; 4.732)	(22; 17)
C.V.	Constant	$(5, 4)$	GED	S.R.	24.672	1
C.V.	Constant	$(3, 3)$	Normal	Adj. R.V.	4.704	1
EGARCH						
HAIC	Zero	$(1, 5)$	GED	(S.R.; Adj. R.V.)	(24.264; 4.248)	(73; 89)
HBIC	Zero	$(1, 2)$	Normal	(S.R.; Adj. R.V.)	(24.022; 3.733)	(27; 25)
C.V.	Constant	$(1, 1)$	GED	S.R.	23.904	1
C.V.	Zero	$(1, 1)$	GED	Adj. R.V.	3.604	1
GJR-GARCH						
HAIC	Zero	$(1, 1)$	GED	(S.R.; Adj. R.V.)	(24.695; 4.872)	(70; 73)
HBIC	Zero	$(1, 1)$	Normal	(S.R.; Adj. R.V.)	(24.517; 4.601)	(24; 31)
C.V.	Constant	$(5, 3)$	Normal	S.R.	24.194	1
C.V.	Constant	$(5, 1)$	Normal	Adj. R.V.	4.485	1

The table presents the optimal model specifications for the GARCH-family models applied to the DAX, based on HAIC, HBIC, and cross-validation (CV). The models are evaluated on the training set using both squared returns (S.R.) and adjusted realized volatility (Adj. R.V.). The information criteria values are obtained by fitting the models on the entire training data and generating variance forecasts. The ranking-column is related to the model performance inside its own class, i.e., an EGARCH model ranked 1 is the best model among all the EGARCH models for a given volatility proxy.

Tab. 5: Cross-Validation Results—Train Set (S&P500)

Model Specification				Evaluation		
—	Mean	(P, Q)	Dist.	Vol-Proxy	Avg. MSE	Rank
ARCH						
HAIC	Constant	$(0, 11)$	GED	(S.R.; Adj. R.V.)	(23.483; 8.070)	(88; 97)
HBIC	Constant	$(0; 10)$	GED	(S.R.; Adj. R.V.)	(23.417; 8.150)	(67; 99)
C.V.	Constant	$(0, 29)$	Normal	S.R.	22.984	1
C.V.	Zero	$(0, 30)$	Normal	Adj. R.V.	6.503	1
GARCH						
HAIC	Constant	$(1, 2)$	GED	(S.R.; Adj. R.V.)	(22.874; 7.280)	(63; 80)
HBIC	Constant	$(1, 1)$	GED	(S.R.; Adj. R.V.)	(22.87; 7.06)	(57; 58)
C.V.	Constant	$(2, 3)$	Normal	S.R.	22.493	1
C.V.	Zero	$(3, 1)$	Normal	Adj. R.V.	6.245	1
EGARCH						
HAIC	Constant	$(1, 5)$	GED	(S.R.; Adj. R.V.)	(21.948; 4.923)	(65; 61)
HBIC	Constant	$(1, 1)$	GED	(S.R.; Adj. R.V.)	(21.215; 4.233)	(4; 23)
C.V.	Zero	$(2, 1)$	GED	S.R.	21.193	1
C.V.	Zero	$(4, 1)$	Normal	Adj. R.V.	3.911	1
GJR-GARCH						
HAIC	Constant	$(2, 2)$	GED	(S.R.; Adj. R.V.)	(22.781; 8.593)	(38; 65)
HBIC	Constant	$(1, 1)$	GED	(S.R.; Adj. R.V.)	(23.00; 8.156)	(68; 49)
C.V.	Constant	$(5, 2)$	Normal	S.R.	22.335	1
C.V.	Constant	$(5, 1)$	Normal	Adj. R.V.	7.326	1

This table presents the optimal model specifications for the GARCH-family models applied to the SP500, based on HAIC, HBIC, and cross-validation (C.V.). The models are evaluated on the training set using both squared returns (S.R.) and adjusted realized volatility (Adj. R.V.). The information criteria values are obtained by fitting the models to the entire training dataset and generating variance forecasts. The "Ranking" column refers to each model's performance within its class, meaning that an EGARCH model ranked 1 is the best EGARCH model for a given volatility proxy. Values in tuples for $\overline{MSE}$ and Rank correspond to the volatility proxy, as indicated in the "Vol Proxy" column.

Tab. 6: Mean Squared Error and MZ-R^2—Test Set Results (GARCH-Family)

Model	$\overline{MSE}\left(\%^4\right)$	$\overline{R}^2_{MZ}\,(\%)$	Rank: $\overline{MSE}$	Rank: $\overline{R}^2_{MZ}$
Security: DAX—Volatility Proxy: (S.R.; Adj. R.V.)				
ARCH	(8.230; 1.529)	(7.89; 27.08)	(4; 4)	(3; 4)
GARCH	(8.131; 1.344)	(7.55, 27.41)	(3; 3)	(4; 3)
EGARCH	(7.655; 1.078)	(12.38; 41.61)	(2; 2)	(2; 1)
GJR-GARCH	(7.641; 1.052)	(12.43; 38.04)	(1; 1)	(1; 2)
Security: S&P500—Volatility Proxy: (S.R.; Adj. R.V.)				
ARCH	(4.741; 0.941)	(8.85; 31.40)	(4; 2)	(4; 4)
GARCH	(4.718; 0.897)	(8.99; 31.87)	(3; 1)	(3; 3)
EGARCH	(4.613; 0.946)	(11.96; 40.02)	(1; 3)	(1; 1)
GJR-GARCH	(4.705; 0.972)	(10.46; 38.01)	(2; 4)	(2; 2)

The table presents the average MSE and R^2.The results are based solely on the test set. The first element in the tuples represents the performance and rank of the models using Squared Return as the volatility proxy, while the second element pertains to forecasts using Adjusted Realized Volatility. Models ranked first indicate the best performance among all considered. The model specifications correspond to those ranked first in Tab. 4 for the DAX and acTab. 5 for the SP500.

