Master Thesis

Forecasting Volatility in Financial Time Series
A Machine Learning Approach

Author

Dominik Bruckmeier

Supervisor

Prof. Dr. Christian Heumann
Department of Statistics

Department of Statistics

Ludwig-Maximilians-Universität München

Munich, 19th of December 2022

Abstract

Forecasting volatility in financial markets plays a central role in financial econometrics, as it is a key factor for decision makers. However, the conventional econometric models used to predict volatility have been repeatedly criticized for their inadequate predictive quality. It is therefore obvious to use machine learning techniques for volatility forecasting. Nevertheless, applying machine learning models to predict volatility faces several problems, such as sparse data, low signal in the data, varying data structure, and the fact that most machine learning methods are constructed for independently and identically distributed data. We show that high-frequency data improves signal strength and adds value even when the data is sparse. We also demonstrate that time series can be put into a supervised machine learning format using Takens' Embedding Theorem. Furthermore, we analyze statistical learning theory for time series and volatility processes and conclude that machine learning models are suitable for volatility forecasting. We also consider key aspects of the volatility literature such as the properties of different volatility proxies, data frequency, and forecast horizon.

In the empirical analysis, we compare the forecasting performance of regression trees, random forests, and support vector regression to the GARCH and HAR-RV models for one-step-, three-steps-, and five-steps-ahead forecasts over different forecast horizons. Results based on realized variance as a proxy for volatility show that the HAR-SVR model performs best for one-step- and three-steps-ahead forecasts over different time horizons. For five-steps-ahead forecasts, we find that the GARCH model performs best for short horizons and the HAR-TREE model is superior for intermediate and longer horizons. The analysis also shows that machine learning models are particularly better for predicting highly volatile phases. In addition, it turns out that features based on economic theory can improve data quality and, thus, forecast performance.

Contents

List of Figures

List of Tables

List of Abbreviations

ADF-Test	Augmented Dickey Fuller Test
AR	Autoregressive
ARCH	Autoregressive Conditional Heteroskedasticity
ARFIMA	Autoregressive Fractionally Integrated Moving Average
DAX	German Stock Index
GARCH	Generalized Autoregressive Conditional Heteroskedasticity
HAR	Heterogeneous Autoregressive
IID	Independent and Identically Distributed
kNN	K Nearest Neighbors
KPSS-Test	Kwiatkowski–Phillips–Schmidt–Shin Test
LASSO	Least Absolute Shrinkage and Selection Operator
MBB	Moving-Block-Bootstrap
ML	Machine Learning
NN	Neural Network
PP-Test	Phillips-Perron Test
RF	Random Forest
RV	Realized Variance
SVR	Support Vector Regression
S&P 500	Standard & Poor's 500 Index
VIX	CBOE Volatility Index

1. Introduction

1.1. Motivation

When analyzing financial market time series, the focus is on returns, i.e., the change in the price of an asset over a given period of time, which is assumed to be random. Since the famous thesis of Bachelier (1900) *„The Theory of Speculation“*, it is known that the modeling and, especially, the prediction of returns is almost impossible. Bachelier (1900) attributes this to the infinite number of events from the past, current circumstances, and future expectations, often independent of each other, which affect returns and, thus, make an exact forecast impossible. Based on this thesis, the *„Efficient Markets Theory“* and the *„Random Walk Hypothesis“* were developed, which assume that markets are perfectly information efficient and that all information is contained in prices. Deviations from this are merely snapshots that are quickly adjusted by the market. Accordingly, stocks and stock markets follow a purely random process that does not allow for systematic, autoregressive forecasting of returns.[1]
Another central object of investigation in financial market analysis is the fluctuation of returns, i.e., volatility, which can not directly be observed from the market and is, thus, a latent variable and must be approximated.[2] It has been long established that volatility, unlike returns, can be predicted. According to Danielsson (2011), the importance of volatility stems from the fact that it is needed for practical and theoretical applications in various areas of finance. As Poon and Granger (2003) describe, modeling and forecasting volatilities of returns is important to investors because they use volatility as an input variable for investment decisions, such as designing portfolios, determining fair option prices, or trading volatility themselves. Risk analysts use volatility as a key concept to assess potential downside risks, regulators use it to determine minimum capital requirements for financial institutions, and policymakers use it to assess general economic uncertainties. In each of these applications, forecasting plays a central role

[1]See, for example, Fama (1970) and Fama (1965).
[2]The volatility is assumed to be latent since we only have one return observation per day but the returns change many times within a trading day.

with econometrics providing various statistical methods to accomplish the task.
While the simplest methods to estimate volatility and make forecasts are moving average models, their extensions, as well as the random walk models, Andersen et al. (2013) report that the most widely used models are autoregressive processes with conditional heteroskedasticity (ARCH) by Engle (1982) and its generalization (GARCH) by Bollerslev (1986). As high-frequency data have become increasingly available in finance in recent years, the development of new volatility proxies and models have come about. Andersen et al. (2013) find that ARCH and GARCH models do not perform well in a high frequency environment, and therefore, appear to be unsuitable. To take better advantage of high frequency data, Corsi (2009) develops the heterogeneous autoregressive model (HAR), which, Clements and Preve (2021) find, to be one of the most popular models in the high frequency data environment. Together, the ARCH, GARCH, and HAR models provide the classic econometric toolbox for modeling and forecasting volatility.
Despite their prevalence, these methods are often criticized[3] for not being able to generate accurate forecasts of volatility, and it seems obvious to use modern machine learning (ML) techniques, which are specially designed to achieve high prediction accuracy, to forecast volatility. Although it seems natural to use machine learning methods for these forecasts, there are some challenges. Volatility forecasting is a classical problem in time series analysis, which deals not only with general properties of time series, but also with special properties of financial market time series i.e., *stylized facts*. These special properties of financial market time series differ significantly from the data environments in which ML methods are successfully applied. As ML methods have been increasingly used for forecasting time series in recent years, it has been shown that these methods are quite competitive with classical methods.[4] Therefore, this thesis deals with the question of whether machine learning algorithms can and should be used to perform volatility forecasts.

1.2. Related Work and Literature

To understand the criticisms of classical econometric models for forecasting volatilities and the challenges of applying machine learning methods in the context of financial market time series, we review a selection of the relevant literature that can provide insight.

[3]See Hansen and Lunde (2011).
[4]See Bontempi et al. (2012).

Classical econometric models for forecasting financial return volatility are mostly based on economic and statistical theory and, as mentioned above, these models are criticized for not generating accurate forecasts of volatility. In a study by Poon and Granger (2003), the authors examine 93 papers dealing with forecasting volatilities. ARCH and GARCH models are compared with simple moving average models, standard deviation models and other volatility models. They find that ARCH and GARCH models often do not perform better than simpler models. According to the authors, the forecasting performance of all the models considered depends on the data frequency, the volatility proxy used, and the forecast horizon. The ARCH and GARCH models are superior for daily data and short forecast horizons, whereas simpler models are superior for a coarser data structure and longer forecast horizons. Overall, the study shows a mixed picture of the forecasting performance of ARCH and GARCH models and it is not clear which models are preferable for forecasting volatility.
Subsequently, a number of extensions to the ARCH and GARCH models have been specially adapted to the stylized facts of financial market time series. According to Hansen and Lunde (2005), the model most commonly used in practice is the simple GARCH(1,1) model. To check whether this model or one of the other numerous extensions is preferable in forecasting volatility, they consider 330 different GARCH models and compare their one-day-ahead out-of-sample forecast performance. They use exchange rate data and financial market data for their comparison. Several conclusions can be drawn from the study. Firstly, for the forecast of exchange rates, none of the models considered significantly outperformed the simple GARCH(1,1) model. In the case of yield data, there are models that are clearly superior to the GARCH(1,1) model. Secondly, models which can reproduce more stylized facts seem to be superior to the simple GARCH(1,1) model. It also appears that for forecasting, normal distribution is more suitable compared to the t-distribution. Despite these results, there are several issues to consider. Poon (2005) argues that the forecasting performance of the of the ARCH and GARCH extensions models depend on whether the time series under consideration has exactly the properties for which the model was constructed. This also explains the different results with regard to different data sets.
The view of predictive performance by ARCH and GARCH models as insufficient is not shared by all experts. Andersen and Bollerslev (1998) attribute the poor out-of-sample forecasting performance to poor volatility proxies. They show that classical volatility proxies such as squared returns, despite being unbiased estimators of true underlying volatility, are very noisy. Therefore, they propose the use of a more efficient proxy, *real-*

ized variance (RV), which is based on intra-day squared returns of high frequency data. Andersen and Bollerslev (1998) argue that ARCH and GARCH class models should be able to explain about 60% of volatility when RV is used as the volatility proxy. They therefore suggest continuing to use the classical models and use RV for forecast evaluation.

Building on this volatility proxy, further research directly modeled realized variance. Probably the best-known model is the heterogeneous autoregressive (HAR) model by Corsi (2009), which directly models realized variance as a linear function of daily, weekly, and monthly past realized variances. The great popularity of HAR is due to its simple structure and application, as well as its good forecasting performance and ease of economic interpretability.[5] Corsi (2009) compared the HAR model with a simple autoregressive (AR) model and an autoregressive fractionally integrated moving average (ARFIMA) model in terms of their one-day, one-week, and two-week out-of-sample forecast performance for different data sets like exchange rates and financial market data. The results show that the HAR model is superior to the simple AR models over all forecast horizons. In contrast, there are only marginal differences between the HAR model and the more complex ARFIMA model. Specifically, the HAR model performs slightly better for one-day-ahead forecast, while the ARFIMA model performs slightly better for larger horizons.

Audrino and Knaus (2016) compare the out-of-sample prediction performance of the HAR model with a Least Absolute Shrinkage and Selection Operator (LASSO) version of the HAR model. They use different stock data over a period of nine years and show that the simple HAR model and the LASSO version perform almost identically. From these results, the authors conclude that there are no significant differences between the two models in terms of prediction and that neither model is clearly preferable to the other. Similar results are obtained by other studies, such as Vortelinos (2017) who shows that the prediction performance of HAR models is not only better than that of GARCH models but also outperforms neural networks and other more complex models.

However, there are also studies that show a different picture of the HAR model's forecasting performance. Wang et al. (2013) compare simple AR models and modified AR versions that capture structural breaks in the time series with the HAR model to predict the realized variance between one and three days out-of-sample. They show that the adjusted AR models perform better than the HAR model.

Izzeldin et al. (2019) compare the forecasting performance of the HAR with that of the

[5]See Clements and Preve (2021).

ARFIMA model. They consider different market phases, forecast horizons, and data frequencies over a ten-year period. In this setup, the HAR forecasts are less sensitive to changing market conditions, but the forecast performance of the ARFIMA model is better with increasing data frequency. Specifically, Izzeldin et al. (2019) find the ARFIMA model to be slightly better for short forecast horizons, while the HAR model is better for longer forecast periods, contrary to the results by Corsi (2009). Overall, the authors conclude from their findings that no model is significantly superior to the other.

Overall, prior studies show a very mixed picture regarding the performance of volatility forecasting by classical econometric models. As described above, it seems obvious to use machine learning methods for this task and obtain more precise volatility forecasts. However, the use of ML in the context of financial market time series is associated with various difficulties. As Israel et al. (2020) note, finance is fundamentally different from the research fields in which ML has been successfully applied. According to the authors, finance is not a big data environment because there is usually only a single time series as an observation. Further, the time series data considered have a relatively low signal-to-noise ratio, which may favor overfitting and, in addition, financial market time series often exhibit rapidly changing dynamics that may be due to non-stationary behaviour. Lommers et al. (2021) and Athey and Imbens (2019) argue that in financial market analysis, inference concepts are the focus of research, to learn information from the underlying a-priori specified data generating process as well as relationships between variables. On the other hand, out-of-sample forecasting performance is the focus of research in the ML area. Forecasting performance should be optimized to obtain the highest possible prediction quality and generalizability over the data and it is not assumed that the models describe the data generating process. According to Lommers et al. (2021), the main difference in the research paradigms is that classical financial market analysis is hypothesis-driven whereas ML is data-driven. Athey and Imbens (2019) also address the question of whether and to what extent machine learning methods can be used in the general context of economic research. They argue that besides all the challenges, especially in the area of forecasting, a large component of econometrics can benefit greatly from ML techniques. However, the authors clarify that, even if a benefit is conceivable, there is no a-priori guarantee for superior forecasting performance of ML models compared to classical methods. This can be derived from the well-known *No-Free-Lunch-Theorem*[6] from machine learning, which states that there is no universally

[6]See, for example, Shalev-Shwartz and Ben-David (2014).

best learning procedure.
Besides the potential opportunities offered by machine learning in certain areas of financial econometrics, Athey and Imbens (2019) argue that the problems mentioned are one reason why econometrics has not yet made greater use of ML methods. López de Prado (2019) shares this assessment, stating that by the end of 2018, a total of 13,772 publications related to economics, statistics, and probability theory had been published. Among these publications, there are only 89 articles related to machine learning. Although machine learning has not yet fully arrived in the econometric toolbox, there are now a number of articles dealing with the application and benchmarking of these methods in the financial market domain. Some of these papers deal with volatility forecasting, with RV at the core of all studies.
Masini et al. (2021) compare various machine learning techniques such as neural networks (NN), random forests (RF), and an extended regularized HAR model (HAR-LASSO) with the classical HAR model. They show that all ML models except the RF outperform the HAR benachmark model over all time periods considered, with the HAR-LASSO model performing best, followed by NN and RF.
Christensen et al. (2021) perform a similar comparison. They consider regularized models, regression trees, RFs, NNs as well as other ML methods and compare them with the classical HAR model in terms of their one-day-ahead out-of-sample forecast performance. The authors use data spanning 16 years and consider two different modeling approaches. In the first approach, only pure autoregressive modeling is used. In the second approach, additional explanatory variables are included in the models. In both approaches, the ML methods outperform the HAR model. Overall, however, NNs and RF are the best performing models. This result is interesting because, although it essentially confirms the results of Masini et al. (2021), RF now performs significantly better.
Rahimikia and Poon (2020) compare the forecasting performance of ML methods such as various NNs with an extended version of the HAR model. For the prediction, they use order book data and news data of 28 stocks over a period of 9 years. The authors find that ML methods show superior forecast performance, but the forecast quality depends on which market phase is forecasted. It appears that for less excited market phases with low volatility, which accounts for 90% of the data used, ML models are significantly better. Whereas in excited market phases with high volatility, the HAR model tends to perform better.
Somewhat different results are shown by Bucci (2020), who compares different NNs with ARFIMA models with respect to their forecast performance of the RV over different hori-

zons. The author takes into account the respective market phases in order to control for different volatility phases. From the results of the study, the authors claim that NNs are superior to ARFIMA models in forecasting normal volatility phases as well as high volatility phases. These findings are contrary to the results of Rahimikia and Poon (2020), which may be due to several reasons. One is that Bucci (2020) uses significantly more explanatory variables in his models than Rahimikia and Poon (2020), which may lead to better generalization of the ML models and thus better forecasting performance. Another reason may be that Bucci (2020) uses significantly more and different NNs than Rahimikia and Poon (2020). A closer look at the evaluation shows that a change in model ranking occurs when forecasting highly volatile phases. Although NNs still perform better than ARFIMA models, this is now a different NN than for phases with moderate volatility. It even shows that the models performing best in normal volatility phases are now outperformed by the ARFIMA models for longer forecast horizons in more volatile market phases.
In a study by De Stefani et al. (2017), the authors examine out-of-sample forecast performance over different forecast horizons of different volatility proxys. They use various ML methods such as k-nearest neighbors (kNN) regression, support vector regression (SVR), and NNs to compare against the classical GARCH model. The authors use two approaches to compare the models. The first forecasts the different volatility proxies with a purely autoregressive structure and the other includes different volatility proxies as explanatory variables in the models. The results show that the ML methods are superior to the classical GARCH model in both scenarios and that the SVR model performs best. Moreover, it is shown that the inclusion of additional volatility proxies can significantly improve forecasting performance.
There are other studies that use ML methods in the context of volatility and may argue for a successful application. Zhang et al. (2022) show superior forecasting performance of different ML methods at forecast horizons of a few minutes in the high frequency domain. Li and Tang (2021) find that a simple ensemble of different ML models perform best and are able to forecast the RV for longer forecast horizons better than the HAR benchmark model. Carr et al. (2019) use option pricing data as well as data from the S&P 500 to obtain better volatility indexation with ML models than with the VIX[7]. Luong and Dokuchaev (2018) combine ML models with a classic HAR model by using RF to forecast the direction and magnitude of volatility and incorporating this as infor-

[7]The Volatility Index (VIX) uses option prices to indicate the market's expected short-term volatility over the next 30 days for the S&P500.

mation into a HAR model. This additional information significantly improves forecast performance compared to that of a simple HAR model.
From the literature review, some necessities and challenges arise from which the problem definition and research question as well as associated implications are derived. This is the subject of the following subsection.

1.3. Problems and Research Question

The literature review reveals several problems from which the research question of this master thesis is derived. Drawing from the vast literature, it can be stated that forecast performance of different classical volatility models depend on data frequency, the volatility proxy, and forecast horizon. In addition, it has become clear that machine learning methods cannot simply be applied to the field of financial econometrics. The main reasons for this are that finance is not a big data environment, different data structures are available, and different research paradigms are underlying. Nevertheless, there are areas in finance where ML techniques can be of great use, such as forecasting. Some of the studies reviewed use ML models to forecast volatility in financial markets, but none of the papers specifically address the dependence of forecast performance on data frequency, volatility proxy, and forecast horizon. Moreover, the models are applied without addressing differences in data structures and without considering possible implications for ML procedures. This can be problematic because time series data have a different structure than data considered in supervised machine learning. Furthermore, time series data violate the assumption that data are independent and identically distributed, which is central to theoretical statements regarding ML algorithms.
The central research question of this thesis is *whether machine learning models can and should be used to perform volatility forecasts.* From the literature review, several problems arise with respect to this question, which are also considered in this thesis:

- *Can machine learning techniques significantly improve forecasting performance compared to econometric volatility models?*

- *What impact do different volatility proxies have on forecast performance and model ranking?*

- *What influence does data frequency and economic theory have on modeling and forecast quality?*

- *What influence do different forecast horizons have on forecast performance?*

In addition, differences between time series and ML data structures will be discussed and an explanation with theoretical justification will be given on how time series data can be brought into a supervised machine learning data setup. Also, some recent results concerning generalization bounds and stability bounds of machine learning methods in dependent data are discussed and put into context.
To answer the research questions posed in this paper, we proceed in two main steps. First, we discuss the properties of time series data, and more specifically financial market time series data, as well as volatility in order to form a basic understanding. For this purpose, we will also take a closer look at econometric models and machine learning techniques. In the second step, we conduct an empirical study and use daily data of the DAX. We approximate daily volatility in two ways, by first using daily data with one observation per day and secondly, using high frequency data with several observations per day. This two-pronged approach allows us not only to approximate and model volatility differently but also to determine the influence of data frequency and the volatility proxy on forecast quality.

1.4. Overview

This master thesis consists of seven chapters. In Chapter 2, we first discuss the theoretical concepts of time series. Then, the characteristics of financial market time series are described. After that, volatility is discussed in detail. In Chapter 3, the relevant econometric models are presented and their properties are described. In Chapter 4, we start with some necessary adjustments to apply time series in supervised machine learning and discuss recent research results regarding algorithmic properties of dependent data. Afterwards, the used algorithms are presented and necessary adaptations are analyzed. In Chapter 5, model evaluation is discussed where the focus is on various error measures and evaluation strategies for time-dependent data. Chapter 6 presents, evaluates, and discusses the empirical results. In Chapter 7, a conclusion is drawn and reference is made to the questions of the thesis. Subsequently, an outlook on future research projects is given.

2. Theoretical Background

Time series analysis has had a major impact on research and practice over the last five decades and is used in a wide variety of fields such as medicine, physics, economics, finance, and others. The goal is to understand the characteristics and dynamics of the series in order to derive regularities and make inferences. To do so, one tries to find a stochastic model that describes the data as well as possible and can help to understand the underlying process. Another elementary objective is the prediction of future states of the time series. The importance of forecasting time series is of particular significance for business and science. This is shown by the fact that many variables of interest evolve over time, such as business cycles, price trends, unemployment rates, new infections, or the electrocardiography, and it is important for optimal decision-making to have an estimate of the series' future behavior.
In this paper, we deal with forecasting volatility of the DAX. In econometrics, this is a task of time series analysis, which we want to compare with machine learning methods in order to answer the research questions outlined above. To achieve this, it is essential to understand the theory of time series analysis, so that the peculiarities of the series can also be accounted for in modeling and forecasting. Therefore, this chapter is first devoted to the general theory of time series which is essential for a basic understanding. Next, we explicitly consider financial market time series and their additional peculiarities. Finally, these results will be used to take a closer look at volatility and to discuss it intensively.