A.2.2. ML-Models

For the machine learning models we used the following grid search (for lagged values of the target variable ranging from 1 to 5):

- SVR: $\varepsilon \in \{0.005, 0.01, 0.5, 0.8\}$; kernel$\in$ {linear; rbf}, and $C \in \{1, 10, 25, 50\}$. The chosen set was (given the volatility proxy used):
  - $S.R._{DAX}$: $\varepsilon = 0.5$; kernel= rbf; $C = 10$.
  - ${}_{adj.}R.V._{DAX}$: $\varepsilon = 0.005$; kernel = rbf; $C = 1$.
  - $S.R._{S\&P500}$: $\varepsilon = 0.01$; kernel = rbf; $C = 1$.
  - ${}_{adj.}R.V._{S\&P500}$: $\varepsilon = 0.005$; kernel = rbf; $C = 10$.
- XGBoost: n_estimator $\in \{50, 100, 200\}$, max_depth $\in \{3, 5, 7\}$, learning_rate $\in \{0.01, 0.05, 0.1\}$, subsample $\in \{0.8, 0.1\}$, colsample_bytree $\in \{0.8, 1.0\}$, min_child_weight $\in \{3, 7\}$.
  - $S.R._{DAX}$: n_estimator = 50; max_depth = 7; learning_rate = 0.05; subsample = 1.0; colsample_bytree = 1.0; min_child_weight = 3

- ${}_{adj.}R.V._{DAX}$:n_estimator = 50; max_depth = 7; learning_rate = 0.1; subsample = 1.0; colsample_bytree = 0.8; min_child_weight = 3
- $S.R._{S\&P500}$: n_estimator = 50; max_depth = 7; learning_rate = 0.1; subsample = 0.8; colsample_bytree = 0.8; min_child_weight = 3
- ${}_{adj.}R.V._{S\&P500}$: n_estimator = 100; max_depth = 7; learning_rate = 0.1; subsample = 1.0; colsample_bytree = 0.8; min_child_weight = 3

- LSTM: batch_size $\in \{1, 32, 64\}$; epochs$\in \{1, 10, 20\}$, dropout$\in \{0; 0.20\}$, layers $\in \{1; 2\}$, units $\in \{32; 64\}$. The model has always the same dropout rate after all layers. The last activation function is linear and outputs 5 values, i.e., one for each day of the forecast horizon. An epoch of, for example, 10, means that the model trains all epochs from 1 to 10, so that the optimal epoch is selected between these values.

  If more than one layer is used, we try all combinations of units, i.e $\{32; 32\}$, $\{32; 64\}$, $\{64; 32\}$, $\{64; 64\}$.

  - $S.R._{DAX}$: batch_size = 64; epochs= 5, dropout= 0, layers = 1, units = 64.
  - ${}_{adj.}R.V._{DAX}$:batch_size = 32; epochs= 20, dropout= 0, layers = 2, units = $(64, 64)$.
  - $S.R._{S\&P500}$: batch_size = 1; epochs= 9, dropout= 0.2, layers = 2, units = $(64, 64)$.
  - ${}_{adj.}R.V._{S\&P500}$: batch_size = 1; epochs= 9, dropout= 0.2, layers = 2, units = $(32, 64)$

Tab. 7: Train-Set Evaluation (ML-Models)

	$\overline{MSE}\left(\%^4\right)$	Lag Order	Rank: $\overline{MSE}$
DAX—Volatility Proxy (S.R.; Adj. R.V.)			
SVR	(27.446; 4.120)	(5; 2)	(1; 3)
XGBoost	(27.955; 3.999)	(5; 5)	(2; 2)
LSTM	(29.00; 3.889)	(5; 4)	(3; 1)
S&P500—Volatility Proxy (S.R.; Adj. R.V.)			
SVR	(24.846; 3.744)	(5; 4)	(1; 2)
XGBoost	(25.130; 3.760)	(5; 5)	(2; 3)
LSTM	(25.803; 3.688)	(5; 5)	(3; 1)

Table presents the performance of the top-performing model specifications in the training set, as determined by the grid search, based on $\overline{MSE}$ values. The first value in each tuple corresponds to the model's performance when using the Squared Returns (S.R.) as the target, while the second value reflects the performance with Adjusted Realized Volatility (Adj. R.V.) as the target. The performance metrics for the SVR and XGBoost models were evaluated using cross-validation, while the performance of the LSTM model was assessed using an 80%-20% train-validation split.

Tab. 8: Test Set Evaluation (ML-Models)

	$\overline{MSE}\left(\%^4\right)$	Rank. $\overline{MSE}$	$\overline{R^2}_{MZ}\,(\%)$	Rank. $\overline{R^2}_{MZ}$
DAX—Volatility Proxy (S.R.; Adj. R.V.)				
SVR	(10.245; 0.969)	(3; 1)	(11.13; 36.74)	(1; 1)
XGBoost	(9.286; 1.052)	(2; 2)	(3.061; 33.59)	(3; 2)
LSTM	(9.178; 1.142)	(1; 3)	(3.99; 33.07)	(2; 3)
S&P500—Volatility Proxy (S.R.; Adj. R.V.)				
SVR	(5.574; 0.573)	(1; 1)	(5.60; 33.68)	(2; 1)
XGBoost	(5.793; 0.596)	(2; 2)	(3.70; 31.76)	(3; 2)
LSTM	(5.845; 0.602)	(3; 3)	(5.67; 28.01)	(1; 3)

Table presents the performance of the top-performing model specifications in the test set, as determined by the grid search, based on $\overline{MSE}$ values. The first value in each tuple corresponds to the model's performance when using the Squared Returns (S.R.) as the target, while the second value reflects the performance with Adjusted Realized Volatility (Adj. R.V.) as the target.

A.3. Test for Average Superior Predictive Ability (Results: P-Values)

In this section we present the tables with the results of the p-values with the tests for average superior predictive ability. The section contains 2 tables, the first presents the results for the models used to forecasts the $S.R.$ and the ${}_{adj.}R.V.$ of the DAX, and the second presents the results for the S&P500.

Tab. 9: aSPA-test P-Values (DAX)

DAX (aSPA p-Values)							
Volatility Proxy: $S.R.$							
$j \backslash i$	ARCH	GARCH	EGARCH	GJR-GARCH	SVR	XGBoost	LSTM
ARCH	—	0.733	0.943	0.948	0.123	0.133	0.152
GARCH	0.267	—	0.962	0.972	0.109	0.091*	0.106
EGARCH	0.057*	0.038**	—	0.609	0.078*	0.039**	0.045**
GJR-GARCH	0.052*	0.028**	0.391	—	0.077*	0.035**	0.041**
SVR	0.877	0.891	0.922	0.923	—	0.8192	0.8442
XGBoost	0.867	0.909	0.961	0.965	0.181	—	0.9891
LSTM	0.848	0.894	0.955	0.9586	0.156	0.011**	—
Volatility Proxy: ${}_{adj.}R.V.$							
ARCH	—	0.926	0.937	0.953	0.957	0.96	0.925
GARCH	0.074*	—	0.987	0.935	0.943	0.935	0.863
EGARCH	0.063*	0.103	—	0.733	0.75	0.56	0.29
GJR-GARCH	0.047**	0.065*	0.267	—	0.721	0.501	0.177
SVR	0.043**	0.057*	0.250	0.279	—	0.097*	0.015**
XGBoost	0.04**	0.065*	0.440	0.499	0.903	—	0.102
LSTM	0.075*	0.137	0.710	0.832	0.985	0.898	—

Table presents the p-values from the test for Average Superior Predictive Ability (aSPA) applied to the DAX $\overline{MSE}$. The null hypothesis asserts that the forecasts from model i (columns) are at least as accurate as those from model j (rows). If the null hypothesis is rejected, model j outperforms model i in terms of the average Mean Squared Error ($\overline{MSE}$).

Tab. 10: aSPA-test P-Values (S&P500)

S&P500 – aSPA (P-Values)							
Volatility Proxy: $S.R.$							
$j \backslash i$	ARCH	GARCH	EGARCH	GJR-GARCH	SVR	XGBoost	LSTM
ARCH	—	0.82	0.835	0.607	0.0^{***}	0.0^{***}	0.0^{***}
GARCH	0.18	—	0.801	0.541	0.0^{***}	0.0^{***}	0.0^{***}
EGARCH	0.165	0.199	—	0.066^{*}	0.0^{***}	0.0^{***}	0.0^{***}
GJR-GARCH	0.393	0.459	0.934	—	0.0^{***}	0.0^{***}	0.0^{***}
SVR	1.0	1.0	1.0	1.0	—	0.0^{***}	0.0^{***}
XGBoost	1.0	1.0	1.0	1.0	1.0	—	0.057^{*}
LSTM	1.0	1.0	1.0	1.0	1.0	0.943	—
Volatility Proxy: $_{adj.}R.V.$							
ARCH	—	0.97	0.481	0.412	0.998	0.997	0.996
GARCH	0.03^{**}	—	0.315	0.286	0.998	0.997	0.996
EGARCH	0.519	0.685	—	0.348	1.0	1.0	0.999
GJR-GARCH	0.588	0.714	0.652	—	0.996	0.995	0.994
SVR	0.002^{***}	0.002^{***}	0.0^{***}	0.004^{***}	—	0.178	0.005^{***}
XGBoost	0.003^{***}	0.003^{***}	0.0^{***}	0.005^{***}	0.822	—	0.4
LSTM	0.004^{***}	0.004^{***}	0.001^{***}	0.006^{***}	0.995	0.6	—

Table presents the p-values from the test for Average Superior Predictive Ability (aSPA) applied to the SP500 $\overline{MSE}$. The null hypothesis asserts that the forecasts from model i (columns) are at least as accurate as those from model j (rows). If the null hypothesis is rejected, model j outperforms model i in terms of the average Mean Squared Error ($\overline{MSE}$).

A.4. Final Portfolio Values (Value at Risk—Application)

Tab. 11: End-Value (Portfolio)

	DAX (S.R.; Adj. R.V.)		S&P500 (S.R.; Adj. R.V.)	
Model	Portfolio Value in €	Rank	Portfolio Value in $	Rank
ARCH	(82,991.11; 82,991.13)	(1; 2)	(82,276.87; 82,277.08)	(2; 3)
GARCH	(82,984.78; 82,984.24)	(4; 5)	(82,272.51; 82,277.35)	(3; 2)
EGARCH	(82,985.20; 82,984.54)	(3; 4)	(82,285.91; 82,282.63)	(1; 1)
GJR-GARCH	(82,986.91; 82,988.25)	(2; 3)	(82,265.77; 82,265.86)	(4; 4)
SVR	(82,555.11; 82,937.17)	(6; 6)	(81,649.97; 82,111.01)	(5; 7)
XGBoost	(82,396.41; 82912.71)	(7; 7)	(81,384.72; 82,146.76)	(7; 6)
LSTM	(82,623.20; 82991.99)	(5; 1)	(81,427.00; 82,168.50)	(6; 5)

Table illustrate the final value of our fictive portfolios for the DAX and SP500. The first element in the tuple represents the total value when the VaR-boundaries were created using forecasts made for the $S.R.$ and the second element whent the forecasts were made using the ${}_{adj.}R.V.$.