2.1. Fundamental Theory of Time Series

Definition of Time Series

We know so far that a time series is the measurement of a variable evolving over time. Kirchgässner et al. (2012) describe a time series as the observed realizations $\{r_t\}_{t\in\mathbb{T}}$ of a time ordered stochastic variable $\{R_t\}_{t\in\mathbb{T}}$. To better understand this concept, we need to use definitions from the theory of stochastic processes.
Following Webel and Wied (2016), we assume that $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space, $(E, \mathcal{E})$

is a measurable space or state space, and $\mathbb{T} \neq \emptyset$ is an index set. A family $\mathcal{R} = \{R_t\}_{t\in\mathbb{T}}$ of measurable mappings

$$\begin{aligned} R_t : (\Omega, \mathcal{A}, \mathbb{P}) &\to (E, \mathcal{E}) \\ \omega &\mapsto R_t(\omega) \end{aligned}$$

is called a stochastic process. If we fix $t_0 \in \mathbb{T}$, then the stochastic process is a random variable. On the other hand, if we interpret the index $t \in \mathbb{T}$ as time and fix $\omega_0 \in \Omega$, we call the mapping a path, trajectory, or realization, which is then called a time series

$$\begin{aligned} R(\omega_0) : \mathbb{T} &\to (E, \mathcal{E}) \\ t &\mapsto R_t(\omega_0). \end{aligned}$$

Considering a time series as a realization of a stochastic process shows that the series is just an explicit realization of an arbitrary number of realizations of a stochastic process with the same properties.[1] Stochastic processes can be classified in terms of their index set as well as their state space. We call the stochastic process discrete for a countable index set and we call the process continuous for an uncountable index set. In the present work, we mostly assume a discrete-time process which will be noted as $\{R_t\}_{t\in\mathbb{T}}$. If continuous-time processes are considered, this is explicitly mentioned and is noted as $\{R(t)\}_{t\in\mathbb{T}}$. Analogously, stochastic processes can be classified with respect to their state space. In the case of $(E, \mathcal{E}) = (\mathbb{R}, \mathcal{B})$ with $\mathcal{B}$ as the Borel σ-algebra, the stochastic process is real-valued. It is essential to note from the definition that the order of the sequence of observations in a time series is relevant and that successive values are generally not independent of each other.

Moments of Time Series

From the interpretation of a time series as a realization of a stochastic process, the question arises of how important values like expected value, variance, or covariance can be determined. For this, we assume that $\{R_t\}_{t\in\mathbb{T}}$ has a finite second moment $\mathbb{E}[R_t^2] < \infty$. Then the mean (2.1), variance (2.2), autocovariance (2.3), and autocorrelation (2.4)-

[1]See Kirchgässner et al. (2012).

functions of $\{R_t\}$ at time t can be stated, as in Kirchgässner et al. (2012), as

$$\mu_t = \mathbb{E}[R_t] \tag{2.1}$$

$$\sigma_t^2 = \mathbb{E}[(R_t - \mu_t)^2] \tag{2.2}$$

$$cov(R_t, R_{t-\tau}) = \mathbb{E}[(R_t - \mu_t)(R_{t-\tau} - \mu_{t-\tau})] \tag{2.3}$$

$$corr(R_t, R_{t-\tau}) = \frac{cov(R_t, R_{t-\tau})}{cov(R_t, R_t)}. \tag{2.4}$$

A short note on the autocorrelation function. This is a quantity that measure serial correlation, that is, correlation over a fixed period of time τ. A significant autocorrelation value indicates how the observation at time t depends on all past τ observations.

Stationarity

As mentioned by Webel and Wied (2016), moments of time series can be described by the joint distribution of the process. This is because the distribution of a stochastic process, under certain conditions, describes the process completely. In time series analysis, the focus is especially on processes that exhibit the same properties over time. This means that the joint distribution should be invariant to time shifts. This invariance property of the distribution describes the *strict stationarity* condition. More formally, we define this property following Webel and Wied (2016). A stochastic process $\{R_t\}_{t\in\mathbb{T}}$ is strictly stationary if for all $n \in \mathbb{N}$, $h \in \mathbb{R}$, all $t_1, \ldots, t_n$, and $r_1, \ldots, r_n \in \mathbb{R}$,

$$F_{t_1,\ldots,t_n}(r_1, \ldots, r_n) = F_{t_1+h,\ldots,t_n+h}(r_1, \ldots, r_n)$$

holds. As Hassler (2007) states, strict stationarity means that the properties of the distribution do not depend on time. Only the distance between the individual components is decisive. Accordingly, strict stationarity implies that expected value and variance are independent of time and that autocovariance depends only on the time difference between two components. This means, for expected value (2.1) $\mu_t = \mu_R$ holds for all $t \in \mathbb{T}$, and for variance (2.2) $\sigma_t^2 = \sigma_R^2$ and autocovariance (2.3) $cov(R_t, R_{t-\tau}) = cov(R_\tau, R_0)$ holds for all $\tau, t \in \mathbb{T}$ with $\tau < t$.[2]

It is often very difficult to check the condition for strict stationarity in practice, so one considers processes whose expected values are independent of time, where $\mu_t = \mu_R$ for all $t \in \mathbb{T}$ and the autocovariance function depends only on the time difference

[2]As mentioned above, we assume that the second moments exist and are finite: $\mathbb{E}[R_t^2] < \infty$. This allows us to specify the variance and autocovariance functions.

$cov(R_t, R_{t-\tau}) = cov(R_0, R_\tau)$ with $\tau, t \in \mathbb{T}$ and $\tau < t$. These processes are considered *weakly stationary*. A weakly stationary process is in general not strict stationary but conversely any strictly stationary process is also weakly stationary. In the following, we only use the term stationarity to mean weak stationarity unless strict stationarity is explicitly mentioned. As described earlier, the stationarity assumption affects the moments of a process. In addition to the described effects, it is also true that the autocorrelation of a stationary process is symmetric $corr(R_t, R_{t-\tau}) = corr(R_t, R_{t+\tau})$ and less than or equal to one in magnitude $|corr(R_t, R_{t+\tau}) \leq 1|$ for all τ.
At this point, we discuss another implicit assumption. As Kirchgässner et al. (2012) describe, another key assumption in time series analysis is *ergodicity*. This assumption states that, for a time series with a finite number of observations $\{r_t\}_{t=1}^{T}$, the empirical moments converge for $T \to \infty$ to the true moments. This assumption is necessary because, in practice, we have only one realization of the stochastic process and inference would thus be difficult. For example, if one wants to estimate the expected value, variance, and autocovariance of a process using empirical analogues, it would be advantageous to have more than one realization at time t. Ergodicity cannot be tested empirically and must be assumed. However, this only makes sense if we assume that the first two moments of the process are constant over time. Thus, a process is ergodic if it is stationary.

Filtration, Information and Martingale

One of the last two purely theoretical explanations concerns filtrations. This concept stems from probability theory and serves in the present context as a set of information. We again follow Webel and Wied (2016) for the definition and assume that $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space, $\mathbb{T} \subseteq \mathbb{R}$ an index set. Further, let $\mathcal{I} = \{\mathcal{A}_t\}_{t \in \mathbb{T}}$ be an isotonic family of sub-σ algebras with $\mathcal{A}_t \subset \mathcal{A}$ for all $t \in \mathbb{T}$ and $\mathcal{A}_s \subset \mathcal{A}_t$ for all $s, t \in \mathbb{T}$ and $s < t$. Then $\mathcal{I}$ is a filtration. We see from the definition of a filtration that it is an increasing sequence of σ-algebras on measurable spaces. Moreover, we see that filtrations can be interpreted as information at different times, where the information at an earlier time is contained in the information set at a later time, and thus no information is lost. However, we are interested in the information that arises from the stochastic process itself, that is, somewhat simply speaking, the information about the past of the process. This leads us to the concept of natural filtration. This is the smallest σ-algebra generated by the past of the process up to time t, and by definition, for which the stochastic process is measurable for all $t \in \mathbb{T}$.[3] We can define this special filtration as $\mathcal{I}_t = \sigma(\{R_s\}_{s \leq t})$.[4] To better use and understand this initially abstract concept, we follow the idea of Kirchgässner et al. (2012) and take natural filtration $\mathcal{I}_t$ simply as a set of information about the process up to time t and imagine that we know the current value of the process as well as all past values. An example can be found in the appendix of this paper for further explanations.

The last theoretical concept we consider concerns martingales. Hassler (2007) describes a process as a martingale if the best prediction for a future value is its value today. For an exact definition, we follow Hassler (2007) as well as Webel and Wied (2016). We assume that $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space, $\mathbb{T}$[5] is an index set, $\mathcal{I}$ is a filtration, and $\{R_t\}_{t \in \mathbb{T}}$ is a stochastic process on the given probability space. Further, let $\{R_t\}_{t \in \mathbb{T}}$ be measurable for all $t \in \mathbb{T}$ with respect to filtration $\mathcal{I}$ and let $\{R_t\}_{t \in \mathbb{T}}$ be absolutely integrable. Then $\{R_t\}_{t \in \mathbb{T}}$ is a martingale if

$$\mathbb{E}[R_t | \mathcal{I}_s] = R_s$$

for all $s, t \in \mathbb{T}$ with $s < t$. From the definition of a martingale, it is important to note that conditional expectation is itself a random variable. This is best illustrated by the example of a random walk, which represents a martingale. The corresponding explanations can be found in the appendix of this thesis. The chosen example of the random walk is also intuitive for the interpretation of a martingale and the importance of this concept. On the one hand, they can be interpreted as a condition for a fair game. On the other hand, they show a generalization of a sum of independently and identically distributed random variables for which versions of the limit theorems exist.

As Hassler (2007) describes, it is also interesting that the expectation of the value at time $t+1$ is given by the value at time t. Thus, there is no expectation of an increase in the value from t to $t+1$. This observation allows us to formulate the martingale concept in terms of differences, which is similar to the martingale definition except that now $\mathbb{E}[R_{t+1} | \mathcal{I}_t] = 0$ must hold. The connection between martingales and martingale differences is thus obvious. If $\{R_t\}_{t \in \mathbb{T}}$ is a martingale, then $\{P_t = R_t - R_{t-1}\}_{t \in \mathbb{T}}$ is a martingale difference. Thus, a martingale difference indicates that the past of the process

[3]As Webel and Wied (2016) note, a process R_t is adapted if R_t is for all $t \in \mathbb{T}$ $\mathcal{A}_t$-measurable. It is therefore clear that any stochastic process is adapted to its natural filtration.

[4]See Hassler (2007).

[5]Note that the index set here is arbitrary and the definition is applicable for continuous as well as for discrete-time processes. In the continuous case, one chooses $\mathbb{T} \subseteq [0, \infty)$ with corresponding state space $(E, \mathcal{E}) = (\mathbb{R}, \mathcal{B})$. In the discrete-time case, on the other hand, $\mathbb{T} = \mathbb{N}_0$ can be chosen.

has no effect on forecast quality. One can now show some properties of martingale differences, which essentially state that martingale differences are, on average, zero and free of serial correlation. Nevertheless, they do not have to be independent or stationary. For a further discussion and a mathematical justification of these statements, we refer to Hassler (2007).

Basic Time Series Models

Having laid out the main general theoretical concepts for stochastic processes, we now briefly consider the basic processes that are relevant to the work. In doing so, we essentially follow the descriptions of Brockwell and Davis (2002).

Probably the simplest model of time series analysis is a process consisting of independent and identically distributed components that does not contain any trends, seasonality structures, or cycles. These processes are called *IID noise* and are usually specified with a mean of zero and constant variance. Formally, we describe such a process as

$$\{R_t\} \sim IID(0, \sigma^2). \tag{2.5}$$

IID processes are thus stationary by definition and are relatively uninteresting as a separate model class, but they form an important basis for more complex models. A less restrictive assumption about the properties of the process $\{R_t\}$ leads us to a very similar process. If, instead of independence of the components, we assume only uncorrelatedness, we arrive at *white noise* processes, which can be formally described as

$$\{R_t\} \sim WN(0, \sigma^2). \tag{2.6}$$

From the properties of the processes, it can be seen that every IID noise process is a white noise process, but the reverse is generally not true. Thus, the statements about stationarity are also apply to the white noise process.

Another simple but important model is the *random walk* model, which is the basis for pricing processes as described in the motivation section. The model with zero mean describes an additive concatenation of independently and identically distributed random components. Formally, the model can be described as

$$R_t = R_{t-1} + \eta_t \tag{2.7}$$

where $\eta_t \sim IID(0, \sigma^2)$ is assumed.[6] As can be easily verified, the random walk model is an example of a non-stationary process, since its variance and autocovariance functions depend on time. Similar to the random walk model, there is a random walk model with drift. This model is identical to the simple random walk except that a drift term in the form of a constant is included in the model. The random walk with drift can be formally represented as

$$R_t = \mu + R_{t-1} + \eta_t \tag{2.8}$$

with $\eta_t \sim IID(0, \sigma^2)$. The random walk model with drift also describes a non-stationary process. In this model, the drift term indicates the direction of the trend, i.e. where the process is moving. For $\mu > 0$, the process shows an upward trend and for $\mu < 0$, a downward trend. For $\mu = 0$, the simple random walk model is obtained. Both random walk models considered here represent processes in discrete time. However, we also consider a model from continuous time and, therefore, briefly discuss *Brownian motions*. According to Webel and Wied (2016), these continuous-time processes can be approximatively derived from a rescaled random walk and thus form the continuous counterpart to the random walk. These processes are an elementary part of stochastic analysis and financial mathematics. In this paper, however, we only discuss one continuous time model to gain derivations for volatility. For a detailed discussion of the underlying theory, we refer to the literature such as Hassler (2007) or Webel and Wied (2016). To define a Brownian motion, we assume that $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space and $\mathcal{W} = \{W(t)\}_{t \geq 0}$ is a stochastic process on it. We call $\mathcal{W}$ a Brownian motion with drift $\mu \in \mathbb{R}$ and variance $\sigma^2 > 0$ if it satisfies the following properties:

$$\mathbb{P}(W(0) = 0) = 1 \tag{2.9}$$

The increments $W(t_1) - W(t_0), \dots, W(t_n) - W(t_{n-1})$ (2.10)
with $0 \leq t_0 \leq t_1 \leq \cdots \leq t_n$ are independent for any n

The increments $W(t+s) - W(s) \sim \mathcal{N}(\mu t, \sigma^2 t)$ are normal distributed (2.11)
for all $s \in [0, \infty)$ and all $t > 0$.

Some properties can be read directly from the definition of the process. The Brownian motion has stationary increments, but is not stationary itself. Moreover, the definition implies that $W(t) \sim \mathcal{N}(\mu t, \sigma^2 t)$ holds. There are other interesting properties of these

[6]In some literature $\eta_t \sim WN(0, \sigma^2)$ is assumed instead of IID noise.

processes which we do not go into here but refer to the literature mentioned. Of special importance to the topic of the present work are standardized Brownian motions, for which $\mu = 0$ and $\sigma^2 = 1$ must hold from the above definition. These processes play a special role in financial mathematics since they are essential to the definition of Ito integrals and Ito processes.
All these technical definitions and concepts are central to a basic understanding of the processes and derivations considered in this paper. We deal with volatility processes, also called ARCH or GARCH processes. They represent a prominent class of processes that use all of the concepts outlined here. A basic understanding of these processes is also central to the application of ML methods to this class of processes.
All statistical tests used in the paper, such as the Ljung-Box test for uncorrelatedness, the Dickey-Fuller test for non-stationarity or the Jarque-Bera test for normal distribution, are no longer explicitly described here. These tests represent basics in time series analysis and can be referenced in prior literature.

2.2. Financial Time Series

Having laid out the essential technical foundations for time series analysis, let us now turn to the additional special features of financial market time series. The aim of this work is to forecast the volatility of the DAX. According to Held (2018), the DAX is an index which is a measure for the stock market and lists so far the 30 largest German companies and thus approximately 80% of the share capital of domestic listed companies.[7] The index is calculated according to the Laspeyres price index formula, whereby the weighting of a share depends on the stock issuer's free float market capitalization, which is the difference between the issued shares and the shares held by the company times the price per share.
We now derive a model equation to describe the return behaviour of the DAX. This equation will play a central role in the further analysis of this thesis, since most derivations and calculations are based on it. The analysis of this process will then allow us to work out all additional features of financial market time series that are important for modeling and forecasting. It should be noted here that we consider all subsequent time series models as models for observed data. The reason for doing so is to simplify the notation and maintain the distinction between realization and process in this context.

[7]On September 20, 2021, the DAX was adjusted to include the 40 largest companies in the German economy.

The Return Process

For the derivation of the return model, we follow the explanations of Fama (1965). He argues that the price p_t of a financial asset follows a random walk model with drift as described in equation (2.8) so that

$$p_t = p_{t-1} + \mu_t + \epsilon_t \tag{2.12}$$

where p_t is the price of the financial asset at time t, μ_t is the conditional mean also called a drift term, and ϵ_t is a white noise process with $\mathbb{E}[\epsilon_t] = 0$, $\mathbb{E}[\epsilon_t^2] = \sigma_\epsilon^2$ and $\mathbb{E}[\epsilon_t \epsilon_{t-\tau}] = 0$ for $\tau \neq 0$ which is also called random innovation-term.
The return on financial assets at time t can now be defined as the change in price from time $t-1$ to time t, $r_t = \Delta p_t = p_t - p_{t-1}$. If we now take into account that the price process follows a model from equation (2.12), we obtain the return process

$$r_t = \mu_t + \epsilon_t. \tag{2.13}$$

As Danielsson (2011) describes, return is defined as the change in prices over a given time interval. In finance, one looks at the relative change of prices, for which there are various calculation methods. In financial market time series analysis, it is common to use either discrete[8] or the continuous returns. Here we use continuous returns, which is defined as

$$r_t = ln\left(\frac{p_t}{p_{t-1}}\right). \tag{2.14}$$

As Kirchgässner et al. (2012) note, the advantage of continuous returns is its symmetric behavior. This means that from the same base value, an increase of 50% and a subsequent decrease of 50% leads back to the initial base value, which is not true for discrete returns. Moreover, the difference between the two return calculations is only relevant when large returns are achieved, which is not the case with financial market returns. This follows from the fact that $ln(1+r_t) \approx r_t$ is true for small values of r_t which can be verified with the first order Taylor approximation.[9] One more note on the derivation of the return process from equation (2.13). The derivation performed is also valid for continuous returns, for which we have to use the logarithm of prices $ln(p_t)$ instead of simple prices

[8]The or discrete return is defined as $r_t = \frac{p_t - p_{t-1}}{p_{t-1}}$

[9]Using $f(r_t) = ln(1+r_t)$ we get $f(r_t) \approx ln(1+0) + \frac{1}{1+0}(r_t - 0) = r_t$

p_t. One practical thing needs to be mentioned at this point. The returns in this paper are not calculated as the log difference between the closing prices of today and yesterday, but rather as the log difference between the closing price and the opening price on the current day. The reason for this is because of potential overnight effects that cause yesterday's closing price to differ from today's opening price. Such effects may be triggered by news that influence asset or stock prices arriving after the end of trading on the market, and are priced in overnight. These effects generate overnight returns and create additional volatility, which is often not accounted for in the construction of volatility proxies. This affects one of the proxies used in this paper, realized variance.[10] However, the derivations here remain valid, as they are performed for general time points.

Stylized Facts

As briefly mentioned at the beginning of this paper, financial market time series exhibit various peculiarities beyond those of time series that must be considered in modeling and forecasting. According to Lommers et al. (2021), these peculiarities are due to the fact that financial markets do not follow natural laws but rather reflect the result of human actions, making markets irrational and difficult to predict. In addition, there are other exogenous shocks such as natural disasters, pandemics, and political events that are hard to predict but have an impact on financial markets. The resulting peculiarities of financial market time series are summarized under the term stylized facts.
Stylized facts are based on empirical observations of financial market time series. To present the most important stylized facts for the purpose of this paper, we consider daily data of the DAX[11] over a period from January 2000 to June 2022 with 5698 observations. Figure 2.1a shows the described time series of the DAX. This is an example of a non-stationary time series that seems to follow a positive trend. However, we are interested in returns of the DAX which is presented in Figure 2.1b. First, it should be mentioned that the return time series of the DAX moves around zero and does not follow a trend. The mean value over the observation period is -0.0288% and the return series seems to show a stationary behavior. An ADF test confirms that the hypothesis of a non-stationary return series must be rejected for a significance level of $\alpha = 0.01$. Another important observation is that very large fluctuations and very small fluctuations in the returns are clustered, meaning that large fluctuations follow previous large fluctuations and that

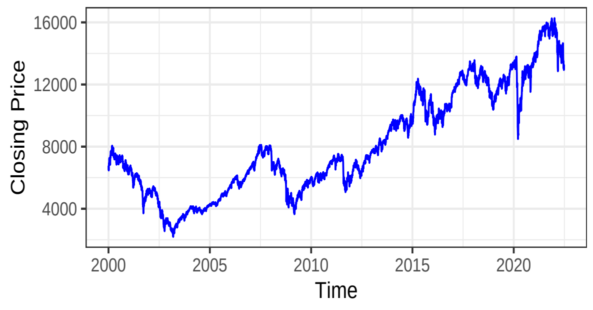
[10]See Shephard and Sheppard (2010).
[11]Data used from the Realized Library of the Oxford-Man Institute of Quantitative Finance, see Heber and Sheppard (2009).

small fluctuations tend to follow small fluctuations. This clustering of return fluctuations implies that the variance of returns is subject to variation over time and is not constant. These emerging patterns of fluctuation are called *volatility clusters* and lead to *excess kurtosis*, as shown in Figure 2.1c. We observe that the distribution has more mass at the edges and in the center compared to a normal distribution. A distribution with such properties is *leptokurtic* and the thick edges of a distribution are termed *heavy tails* in the literature. In the present case, the kurtosis is 5.3 and is significantly larger than the kurtosis of a normal distribution with 3. Thus, it is expected that the returns are not normally distributed. This expectation is confirmed by the Jarque-Bera statistic. The null hypothesis of a normal distribution of returns can be rejected at any conventional significance level. The heavy tails and non-normality of the returns pose problems because most models in finance are based on the normal distribution assumption, which is obviously inadequate for the data at hand. This inadequate distribution assumption has particular consequences in the area of risk modeling and forecasting as it can lead to underestimation of risk. If a normal distribution of returns were assumed, this would suggest that extreme return values occur only very rarely. Empirically, however, this is not the case as we see, since heavy tails are an expression of extreme returns.