A.5. Evaluation: Final Ranking

Tab. 12: Final Ranking

	Rank. $\overline{MSE}$	Rank. $\overline{R^2}_{MZ}$	Rank. F.V.	Total Rank
DAX—Volatility Proxy (S.R.; Adj. R.V.)				
ARCH	(4; 7)	(4; 7)	(1; 2)	(3; 6)
GARCH	(3; 6)	(5; 6)	(4; 5)	(4; 7)
EGARCH	(2; 4)	(2; 1)	(3; 4)	(2; 2)
GJR-GARCH	(1; 3)	(1; 2)	(2; 3)	(1; 1)
SVR	(7; 1)	(3; 3)	(6; 6)	(5.5; 3)
XGBoost	(6; 2)	(7; 4)	(7; 7)	(7; 5)
LSTM	(5; 5)	(6; 5)	(5; 1)	(5.5; 4)
S&P500—Volatility Proxy (S.R.; Adj. R.V.)				
ARCH	(4; 5)	(4; 6)	(2; 3)	(4; 6)
GARCH	(3; 4)	(3; 4)	(3; 2)	(3; 2)
EGARCH	(1; 6)	(1; 1)	(1; 1)	(1; 1)
GJR-GARCH	(2; 7)	(2; 2)	(4; 4)	(2; 4.5)
SVR	(5; 1)	(6; 3)	(5; 7)	(5; 3)
XGBoost	(6; 2)	(7; 5)	(7; 6)	(7; 4.5)
LSTM	(7; 3)	(5; 7)	(6; 5)	(6; 7)

Table presents the ranking of the models according to each one of our evaluation methods. The last column (Total Rank), was created by suming the individual ranks. Then rebasing the ranking such that it goes from 1 to 7. When two models the same ranking, the value value is given as decimals. For example, if two models have ranking 4, the value displayed will be 4.5, and 4.5.

A.6. GARCH-Models Estimation Output (Coefficients)

A.6.1. Security: S&P500

Volatility Proxy: Squared Return The tables below present the results of the regression estimation for the GARCH-family of models using the $S.R.$ as the volatility proxy of the S&P500 returns. The tables have the following order: ARCH, GARCH, EGARCH, GJR-GARCH. For all the models, the α coefficients are related to past values of ε_t^2 and β coefficients are related to past values of the volatility. In the case of the GJR-GARCH and EGARCH the γ coefficients are responsible for accounting for the

leverage effect.

Tab. 13: ARCH(29)—Estimation Results (S&P500)

Target: Squared Return; Security: S&P500			
–	Coefficient	Std. Error	P-value
μ	0.0743***	0.0137	0.0000
ω	0.1444***	0.0313	0.0000
α_1	0.1209***	0.0345	0.0005
α_2	0.1636***	0.0304	0.0000
α_3	0.1072***	0.0289	0.0002
α_4	0.1514***	0.0454	0.0008
α_5	0.0527**	0.0233	0.0236
α_6	0.0491**	0.0232	0.0343
α_7	0.0689**	0.0328	0.0354
α_8	0.0370	0.0227	0.1029
α_9	0.0334	0.0214	0.1190
α_{10}	0.0402*	0.0219	0.0657
α_{11}	0.0115	0.0200	0.5655
α_{12}	0.0000	0.0186	1.0000
α_{13}	0.0000	0.0178	1.0000
α_{14}	0.0181	0.0213	0.3952
α_{15}	0.0000	0.0273	1.0000
α_{16}	0.0000	0.0210	1.0000
α_{17}	0.0115	0.0219	0.5998
α_{18}	0.0000	0.0210	1.0000
α_{19}	0.0071	0.0197	0.7181
α_{20}	0.0000	0.0166	1.0000
α_{21}	0.0000	0.0173	1.0000
α_{22}	0.0116	0.0161	0.4686
α_{23}	0.0000	0.0138	1.0000
α_{24}	0.0000	0.0444	1.0000
α_{25}	0.0000	0.0325	1.0000
α_{26}	0.0000	0.0102	1.0000
α_{27}	0.0081	0.0168	0.6297
α_{28}	0.0000	0.0226	1.0000
α_{29}	0.0343**	0.0156	0.0280

Table presents the estimation results for the ARCH(29) model with a constant mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 14: GARCH(2, 3)—Estimation Results (S&P500)

Target: Squared Return; Security: S&P500			
–	Coefficient	Std. Error	P-value
μ	0.0753***	0.0135	0.0000
ω	0.0580***	0.0149	0.0001
α_1	0.1161***	0.0341	0.0007
α_2	0.1750***	0.0302	0.0000
α_3	0.0321	0.0394	0.4152
β_1	0.0384	0.1186	0.7459
β_2	0.6065***	0.0889	0.0000

Table presents the estimation results for the GARCH(2, 3) model with a constant mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 15: EGARCH(2, 1)—Estimation Results (S&P500)

Target: Squared Return; Security: S&P500			
–	Coefficient	Std. Error	P-value
ω	0.0102**	0.0042	0.0148
α_1	0.2029***	0.0239	0.0000
γ_1	-0.1758***	0.0216	0.0000
β_1	0.9678***	0.0825	0.0000
β_2	0.0000	0.0819	1.0000
ν	1.2859***	0.0519	0.0000

Table presents the estimation results for the EGARCH(2, 1) model with a zero mean and GED-distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 16: GJR-GARCH(5, 2)—Estimation Results (S&P500)

Target: Squared Return; Security: S&P500			
–	Coefficient	Std. Error	P-value
μ	0.0383***	0.0147	0.0093
ω	0.0570	0.0430	0.1848
α_1	0.0397	0.1108	0.7201
α_2	0.0285	0.0676	0.6736
γ_1	0.1558	0.1932	0.4202
γ_2	0.2452***	0.0705	0.0005
β_1	0.0000	2.0902	1.0000
β_2	0.6760*	0.3573	0.0585
β_3	0.0000	2.8673	1.0000
β_4	0.0000	0.2955	1.0000
β_5	0.0108	1.1141	0.9923

Table presents the estimation results for the GJR-GARCH(5,2) model with a constant mean and GED-distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Volatility Proxy: Adj. Realized Volatility The tables below present the results of the regression estimation for the GARCH-family of models using the ${}_{adj.}R.V.$ as the volatility proxy of the S&P500 returns. The tables have the following order: ARCH, GARCH, EGARCH, GJR-GARCH. For all the models, the α coefficients are related to past values of ε_t^2 and β coefficients are related to past values of the volatility. In the case of the GJR-GARCH and EGARCH the γ coefficients are responsible for accounting for the leverage effect.

Tab. 17: ARCH(30)—Estimation Results (S&P500)

Target: Adj. Realized Volatility; Security: S&P500			
–	Coefficient	Std. Error	P-value
ω	0.1492***	0.0332	0.0000
α_1	0.1103***	0.0312	0.0004
α_2	0.1593***	0.0294	0.0000
α_3	0.1074***	0.0282	0.0001
α_4	0.1440***	0.0437	0.0010
α_5	0.0554**	0.0243	0.0228
α_6	0.0514**	0.0237	0.0302
α_7	0.0664**	0.0315	0.0350
α_8	0.0370*	0.0225	0.0996
α_9	0.0380*	0.0222	0.0867
α_{10}	0.0407**	0.0207	0.0495
α_{11}	0.0079	0.0178	0.6585
α_{12}	0.0000	0.0194	1.0000
α_{13}	0.0000	0.0166	1.0000
α_{14}	0.0183	0.0215	0.3948
α_{15}	0.0010	0.0262	0.9696
α_{16}	0.0000	0.0211	1.0000
α_{17}	0.0044	0.0213	0.8350
α_{18}	0.0000	0.0226	1.0000
α_{19}	0.0125	0.0208	0.5457
α_{20}	0.0000	0.0168	1.0000
α_{21}	0.0000	0.0171	1.0000
α_{22}	0.0175	0.0161	0.2766
α_{23}	0.0000	0.0128	1.0000
α_{24}	0.0000	0.0421	1.0000
α_{25}	0.0000	0.0353	1.0000
α_{26}	0.0000	0.0110	1.0000
α_{27}	0.0048	0.0170	0.7783
α_{28}	0.0000	0.0199	1.0000
α_{29}	0.0376**	0.0160	0.0184
α_{30}	0.0000	0.0218	1.0000

Table presents the estimation results for the ARCH(30) model with a zero mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 18: GARCH(3, 1)—Estimation Results (S&P500)

Target: Adj. Realized Volatility; Security: S&P500			
–	Coefficient	Std. Error	P-value
ω	0.0278***	0.0070	0.0001
α_1	0.1544***	0.0311	0.0000
β_1	0.8286***	0.2362	0.0005
β_2	0.0000	0.2117	1.0000
β_3	0.0000	0.1298	1.0000

Table presents the estimation results for the GARCH(3, 1) model with a zero mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 19: EGARCH(4, 1)—Estimation Results (S&P500)

Target: Adj. Realized Volatility; Security: S&P500			
–	Coefficient	Std. Error	P-value
ω	0.0106	0.0246	0.6667
α_1	0.2161	0.5633	0.7012
γ_1	-0.1615	0.3614	0.6549
β_1	0.9472	3.9002	0.8081
β_2	0.0000	1.6133	1.0000
β_3	0.0000	6.3954	1.0000
β_4	0.0168	4.2102	0.9968

Table presents the estimation results for the EGARCH(4, 1) model with a zero mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 20: GJR-GARCH(5, 1)—Estimation Results (S&P500)

Target: Adj. Realized Volatility; Security: S&P500.0000			
–	Coefficient	Std. Error	P-value
μ	0.0386***	0.0131	0.0033
ω	0.0299*	0.0165	0.0695
α_1	0.0370	0.0335	0.2696
γ_1	0.2179	0.1416	0.1237
β_1	0.8181	0.8648	0.3442
β_2	0.0000	0.4740	1.0000
β_3	0.0000	1.0245	1.0000
β_4	0.0000	0.9282	1.0000
β_5	0.0127	0.2598	0.9610

Table presents the estimation results for the GJR-GARCH(5,2) model with a constant mean and normally distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

A.6.2. Security: DAX

Volatility Proxy: Squared Return The tables below present the results of the regression estimation for the GARCH-family of models using the $S.R.$ as the volatility proxy of the DAX returns. The tables have the following order: ARCH, GARCH, EGARCH, GJR-GARCH. For all the models, the α coefficients are related to past values of ε_t^2 and β coefficients are related to past values of the volatility. In the case of the GJR-GARCH and EGARCH the γ coefficients are responsible for accounting for the leverage effect.