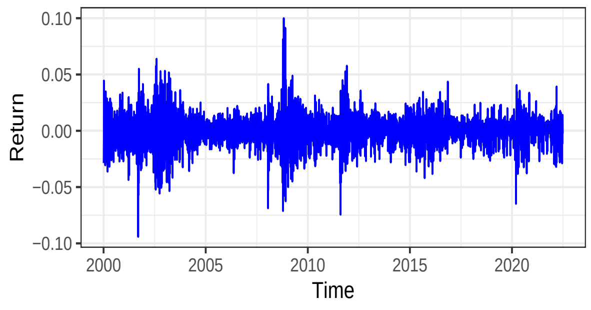
We now consider Figure 2.1d, which presents autocorrelation of the returns and is intended to provide information on whether the random walk hypothesis and the return process (2.13) derived from it appear reasonable for the data at hand. Most of the lags are within the confidence interval, but there are some that go to the limit or beyond. The Ljung-Box test indicates no significant autocorrelation up to lag four. Above a lag order of four, the null hypothesis that the returns are not jointly autocorrelated must be rejected.

The result of the Ljung-Box test for higher order autocorrelation seems somewhat strange against the background of the graph and the level of the individual autocorrelation. As Danielsson (2011) describes, this apparent autocorrelation may be due to synchronization biases in the financial market data. To account for possible autocorrelation and to obtain a mean model for process (2.13) it seems reasonable to fit an ARIMA model, where the choice of the model happens stepwise with the help of information criteria. This step is necessary to obtain residuals from the model that no longer exhibit autocorrelation and are zero on average. One would then proceed with the residual series instead of the observed series to model volatility.[12]

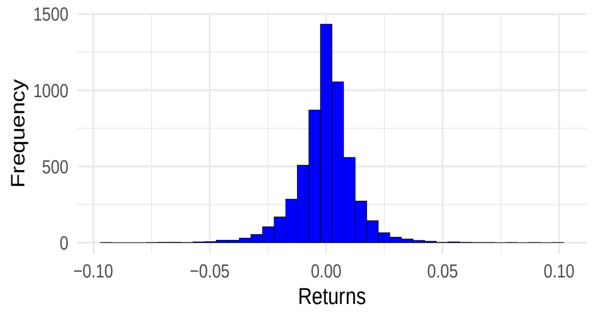
[12]See Kirchgässner et al. (2012).



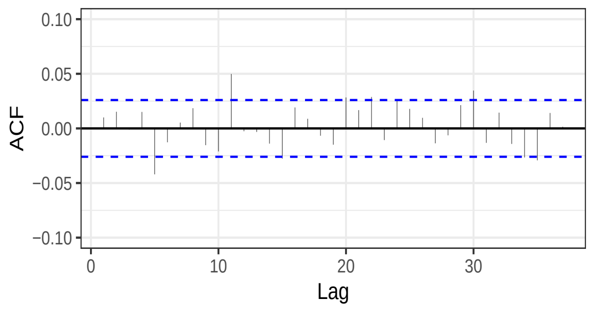
(a) DAX price series



(b) DAX return series



(c) DAX returns histogram



(d) Autocorrelation of DAX returns

Figure 2.1.: Different DAX return plots. The returns are calculated as the first difference of logarithms of the DAX open to close prices.

Interestingly, the optimal model here is an ARIMA$(0, 0, 0)$ model with zero mean, which can be written as

$$r_t = \epsilon_t. \tag{2.15}$$

The results so far suggest that the return process follows a white noise process similar to equation (2.15). However, it does not seem plausible that the data can be explained by this model, because it is clear from Figure 2.1b that the fluctuation in returns, volatility, is larger in some periods than in others and therefore the variance of the process is not constant. An indicator for existing dependencies caused by volatility clusters is significant autocorrelation in the squared returns which can be seen in Figure 2.2a. The dependence of the squared returns reflects a non-constant, time-dependent variance of the return process and shows not only that the process does not follow a white noise process but also gives evidence of an *autoregressive conditionally heteroskedastic*

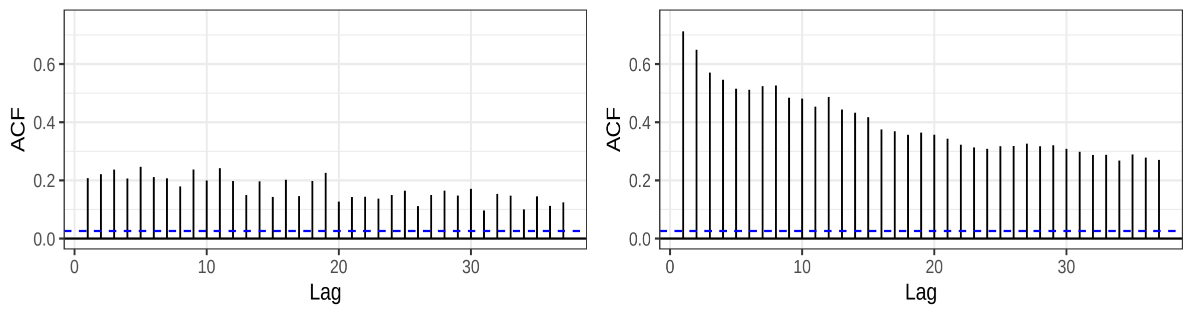
structure of the variance.
These results have several implications. First, it confirms Danielsson (2011) conclusion and suggests that the returns are not autocorrelated. Second, the model choice confirms the derived return process (2.13), which can be simplified even further as stated in equation (2.15) but the dependence of the squared returns must still be considered.
To take the described dependencies into account, special attention must be paid to volatility. This is the central variable of this thesis and is therefore analyzed separately in the following chapter. It should be noted that there are many other properties of financial market time series that are summarized under the term stylized facts. However, these are not relevant for this thesis and we therefore refer to the literature such as Poon (2005) for further details.

2.3. Volatility in Financial Markets

As described in the previous chapter, volatility plays a central role in this thesis. Until now, volatility has been understood as the fluctuation of returns, which serves as a measure of return uncertainty in financial markets. Mathematically, volatility is measured by either the variance or standard deviation of returns. As Danielsson (2011) notes, a further distinction is made between unconditional and conditional volatility. In the literature, a distinction is also made depending on the model, but volatility is usually defined in terms of conditional variance or standard deviation. This is also evident from the stylized facts, where we observe that returns exhibit heteroskedasticity leading to volatility clusters. Simple variance or standard deviation describes the unconditional volatility over the entire observation period, which is understood as the mean constant value of the fluctuation in returns. This does not take into account the time dependent structure so that given a set of information, the conditional volatility in the form of the conditional variance or standard deviation is important. Linguistically, we follow the literature and will not distinguish between conditional and unconditional variance, because it is obvious from the derived equations which form is meant.
Volatility is a latent variable that has to be approximated, for which there are various methods.[13] The fact that volatility is an unobservable variable has a significant impact on modeling and forecasting. The choice of a volatility proxy is therefore an essential factor, as it has a decisive impact on forecast evaluation and model comparison. Research by Andersen and Bollerslev (1998), Lopez (2001), Hansen and Lunde (2006a), Liu et al.

[13]See Poon (2005).

(2015) and others show that common volatility proxies, which have desirable statistical properties such as unbiasedness, can be very noisy and are often the reason for poor forecast performance of classical volatility models and may not lead to optimal model selection.
In this paper, we consider two different proxies and use them to assess and compare the forecasting performance of the subsequent models. We only consider volatility models that have an autoregressive structure.



(a) DAX daily squared returns (b) DAX realized 5-min variance of returns

Figure 2.2.: Autocorrelation of DAX volatility proxies. Calculated based on daily squared returns and intraday squared returns with high frequency data

Daily Squared Returns Volatility Proxy

In the classical literature on modeling and forecasting financial market volatilities, the most widely used volatility proxy is daily squared returns r_t^2. To understand why this proxy is used and what its properties are, we start from the yield process equation (2.15). As described in the previous section, due to observed volatility clusters and the resulting time-dependent variance of returns as well as the dependencies in squared returns, it is implausible that this process adequately describes returns. In consideration of the mentioned properties, we follow the explanations of Hassler (2007) and assume that the random innovation term ϵ_t in equation (2.15) can be multiplicatively decomposed into a pure random process scaled by a volatility process. Mathematically, this can be represented by equation

$$r_t = \sigma_t z_t \tag{2.16}$$

where $z_t \sim IID(0,1)$ is a pure random process, σ_t^2 is the volatility process or variance which is assumed to be independent of z_t. Further, we assume that volatility σ_t^2 can be modeled by the past of the process, i.e.

$$\sigma_t^2 = f(r_{t-1}, r_{t-2}, \dots) \tag{2.17}$$

holds. With equation (2.17) we can now show that the process from equation (2.16) represents a martingale difference. We consider the amount of information $\mathcal{I}_{t-1}$ which is generated by the past of our process. Further we assume that r_t is integrable and therefore

$$\begin{aligned} \mathbb{E}[r_t|\mathcal{I}_{t-1}] &= \mathbb{E}[\sigma_t z_t|\mathcal{I}_{t-1}] \\ &= \sigma_t \mathbb{E}[z_t|\mathcal{I}_{t-1}] \\ &= \sigma_t \mathbb{E}[z_t] \\ &= 0 \end{aligned} \tag{2.18}$$

holds, where the second equality follows from the independence of σ_t and z_t and that σ_t is $\mathcal{I}_{t-1}$ measurable. The third and fourth equality come from the fact that by construction z_t is independent of r_{t-j} for $j > 0$ and z_t is zero on average. Overall, this shows that the return process from equation (2.16) is a martingale difference. This property now allows us to analyze squared returns r_t^2 as a proxy of volatility. It applies

$$\begin{aligned} var(r_t|\mathcal{I}_{t-1}) &= \mathbb{E}[r_t^2 \mid \mathcal{I}_{t-1}] \\ &= \mathbb{E}[\sigma_t^2 z_t^2 \mid \mathcal{I}_{t-1}] \\ &= \sigma_t^2 \mathbb{E}[z_t^2 \mid \mathcal{I}_{t-1}] \\ &= \sigma_t^2 \end{aligned} \tag{2.19}$$

where the first equality is a consequence of the martingale property from (2.18). The third equality follows again from the independence of σ_t^2 and z_t and that σ_t^2 is $\mathcal{I}_{t-1}$ measurable and the fourth equality holds since z_t^2 is independent of r_{t-j} for $j > 0$ and has a variance of one. Overall, this shows that squared returns is an unbiased estimator of the conditional variance of returns, justifying the use of squared returns as a proxy for volatility. However, there is also much criticism towards this proxy. Andersen and Bollerslev (1998) show that while squared returns is a conditionally unbiased estimator, it is by construction subject to strong fluctuations. According to the authors, this leads to significant underestimation in forecasting performance of models and possibly

to wrong conclusions being drawn when comparing models.
Hansen and Lunde (2005) come to a similar conclusion. In their study, they refer to squared returns as a noisy proxy that is largely responsible for the poor forecasting performance of volatility models. In another study, Hansen and Lunde (2006a) show that using squared returns as the volatility proxy in choosing the optimal forecasting model leads to inconsistent model rankings. They demonstrate that with probability converging to one as the sample size increases, a non-optimal forecasting model is incorrectly chosen as the best model.
To better understand why squared returns is an imprecise and noisy volatility proxy, we follow the approach of Lopez (2001). For this, we assume that $r_t = \epsilon_t = \sigma_t z_t$ with $z_t \sim \mathcal{N}(0,1)$. From statistical theory, we know that the square of a standard normal distributed random variable follows a chi-squared distribution with one degree of freedom, thus $z_t^2 \sim \chi^2_{(1)}$ holds. In addition, we know that equation (2.19) is true for the assumed case. The median of a $\chi^2_{(1)}$ distributed random variable is 0.455, which in the present case means that squared returns r_t^2 are smaller than $\frac{1}{2}\sigma_t^2$ in 50% of the time. Now the question arises of what is the probability that squared returns deviate from variance σ_t^2 by more than 50%. This probability can be determined by the $\chi^2_{(1)}$ distribution of z_t^2. From this we obtain

$$\begin{aligned}
\mathbb{P}\left(r_t^2 \in \left[\frac{1}{2}\sigma_t^2, \frac{3}{2}\sigma_t^2\right]\right) &= \mathbb{P}\left(\frac{1}{2}\sigma_t^2 \leq r_t^2 \leq \frac{3}{2}\sigma_t^2\right) \\
&= \mathbb{P}\left(\frac{1}{2} \leq \frac{r_t^2}{\sigma_t^2} \leq \frac{3}{2}\right) \\
&= \mathbb{P}\left(\frac{1}{2} \leq z_t^2 \leq \frac{3}{2}\right) \\
&= \mathbb{P}\left(z_t^2 \in \left[\frac{1}{2}, \frac{3}{2}\right]\right) \\
&= 0.2588
\end{aligned}$$

which indicates the probability that squared returns do not deviate more than 50% from variance σ_t^2. To answer our question, we need to determine the converse probability.

Thus, we obtain

$$\begin{aligned}\mathbb{P}\left(r_t^2 \notin \left[\frac{1}{2}\sigma_t^2, \frac{3}{2}\sigma_t^2\right]\right) &= \mathbb{P}\left(z_t^2 \notin \left[\frac{1}{2}, \frac{3}{2}\right]\right)\\ &= 1 - \mathbb{P}\left(z_t^2 \in \left[\frac{1}{2}, \frac{3}{2}\right]\right)\\ &= 0.7412.\end{aligned}$$

This probability indicates that squared returns deviate from variance σ_t^2 by more than 50% in 74.12% of the time. These results confirm the empirical results and show that squared returns are an unreliable estimator of the volatility. Poon and Granger (2003) note that, in addition to the problems already discussed, the use of this volatility proxy also means that the interpretation of the results is significantly affected by it and can lead to incorrect conclusions. They therefore suggest the use of absolute returns as the volatility estimator. This is more robust to extreme values but is plagued by similar problems as squared returns which is why we do not consider it in more detail. Overall, it appears that the volatility proxies widely used in the literature, such as squared or absolute returns, appear to be only partially suitable for approximating volatility. One reason for using these proxies may have been the low availability of high frequency data. The high costs of high frequency data may also be a possible reason for using noisy volatility proxies.

Realized Variance

The second volatility proxy we consider in this paper, *realized variance*, is based on high frequency data and also uses squared returns. As a measure of variation in returns, squared returns are sampled in regular intervals and summed within a day.[14] The choice of realized variance as a second volatility proxy can be justified by its robust empirical performance in many studies. Andersen and Bollerslev (1998) argue that employing realized variance as a volatility proxy leads to significantly better forecasting performances of GARCH models and therefore should be used. Blair et al. (2010) demonstrate that one-step-ahead out-of-sample forecasts improve significantly when realized variance is used as a volatility proxy instead of squared returns. They report an increase R^2 of regressions of realized volatility on one or more volatility forecasts by a factor of three

[14]The sampled squared returns from a regular interval like 5, 10, or 15 minutes are called intraday squared returns.

to four. Hansen and Lunde (2006a) show that among different proxies, realized variance best approximates the true variance and leads to a superior model ranking. In addition, Danielsson (2011) argues that realized variance offers several advantages as it is a non-parametric approach that uses high frequency data and can be used with classical models.
To define realized variance, we mainly follow Hautsch (2011), Poon (2005), and Zivot (2011). The basic idea is that the returns can be observed not only once a day but theoretically at an arbitrarily high frequency. To account for this, we denote m as the trading frequency, i.e. the number of sampled returns per time interval of length $\Delta = \frac{1}{m}$ and T as the number of observed days. Then $mT = \frac{T}{\Delta}$ is the number of all observations. In an example using the DAX, 8.5 hours are traded per day. If we assume that returns are sampled every five minutes, then we have $m = \frac{8.5hrs}{5min} = \frac{510min}{5min} = 102$, meaning that there are 102 five-minute intervals per trading day. To generally determine the continuous intraday returns from time $t-1+(j-1)\Delta$ to $t-1+j\Delta$ we define $r_{t-1+j\Delta} = p_{t-1+j\Delta} - p_{t-1+(j-1)\Delta}$ for $j = 1, \dots, m$. Summing all returns for $j = 1, \dots, m$, we obtain the returns for day t as $r_t = r_{t-1+\Delta} + r_{t-1+2\Delta} + \dots + r_{t-1+(m-1)\Delta} + r_{t-1+m\Delta}$. By squaring the individual intraday returns and summing them up accordingly, we then obtain the definition of realized variance for day t as

$$RV_t^{(m)} = \sum_{j=1}^{m} r_{t-1+j\Delta}^2, \quad t = 1, \dots, T. \tag{2.20}$$

From the definition of realized variance, the similarities and differences to squared returns become clear. Instead of using only one squared return observation per day as a volatility proxy, m squared returns are used here for this purpose. Similar to squared returns, several questions arise: How can we use the realized variance as a volatility proxy and what exactly we are approximating? What are the properties of this proxy? To answer these questions, we need methods from stochastic calculus and continuous-time stochastic processes. To keep mathematically demanding methods to a minimum, we try to create an intuitive approach which, should still explain all essential formal results. Similar to how (2.13) was derived for discrete-time processes, this can be done for continuous-time processes. For this purpose, we assume that the log-price process follows an Ito process or Ito diffusion, which can be thought of as the continuous counterpart to the random walk model from (2.12) since the Brownian motion is a special case of an Ito process. This is not mathematically correct, but should clarify the intuition behind the concept. However, the price process can therefore be represented as a martingale of the form

$$p(t) = p(0) + \int_0^t \mu(s)\mathrm{d}s + \int_0^t \sigma(s)\mathrm{d}W(s) \tag{2.21}$$

where $\mu(s)$ is a drift process, $W(s)$ is a standard Brownian motion, and $\sigma(s)$ is a volatility process which is strictly positive and square-integrable. To determine the continuous yield over interval $[0, t]$, we determine $r(0, t) = p(t) - p(0)$ and get

$$r(0, t) = \int_0^t \mu(s)\mathrm{d}s + \int_0^t \sigma(s)\mathrm{d}W(s). \tag{2.22}$$

If we assume in addition that the present series has a zero mean and no jumps, then it follows that the returns are generated by a continuous time martingale

$$r(0, t) = \int_0^t \sigma(s)\mathrm{d}W(s). \tag{2.23}$$

This representation can be seen as a continuous counterpart to the yield process from equation (2.16). Again, latent volatility $\sigma(s)$ scales the random process $\mathrm{d}W(s)$, but now continuously over time.[15] Now, to establish the connection to realized variance, we first consider the partitioning of interval $[0, t]$ with $P_n([0, t]) : 0 = s_o < s_1 < \cdots < s_n = t$. For a stochastic function g, we define the quadratic variation over this partitioning as

$$QV_n(0, t) = \sum_{i=1}^{n} (g(s_i) - g(s_{i-1}))^2 \tag{2.24}$$

and say that g is of finite quadratic variation if the limit $QV_n(0, t) \xrightarrow{m.s.} QV(0, t)$ in mean square exists. Quadratic variation measures how rough, jagged, or irregular the realization of the process over interval $[0, t]$ is. It is distinct from the variance of the process because quadratic variation is dependent on the sample path, whereas variance is a mean value over all possible sample paths. Turning to the yield process (2.23), its quadratic variation is given by

$$QV(0, t) = \int_0^t \sigma^2(s)\mathrm{d}s. \tag{2.25}$$

[15]The processes are given in integral representation because this is simpler for the derivations. Often Ito processes are also represented in differential notation. For the processes (2.21) and (2.22), this is $\mathrm{d}p(t) = \mu(t)\mathrm{d}t + \sigma(t)\mathrm{d}W(t)$. For process (2.23) it follows $\mathrm{d}p(t) = \sigma(t)\mathrm{d}W(t)$.

The quadratic variation from equation (2.23) is also valid for more general Ito processes like (2.22) because the drift process has a quadratic variation of zero. Thus, only the quadratic variation of a standard Brownian motion over interval $[0, t]$ has influence on the variation of the return process. A proof that the quadratic variation corresponds exactly to the expression (2.25) can be found in the appendix of this thesis. In addition, squared variation is equal to integrated variance, which can be considered analogous to variance of the returns from discrete-time models. More formally, it holds that $QV(0, t) = IV(0, t)$ since the return process (2.23) is continuous.