Tab. 21: ARCH(11)—Estimation Results (DAX)

Target: Squared Return; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0793***	0.0185	0.0000
ω	0.3635***	0.0487	0.0000
α_1	0.0454*	0.0239	0.0577
α_2	0.1457***	0.0285	0.0000
α_3	0.1784***	0.0312	0.0000
α_4	0.0830***	0.0289	0.0041
α_5	0.0685***	0.0214	0.0014
α_6	0.0745***	0.0246	0.0024
α_7	0.0413*	0.0238	0.0831
α_8	0.0726***	0.0250	0.0037
α_9	0.0278	0.0197	0.1572
α_{10}	0.0521**	0.0211	0.0134
α_{11}	0.0631**	0.0279	0.0235
ν	1.2647***	0.0446	0.0000

Table presents the estimation results for the ARCH(11) model with a constant mean and GED-distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 22: GARCH(5, 4)—Estimation Results (DAX)

Target: Squared Return; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0773***	0.0055	0.0000
ω	0.0805	0.0613	0.1887
α_1	0.0386*	0.0221	0.0815
α_2	0.1083***	0.0371	0.0035
α_3	0.1492*	0.0844	0.0769
α_4	0.0290	0.1219	0.8118
β_1	0.0000	0.8787	1.0000
β_2	0.0082	0.1159	0.9435
β_3	0.3404	0.2321	0.1426
β_4	0.0000	0.4835	1.0000
β_5	0.2941***	0.1051	0.0052
ν	1.2821***	0.0448	0.0000

Table presents the estimation results for the GARCH(5, 4) model with a constant mean and GED-distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 23: EGARCH(1, 1)—Estimation Results (DAX)

Target: Squared Return; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0463***	0.0155	0.0029
ω	0.0085**	0.0040	0.0342
α_1	0.1299***	0.0205	0.0000
γ_1	-0.1348***	0.0176	0.0000
β_1	0.9789***	0.0057	0.0000
ν	1.3367***	0.0471	0.0000

Table presents the estimation results for the EGARCH(1, 1) model with a constant mean and GED-distributed innovations. The model was fitted using the return series of the SP500 over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Tab. 24: GJR-GARCH(5, 3)—Estimation Results (DAX)

Target: Squared Return; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0222	0.0183	0.2238
ω	0.0507	0.0363	0.1623
α_1	0.0000	0.0697	1.0000
α_2	0.0000	0.0471	1.0000
α_3	0.0062	0.0594	0.9162
γ_1	0.1241	0.0976	0.2038
γ_2	0.0852	0.1385	0.5386
γ_3	0.0520	0.1179	0.6589
β_1	0.6178	1.0918	0.5715
β_2	0.0000	1.4772	1.0000
β_3	0.0000	0.8913	1.0000
β_4	0.0000	0.7682	1.0000
β_5	0.2127	0.5706	0.7093

Table presents the estimation results for the GJR-GARCH(5,3) model with a constant mean and normally distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Squared Return in terms of average Mean Squared Error within its own class.

Volatility Proxy: Adj. Realized Volatility The tables below present the results of the regression estimation for the GARCH-family of models using the ${}_{adj.}R.V.$ as the volatility proxy of the DAX returns. The tables have the following order: ARCH, GARCH, EGARCH, GJR-GARCH. For all the models, the α coefficients are related to past values of ε_t^2 and β coefficients are related to past values of the volatility. In the case

of the GJR-GARCH and EGARCH the γ coefficients are responsible for accounting for the leverage effect.

Tab. 25: ARCH(10)—Estimation Results (DAX)

.0000	Target: Adj. Realized Volatility; Security: DAX		
–	Coefficient	Std. Error	P-value
ω	0.4193***	0.0529	0.0000
α_1	0.0470*	0.0245	0.0553
α_2	0.1321***	0.0277	0.0000
α_3	0.1656***	0.0324	0.0000
α_4	0.0985***	0.0365	0.0069
α_5	0.0690***	0.0219	0.0016
α_6	0.0644***	0.0232	0.0055
α_7	0.0470**	0.0225	0.0371
α_8	0.0797***	0.0256	0.0018
α_9	0.0363*	0.0209	0.0815
α_{10}	0.0668***	0.0225	0.0030

Table presents the estimation results for the ARCH(10) model with a zero mean and normally distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 26: GARCH(3, 3)—Estimation Results (DAX)

Target: Adj. Realized Volatility; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0592***	0.0195	0.0024
ω	0.0587***	0.0207	0.0046
α_1	0.0398*	0.0215	0.0643
α_2	0.0745***	0.0261	0.0043
α_3	0.0743*	0.0387	0.0545
β_1	0.4300	0.2867	0.1337
β_2	0.0000	0.3264	1.0000
β_3	0.3528**	0.1396	0.0115

Table presents the estimation results for the GARCH(3, 3) model with a zero mean and normally distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 27: EGARCH(1, 1)—Estimation Results (DAX)

Target: Adj. Realized Volatility; Security: DAX			
–	Coefficient	Std. Error	P-value
ω	0.0147***	0.0037	0.0001
α_1	0.1287***	0.0200	0.0000
γ_1	-0.1371***	0.0178	0.0000
β_1	0.9764***	0.0059	0.0000
ν	1.3572***	0.0470	0.0000

Table presents the estimation results for the EGARCH(1, 1) model with a zero mean and GED-distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Tab. 28: GJR-GARCH(5, 1)—Estimation Results (DAX)

Target: Adj. Realized Volatility; Security: DAX			
–	Coefficient	Std. Error	P-value
μ	0.0231	0.0180	0.2004
ω	0.0342***	0.0132	0.0095
α_1	0.0000	0.0181	1.0000
γ_1	0.1862***	0.0612	0.0024
β_1	0.7883	0.5093	0.1217
β_2	0.0000	0.6958	1.0000
β_3	0.0000	0.3093	1.0000
β_4	0.0000	0.2391	1.0000
β_5	0.0967	0.2441	0.6920

Table presents the estimation results for the GJR-GARCH(5,1) model with a constant mean and normally distributed innovations. The model was fitted using the return series of the DAX over the entire training set and achieved the best performance at forecasting the Adj. Realized Volatility in terms of average Mean Squared Error within its own class.

Bibliography

Abad, Pilar, Sonia Benito, and Carmen López (2014), A comprehensive review of Value at Risk methodologies, *The Spanish Review of Financial Economics* 12.1, 15–32.

Ampadu, Samuel, Eric T. Mensah, Eric N. Aidoo, Alexander Boateng, and Daniel Maposa (2024), A comparative study of error distributions in the GARCH model through a Monte Carlo simulation approach, *Scientific African* 23, e01988.

Andersen, Torben G. and Tim Bollerslev (1998), Answering the Skeptics: Yes, Standard Volatility Models do Provide Accurate Forecasts, *International Economic Review* 39.4, 885–905.

Arlot, Sylvain and Alain Celisse (Jan. 2018), A survey of cross-validation procedures for model selection, *Statistics Surveys* 4.none.

Athey, Susan and Guido Imbens (2019), Machine Learning Methods That Economists Should Know About, *Annual Review of Economics*.

Awad, Mariette and Rahul Khanna (2015), Support Vector Regression, in: *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*, Apress, Berkeley, CA, pp. 67–80.

Black, Fischer and Myron Scholes (1973), The Pricing of Options and Corporate Liabilities, *Journal of Political Economy* 81.3, 637–654.

Bollerslev, Tim (1986), Generalized autoregressive conditional heteroskedasticity, *Journal of Econometrics* 31.3, 307–327.

– (1988), On The Correlation Structure For The Generalized Autoregressive Conditional Heteroskedastic Process, *Journal of Time Series Analysis* 9.2, 121–131.

Breiman, Leo (1996), Bagging predictors, *Machine Learning* 24.2, 123–140.

– (2001), Random Forests, *Machine Learning* 45.1, 5–32.

Brooks, Chris and Simon P. Burke (2003), Information criteria for GARCH model selection, *The European Journal of Finance* 9.6, 557–580.

Chen, Shiyi, Wolfgang K. Härdle, and Kiho Jeong (2010), Forecasting volatility with support vector machine-based GARCH model. *Journal of Forecasting* 29.4, 406–433.

Chen, Tianqi and Carlos Guestrin (Aug. 2016), XGBoost: A Scalable Tree Boosting System, in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, ACM.

Christie, Andrew A. (1982), The stochastic behavior of common stock variances: Value, leverage and interest rate effects, *Journal of Financial Economics* 10.4, 407–432.

Corsi, Fulvio (2009), A Simple Approximate Long-Memory Model of Realized Volatility, *Journal of Financial Econometrics* 7.2, 174–196.

Diebold, Francis X. and Roberto S. Mariano (2002), Comparing Predictive Accuracy, *Journal of Business Economic Statistics* 20.1, 134–144.

Drucker, Harris, Christopher J. C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik (1996), Support Vector Regression Machines, in: *Advances in Neural Information Processing Systems*, M.C. Mozer, M. Jordan, and T. Petsche (eds.), vol. 9, MIT Press.

Engle, R.F. and A.J. Patton (2001), What good is a volatility model?, *Quantitative Finance* 1.2, 237–245.

Engle, Robert F. (1982), Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation, *Econometrica* 50.4, 987–1007.

Engle, Robert F. and Tim Bollerslev (1986), Modelling the persistence of conditional variances, *Econometric Reviews* 5.1, 1–50.

European Central Bank (2014), *Nonlinearities in Macroeconomics and Finance*, Accessed: 2024-09-08.

Fama, Eugene F. (1965), Random Walks in Stock Market Prices, *Financial Analysts Journal* 21.5, 55–59.

– (1970), Efficient Capital Markets: A Review of Theory and Empirical Work, *The Journal of Finance* 25.2, 383–417.

Francq, Christian and Jean-Michel Zakoian (2019), *GARCH models : structure, statistical inference and financial applications*, Second edition, John Wiley 38; Sons, Hoboken, NJ.

Gaunt, Clive and Philip Gray (2003), Short-Term Autocorrelation in Australian Equities, *Australian Journal of Management* 28.1, 97–117.

Gers, F.A., J. Schmidhuber, and F. Cummins (1999), Learning to forget: continual prediction with LSTM, in: *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, 850–855 vol.2.

Glosten, Lawrence R., Ravi Jagannathan, and David E. Runkle (1993), On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks, *The Journal of Finance* 48.5, 1779–1801.

Hansen, Peter R. and Asger Lunde (2005), A forecast comparison of volatility models: does anything beat a GARCH(1,1)?, *Journal of Applied Econometrics* 20.7, 873–889.

Hastie, Trevor (2009), *The elements of statistical learning : data mining, inference, and prediction*, Robert Tibshirani and Jerome H.Translators Friedman (eds.), Springer, New York, NY.

He, Changli, Timo Teräsvirta, and Hans Malmsten (2002), Moment Structure of a Family of First-Order Exponential GARCH Models, *Econometric Theory* 18.4, 868–885.

Henrique, Bruno Miranda, Vinicius Amorim Sobreiro, and Herbert Kimura (2019), Literature review: Machine learning techniques applied to financial market prediction, *Expert Systems with Applications* 124, 226–251.