The relationship between realized variance and squared variation is that realized variance can be used as a consistent estimator of squared variation of the return process. Thus, daily realized variance $RV_t^{(m)}$ in equation (2.20) converges in probability to the daily squared variation of the return process $QV(t-1, t) = QV(0, t) - QV(0, t-1)$ for increasing sample frequency $m \to \infty$. More formally

$$\lim_{m\to\infty} \mathbb{P}\left(\left| \int_{t-1}^{t} \sigma^2(s)\mathrm{d}s - \sum_{j=1}^{m} r_{t-1+j\Delta}^2 \right| \geq \epsilon \right) = 0 \tag{2.26}$$

holds. According to Poon (2005), the convergence statement follows from the convergence theory for martingales and a proof can be found in Karatzas and Shreve (2012). Moreover, the convergence result from (2.26) implies that at time t, volatility is theoretically observable as long as sample frequency m is high enough. Further, from the previous derivations, we can state that squared variation measures the variation of the realized return process and that realized variance is a consistent estimator for it. As Andersen et al. (2001) note, it is not generally true that quadratic variation coincides with daily conditional variance. Under the conditions that the price process is square-integrable, the mean process is continuous, and the daily mean process is a predeterminate process which is, given information $\mathcal{I}_t$, independent of the daily return process, then it can be shown that the conditional variance of the daily return process is equal to the conditional expected value of the daily squared variation. Based on these conditions and assuming the return process from (2.23), we get

$$\begin{aligned} var\left(\int_{t-1}^{t} \sigma(s)\mathrm{d}W(s) \middle| \mathcal{I}_{t-1} \right) &= \mathbb{E}\left[\int_{t-1}^{t} \sigma^2(s)\mathrm{d}s \middle| \mathcal{I}_{t-1} \right] \\ &= \mathbb{E}\left[\sum_{j=1}^{m} r_{t-1+j\Delta}^2 \middle| \mathcal{I}_{t-1} \right]. \end{aligned} \tag{2.27}$$

Under the assumptions, it then also follows that realized variance is an unbiased estimator of the conditional variance of returns. The previous results even hold for equation (2.23) when $\mu(t) = 0$. The results are also reflected in the asymptotic theory for realized variance. When the mean and volatility processes are independent of the Brownian motion and assuming that the price process contains no jumps, Andersen et al. (2003) show that standardized returns are normally distributed, i.e., $\frac{r_t}{\sqrt{RV_t^{(m)}}} \sim \mathcal{N}(0,1)$ holds. Barndorff-Nielsen and Shephard (2002) show that the asymptotic distribution of realized variance is also standard normally distributed when the mean and variance processes are independent of the Brownian motion and can be specified as

$$\sqrt{m}\left(\frac{RV_t^m - IV}{2\sqrt{IQ}}\right) \sim \mathcal{N}(0,1) \tag{2.28}$$

where $IV = IV(t-1,t)$ which is interchangeable with $QV(t-1,t)$ and $IQ = IQ_t(t-1,t)$ is the integrated quarticity which can be consistently estimated in a similar fashion as the integrated variance with $RQ_t^{(m)} = \frac{m}{3}\sum_{j=1}^{m} r_{t-1+j\Delta}^4$.

High-frequency data and continuous price theory offer many new possibilities for modeling and forecasting volatility. However, the key assumptions in the models are often not tenable in reality and sampling at a high frequency poses some challenges. One problem is that all presented results on realized variance, especially the consistency property, were derived under the assumption that the price process is a martingale and an arbitrarily high sample frequency m can be chosen such that $\Delta \to 0$ holds. The assumption that continuous price data are available in any sample frequency is not true in practical applications and the convergence statement (2.26) does not hold and therefore realized variance is a biased estimator.[16] As mentioned by Hansen and Lunde (2006b), in addition to the discretization of prices, there are other effects such as bid-ask bounces, noise trading, different trading mechanisms, trading volumes, and asymmetric or asynchronous trading in the markets that cause distortions in the calculation of realized variance.[17] These effects are summarized in the literature as *market microstructure effects* and, as Hansen and Lunde (2006b) show, cause spurious correlations in intraday returns. The authors find that increasing sample frequency increases the bias of realized variance. To

[16]See Hautsch (2011).

[17]Bid-ask bounces occur when the price of an asset jumps back and forth between the bid and ask price.
Noise trading corresponds to rumor-based trading, rather than data-based trading, and do not follow rationality principles.
Asynchronous trading arises because different stocks or assets have different trading frequencies.

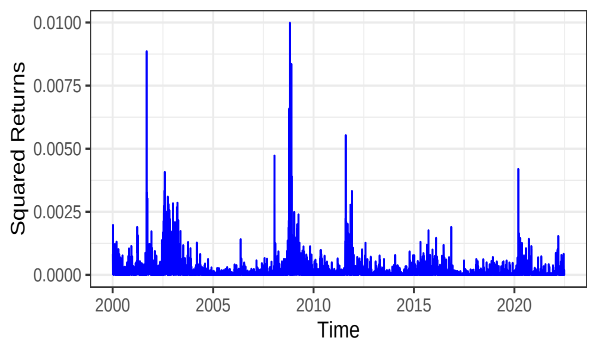
illustrate this, Hautsch (2011) states that in the presence of market microstructure effects, the observable price process is given by $p_t = p_t^* + v_t$. If $v_t \sim IID(0, \omega^2)$ is assumed, it can be shown that

$$\mathbb{E}[RV_t^{(m)}] = IV(t-1,t) + 2m\omega^2$$

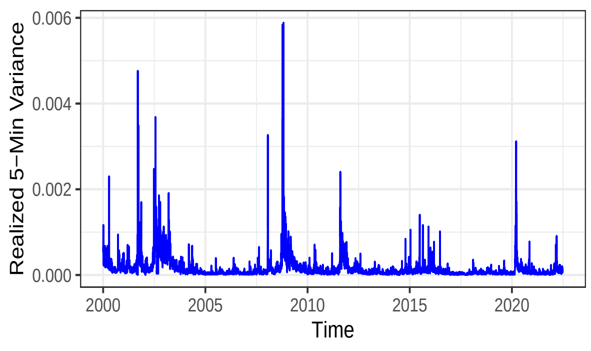
holds. Thus, it is clear that for a higher sample frequency m, the bias of realized variance increases and noise becomes more dominant in the estimation. To reduce the bias of realized variance, Hansen and Lunde (2006b) propose the *sparse sampling principle.* Instead of sampling at the maximum possible frequency, a coarser frequency such as five-, ten- or fifteen-minute frequency should be used. However, the reduction of bias comes at the cost of higher variance leading to a less precise estimation. Overall, there is a bias-variance tradeoff with respect to the sample frequency m that must be taken into account when choosing realized variance.
In the literature, realized variance is usually based on a five-minute frequency. Why this frequency is a balanced choice is shown by Liu et al. (2015), who compare different volatility proxies based on high frequency data and compare them with the realized variance on a five-minute basis. They use 400 volatility proxies with frequencies between one second and fifteen minutes for different assets and stocks from the U.S. and U.K. over a period of eleven years. The proxies are then compared with a data-based approach as well as with respect to their forecasting performance using simple autoregressive forecasting models. The authors find no clear evidence that one of the considered proxies systematically outperforms realized variance on a five-minute basis. There are indeed proxies that perform better in some applications, but these are usually very complex volatility proxies that were constructed specifically for the problem at hand. Moreover, very complex proxies perform particularly well in the U.S. markets, but this good performance is not transferable to other markets. Overall, realized variance on a five-minute basis is a good approximation that is very difficult to outperform by other proxies. Based on presented results, in this work we calculate the realized variance with a sample frequency of five minutes.

Volatility of the DAX

Similar to daily squared returns, Figure 2.2b shows the autocorrelation of realized variance sampled at five minutes. The autocorrelation structure of realized variance appears to be significantly stronger and more persistent than that of squared returns. This would argue for autoregressively modeling realized variance directly instead of the squared returns. We take up this idea in the following chapters not only for econometric models but also for machine learning models. Now we consider the volatility approximation



(a) DAX volatility measured with daily squared returns



(b) DAX volatility measured with realized 5-min variance

Figure 2.3.: DAX Volatility based on daily squared returns and intraday squared returns with five minutes frequency

of the two estimators presented in Figure 2.3. First, it should be noted that strongly and weakly volatile phases are captured in both charts at the same points in time. For example, a spike in volatility can be observed in the early 2000s, which may be related to the dot-com bubble. Also, effects of the 2007/2008 financial crisis of and the beginning of the corona pandemic in March 2020 can be clearly seen in both figures. It is remarkable that the swings are perceived at the same time, but at completely different levels. This becomes clear when we see the different scales of volatility on the charts. On average, squared returns seem to react much more strongly to financial markets shocks than realized variance. This can lead to the interpretation that both volatility proxies assess exogenous shocks to the market quite differently.

From these observations, the question arises, which of the two volatility proxies can be considered a better approximation of quality. This question is difficult to answer. As Lopez (2001) states, the noise in squared returns decreases for an increasing observation period, so the proxy becomes more accurate. However, from a statistical and empirical perspective, realized variance has more desirable properties and show better performance in many studies compared to squared returns. Also, there is consensus in the academic literature that bias in realized variance due to market microstructure effects no longer matters with realized variance on a five-minute basis and realized variance is preferable

compared to squared returns. All in all, it is interesting and important to explore the difference of the two volatility proxies as financial institutions base their risk management on these proxies. In this paper, we want to check whether different proxies come to different model rankings or if they give similar results.
At this point we would like to make a short practical remark, which is very rarely mentioned in the literature on volatility in financial markets. While studying volatility and its approximation, we notice that how one measures returns makes a significant difference. If one were to determine returns as the log difference of closing prices at t and $t-1$ and then square these to obtain the volatility proxy, this proxy cannot be used for comparison with realized variance. The reason is that this measurement captures overnight effects, because the closing price at $t-1$ is not necessarily the same as the opening price at t. If this is the case, these effects have an impact on returns and can create additional volatility. Realized variance, on the other hand, uses squared returns of the respective day, i.e., between opening prices and closing prices at time t, but between trading days the overnight effects are not taken into account. Thus, to compare both volatility proxies, returns must be measured as the log difference between the closing price and the opening price at time t.

Volatility and Risk

To conclude this chapter, we briefly discuss the differences and similarities between volatility and risk. According to Poon and Granger (2003), this is necessary because it is often difficult to differentiate between volatility and risk. From a heuristic point of view, volatility is an objective measure of uncertainty that also includes positive events. Risk, in contrast, is a subjective measure that relates only to negative events and depends on the individual utility function of investors. The uncertainty together with the utility function describe the risk attitudes of investors. Following this logic volatility serves as a central input variable in determining risk. According to Poon (2005) and Danielsson (2011), in the context of financial time series analysis, volatility is a measure of the spread of a distribution but not of its shape, which is the reason why volatility is generally distinct from risk. Risk depends on the underlying distribution of returns. Volatility can be equated with risk if one assumes that returns are normally distributed or log-normally distributed, as those distributions are fully described by the mean and variance. However, if other distributions are used with additional parameters, these must also be considered. This means that, in general, volatility is not equal to risk.

3. Econometric Forecasting Models

Volatility forecasting concerns all areas in which financial risk exists and has occupied the scientific and business communities for many decades. The strong significance of volatility forecasting in risk analysis and risk management comes from the fact that volatility is the central input variable in determining risk. Accordingly, there numerous of models to predict volatility. They must also account for stylized facts, especially the facts that fluctuation of returns over time has variations and that clusters occur. As we have seen, this is expressed by the observation that although returns are uncorrelated, squared returns show significant and persistent autocorrelation. This dependence of squared returns is used by the *Autoregressive Conditional Heteroskedasticity (ARCH)* model introduced by Engle (1982). These models use past squared return observations to model conditional variance to reflect the time-dependent structure of volatility, which is also the basis for its name.
Since the introduction of ARCH models, there has been a boom in research from which numerous generalizations and extensions have emerged, such as the GARCH model by Bollerslev (1986). More modern approaches and models were gradually developed due to the advent and greater availability of high frequency data. Volatility proxies such as realized variance have become increasingly important in modeling and forecasting. As Figure 2.2b shows, realized variance has a similar autocorrelation structure as squared returns, but much stronger and more persistent. Researchers built upon this characteristic, leding to autoregressively modeling realized variance and using it for forecasting. Probably the best known and most widely used model for this purpose is the *Heterogeneous Autoregressive Model of Realized Volatility (HAR-RV)* by Corsi (2009). Instead of a simple autoregressive model, this model uses a parsimonious, cumulative approach to model and forecast realized variance. It attempts to describe volatility using the average level of realized variance over the last day, week, and month.

3.1. The ARCH Model

To introduce the ARCH models, we follow Hassler (2007) and Kirchgässner et al. (2012). The idea of these models is based on a return process that can be decomposed multiplicatively as in equation (2.16) and, in addition, it is assumed that conditional variance can be modeled as a function of the past of the process, as in equation (2.17). Thus, we can specify an ARCH(q) model with the following equations

$$\begin{aligned} r_t &= \sigma_t z_t \\ z_t &\sim IID(0,1) \\ \sigma_t^2 &= \alpha_0 + \sum_{i=1}^{q} \alpha_t r_{t-i}^2. \end{aligned} \tag{3.1}$$

There are several things to consider with this definition. First, the conditional mean is zero since ARCH processes are martingale differences, which is a consequence from equation (2.18). Furthermore, it follows from the martingale difference property and the law of total expectation that the return process is uncorrelated, where

$$\begin{aligned} cov(r_t, r_{t+\tau}) &= \mathbb{E}[r_t r_{t+\tau}] \\ &= \mathbb{E}[\mathbb{E}[r_t r_{t+\tau} | \mathcal{I}_{t+\tau-1}]] \\ &= \mathbb{E}[r_{t+\tau} \mathbb{E}[r_t | \mathcal{I}_{t+\tau+1}]] \\ &= 0. \end{aligned} \tag{3.2}$$

Overall, it shows that we have a serially uncorrelated process with zero mean. Such processes are described as white noise processes, but they are not necessarily independent over time. Moreover, it follows from equation (2.19) that

$$var(r_t | \mathcal{I}_{t-1}) = \sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i r_{t-i}^2 \tag{3.3}$$

holds. Since we model conditional variance, which is strictly greater than zero, we have to impose some conditions on the model parameter. To ensure that $\sigma_t^2 > 0$ applies, $\alpha_0 > 0$ and $\alpha_i \geq 0$ must hold for all $i = 1, \ldots, q$. In addition to the positivity condition for conditional variance, a finite and positive variance expression of the ARCH process exists exactly when it is stationary. Given the definition (3.1) and that the non-negativity conditions hold, the stationarity condition for the ARCH(q) model can be written as

$$\sum_{i=1}^{q} \alpha_i < 1. \tag{3.4}$$

A proof that the stationarity condition (3.4) is necessary and sufficient can be found in Hassler (2007). From the definition of the ARCH process, we observe that it is able to capture volatility clusters. If a large shock in returns occurs, subsequent values of conditional variance are also larger because equation (3.3) is a monotonically increasing function of past squared returns. The same applies to small shocks. The parameters α_i and the order q of the model plays a crucial role in modeling the volatility clusters. The coefficients α_i indicate how strong the volatility clusters are. The larger the values of the model parameters, the more clearly volatility clusters emerge. The model order q indicates how many parameters are necessary to reproduce volatility patterns with the model. A larger value of q implies that more and larger clusters are present in the data. However, it must be considered that the model order also determines the number of parameters in the model and a large q would contradict the parsimony principle.
Another aspect to note from the definition (3.1) is that z_t is assumed to be an IID random process and a distribution is not explicitly described. This means that although Engle (1982) assumes in his original paper that $z_t \sim \mathcal{N}(0, 1)$ holds, this assumption is not mandatory so that it is also possible to choose other distributions that may better describe fat tails such as a t-distribution. Nevertheless, even assuming a normal distribution of the innovations, it is possible to approximate leptokurtic distributions. This follows from the fact, that under this assumption, the conditional distribution given the information of the past process is given as $r_t|\mathcal{I}_{t-1} \sim \mathcal{N}(0, \sigma_t^2)$. This does not imply that the joint nor the marginal distribution are normal. To show that leptokurtic distributions can be approximated under the assumption $z_t \sim \mathcal{N}(0, 1)$ kurtosis must be determined. We know that for a standard normally distributed random variable, the kurtosis is $\mathbb{E}[z_t^4] = 3$ and, in addition, we assume $\mathbb{E}[\sigma_t^4] < \infty$. This gives us

$$\begin{aligned} \mathbb{E}[r_t^4] = \mathbb{E}[\sigma_t^4 z_t^4] &= 3\mathbb{E}[\sigma_t^4] \geq 3\mathbb{E}[\sigma_t^2]^2 \\ \Rightarrow \qquad \frac{\mathbb{E}[r_t^4]}{\mathbb{E}[\sigma_t^2]^2} &\geq 3. \end{aligned} \tag{3.5}$$

Here, the second equality results from the independence of σ_t and z_t as well as the previously mentioned kurtosis. The inequality follows from Jensen's inequality or also

from the definition of variance.[1]
The estimation of ARCH models is mostly done with the maximum likelihood method. We do not discuss this method here in the context of ARCH models, but rather in the following chapter on generalized ARCH models. The reason is that the likelihood functions are almost identical, and the ARCH likelihood function can be derived from the GARCH likelihood function. Before we move on to the generalized ARCH models, the question arises why a generalization of the models is necessary at all. The reason for the generalization of ARCH models can be easily understood from Figure 2.2a. Here we see that the autocorrelation function of squared returns is present for a relatively long time, i.e., it has a high persistence or long memory. This means that we would have to choose a high model order q to capture volatility clusters, which contradicts the sparsity principle. Moreover, due to the need for a high model order q, problems with the non-negativity constraints may arise if the estimates are not properly restricted. The restrictions may result in the dynamics being only partially or insufficiently captured, resulting in inflexible conditional variance dynamics for the model.

3.2. The GARCH Model

The introduction of the generalized ARCH models (GARCH) by Bollerslev (1986) is largely due to the practical needs and associated problems of the ARCH model mentioned. To describe the model, we essentially follow the explanations of Hassler (2007), Kirchgässner et al. (2012) and Tsay (2005). The idea behind the GARCH models is easiest to describe by thinking of the ARCH model as an AR process for volatility. Accordingly, one can think of the GARCH model as an ARMA process for volatility. Therefore, the main difference between GARCH and ARCH models is that conditional variance additionally depends on the lagged values of the process itself. Thus, the GARCH(p,q) process can be defined as

$$\begin{aligned} r_t &= \sigma_t z_t \\ z_t &\sim IID(0,1) \\ \sigma_t^2 &= \alpha_0 + \sum_{i=1}^{q} \alpha_t r_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2. \end{aligned} \tag{3.6}$$

[1] Jensen's Inequality: For a convex function g and a random variable X, $\mathbb{E}[g(X)] \geq g(\mathbb{E}[X])$ holds. Note the similarity between Jensen's inequality and variance $var(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2 \geq 0$.

The most important difference is the volatility function, which is calculated by the GARCH model according to (2.19) and is given as

$$var(r_t|\mathcal{I}_{t-1}) = \sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i r_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2. \tag{3.7}$$

Since we again use conditional variance for modeling, it must be greater than zero. To ensure this, in addition to the conditions for the ARCH model $\alpha_0 > 0$ and $\alpha_i \geq 0$ for all $i = 1, \ldots, q$, we also need restrictions for the GARCH parameters. Thus, for these, $\beta_i \geq 0$ for all $i = 1, \ldots, p$ must also hold. Furthermore, very similar to the stationarity condition of the ARCH model from (3.4), GARCH models must consider the appropriate model parameters. Thus we call a GARCH(p,q) model that satisfies the non-negativity conditions stationary with finite existing variance, if

$$\sum_{j=1}^{q} \alpha_j + \sum_{i=1}^{p} \beta_i < 1. \tag{3.8}$$

holds. In the last section, we mentioned that the GARCH(p,q) model allows a more parsimonious parameterization than the ARCH(q) model. This may not be obvious at first glance since the GARCH model has $p + q$ parameters and the ARCH model has only q. The reason for the parsimony of the GARCH model compared to the ARCH model is that a stationary GARCH model can be transformed into an ARCH(∞) model. Thus, a GARCH process can also be written as

$$\sigma_t^2 = \nu_0 + \sum_{i=1}^{\infty} \nu_i r_{t-i}^2 \tag{3.9}$$

with $\nu_i \geq 0$ and $\sum_{i=1}^{\infty} |\nu_i| < \infty$. A derivation of this result for a GARCH(1,1) process can be found in the appendix of this paper. For a general proof we refer to Hassler (2007). Overall, the representation (3.9) shows that GARCH with only $p+q$ parameters is a more flexible representation of an infinite ARCH process. This shows that GARCH processes model an infinitely long dependence of volatility on the past of the process itself. In doing so, the models use exponentially decreasing weights, i.e., $\nu_i \to 0$ for $i \to \infty$, so that squared returns far in the past are assigned a lower weight than more recent ones. Moreover, the fact that GARCH processes can be represented as ARCH(∞) processes has the consequence that all the results already described for stationary ARCH processes also apply to GARCH processes. Thus, for GARCH processes, it also holds that they are martingale differences and, under the assumption that $z_t \sim \mathcal{N}(0,1)$ applies, kurtosis can exceed the value three. It is also true for GARCH models that the normal distribution assumption is not mandatory. Moreover, the choice of the model order p, q is now also essential. Indeed, the larger the sum $\sum_{j=1}^{q} \alpha_j + \sum_{i=1}^{p} \beta_i$ is, the more often phases with low volatility and phases with high volatility alternate.

Maximum Likelihood Estimation

Having introduced the ARCH and GARCH models and discussed their essential properties, we now consider the estimation of the models. The estimation of GARCH models is a challenging task that involves many problems. Usually, ARCH and GARCH models are estimated using the maximum likelihood method but there is no analytically closed form for optimization and, thus, numerical methods must be used. The likelihood functions for ARCH and GARCH models are almost identical and differ only by the number of parameters, since a GARCH$(0, q)$ is an ARCH(q) model. The derivations presented here essentially follow Tsay (2005), Lee and Hansen (1994), and Zivot (2009).
In general, the likelihood function for time series models is somewhat more cumbersome to determine than for data that satisfy the IID assumption. Technically speaking, this means that for dependent data, the joint density cannot be represented as a product of marginal densities. The trick for dependent data is to factorize the density based on the equation $f(x|y) = \frac{f(x,y)}{f(y)}$. Thus, we can specify joint density in the present case as

$$\begin{aligned}
f_\theta(r_0, r_1, \dots, r_T) &= f_\theta(r_T | r_{T-1}, \dots, r_0) f_\theta(r_0, \dots, r_{T-1}) \\
&= f_\theta(r_T | r_{T-1}, \dots, r_0) f_\theta(r_{T-1} | r_{T-2}, \dots, r_0) f_\theta(r_0, \dots, r_{T-2}) \\
&\vdots \\
&= \prod_{t=1}^{T} f_\theta(r_t | r_{t-1}, \dots, r_0) f_\theta(r_0) \\
&= \prod_{t=1}^{T} f_\theta(r_t | \mathcal{I}_{t-1}) f_\theta(r_0).
\end{aligned} \tag{3.10}$$

As Tsay (2005) notes, for a large sample size T the density of r_0 can be neglected and, since maximizing the conditional likelihood is equivalent to maximizing its logarithm, we can generally state the conditional likelihood and the conditional log-likelihood for the full data as

$$\mathcal{L}(\theta|r_0,\ldots,r_T) = \prod_{t=1}^{T} f_\theta(r_t|\mathcal{I}_{t-1}) \tag{3.11}$$

$$\log(\mathcal{L}(\theta|r_0,\ldots,r_T)) = l(\theta|r_0,\ldots,r_T) = \sum_{t=1}^{T} log(f_\theta(r_t|\mathcal{I}_{t-1})). \tag{3.12}$$

To better analyze the estimation of GARCH models, we consider a GARCH$(1,1)$ model of the form $\sigma_t^2 = \alpha_0 + \alpha_1 r_{t-1}^2 + \beta_1 \sigma_{t-1}^2$ and assume that the returns given past information are normally distributed, so $r_t|\mathcal{I}_{t-1} \sim \mathcal{N}(0, \sigma_t^2)$ holds. This can be justified by the fact that this model specification is often sufficient to obtain a good model fit.[2] The distribution assumption may often not be true, but according to Zivot (2009), it has been shown that the maximization of the Gaussian likelihood results in the quasi-maximum likelihood estimate, which is consistent and asymptotic normally distributed given a proper model specification.[3] Thus, we can write the parameter vector as $\theta = (\alpha_0, \alpha_1, \beta_1)^\top$, where we actually do not know the value of σ_1^2 but take it as given. In practice, this parameter is either estimated or randomly determined. Thus, we can obtain the conditional log-likelihood from (3.12) as

$$l(\theta|r_0,\ldots,r_T) \propto -\frac{1}{2}\sum_{t=1}^{T}\left(ln(\sigma_t^2) + \frac{r_t^2}{\sigma_t^2}\right) \tag{3.13}$$

where $\sigma_t^2 = \sigma_t^2(\theta)$ depends on unknown parameters because of the GARCH$(1,1)$ specification. To maximize the conditional log-likelihood, we can now perform the derivative of this function with respect to the parameter vector θ and obtain the score function

$$\begin{aligned}\frac{\partial l(\theta|r_0,\ldots,r_T)}{\partial\theta} &= s(\theta|r_0,\ldots,r_T) \\ &= -\frac{1}{2}\sum_{t=1}^{T}\left(\frac{1}{\sigma_t^2(\theta)} - \frac{r_t^2}{(\sigma_t^2(\theta))^2}\right)\frac{\partial\sigma_t^2(\theta)}{\partial\theta} \\ &= \sum_{t=1}^{T}\frac{\partial l(\theta|r_0,\ldots,r_T)}{\partial\theta}\frac{\sigma_t^2(\theta)}{\partial\theta}\end{aligned} \tag{3.14}$$

[2]See Hansen and Lunde (2005).
[3]See Lee and Hansen (1994).

where the derivative $\frac{\partial \sigma_t^2(\theta)}{\partial \theta}$ must be partially differentiated according to the parameters of the vector θ. Thus, one obtains

$$\frac{\partial \sigma_t^2(\theta)}{\partial \alpha_0} = 1 + \beta_1 \frac{\partial \sigma_{t-1}^2(\theta)}{\partial \alpha_0} \tag{3.15}$$

$$\frac{\partial \sigma_t^2(\theta)}{\partial \alpha_1} = r_{t-1}^2 + \beta_1 \frac{\partial \sigma_{t-1}^2(\theta)}{\partial \alpha_1} \tag{3.16}$$

$$\frac{\partial \sigma_t^2(\theta)}{\partial \beta_1} = \sigma_{t-1}^2 + \beta_1 \frac{\partial \sigma_{t-1}^2(\theta)}{\partial \beta_1}. \tag{3.17}$$

Overall, there is no closed analytic form, so maximization of the conditional log likelihood must be performed using numerical methods. However, according to Zivot (2009), there are still some problems with the optimization of the objective function. Especially the initial values, the choice of the optimization algorithm, and the choice of the convergence criterion are crucial for the stability of the estimation. In addition, the likelihood function for models with many parameters is not always well-behaved, so that optimization methods exhibit convergence problems and may only find local maxima. Overall, the facts described here must be considered when modeling and forecasting volatility with GARCH models. Attention should be paid to more parsimonious models under a normal distribution assumption. This ensures that a more stable solution is obtained when optimizing the likelihood, which represents a consistent and asymptotic normally distributed estimate, and this approach is also supported by the studies mentioned.

3.3. The HAR-RV Model

The ARCH and GARCH models considered so far use past squared returns and past values of the process to forecast volatility. However, according to Andersen et al. (2003) these models are unsuitable for high frequency data such as realized variance and often show poor forecasting performance. The authors were able to show that simple time series models that directly model realized variance are significantly superior to GARCH models in forecasting. Moreover, as we discussed in chapter 2, realized variance has better properties than squared returns from a statistical point of view, such that Andersen and Bollerslev (1998) suggest using this proxy for model evaluation for GARCH forecasts. Thus, it also seems obvious to model realized variance directly and make forecasts.

Based on this idea, Corsi (2009) proposed the *heterogeneous autoregressive model for*

realized variance (HAR-RV) which is the best known and most widely used model for realized variance. According to Corsi (2009), the HAR-RV model is an additive cascade model that uses different volatility frequencies, starting from the lowest to the highest. The economic justification of the model comes from the *heterogeneous market hypothesis* introduced by Müller et al. (1993). This hypothesis essentially states that there are various reasons why heterogeneities occur in financial markets, such as different market players, constraints, time horizons, and geographical location, as well as many other factors. For volatility modeling specifically, different market participants and their different time horizons play a particularly important role. It is assumed that different market actors have different time horizons and, therefore, perceive, react to, and cause volatility on the markets differently. Depending on the different time horizons, market participants can be divided into short-term traders who have a daily or higher trading frequency, medium-term traders who adjust their positions approximately weekly and long-term traders who adjust their positions only monthly. Based on this classification, only one volatility component is relevant for each market participant. As Corsi (2009) explains, this can be justified by the fact that long-term volatility is important for high-frequency traders, as it influences future trends and risks. Thus, they adjust their trading strategies to long-term volatility, causing short-term volatility. Meanwhile, short-term volatility does not affect the trading behavior of long-term investors. Overall, this means that long-term volatility affects short-term volatility, but short-term volatility does not affect long-term volatility.
To formalize this idea, let $\tilde{\sigma_t}^{(j)}$ for $j \in \{d, w, m\}$ be the latent volatility for one day $\tilde{\sigma_t}^{(d)}$, one week $\tilde{\sigma_t}^{(w)}$, and one month $\tilde{\sigma_t}^{(m)}$. The frequency of the return process is determined by the highest frequency of volatility. In the case described, this is daily volatility, which coincides with integrated volatility and thus $\tilde{\sigma_t}^{(d)} = \sigma_t^{(d)}$ holds. Therefore, the daily return process can be expressed as

$$r_t = \sigma_t^{(d)} z_t \tag{3.18}$$

where $z_t \sim \mathcal{N}(0, 1)$ is assumed. To account for volatility components and the dependence of long-term volatility on short-term volatility, the volatility processes $\tilde{\sigma_t}^{(j)}$ for $j \in \{d, w, m\}$ are assumed to have an autoregressive structure and involve the expectation of the next lower frequency. Corsi (2009) proposes the following formulation for it

$$\begin{aligned}
\tilde{\sigma_t}^{(m)} &= c^{(m)} + \phi^{(m)} RV_{t-1}^{(m)} + \tilde{\omega_t}^{(m)} \\
\tilde{\sigma_t}^{(w)} &= c^{(w)} + \phi^{(w)} RV_{t-1}^{(w)} + \gamma^{(w)} \mathbb{E}_{t-1}[\tilde{\sigma_t}^{(m)}] + \tilde{\omega_t}^{(w)} \\
\tilde{\sigma_t}^{(d)} &= c^{(d)} + \phi^{(d)} RV_{t-1}^{(d)} + \gamma^{(d)} \mathbb{E}_{t-1}[\tilde{\sigma_t}^{(w)}] + \tilde{\omega_t}^{(d)}
\end{aligned} \tag{3.19}$$

where $\tilde{\omega_t}^{(j)}$ are serially independent innovation terms and $c^{(j)}$ are constants for $j \in \{d, w, m\}$. $RV_{t-1}^{(d)}$, $RV_{t-1}^{(w)}$ and $RV_{t-1}^{(m)}$ denote the daily, weekly, and monthly realized variance, respectively. It should be noted that a typical financial market is assumed with 5 trading days per week and 22 trading days per month. Accordingly, the realized variances can be determined as

$$\begin{aligned}
RV_{t-1}^{(d)} &= \sum_{j=1}^{m} r_{t-2+j\Delta}^2 \\
RV_{t-1}^{(w)} &= \frac{1}{5} \sum_{j=1}^{5} RV_{t-j} \\
RV_{t-1}^{(m)} &= \frac{1}{22} \sum_{j=1}^{22} RV_{t-j}.
\end{aligned} \tag{3.20}$$

To derive the model, forward substitution must be done for each volatility process and $\tilde{\sigma_t}^{(d)} = \sigma_t^{(d)}$ must be considered. This gives

$$\tilde{\sigma_t}^{(d)} = c + \beta^{(d)} RV_{t-1}^{(d)} + \beta^{(w)} RV_{t-1}^{(w)} + \beta^{(m)} RV_{t-1}^{(m)} + \tilde{\omega_t}^{(d)} \tag{3.21}$$

where $c = c^{(d)} + \gamma^{(d)} c^{(w)} + \gamma^{(d)} \gamma^{(w)} c^{(m)}$ and $\beta^d = \phi^d, \beta^{(w)} = \gamma^{(d)} \phi^{(w)}, \beta^{(m)} = \gamma^{(d)} \gamma^{(w)} \phi^{(m)}$. This model represents a three-factor model, with each factor corresponding to realized variance at different time horizons. Taking advantage of the fact that ex-post volatility is realized variance plus an error term and thus $\sigma_t^{(d)} = RV_t^{(d)} + \omega_t^{(d)}$, equation (3.21) can be rewritten as

$$RV_t^{(d)} = c + \beta^{(d)} RV_{t-1}^{(d)} + \beta^{(w)} RV_{t-1}^{(w)} + \beta^{(m)} RV_{t-1}^{(m)} + \omega_t \tag{3.22}$$

where $\omega_t = \tilde{\omega_t}^{(d)} - \omega_t^{(d)}$ applies. Equation (3.22) is the HAR-RV model proposed by Corsi (2009). The model now offers the advantage of directly modeling and forecasting realized variance. Moreover, with this model it is possible to account for the high persistence of realized variance from Figure 2.2b without using an overparameterized model. For comparison, a simple AR model would have to account for 22 lagged variables of realized variance to do so. The economic rationale for restrictions allows the HAR-RV model to capture these dependencies with only three parameters. From this we see that only the average volatility level of the last day, the last week and the last month is of importance for modeling and forecasting.

Corsi (2009) also conducted a model comparison, showing that the HAR-RV model has significantly better forecasting performance than a simple AR model and is on par with a more complex ARFIMA model. As described by Clements and Preve (2021), many other studies find that the HAR-RV model is also able to capture the strong persistence of volatility. In addition, the model was found to have superior forecasting performance in many cases, which led to its acceptance in the literature. As Clements and Preve (2021) argue, despite the relatively simple structure of the model, it captures all relevant characteristics of volatility, has good forecasting performance, and provides an economic interpretation. Further, the authors describe that the simplicity is also reflected in the fact that the model can be estimated using ordinary least squares (OLS). Given the observations $RV_1, \dots, RV_T$, the OLS estimator for $\beta = (c, \beta^{(d)}, \beta^{(w)}, \beta^{(m)})^\top$ is the solution to the minimization problem

$$\min_{\hat{c}, \hat{\beta}^{(d)}, \hat{\beta}^{(w)}, \hat{\beta}^{(m)}} \sum_{t=23}^{T} \Big(RV_t - \hat{c} - \hat{\beta}^{(d)} RV_{t-1}^{(d)} - \hat{\beta}^{(w)} RV_{t-1}^{(w)} - \hat{\beta}^{(m)} RV_{t-1}^{(m)} \Big)^2 . \tag{3.23}$$

If the errors ω_t of the model (3.22) are homoscedastic, independent, and normally distributed, then the OLS estimator is asymptotically consistent.

The aim of this paper is to forecast the volatility of the DAX. For this purpose, we have so far considered the classical econometric models, which are based either on the observation of empirical facts, such as stylized facts, or on economic theory. All these models are used going forward as benchmark models in comparison to methods from machine learning for the evaluation of the volatility forecast.

4. Machine Learning Algorithms

When it comes to forecasting volatility, it seems obvious to use methods from machine learning since they were designed specifically for prediction tasks. This is in line with the argumentation of Lommers et al. (2021) and Athey and Imbens (2019), who describe that the research paradigms of finance and machine learning are different, but also that forecasting is a fundamental part of financial econometrics where machine learning approaches can be beneficial. However, to apply machine learning techniques in the context of financial market time series, additional issues must be considered.
Israel et al. (2020) note that ML methods have been successfully used predominantly in areas where large amounts of data are available. In contrast, finance is not a big data environment, where often only a single time series is available, which has a low signal-to-noise ratio, is characterized by large dynamics, and may be non-stationary.
A possible solution for these problems is high frequency data, which offer several advantages and, therefore, are used in this thesis. First, high frequency data allow several observations per day, thus increasing the amount of available data. Second, high frequency data makes it is possible to determine more precise and informative quantities, such as realized variance, which also have a higher signal-to-noise ratio[1] due to the higher measurement frequency. As Israel et al. (2020) state, financial market returns exhibit a signal-to-noise ratio close to zero, which also infers their unpredictability. Specifically, volatility measured by realized variance determined from high frequency data has a ratio of 0.5329, whereas daily squared returns in the available data has a signal-to-noise ratio of 0.3554. This confirms the argumentation that high frequency data can at least partially solve the problem of low signal-to-noise ratio and shows that volatility approximated by realized variance is more suitable than squared returns for forecasting with machine learning. The problem of large dynamics and the possibility of non-stationarity of the series is present not only for machine learning approaches, but also in classical models. For example, GARCH models have problems with structural breaks in the data and often their parameter values show that the processes are close to non-stationarity.

[1]The signal-to-noise ratio is here calculated approximately as $SNR = \frac{\mu}{\sigma}$.

However, these problems have recently become more present in the machine learning literature and we discuss them in more detail in the following chapter.
Another important aspect is that time series have a different data structure than data used for machine learning. This particularly concerns the time-ordered structure of financial market time series and their associated dependencies. The special structure of time series makes it necessary to embed them in other spaces, like the Euclidean space, in order to apply them to ML methods. However, the temporal structure of the data also has implications for evaluation procedures, such as cross-validation or bootstrap procedures, which must be adapted for time series. We consider these problems and possible solutions in more detail in the following chapters.
Further reasons why ML methods may be suitable for forecasting return volatilities result from the stylized facts. We see from Figure 2.1 that returns exhibit heavy tails. When modeling volatility with classical econometric models such as the GARCH model, a priori a distribution and functional form must be specified that generates these data and accounts as accurately as possible for the property of heavy tails. While the distributional assumption and the specification of the relationship's form allow for inference, it can also lead to further problems if the assumptions do not adequately reflect reality. ML methods, in contrast, do not a priori assume a distribution and a functional form, but try to approximate a function based on the data which fits the data best. While this often prevents inference, it has the advantage that no unrealistic distribution assumptions have to be made and a non-linear modeling is possible.
All the described problems and possible solutions are the main reasons why we use machine learning methods to forecast the volatility of financial market returns in this paper. Another reason is that procedures from ML have not yet arrived in the econometric toolbox. This may be due to various reasons, but the advantages of these methods for specific areas of econometrics should be taken up and their potential exploited, which this work aims to illustrate. Now the question remains, which ML algorithms can be used to forecast volatility? Since we have 5.698 observations in our data, this rules out methods from Deep Learning since these require far more data for an efficient application. The choice of models is essentially based on the studies evaluated. Studies in ML research by Christensen et al. (2021), Masini et al. (2021), and De Stefani et al. (2017) show that, in addition to neural networks, models such as trees, random forests, and support vector machines may also be suitable for forecasting return volatilities. Before we consider these ML models and the necessary adaptations for their application in the context of time series, we first discuss the theory of machine learning for time series.

From our point of view, this is necessary because none of the prior research deals with this topic and it seems to be taken for granted that ML methods are applicable for time series. Thereby, the different data structure and learnability guarantees will be in the focus of the discussion.

4.1. Basic Theory of Machine Learning for Time Series

The Data Structure

The aim of this paper is to compare volatility forecasts with econometric methods and machine learning techniques. In the context of machine learning theory, this is a supervised learning problem. This describes, according to Shalev-Shwartz and Ben-David (2014), that we guide the machine learning models to learn with the help of data where the variable of interest is known. The data is split into two sets: first, training data to learn from, and, second, unseen test data to check how accurately the machine learned. The goal is to make predictions of a variable of interest y using various features x, taking into account the learned patterns from the data.
Following Shalev-Shwartz and Ben-David (2014), we can formalize this concept by assuming that we have an input pair $(\mathbf{x}_i, y_i)$ which is independently and identically drawn from an unknown distribution $\mathbb{P}$. Thereby $\mathcal{X}^d$ is the domain set or instance space and the points of the domain set $\mathbf{x}_i \in \mathcal{X}^d$ are usually vectors of covariates, which are also called features or instances. The target variable $y_i \in \mathcal{Y}$ is also called a label and $\mathcal{Y}$ is correspondingly called a label set. Thus, we can specify the total training data as $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\} \subseteq \mathcal{X}^d \times \mathcal{Y}$. The goal now is to learn a function $f(\mathbf{x}_i)$ such that for new data $f(\mathbf{x}_i) \approx y_i$ holds with high probability.
From the supervised learning setup, some differences in the structure and givenness of a time series can now be seen. First, it becomes apparent that the data structure is different from that of a time series. A time series is a time ordered sequence of the same observation $\{r_t\}_{t\in\mathbb{T}} = \{r_1, \dots, r_T\}$ which usually has autocorrelation and cannot be directly represented as a pair of observations as described. An obvious way to put time series into a supervised learning data format is, for example, if we think of predicting the value of r_{t+1} and assuming that its last three observations $\{r_t, r_{t-1}, r_{t-2}\}$ are important. Thus, we can write $y = r_{t+1}$ and $\mathbf{x} = (r_t, r_{t-1}, r_{t-2})^\top$ in the previous notation. Following this logic we use $\mathbf{x} = (r_{t+1}, r_t, r_{t-1})^\top$ to predict $y = r_{t+2}$. This rationale can be continued for the whole observed time series. Thus, we get a vector of labels with $\mathbf{y} \in \mathbb{R}^{T-n-1}$ and

a matrix of covariates $\mathbf{X} \in \mathbb{R}^{(T-n-1)\times n}$ and can represent them as

$$\mathbf{y} = \begin{pmatrix} r_{n+1} \\ r_{n+2} \\ \vdots \\ r_{T-1} \\ r_T \end{pmatrix} \qquad \mathbf{X} = \begin{pmatrix} r_n & r_{n-1} & \dots & r_1 \\ r_{n+1} & r_n & \dots & r_2 \\ \vdots & \vdots & \vdots & \vdots \\ r_{T-2} & r_{T-3} & \dots & r_{T-n-2} \\ r_{T-1} & r_{T-2} & \dots & r_{T-n-1} \end{pmatrix}. \tag{4.1}$$

With this idea, it is possible to put the time series into a data format that allows it to be used for supervised machine learning applications. However, if we look more closely at representation (4.1), we notice that the series $\{r_1, \dots, r_T\}$ is divided into $T - n - 1$ series, each of which is represented by n past values from the original T observations. This intuitive structuring of the data can be theoretically justified with Takens (1981) *embedding theorem*. This theorem is mathematically very demanding and extends beyond the scope of this paper, so we do not present and prove it. Instead, we describe and discuss its main implications, following the explanations of Bontempi et al. (2012) and Jemwa (2003).
Takens' embedding theorem forms the basis of nonlinear time series from the perspective of *dynamic system theory*. The dynamic system theory offers an alternative view of a time series. Instead of assuming that a time series is a realization of a stochastic process, one assumes that the series and its randomness is generated by a non-linear deterministic system. The resulting behavior is *deterministic chaos*. The essential statement of Takens' theorem is, that given an observed time series r_t, it is possible to reconstruct an equivalent[2] state space of the dynamics that generates the time series. This implies that, for a deterministic system, time series r_t can be used to reconstruct the state of the system at a given time. Therefore, in this approach, a time series is viewed as an observation of a dynamic system whose states $s(t)$ vary over time within state space Γ, according to the relationship

$$s(t) = \mathcal{H}^t(s(t_0)) \tag{4.2}$$

with $\mathcal{H} : \Gamma \to \Gamma$ as a function which describes the dynamics. The relationship between the observed noise-free time series and the dynamic system that generates the series

[2]The equivalence here is meant as diffeomorphic equivalent, which means that there exists a bijective continuously differentiable function ϕ whose inverse is also differentiable.

depends on the measurement and can be stated as

$$r_t = \mathcal{G}(s(t)). \tag{4.3}$$

where $\mathcal{G} : \Gamma \to \mathbb{R}^D$ is a measurement function[3]. Since functions $\mathcal{H}$ and $\mathcal{G}$ are usually unknown, one cannot in general reconstruct the original states of the system.[4] However, it is possible under certain conditions to reconstruct an equivalent state space if all available information is contained in the observed time series r_t. This statement is Takens (1981) *embedding theorem* and provides the formal basis for embedding a time series into a lower dimensional Euclidean space. The embedding theorem implies that a mapping $\phi : \Gamma \to \mathbb{R}^n$ exists, which describes the relation between a finite time window of the time series according to

$$\phi(s(t)) = \{\mathcal{G}(\mathcal{H}^{-d}(s(t))), \dots, \mathcal{G}(\mathcal{H}^{-d-n+1}(s(t)))\} = \{r_{t-d}, \dots, r_{t-d-n-1}\} \tag{4.4}$$

and the state of the dynamic system that creates the series. Thereby $\{r_{t-d}, \dots, r_{t-d-n-1}\}$ is an *embedding vector* or *Takens' vector*, d is the lag time, and n is the number of past observations that are considered. Takens (1981) showns that if ϕ is an embedding[5] this implies a smooth function $f : \mathbb{R}^n \to \mathbb{R}$ in the reconstructed state space such that

$$r_t = f(r_{t-d}, r_{t-d-1}, \dots r_{t-d-n+1}) \tag{4.5}$$

holds. This representation implies that reconstructed states can be used for any purpose of time series analysis such as estimating f. This approach to reconstructing the state space implies and justifies the representation of the observed time series as in (4.1). It should be noted, however, that we have implied $d = 0$ in our intuitive derivation of the data structure (4.1), since the vector $\mathbf{y}$ is in Takens' embedding theorem actually the first column of the matrix $\mathbf{X}$. To better understand this abstractly formulated approach, a visual elaboration describing the steps can be found in the appendix of this thesis.