Hirschman, Daniel (2016), Stylized Facts in the Social Sciences, *Sociological Science* 3.26, 604–626.

Hochreiter, Sepp and Jürgen Schmidhuber (1997), Long Short-Term Memory, *Neural Comput.* 9.8, 1735–1780.

Hol, Eugenie and Siem Jan Koopman (2002), *Stock Index Volatility Forecasting with High Frequency Data*, eng, Tinbergen Institute Discussion Paper 02-068/4, Amsterdam and Rotterdam.

Hu, Min, Dayong Zhang, Qiang Ji, and Lijian Wei (2020), Macro factors and the realized volatility of commodities: A dynamic network analysis, *Resources Policy* 68, 101813.

Hyndman, Rob J. and George Athanasopoulos (2021), *Forecasting: Principles and Practice*, 3rd, `https://OTexts.com/fpp3`, OTexts, Melbourne, Australia.

Hyndman, Rob J. and Anne B. Koehler (2006), Another look at measures of forecast accuracy, *International Journal of Forecasting* 22.4, 679–688.

James, Gareth, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Taylor (2023), *An introduction to statistical learning : with applications in Python*, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Translators Taylor (eds.), Springer, New York, NY, U.S.A.

Jegadeesh, Narasimhan (1990), Evidence of Predictable Behavior of Security Returns, *The Journal of Finance* 45.3, 881–898.

Kambouroudis, Dimos S., David G. McMillan, and Katerina Tsakou (2016), Forecasting Stock Return Volatility: A Comparison of GARCH, Implied Volatility, and Realized Volatility Models, *Journal of Futures Markets* 36.12, 1127–1163.

Linsmeier, Thomas J. and Neil D. Pearson (1996), *Risk measurement: an introduction to value at risk*, ACE Reports 14796, University of Illinois at Urbana-Champaign, Department of Agricultural and Consumer Economics.

Liu, Yang (2019), Novel volatility forecasting using deep learning–Long Short Term Memory Recurrent Neural Networks, *Expert Systems with Applications* 132, 99–109.

Mandelbrot, Benoit (1963), The Variation of Certain Speculative Prices, *The Journal of Business* 36.4, 394–419.

Martens, Martin (2002), Measuring and forecasting S&P 500 index-futures volatility using high-frequency data, *Journal of Futures Markets* 22.6, 497–518.

Martin, Ian (Jan. 2021), On the Autocorrelation of the Stock Market*, *Journal of Financial Econometrics* 19.1, 39–52.

Masini, Ricardo P., Marcelo C. Medeiros, and Eduardo F. Mendes (2023), Machine learning advances for time series forecasting. *Journal of Economic Surveys* 37.1, 76–111.

Mincer, Jacob and Victor Zarnowitz (1969), The Evaluation of Economic Forecasts, in: *Economic Forecasts and Expectations: Analysis of Forecasting Behavior and Performance*, National Bureau of Economic Research, Inc, 3–46.

Morgan, J.P., N.Y.) J.P. Morgan Chase & Co. (New York, J. Longerstaey, Reuters Limited, Reuters Ltd, M. Spencer, and Morgan Guaranty Trust Company of New York (1996), *RiskMetrics: Technical Document*, J. P. Morgan.

Nelson, Daniel B. (1991), Conditional Heteroskedasticity in Asset Returns: A New Approach, *Econometrica* 59.2, 347–370.

Patton, Andrew J. (2011), Volatility forecast comparison using imperfect volatility proxies, *Journal of Econometrics* 160.1, Realized Volatility, 246–256.

Peng, Yaohao, Pedro Henrique Melo Albuquerque, Jader Martins Camboim de Sá, Ana Julia Akaishi Padula, and Mariana Rosa Montenegro (2018), The best of two worlds:

Forecasting high frequency volatility for cryptocurrencies and traditional currencies with Support Vector Regression, *Expert Systems with Applications* 97, 177–192.

Poon, Ser-Huang. and Ser-Huang. Poon (2005), *A practical guide to forecasting financial market volatility*, The Wiley finance series Wiley finance series. Chichester ; Hoboken, NJ : Wiley, c2005.

Quaedvlieg, Rogier (2021), Multi-Horizon Forecast Comparison, *Journal of Business & Economic Statistics* 39.1, 40–53.

Quinlan, J. R. (1986), Induction of decision trees, *Machine Learning* 1.1, 81–106.

Rossi, Riccardo, Andrea Murari, Pasquale Gaudio, and Michela Gelfusa (2020), Upgrading Model Selection Criteria with Goodness of Fit Tests for Practical Applications, *ENTROPY* 22.4, 447–459.

Sewell, Martin (2011), Characterization of financial time series, *Rn* 11.01, 01.

Sheppard, Keven (2024), *arch Documentation*, Accessed: 2024-04-18.

Sherstinsky, Alex (2020), Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network, *Physica D: Nonlinear Phenomena* 404, 132306.

Shmueli, Galit (2010), To Explain or to Predict?, *Statistical Science* 25.3, 289 –310.

Strang, Gilbert (2019), *Linear algebra and learning from data*, Wellesley-Cambridge Press, Wellesley, MA.

Tashman, Leonard J. (2000), Out-of-sample tests of forecasting accuracy: an analysis and review, *International Journal of Forecasting* 16.4, The M3- Competition, 437–450.

Tsay, Ruey S. (2010), *Analysis of financial time series*, 3 rd edition, Wiley, Hoboken, N.J.

Yu, Yong, Xiaosheng Si, Changhua Hu, and Jianxun Zhang (July 2019), A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures, *Neural Computation* 31.7, 1235–1270.