We now consider two additional aspects of this approach. First, the previous derivation assumes that f is a deterministic function that perfectly describes the time series. However, since we usually have to estimate f and cannot assume that f is known, it is

[3]Here we deal with univariate time series so we assume for the measurement function $D = 1$.

[4]Since the two functions $\mathcal{H}$ and $\mathcal{G}$ are unknown, a diffeomorphic equivalence becomes evident here. If one would not demand this it would be senseless to expect a reconstruction.

[5]Bontempi et al. (2012) describes an embedding as "a smooth one-to-one differential mapping with a smooth inverse".

reasonable to assume that a noise term is also present and that the relation is given as

$$r_t = f(r_{t-d}, r_{t-d-1}, \dots r_{t-d-n+1}) + u_t. \tag{4.6}$$

Equation (4.6) is called *nonlinear autoregressive (NAR)* model. The second aspect to consider is the choice of d and n. There are different approaches to determining d and n, like order selection based on the autocorrelation function or information criteria, which we do not discuss in detail in this paper but refer e.g. to Casdagli et al. (1991). Instead, we briefly consider the intuition behind the selection based on the autocorrelation function and thus also describe our approach in this work. The logic in this approach of how n is chosen can be seen from Figure 2.2 and the representation of the autoregressive data structure in (4.1). The idea here is to choose n such that significant autocorrelation structures are included in the embedding vectors. In other words, we choose n such that r_t and r_{t-n} are nearly uncorrelated with each other. By doing so, we can use the autocorrelation function and obtain at least $n = 30$ for daily squared returns r_t^2 and realized variance RV_t.[6] A value of $n = 30$ is not surprising in modeling and forecasting volatility, as this just reflects the high persistence of volatility. Even though this approach is intuitive and described in the literature, there are potential problems with it. As Casdagli et al. (1991) note, the autocorrelation function only measures linear dependencies and may therefore be generally inappropriate. Nevertheless, this approach will be used here and we refer to Casdagli et al. (1991) for a detailed discussion of potential problems with it.

Statistical Learning Theory for Time Series

After discussing the different data structures between time series and supervised learning methods, we now turn to the theoretical basis of machine learning. We focus on upper bounding the probability that learning will lead to an error. The area particularly relevant to such a topic is *statistical learning theory.* The key assumption in standard machine learning is that the data are independent and identically distributed $\mathcal{S} \sim \mathbb{P}^n$, which is not satisfied for time series because of their temporal dependencies and possible non-stationary behavior. Because of these problems, it is therefore questionable whether machine learning methods are suitable for forecasting time series and what learning guarantees exist. To answer these questions, we first describe the most important concepts of

[6]We only choose n with this approach for squared returns. To obtain features with realized variance, we use the same regressors as in (3.20) from the HAR-RV model.

standard statistical learning theory and present recent research results in the context of time series analysis. In doing so, we do not consider the concepts in mathematical detail, but provide a verbal explanation. Nevertheless, to introduce the theory we need some notation, and essentially follow Mohri et al. (2018) and Shalev-Shwartz and Ben-David (2014) for such.

We already formulated the general goal of machine learning at the beginning of this chapter and that remains the same even in the context of time series. We are looking for a function $f : \mathcal{X} \to \mathcal{Y}$ such that for new data, $f(\mathbf{x}_i) \approx y_i$ holds with high probability. Here, we refer to f as the predictor or hypothesis. Set $\mathcal{F}$ denotes the set of all such functions f and is called a hypothesis space or hypothesis class.[7] To find out which machine learning algorithm is best suited for a specific learning problem, an evaluation criterion is needed to compare the performance of the different approaches. For this, we define a measure of failure, also called the loss function, as $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$. The loss function can be written as $\ell(f(\mathbf{x}_i), y_i)$ since it measures the output of chosen hypothesis f and the observed true value. It indicates the costs of failures.[8] Based on the loss function, the *risk function* or the *generalization error* is defined as the expectation of the loss function

$$R_{\mathbb{P}}(f) = \mathbb{E}_{\mathbb{P}}[\ell(f(\mathbf{x}), y)]. \tag{4.7}$$

This quantity gives expected loss if we use f to predict y_i from $\mathbf{x}_i$ given a new observation. The integral of expected value is with respect to distribution $\mathbb{P}$ of a new test point $(\mathbf{x}_i, y_i)$, which is independent of f . Risk is a key measure in machine learning, as it measures the average prediction quality of predictor f in terms of the loss function. Accordingly, optimal risk can theoretically be found by the minimization problem $f^* = \arg\min_{f \in \mathcal{F}} R_{\mathbb{P}}(f)$. In practice, however, true distribution $\mathbb{P}$ is often unknown and, thus, $R_{\mathbb{P}}(f)$ is also unknown and must be estimated. This is done with the mean over the training data as follows

$$\hat{R}_{\mathbb{P}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\mathbf{x}_i, y_i)) \tag{4.8}$$

[7]For example, $\mathcal{F} = \{f(\mathbf{x}_i) : \mathbf{x}_i^\top \boldsymbol{\beta}, \forall \boldsymbol{\beta} \in \mathbb{R}^p\}$ is the hypothesis space of linear functions.

[8]Some of the best known loss functions are the quadratic loss function $\ell(f(\mathbf{x}_i), y_i) = (f(\mathbf{x}_i) - y_i)^2$ or the absolute loss function $\ell(f(\mathbf{x}_i), y_i) = |f(\mathbf{x}_i) - y_i|$.

and is called *empirical risk* or *training error*. Accordingly, *empirical risk minimization* can be performed with $\hat{f}^* = \arg\min_{f \in \mathcal{F}} \hat{R}_{\mathbb{P}}(f)$. The approach of empirical risk minimization is intuitive and, under the assumptions of independent and identically distributed data and a bounded loss function ℓ, it is easy to show that empirical risk is an unbiased estimator of true risk and, based on the law of large numbers, it is also consistent and $\hat{R}_{\mathbb{P}}(f) \xrightarrow{\mathbb{P}} R_{\mathbb{P}}(f)$ holds.
Although the preceding argument partially justifies the use of empirical risk minimization, there are practical problems with it. According to McDonald et al. (2011), these problems arise from minimizing empirical risk, which relates only to the training data, and possibly from ignoring the fact that we are looking for a predictor that has a small true risk. From a practical point of view, this means that we only have a finite amount of data and, therefore, there may be large differences between true risk and empirical risk, especially when $\mathcal{F}$ is large and n is small. Thus, overfitting the data and poor out-of-sample prediction performance can occur with this approach.
There are two strategies to avoid these problems, according to McDonald et al. (2011). First, one can restrict hypothesis space $\mathcal{F}$, and second, one can adapt the optimization problem by penalizing for high model complexity. Nevertheless, the true distribution of the data remains unknown and, therefore, true risk cannot be determined. To circumvent this problem, upper bounds for the risk are considered, which hold with high probability. This resulting theory is called *probably approximately correct (PAC)*. According to McDonald et al. (2011), the core of the PAC model is that for a hypothesis f from a finite hypothesis space $\mathcal{F}$ with probability of at least $1 - \delta$,

$$R_{\mathbb{P}}(f) \leq \hat{R}_{\mathbb{P}}(f) + \Theta(C(\mathcal{F}), \delta, n) \tag{4.9}$$

holds, where $C(\mathcal{F})$ is a measure of model complexity, n is the number of training data, δ is the confidence parameter, and Θ represents a function of all these parameters.[9]
There are a few special features to note with the PAC model. First, the PAC model is a distribution-free approach. Second, the training data and the test data are used to determine the error. Third, the PAC model deals with the learnability issue for a class of learners $\mathcal{F}$ and not a specific learner. Fourth, the model considers only finite hypothesis sets $\mathcal{F}$.
There are also several other learning models that consider infinite hypothesis classes or allow for algorithm-specific generalization bounds. Probably the best-known exten-

[9]For a proof of statement (4.9) we refer to Mohri et al. (2018).

sions which allow infinite hypothesis sets are the Rademacher complexity and the VC (Vapnik-Chervonenkis) dimension. According to Mohri et al. (2018), the general idea of Rademacher complexity and VC dimension is that the infinite case is reduced to a finite case and then proceeds as in the PAC model. The reduction of complexity of hypothesis space $\mathcal{F}$ is done differently between the two approaches. Rademacher complexity is used by Rademacher learning guarantees to measure the complexity of the family of functions and gauge to what degree it can fit random noise. This uses the correlation between random noise in the data $\mathcal{S}$ and the family of functions. The more complex the considered function classes are, the more they correlate with random noise in the data. In contrast, generalization bounds with VC dimension use the growth function for a hypothesis set $\mathcal{F}$. This gives the maximum number of possibilities with which n points can be uniquely classified by hypotheses from $\mathcal{F}$ and, thus, can be interpreted as a kind of size of the hypothesis space. Thereby, we say that hypothesis set $\mathcal{F}$ shatters a set A with n points if it realizes all possible ways of labeling the n points of A, that is, when the growth function equals $2^{|A|} = 2^n$. Based on these considerations, the VC dimension of hypothesis space $\mathcal{F}$ is defined as the maximum size or cardinality of the set it can shatter.[10]
Each of the PAC, Rademacher complexity, or VC dimension approaches allows us to determine learning guarantees in terms of probability bounds, essentially corresponding to the one stated in (4.9). They fundamentally differ only in the last term and can be more complex. However, these bounds do not consider specific algorithms but apply to arbitrary algorithms using $\mathcal{F}$ as a hypothesis space. In order to consider the respective algorithms or whole classes of algorithms with similar properties and to determine generalization bounds based on them, the algorithmic stability is used and the resulting bounds are called stability bounds. To define the stability of an algorithm, it is assumed that the loss function ℓ is bounded and two training data sets $\mathcal{S}$ and $\mathcal{S}'$ are used which differ only with respect to one point. A learning algorithm is said to be uniformly β-stable if the algorithm is trained on both data sets and the respective losses of the corresponding hypotheses do not differ by more than β. The resulting generalization bounds are also similar to (4.9), differing only in the last term and now additionally depending on β. To determine probability bounds for the generalization error in all the described approaches and to measure how quickly empirical risk converges to true risk, concentration inequalities are used. The most frequently used inequalities for this

[10]For example, if we consider $\mathcal{F} = \{f(x) = \mathbb{1}(x \leq \theta), \theta \in \mathbb{R}\}$, then this hypothesis set can shatter two points, but not three.

purpose are the Hoeffding and McDiarmid inequalities.[11] All these inequalities assume that the data are independent and identically distributed, so that the results from the standard theory of statistical learning are not directly transferable to time series.
However, time series are playing an increasingly important role in theory and applications and research in this area has made some progress in recent years. Most generalization bounds for time series assume that the series is stationary and mixing. We have already defined stationarity, but mixing is a concept that deals with how to relax the independence of two events $\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$. Intuitively, a stochastic process is mixing if events in the past and future become independent as the time interval between events increases. Figure 4.1 shows the intuition of mixing.

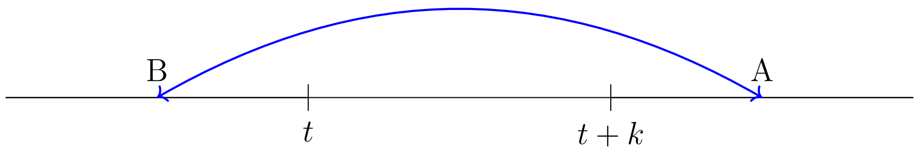


Figure 4.1.: Illustration of the mixing property following Kuznetsov and Mohri (2018)

The dependence of past events B on future event A decreases for increasing k, so that for a sufficiently large k the events are independent. In other words, if the time distance between past and future events becomes large enough, the events approach asymptotic independence and, therefore, mixing can be interpreted as a measure of weak dependence. There is to note that mixing of stochastic processes and the associated asymptotic independence relate to an appropriate metric. It is also interesting to note that IID processes by definition satisfy the conditions of mixing and, therefore, suggests that many statements about learnability of IID processes are also applicable to mixing processes if they achieve asymptotic independence quickly enough. There are many ways to define mixing, with the β-mixing being of particular importance for statistical learning theory for time series. A detailed overview of different types of mixing and their definitions can be found in Bradley (2005).
There are several definitions of β-mixing and the definition below follows that of Mohri and Rostamizadeh (2007). They define β-mixing for a stationary sequence $\{R_t\}$ as

$$\beta(k) = \sup_t \mathop{\mathbb{E}}_{B \in \sigma_{-\infty}^{t}} \left[\sup_{A \in \sigma_{t+k}^{\infty}} \left| \mathbb{P}(A|B) - \mathbb{P}(A) \right| \right] \overset{k \to \infty}{\longrightarrow} 0 \tag{4.10}$$

[11]A detailed overview of the main concentration inequalities can be found in the appendix of Mohri et al. (2018).

where σ_i^j is the σ-algebra generated by random variable R_k for $i \leq k \leq j$ with $i, j \in \mathbb{Z} \cup \{-\infty, \infty\}$.

The definition (4.10) of β-mixing is very general and technical. But it is easy to understand when considering a relaxation of independence. While mathematically inexact, the concept is intuitively appealing. For a β-mixing process, it holds that $\beta(k) \overset{k\to\infty}{\to} 0$ and, since we can write $\mathbb{P}(A|B) - \mathbb{P}(A) = \frac{\mathbb{P}(A\cap B)-\mathbb{P}(A)\mathbb{P}(B)}{\mathbb{P}(B)}$, this implies $\mathbb{P}(A \cap B) \overset{k\to\infty}{\to} \mathbb{P}(A)\mathbb{P}(B)$. Thus, a process is β-mixing if for an increasing k, the joint distribution of events is equal to the product of the individual distributions of events.[12]

In the literature, many well-known time series processes have been shown to be mixing processes. Fryzlewicz and Rao (2011) determine lower and upper limits for mixing parameters of ARCH processes and non-stationary ARCH processes. Carrasco and Chen (2002) and Lindner (2009) describe that GARCH processes are β-mixing. Varied results are also known in the context of machine learning. Karandikar and Vidyasagar (2009) show that under additional assumptions, an algorithm whose predictors risk can be upper bounded in the case of IID data, it is also possible to upper bound the risk of the same predictor when the series at hand is stationary and β-mixing. Based on these results, the authors determine bounds on the generalization error not only in the IID case, but also for β-mixing. Rademacher complexity bounds for dependent, stationary, and β-mixing processes are determined by Mohri and Rostamizadeh (2008). They show its applicability to classification tasks and note that similar generalization bounds can be derived for regression applications. These results were extended by Kuznetsov and Mohri (2014), who show that Rademacher complexity bounds also hold for nonstationary β-mixing processes. For this, they use discrepancy, which describes the difference between path-dependent risks of two probability distributions. In other words, discrepancy can be interpreted as the difference between two probability distributions, which serves as a measure for non-stationarity while accounting for the hypothesis set and the loss function.[13]

Further generalization bounds based on VC dimension and algorithmic stability for stationary mixing data have also been developed. Yu (1994) generalizes the theory of VC dimension for stationary mixing processes, from which bounds for the generalization error can be derived. Mohri and Rostamizadeh (2007) derive stability bounds that directly

[12]See McDonald et al. (2011), who additionally note that for stationary sequences β-mixing can also be written in terms of the total variation norm $\beta(k) = \|\mathbb{P}_t \times \mathbb{P}_{t+k} - \mathbb{P}_{t\otimes t+k}\|_{TV}$, where $\mathbb{P}_t$, $\mathbb{P}_{t+k}$, and $\mathbb{P}_{t\otimes t+k}$ are the restrictions of $\mathbb{P}$ to $\sigma_{-\infty}^t$, σ_{t+k}^∞ and $(\sigma_{-\infty}^t, \sigma_{t+k}^\infty)$ respectively.

[13]For a detailed definition and explanation of the discrepancy approach see Mohri and Muñoz Medina (2012).

generalize the IID stability bounds and demonstrate their applicability to Support Vector Regression and Kernel Ridge Regression. In recent years, many other results have been obtained on the learnability of dependent data, with almost all work assuming the properties of mixing and stationary. Particularly noteworthy is the work of Kuznetsov and Mohri (2015), who note that these assumptions are often not met. Thus, the question arises whether time series that are not stationary and have no mixing properties can be learned at all. To answer this question, new tools for the analysis are needed. Kuznetsov and Mohri (2015) propose using generalization bounds with discrepancy to develop new algorithms. The idea is to optimize the upper discrepancy bounds directly. The resulting optimization problem can then be solved efficiently, as the authors show with an example of kernel-based regression, since a convex optimization problem results. In addition to the development of new algorithms, this theory allows analysis of known algorithms in more detail.
Overall, the literature provides a theoretical basis for using machine learning techniques in a time series context. From the literature described, machine learning algorithms seem to be particularly suitable for volatility forecasting because GARCH processes are β-mixing and there are learning guarantees for stationary and non-stationary β-mixing processes. Altogether, this allows for the interpretation that GARCH processes can be learned regardless of whether the process is stationary or not and provides a theoretical justification for the use of machine learning methods for volatility forecasting.

4.2. Trees

We now introduce and describe the ML models used in the paper. As discussed earlier, the choice of methods considered is justified by the models' performance in the literature and the small amount of data. Essentially, we use trees, random forests, and support vector regression. Our remarks describing the methods essentially follow the books by Hastie et al. (2009) and James et al. (2013).

Decision Trees

Similar to the well-known probability tree, a decision tree is a directed graph with a tree structure running from top to bottom. The tree consists of a *root node* at the top that contains all features or covariates and divides downwards into further *nodes* up to *leaves* that mark the end of the tree. Each node to be split is called a *parent node* and the resulting nodes are called *child nodes*. Based on a single covariate, parent nodes are split into two child nodes using a binary decision rule. The division of parent nodes is chosen such that both child nodes are as heterogeneous as possible, but with the greatest possible homogeneity within the nodes. Then the two child nodes become parent nodes themselves and subdivide in the same way. This procedure is carried out until a stop rule is applied. A simple decision tree with two covariates is shown in Figure 4.2.

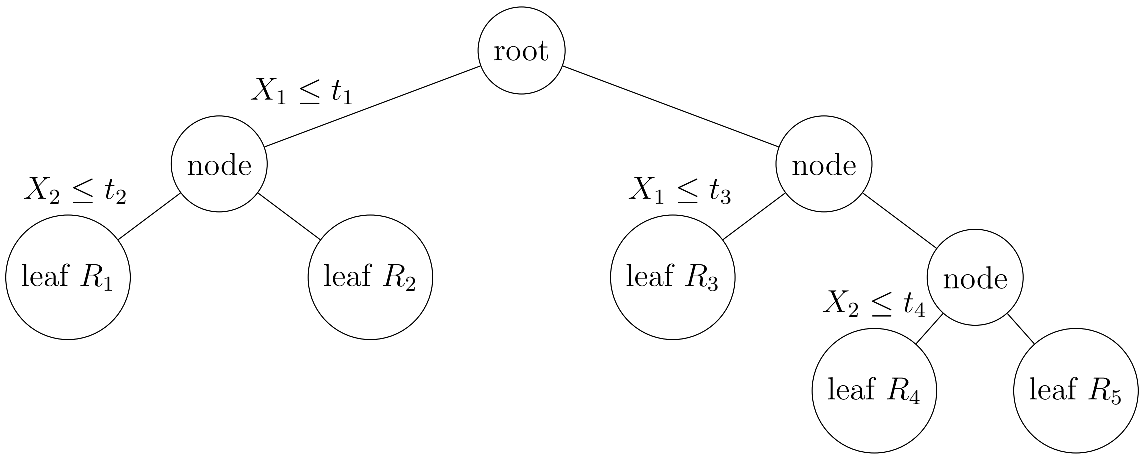


Figure 4.2.: Schematic representation of a decision tree based on Hastie et al. (2009) with two covariates used for binary decisions.

It is possible to further distinguish decision trees. We call it a classification tree when the dependent variable is categorical and a regression tree when the dependent variable is metrically scaled. There are various algorithms that automatically generate decision trees, and here we focus on the classification and regression trees, *CART* algorithm.

Classification and Regression Trees

We assume that we have training data $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ with feature vector $\mathbf{x}_i = (x_{i1}, \dots, x_{id})^\top$. The CART algorithm decomposes the covariate space into d-dimensional hyperrectangles. In other words, the covariate space is divided into disjoint subsets. Thus, CART attempts to approximate the underlying true function using piecewise constant functions. Figure 4.2 below shows the disjoint decomposition of the two-dimensional covariate space for the decision tree from Figure 4.2. In doing so, the algorithm must decide by which covariate to split and determines the split points for it. To better understand this idea, we essentially follow the mathematical description of Hastie et al. (2009).

For this, we suppose the covariate space is decomposed into M d-dimensional non-overlapping hyperrectangles $R_1, R_2, \dots, R_M$ and the variable of interest y_i is modeled in

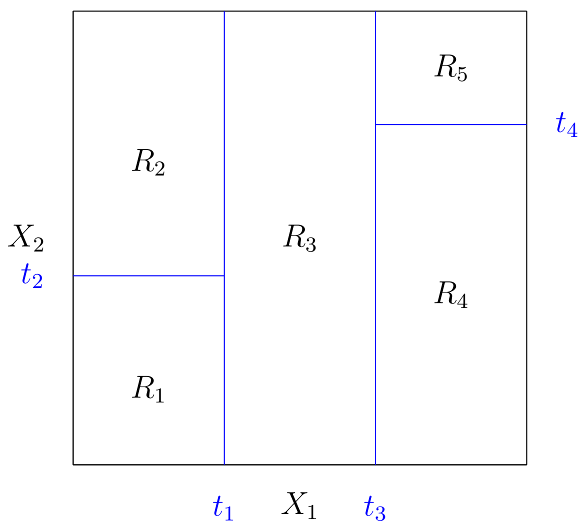


Figure 4.3.: Decomposition of a two-dimensional covariate space into rectangles. The presented decomposition corresponds to the representation of the decision tree from Figure 4.2 (Hastie et al., 2009)

each of the regions R_m as a constant value c_m for $m = 1, \dots, M$ with

$$f(\mathbf{x}_i) = \sum_{m=1}^{M} c_m \mathbb{1}(\mathbf{x}_i \in R_m) \tag{4.11}$$

where $\mathbb{1}()$ is the indicator function and c_m is a constant function of the respective region R_m. To find the best possible model fit for a regression tree T, the sum of squared distances

$$\begin{aligned} SSE(T) &= \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 \\ &= \sum_{m=1}^{M} \sum_{i=1}^{n} (y_i - c_m \mathbb{1}(\mathbf{x}_i \in R_m))^2. \end{aligned} \tag{4.12}$$

is used. The goal of the algorithm is now to find the optimal decomposition of the covariate space in M d-dimensional hyperrectangles and function c_m such that $SSE(T)$ is minimized. For any decomposition of the covariate space this is achieved by choosing

$$\hat{c}_m = \frac{1}{n_{R_m}} \sum_{\mathbf{x}_i \in R_m} y_i \tag{4.13}$$

where n_{R_m} corresponds to the number of observations in region R_m. In other words, this shows that using the mean over all y_i in the respective region R_m minimizes the sum of squared distances. According to James et al. (2013), to find the optimal decomposition of the covariate space into M hyperrectangles such that equation (4.12) is minimized, the algorithm starts at root node R with all observations and chooses the best split covariate $\mathbf{x}_j$ with corresponding split point s. To split all observations with respect to covariates $\mathbf{x}_j$, a binary split is performed, where $R_L(j,s) = \{\mathbf{X} \mid \mathbf{x}_j \leq s\}$ and $R_R(j,s) = \{\mathbf{X} \mid \mathbf{x}_j > s\}$ is chosen so that the largest possible reduction of the sum of squared distances is achieved. To determine the reduction of the sum of squared distances by binary partitioning, the difference of the sum of squared distances in the root node before and after the split must be determined. Therefore, all covariates $X_1, \dots, X_d$ and all possible split points s for the covariates are considered and these are chosen so that the resulting tree has the smallest possible $SSE(T)$. The best split variable $\mathbf{x}_j$ and the corresponding split point s are determined from the minimization problem

$$\min_{j,s}\left(\min_{c_L}\sum_{\mathbf{x}_j \in R_L(j,s)}(y_i - c_L)^2 + \min_{c_R}\sum_{\mathbf{x}_j \in R_R(j,s)}(y_i - c_R)^2\right). \tag{4.14}$$

For an arbitrary choice of j and s, the internal minimization problem is solved by

$$\begin{aligned}\hat{c}_L &= \frac{1}{n_{R_L}}\sum_{\mathbf{x}_i \in R_L(j,s)} y_i \\ \hat{c}_R &= \frac{1}{n_{R_R}}\sum_{\mathbf{x}_i \in R_R(j,s)} y_i.\end{aligned} \tag{4.15}$$

After finding optimal split variable $\mathbf{x}_j$ and corresponding split variable s, the data is split into two disjoint subsets and the described splitting steps are repeated for the two new regions. This procedure could then be carried out until there is only one observation in each leaf, which would result in overfitting to the data. However, it is also conceivable that a tree that is too small, with little depth, would fail to detect significant structures in the data. Thus, it can be seen that tree depth determines the complexity of the tree and must be taken as a tuning parameter that should be determined from the data.
To counter these problems, there are several approaches. Particularly well known is the cost-complexity pruning. In this approach, splitting of the covariate space is stopped as soon as the child nodes contain a fixed number of observations. This initially leads to a very large and deep tree which is then pruned such that a specified objective function

is minimized depending on tree size and goodness of fit of the tree. For a detailed discussion of this subject, we refer the reader to the book by Hastie et al. (2009). In general, trees provide a simple structure that can be easily interpreted and allow easy handling of all covariate types. If the tree is large enough, good performance can be achieved. However, it is problematic that they tend to overfit, since they are unstable and small changes in the data can lead to completely different trees. The main reason for the instability and the resulting high variance is the hierarchical structure of the method. This structure leads to the fact that an error in the first decompositions has an effect until the last decomposition. Overall, this describes the well-known bias-variance tradeoff that applies to trees, as they have a low bias and high variance.

4.3. Random Forest

To eliminate the problems of instability and the resulting large variance of trees, several approaches have been proposed in the literature. According to James et al. (2013), the best-known proposed solutions are based on reducing the high variance for a procedure with low bias. The basic idea is that through an ensemble of trees, the bias remains the same while the variance is reduced. The intuition behind these approaches is easy to understand from a statistical point of view, since for independently and identically distributed random variables $Z_1, \dots, Z_n$ with variance σ^2, the arithmetic mean $\bar{Z}$ has a lower variance $\frac{\sigma^2}{n}$ compared to that of each individual variable. This demonstrates the idea that averaging a set of observations leads to a reduction in variance. Based on this intuition, Breiman (1996) proposes *Bootstrap Aggregation (Bagging)*. In this method, B bootstrap samples are drawn with replacement from existing training data and one tree per bootstrap sample is fitted. To obtain predictions, individual trees are aggregated and the mean of all predictions is used. One problem with this method is that while the prediction function of B trees is from the same distribution, they are not necessarily independent or uncorrelated because they are generated from very similar data. This correlation between the trees influences variance reduction, because the higher the correlation, the higher the variance of the ensemble's prediction function. To generate an ensemble of uncorrelated trees, Breiman (2001) proposes the *random forest* algorithm. The idea behind this approach is to reduce correlation between trees without increasing the variance of each tree too much. *Random feature selection* achieves this by randomly selecting m_{try} from the total d covariates and searching for the optimal split variable. This step leads to a decrease in correlation between individual trees, since the ensemble uses randomly generated and different trees. This intuitive description we want to analyze mathematically more exactly, following the books of Hastie et al. (2009) and James et al. (2013). In addition, an essential adjustment must be made to be able to use the Random Forest algorithm for time series data. This concerns the bootstrap method, which is not applicable in the classical form to time series since these exhibit temporal dependencies and a resampling with replacement would destroy this structure. To take this adaptation into account, we first introduce the algorithm and speak of bootstrap methods in general, and then explain the *Moving Block Bootstrap (MBB)* method, which is used in the present work.

Bootstrap Aggregation

To obtain an ensemble of trees, we assume the training data $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with feature vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})^\top$. From these data, we draw B bootstrap samples $\mathcal{S}_1^*, \ldots, \mathcal{S}_B^*$ with length n, where for all $b = 1, \ldots, B$ a CART T_b^* with prediction function $\hat{f}_b^*$ are created. These trees are grown without pruning and until a minimum node size n_{min} is reached. Finally, we can take the mean over all B prediction functions $\hat{f}_b^*(\mathbf{x})$ and get the bagged predictor

$$\hat{f}_{Bag}^B(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b^*(\mathbf{x}). \tag{4.16}$$

As described heuristically at the beginning of this chapter, the bagging procedure is particularly suitable for models with low bias and high variance. This is especially true for tree models, since they are not pruned for bagging and thus the predicted values deviate only slightly from the actual values, resulting in low bias. At the same time, not pruning trees lead to a possible overfitting which results in high variance. The bootstrap samples $\mathcal{S}_1^*, \ldots, \mathcal{S}_B^*$ are drawn from the original data $\mathcal{S}$ and therefore (should) follow the same distribution. Thus, trees T_b^* and their prediction functions $\hat{f}_b^*$ are also identically distributed, since these are generated from the B identically distributed bootstrap samples. From this fact we get

$$\begin{aligned}
\mathbb{E}\big[\hat{f}^B_{Bag}(\mathbf{x})\big] &= \mathbb{E}\left[\frac{1}{B}\sum_{b=1}^{B}\hat{f}^*_b(\mathbf{x})\right] \\
&= \frac{1}{B}\sum_{b=1}^{B}\mathbb{E}\big[\hat{f}^*_b(\mathbf{x})\big] \\
&= \mathbb{E}\big[\hat{f}^*_b(\mathbf{x})\big]
\end{aligned} \tag{4.17}$$

which means that the expected value of the mean of the B prediction functions $\hat{f}^{tree}_{Bag}$ is equal to the expected value of any prediction function from the ensemble. In other words, the estimator of the bagging method is unbiased. It should be noted that although individual trees and prediction functions are identically distributed, they need not be independent or uncorrelated, since they were generated from the same data. The variance of bagged predictor $\hat{f}^B_{Bag}$ can be calculated under the assumption of positive pairwise correlation ρ between prediction functions as

$$var(\hat{f}^B_{Bag}(\mathbf{x})) = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{4.18}$$

where σ^2 is the variance of the individual prediction functions. A detailed derivation or proof of the equation (4.18) can be found in the appendix of this paper. Note that this equation is only valid for positively correlated trees $\rho > 0$, otherwise it would be possible to get a negative variance expression. However, the interpretation of this equation is interesting. First, we see that the variance due to bagging for $\rho \neq 1$ and $B \neq 1$ is smaller than that of a single tree or prediction function which have variance σ^2. Moreover, for a sufficiently large bootstrap sample B, the second term is negligible, but the first term $\rho\sigma^2$ remains. Overall, this shows that correlation has an impact on variance reduction of the bagging procedure and reducing the correlation can further reduce variance of the bagged predictor. Reducing correlation to lower variance while maintaining accuracy is the goal of the random forest algorithm.

Random Forest Algorithm

The objective of the random forest algorithm is to reduce correlation ρ between individual trees or predictor functions in bagging to further reduce variance (4.18) of the bagged predictor. To achieve this goal, bagging is extended by *random feature selection.* This means that, unlike classical trees, not all d features are considered as potentially optimal split variables. Instead, $m_{try} \leq d$ features get randomly selected and the optimal split variable is searched for among them. This process is performed until the minimum number of observations n_{min} in a child node is reached and no more splitting of this node is performed. This approach seems a bit strange at first but leads to the desired decorrelation. The effect of random feature selection follows a simple logic. To understand this, we follow the example of James et al. (2013) and assume that there is one strong predictor among all covariates and that all other covariates have only a moderate predictive quality. Then, without random feature selection, bagging would generate trees, most of which would choose the strong predictor as the first split variable. As a result, all trees would have a similar structure and the predictions would be strongly correlated with each other, leading not to the desired variance reduction. By randomly selecting potential split variables, frequent selection of the strong predictor is prevented and correlation between forecasts is reduced, resulting in a reduction of variance. The random forest predictor differs from the bagging predictor only by the random feature selection and to distinguish between them, it is written as

$$\hat{f}^B_{RF}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^*_b(\mathbf{x}). \tag{4.19}$$

in the following.
If a random forest is used for prediction, the number of trees and thus the number of bootstrap samples B, as well as the number of random features m_{try} must be determined beforehand. A reference for the choice of these parameters is given by James et al. (2013). They state that an increasing number of trees B does not lead to overfitting to the data as they are generated from different bootstrap samples. Breiman (2001) notes that the generalization error of the random forest for an increasing number of trees B almost surely converges to a lower bound. This means that the number of trees can be chosen arbitrarily large without increasing generalization error. However, this should not be chosen too large, since computational time increases but no significant improvements of the error can be expected. The number of covariates m_{try} for each splitting of nodes is set in most implementations in R[14] to $m_{try} = \lfloor \sqrt{d} \rfloor$ or $m_{try} = \lfloor \frac{d}{3} \rfloor$ as default values. These values are based on the work of Breiman (2003), who claims that these values are close to the optimal values. To optimize them, the author suggests using half and double the default to make optimizations. Taking into account that in the present work time series data are used and that the covariates are highly correlated with each other,

[14] R Core Team (2022)

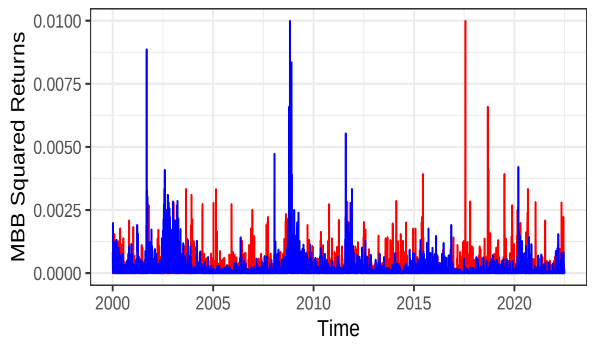
smaller values are preferable according to James et al. (2013). All in all, it seems to make sense to use the default setting and smaller values such as a half or one third of this value to perform optimizations.
We would like to briefly describe another property of the random forest. As described, each tree is generated from a bootstrap sample, using on average only about two thirds of the data.[15] The remaining data is called *out-of-bag* data and can be used to evaluate the prediction performance. For a detailed discussion of out-of-bag data, we refer to Hastie et al. (2009)

Bootstrap for Time Series

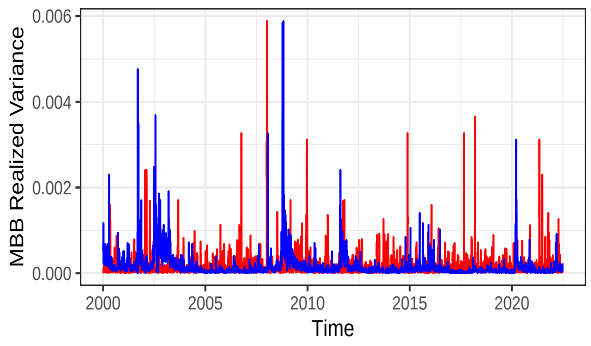
In the description of the random forest algorithm we mentioned that we use bootstrap to generate different trees, but the classic bootstrap procedure must be adapted for the application of the algorithm in the time series context. Suppose we have data $\mathbf{x} = (x_1, \dots, x_n)^\top$ which are independent and identically distributed. In classical bootstrapping[16], we draw n times with replacement from the data and get a bootstrap sample $\mathbf{x}^{1*} = (x_1^{1*}, \dots, x_n^{1*})^\top$. Repeating this process B times we obtain B bootstrap samples $\mathbf{x}^{1*}, \dots, \mathbf{x}^{B*}$ each of length n. This classical approach of the bootstrap method does not work for time series, because the data are assumed to be independent and the order of the data does not matter. Thus, the classical bootstrap would destroy the temporal structure of the series and would not be able to reflect dependencies.
This problem is not new and several extensions have been proposed in the literature to make bootstrap usable for dependent data. A distinction is made between model-based bootstrap methods like the *residual bootstrap* or *autoregressive-sieve bootstrap* and non-parametric bootstrap like the *block bootstrap* introduced by Carlstein (1986). In this paper, we use a method from the block bootstrap called *moving block bootstrap* which was proposed by Künsch (1989). A good overview of bootstrap procedures for dependent data is given by Bühlmann (2002) and Kreiss and Lahiri (2012), which serve as a basis for the following remarks.
The goal of the block bootstrap is to generate pseudodata that retain the dependence structure from the original stationary time series. To achieve this, entire blocks are resampled instead of individual observations. The idea is that for stationary time

[15]For n observations, the probability of an observation being selected for the bootstrap sample is equal to $\frac{1}{n}$ and for not being selected $1 - \frac{1}{n}$. Thus, the probability of not being in the sample is $(1 - \frac{1}{n})^n$ which approaches 0.6321 for $n \to \infty$. Thus, on average, $0.3678 = 36.78\%$ of the data corresponds to out-of-bag data.
[16]See Efron and Tibshirani (1994).

series, individual blocks that are far enough apart are approximately uncorrelated. Resampling is similar to the classical bootstrap, except that whole blocks are resampled, assuming that blocks of sufficiently large size contain the essential dependence structure. Depending on the block bootstrap procedure, the blocks may or may not overlap and the block length may be fixed or random. To make these ideas more tangible, suppose we have a time series $\{x_t\}_{t=1}^{T}$ of length T. In the moving block bootstrap, we divide the data into $N = T - l + 1$ overlapping blocks of length l, where we assume for simplicity that l divides T. Thus, individual blocks can be defined as $L_1 = \{x_1, \ldots, x_l\}, L_2 = \{x_2, \ldots, x_{l+1}\}, \ldots, L_N = \{x_{T-l+1}, \ldots, x_T\}$. From these blocks, $\frac{T}{l}$ blocks are now randomly sampled with replacement. Putting together these blocks yields a bootstrap sample. For example, let us consider the time series $x_t = \{x_1, x_2, \ldots, x_6\}$ of length $T = 6$ and set a block length of $l = 3$, then we obtain $N = T-l+1 = 6-3+1 = 4$ blocks $L_1 = \{x_1, x_2, x_3\}, L_2 = \{x_2, x_3, x_4\}, L_3 = \{x_3, x_4, x_5\}$, and $L_4 = \{x_4, x_5, x_6\}$. Since the original series has length $T = 6$, in this example $\frac{T}{l} = \frac{6}{3} = 2$ blocks must be drawn randomly with replacement where there are a total of $N^2 = 16$ possibilities to assemble the blocks into a bootstrap sample. The following figure shows the moving block bootstrap for squared returns as well as for the realized variance.



(a) Bootstrapped daily squared returns



(b) Bootstrapped realized 5-min variance

Figure 4.4.: Moving block bootstrap for the volatility proxies considered in this thesis. For the resampling we used block length $l = 5$ and only $B = 1$ replications because of the clarity and comparability. The original series are blue and the MBB series are red.

If we compare the bootstrap sample with the original series in Figure 4.4, we notice that the extreme deflections in the original series do not occur at the same times in the bootstrap sample. This is because in the moving block bootstrap method, the blocks

are randomly chosen and assembled. In addition, this procedure assumes that the series must be stationary. Using the ADF or the PP test, it confirms that there is no unit root for the series of volatility proxies. Interestingly, the results of the KPSS test show that the null hypothesis of stationarity must be rejected. These seemingly contradictory results may be an expression of heteroskedasticity and structural breaks that are supposed to be captured by these series. Moreover, it is known that the parameters of GARCH models sum close to unity, suggesting that the volatility processes are close to a unit root which may explain the different test results.
Overall, however, this means that the moving block bootstrap method cannot be considered optimal for series with structural breaks and near a unit root. Even so, the choice of block length l is non-trivial and is the subject of current research. The work of Politis and White (2004) and the correction of this work by Patton et al. (2009) should be mentioned. These papers use a spectral density approach to determine optimal block length. In the present thesis, we treat block length as hyperparameter of the random forest. To check if the results of the tuning process are theoretically justified, we follow the work of Hall et al. (1995), who show that the optimal asymptotic formula for optimal block length is proportional to $n^{\frac{1}{k}}$ for $k \in \{3, 4, 5\}$. According to the authors, the choice of k depends strongly on the context. That is, smaller k values can be used to estimate bias or variance, while larger values can be used for more complex calculations such as confidence intervals. Since volatility forecasting can be considered a complex task for many reasons described, a large k should possibly be chosen for approximation. For $k = 5$, a block length of 5.64 is obtained in the present case.

4.4. Support Vector Regression

The last approach from supervised machine learning we consider are *support vector machines*, or, in the present context *support vector regression*. Mohri et al. (2018) describe this regression approach as inspired by the idea of support vector classification. The idea is to estimate a regression on the data with a tube of radius $\epsilon > 0$, such that most observations lie within this ϵ-*environment*. With this approach, we get two regions in which observations lie. One region describes the data within the ϵ-environment which are not penalized. The other region contains all observations that lie outside the ϵ-environment and are penalized according to the distance to the fitted function. The fitted hyperplane is represented by support vectors, which are all elements from the data outside of the ϵ-tube. Support vectors influence how the ϵ-environment and its shape is determined. For a small ϵ, one allows only a small error tolerance, and accordingly more observations lie outside the ϵ-tube, which means a higher number of support vectors. The reverse is true for large ϵ, since fewer observations lie outside the ϵ-tube, resulting in fewer support vectors. This intuitive description of the idea of support vector regression is shown in Figure 4.4.

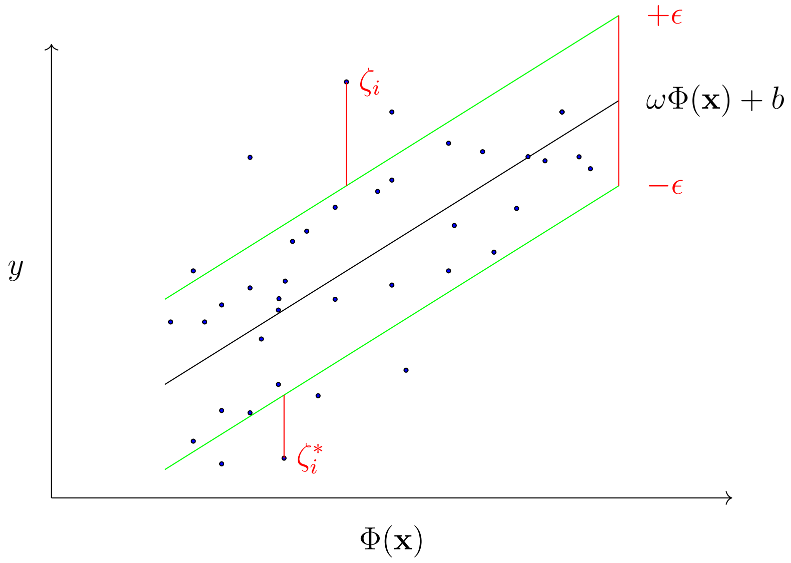


Figure 4.5.: Soft margin support vector regression for a feature mapping $\Phi(\mathbf{x}) = \mathbf{x}$ corresponding to some kernel K. Points inside the green bounds are in the ϵ-environment. ζ_i and ζ_i^* are slack variables that measure the deviation of points outside the ϵ tube to the tube. See Smola and Schölkopf (2004).

To elaborate the concept mathematically, we follow the explanations of Mohri et al. (2018), Awad and Khanna (2015), and Smola and Schölkopf (2004). First, we consider the hypothesis set of all linear functions $\mathcal{F} = \{\mathbf{x} \mapsto \mathbf{w}^\top \Phi(\mathbf{x}) + b : \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\}$. Here Φ is the feature mapping, a function of the input variables, which allows more complex structures of input variable $\mathbf{x}$. Mapping Φ is related to a positive definite kernel K since features $\mathbf{x}$ can be very high dimensional and, therefore, scalar product $\langle \Phi(\mathbf{x})\Phi(\mathbf{x}^{'})\rangle$ can only be determined with much effort and is possibly inefficient. But if there is a kernel function $K(\mathbf{x}, \mathbf{x}^{'}) = \Phi(\mathbf{x})^\top \Phi(\mathbf{x}^{'})$ corresponding to this scalar product, then the scalar product can be replaced by the kernel function, which increases the efficiency of the calculations. This procedure is also called the *kernel trick* and will be used for later derivations.[17] In order to approximate the unknown function with support

[17]A detailed introduction and discussion of kernel functions and its properties can be found in Mohri et al. (2018). Here is just a short example: for $K(\mathbf{x}, \mathbf{x}^{'}) = (\mathbf{x}^\top \mathbf{x}^{'})^2$ with $\mathbf{x} = (x_1, x_2)^\top$ it follows that

vector regression, this approach uses optimization techniques that can be justified from Figure 4.4. As described, the goal is to find a ϵ-environment which contains the most observations and is closest to the fitted function $f(x) = \mathbf{w}^\top \Phi(\mathbf{x}) + b$ while minimizing the prediction error. We first assume that we seek a function with a ϵ-tube that contain all observations. In the case of Figure 4.4 that means that all points lie either in the upper or lower ϵ-environment. To achieve this, we must minimize the length of normal vector $\|\mathbf{w}\|$ to the approximated surface. Additionally the constraints that all deviations are within the ϵ neighborhood shall apply. From this description, we obtain the optimization problem

$$\begin{aligned} &\min_{\mathbf{w},b} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 \right\} \\ \text{s.t.} &\begin{cases} y_i - \mathbf{w}^\top \Phi(\mathbf{x}_i) - b & \leq \epsilon \\ \mathbf{w}^\top \Phi(\mathbf{x}_i) + b - y_i & \leq \epsilon. \end{cases} \end{aligned} \tag{4.20}$$

The formulation of optimization problem (4.20) is due to mathematical convenience and permissible, since $\frac{1}{2}\|\mathbf{w}\|^2$ is a monotonically increasing function over the non-negative domain, and thus $\|\mathbf{w}\|$ and $\frac{1}{2}\|\mathbf{w}\|^2$ share the same minimum value for the given constraints. Minimization problem (4.20) can be represented more compactly, as in Mohri et al. (2018), using the ϵ-sensitive loss function

$$|y_i - (\mathbf{w}^\top \Phi(\mathbf{x}_i) + b)|_\epsilon = \max(0, |y_i - (\mathbf{w}^\top \Phi(\mathbf{x}_i) + b)| - \epsilon) \tag{4.21}$$

which does not consider any losses for values within the ϵ-environment and only penalize points outside the ϵ-tube, the support vectors. Therefore this loss function provides sparse solutions with a relatively small number of support vectors. Thus, it is possible to further rewrite the optimization problem as

$$\min_{\mathbf{w},b} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} |y_i - (\mathbf{w}^\top \Phi(\mathbf{x}_i) + b)|_\epsilon \right\} \tag{4.22}$$

$K(\mathbf{x}, \mathbf{x}') = x_1^2 x_1'^2 + 2x_1 x_1' x_2 x_2' + x_2^2 x_2'^2 = \begin{pmatrix} x_1^2 & \sqrt{2} x_1 x_2 & x_2^2 \end{pmatrix} \begin{pmatrix} x_1'^2 \\ \sqrt{2} x_1' x_2' \\ x_2'^2 \end{pmatrix} = \Phi(\mathbf{x})^\top \Phi(\mathbf{x}')$. This shows that for an two dimensional input space, the second-degree polynomial corresponds to the scalar product of dimension three.

where $|.|_\epsilon$ is the ϵ-sensitive loss and C is a positive numerical value which we consider in more detail below. In the optimization problem (4.20) and (4.22), it is assumed that a function can be found for which all observations lie within the ϵ-environment. This assumption is somewhat restrictive and can be relaxed by using slack variables $\zeta_i, \zeta_i^* \geq 0$ for each point, as shown in Figure 4.4. This makes it possible to allow for additional errors up to the value of slack variables ζ_i, ζ_i^* and to satisfy the otherwise infeasible constraints from Equation (4.20). Note that while each point theoretically has two slack variables, only one of these is non-zero because a point can only be either above or below the ϵ-environment. Considering the slack variables, we obtain the optimization problem

$$\min_{\mathbf{w},b,\zeta_i,\zeta_i^*} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \right\}$$
$$\text{s.t.} \begin{cases} y_i - \mathbf{w}^\top \Phi(\mathbf{x}_i) - b & \leq \epsilon + \zeta_i^* \\ \mathbf{w}^\top \Phi(\mathbf{x}_i) + b - y_i & \leq \epsilon + \zeta_i \\ \zeta_i, \zeta_i^* \geq 0 & \forall i \in \{1, \dots, n\} \end{cases} \tag{4.23}$$

where C denotes an optimization parameter that penalizes observations that lie outside the ϵ-environment. This parameter controls the tradeoff between complexity of the function to be approximated and the amount of allowed deviations. Therefore, C is a tuning parameter that can be used to avoid overfitting.
As described by Smola and Schölkopf (2004), the Lagrange method is used to solve *primal* optimization problem (4.23). This method allows us to construct a Lagrange function from the original objective function with constraints by introducing *dual variables*, which in turn allows us to consider the problem as a *dual* optimization problem. This is possible because the optimization problem at hand (4.23) is convex and, thus, the optimal solutions of the primal problem and dual problem coincide. That is, the value of the optimal solution of the primal problem is given by the solutions of the dual problem. This is advantageous in that optimization problem (4.23) is easier to solve in dual form, where the dual form is crucial for the nonlinear extension of support vector regression. For the primal Lagrange function, we assume that $\alpha_i, \alpha_i^* \lambda_i, \lambda_i^* \geq 0$ are the

Lagrange multipliers, thus we obtain

$$\begin{aligned}\mathbb{L}(\mathbf{w}, b, \zeta_i\zeta_i^*, \alpha_i, \alpha_i^*, \lambda_i, \lambda_i^*) =& \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) - \sum_{i=1}^{n}(\lambda_i\zeta_i + \lambda_i^*\zeta_i^*) \\ &- \sum_{i=1}^{n}\alpha_i^*(\mathbf{w}^\top\Phi(\mathbf{x}_i) + b - y_i - \epsilon - \zeta_i^*) \\ &- \sum_{i=1}^{n}\alpha_i(y_i - \mathbf{w}^\top\Phi(\mathbf{x}_i) - b - \epsilon - \zeta_i).\end{aligned} \tag{4.24}$$

The minimum of primary Lagrangian function (4.24) can now be determined by partial derivatives with respect to the variables which must be set equal to zero. In addition, the *Karush-Kuhn-Tucker (KKT)* conditions, which state that the product of the Lagrange multipliers and the constraints must be equal to zero, must hold. To better understand the role of the conditions and constraints, we look at the optimization step by step and first determine the partial derivatives with respect to primary variables and slack variables $(\omega, b, \zeta_i, \zeta_i^*)$. In order not to overstress the notation, we write $\mathbb{L}$ for the Lagrange function $\mathbb{L}(\omega, b, \zeta_i\zeta_i^*, \alpha_i, \alpha_i^*, \lambda_i, \lambda_i^*)$. Therefore, the partial derivatives can be written as

$$\frac{\partial\mathbb{L}}{\partial\mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(\mathbf{x}_i) = 0 \tag{4.25}$$

$$\frac{\partial\mathbb{L}}{\partial b} = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i) = 0 \tag{4.26}$$

$$\frac{\partial\mathbb{L}}{\partial\zeta_i} = C - \lambda_i - \alpha_i = 0 \tag{4.27}$$

$$\frac{\partial\mathbb{L}}{\partial\zeta_i^*} = C - \lambda_i^* - \alpha_i^* = 0 \tag{4.28}$$

Partial derivatives (4.25), (4.26), (4.27) and (4.28) can now be substituted into the primary Lagrange function (4.24) to obtain the optimization problem in dual form

$$\max_{\alpha_i,\alpha_i^*}\left\{-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_j) - \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n}y_i(\alpha_i - \alpha_i^*)\right\}$$
$$\text{s.t.}\begin{cases}\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0\\ \alpha_i, \alpha_i^* \in [0, C].\end{cases} \tag{4.29}$$

By transforming equations (4.27) and (4.28) to $\lambda_i = C - \alpha_i$ and $\lambda_i^* = C - \alpha_i^*$, respectively, dual variables λ_i, λ_i^* can be eliminated from optimization problem (4.29). It should be noted that dual problem (4.29) is also a convex quadratic problem, which can be solved by various quadratic optimization methods. However, from equation (4.25) follows $\mathbf{w} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(\mathbf{x}_i)$, which shows that $\mathbf{w}$ can be represented as a linear combination of the input variables. Together with $f(\mathbf{x}) = \mathbf{w}^\top\Phi(\mathbf{x}_i) + b$, this gives the *support vector expansion*

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_i) + b. \tag{4.30}$$

which is used for predictions of new inputs. From this representation, we see that $\mathbf{w}$ does not have to be explicitly calculated to evaluate the complete function $f(\mathbf{x})$. Moreover, it shows that product $\Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_j)$ corresponds to kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ and can be replaced by it, which extends the support vector regression to non-linear function approximation.
Let us now consider how offset b can be determined. For this we consider the following KKT conditions

$$\alpha_i(y_i - \mathbf{w}^\top\Phi(\mathbf{x}_i) - b - \epsilon - \zeta_i) = 0 \tag{4.31}$$
$$\alpha_i^*(\mathbf{w}^\top\Phi(\mathbf{x}_i) + b - y_i - \epsilon - \zeta_i^*) = 0 \tag{4.32}$$
$$(C - \alpha_i)\zeta_i = 0 \tag{4.33}$$
$$(C - \alpha_i^*)\zeta_i^* = 0 \tag{4.34}$$

which are also called *complementary slackness* conditions. These conditions allow different statements about the contribution of individual points $(\mathbf{x}_i, y_i)$ for the support vector regression and shows the way to calculate offset b. We follow Bishop and Nasrabadi (2006) for the arguments and statements.

For any $(\mathbf{x}_i, y_i)$ it holds that $\alpha_i\alpha_i^* = 0$

In the case of a general point $(\mathbf{x}_i, y_i)$, it holds that $\alpha_i\alpha_i^* = 0$. This follows from the fact that if we assume $\alpha_i > 0$ and $\alpha_i^* > 0$, then both constraints would apply and the sum of complementary slack conditions $(y_i - \mathbf{w}^\top\Phi(\mathbf{x}_i) - b - \epsilon - \zeta_i) = 0$ and $(\mathbf{w}^\top\Phi(\mathbf{x}_i) + b - y_i - \epsilon - \zeta_i^*) = 0$ equals $\zeta_i + \zeta_i^* = -2\epsilon$. However, since by definition, only one of the slack variables ζ_i, ζ_i^* is non-zero and $\epsilon > 0$, it must be negative, which contradicts the assumption $\zeta_i, \zeta_i^* \geq 0$. Thus, there is no set of dual variables α_i, α_i^* which are simultaneously non-zero. Since we have additionally assumed that $\alpha_i, \alpha_i^* \geq 0$, only one of the Lagrange multipliers α_i, α_i^* can be greater than zero at a time, which means that the fitted function either overestimates or underestimates the true value by more than ϵ.

For $(\mathbf{x}_i, y_i)$ in the ϵ-environment, it holds that $\alpha_i = 0$ and $\alpha_i^* = 0$

For points inside the ϵ-environment the slack variables are zero, i.e., $\zeta_i = 0$ and $\zeta_i^* = 0$ holds. Since these points are not support vectors, they do not contribute to the predictions and we have $\alpha_i = \alpha_i^* = 0$. This is also evident by the complementary slackness conditions, where we have $(y_i - \mathbf{w}^\top \Phi(\mathbf{x}_i) - b - \epsilon) < 0$ and therefore $\alpha_i = 0$ must hold so that the slackness conditions are valid. The same is true for $(\mathbf{w}^\top \Phi(\mathbf{x}_i) + b - y_i - \epsilon) < 0$ which also leads to $\alpha_i^* = 0$.

For $(\mathbf{x}_i, y_i)$ outside the ϵ-environment, where we have $\alpha_i = C$ or $\alpha_i^* = C$ and points on the ϵ-tube, it holds that $\alpha_i, \alpha_i^* \in [0, C]$

From the definition of $\lambda_i, \lambda_i^* \geq 0$ and Equations (4.27) and (4.28), we conclude that $\alpha_i, \alpha_i^* \in [0, C]$ holds in general. For points $(\mathbf{x}_i, y_i)$ above or on the upper ϵ-environment, it follows that $y_i - \mathbf{w}^\top \Phi(\mathbf{x}_i) - b - \epsilon = \zeta_i \geq 0$ is true. If a point $(\mathbf{x}_i, y_i)$ lies on the ϵ-bound, $\zeta_i = 0$ applies which implies $\lambda_i \geq 0$ and, therefore we must have $\alpha_i \in [0, C]$. If we consider points $(\mathbf{x}_i, y_i)$ above the upper ϵ-environment, we have $\zeta_i > 0$ and $\lambda_i = 0$, which implies $\alpha_i = C$. This is analogously true for points on or below the lower ϵ-environment, such that $\alpha_i^* \in [0, C]$ holds for points on the lower ϵ-tube and $\alpha_i^* = C$ for points below the lower ϵ-bound.

All in all, these explanations show that only those points which are support vectors have to be considered and, thus, sparse solutions are available. However, the parsimony of solutions depends essentially on the choice of the ϵ-environment. This controls a tradeoff between prediction accuracy and sparsity. If we allow only small deviations, i.e., a small ϵ, then there are many support vectors since many observations lie outside the ϵ-tube and the solution is less parsimonious. Conversely, if we allow a larger error tolerance, with a larger ϵ, then there are fewer support vectors since many observations lie within the ϵ-environment and the solution is more parsimonious. Overall, it is possible to determine offset b from the previous statements. For an input $\mathbf{x}_j$ within the ϵ-environment with $\zeta_j = 0$ and $0 < \alpha_j < C$, offset b can be calculated from complementary slackness constraints (4.31) and (4.33) by

$$\begin{aligned} b &= y_j - \mathbf{w}^\top \Phi(\mathbf{x}_j) - \epsilon \\ &= y_j - \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j) - \epsilon. \end{aligned} \tag{4.35}$$

Analogously, we obtain this result for an input $\mathbf{x}_j$ with $0 < \alpha_j^* < C$. At this point we want to emphasize again that all mathematical representations shown were made using feature mapping $\Phi(\mathbf{x})$. It has already been noted that product $\Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)$ can be replaced by kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ to obtain an efficient calculation while also extending the support vector regression algorithm for non-linear regression problems. All presented mathematical formulas are still valid, but should be replaced with kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, which we do not explicitly due to the limited scope of this work.
There are, of course, several kernel functions that can be used for prediction, which begs the question, which function should be used? This question is difficult to answer because kernel functions are not hyperparameters and only a few papers use the same machine learning techniques for volatility prediction. Therefore, we follow the work of Peng et al. (2018) and their reasoning and use the *radial basis function* kernel, or *RBF* kernel for short, which is defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right) \tag{4.36}$$

with $\sigma > 0$. Peng et al. (2018) state that this kernel function is one of the most widely used in machine learning because it can represent an infinite dimensional space while depending only on the single parameter σ.

5. Evaluation

To choose optimal hyperparameters for models or to compare different models in terms of their out-of-sample forecasting performance, cross-validation techniques such as K-fold cross-validation are usually used and forecasts are performed. The so generated forecasts of each model are then evaluated and compared using various performance measures such as Mean-Square-Error (MSE), Mean-Absolute-Error (MAE) or others. Since we want to forecast volatility, a latent variable, and we use time series data to do so, various properties of volatility and the dependency structure of the data must also be taken into account.

As already described in the chapter on volatility in financial markets, this is a latent variable that must be approximated. The choice of the volatility proxy can therefore have a considerable influence on the out-of-sample forecast performance of different models. Volatility forecasts depend on various other factors as well. According to Poon and Granger (2003), one of these important factors is current volatility level and the volatility level of the considered forecast horizon. If a model is fitted to data that consistently exhibit moderate volatility and is used to forecast highly volatile periods, then it is unlikely that the model will achieve good forecasting performance. This is because time series models are based on historical data and are, by design, unsuitable for forecasting unpredictable and unprecedented events. Since this paper compares conventional time series models with machine learning methods, which use also purely autoregressive data, it is not about forecasting only highly volatile phases or even shocks, but about a comparison to show whether machine learning methods are suitable and possibly even better suited for volatility forecasting. To forecast shocks, additional explanatory variables should be included as regressors, regardless of whether classical econometric volatility models or machine learning models are used.

Another important factor is volatility structure or its persistence. As shown in Figure 2.2, the high persistence of volatility proxies is an expression of long-lasting significant autocorrelation. Capturing these structures is the task of the models. It is known that simple GARCH models cannot capture high persistence, but as Corsi (2009) notes, the

HAR-RV model is able to capture long-lasting effects. The machine learning models used in this work do not have properties in their construction that specifically take long memory effects into account. To sufficiently account for long-lasting effects, it is therefore necessary to include the most significant autocorrelations in the models. This is attempted in the present work by including a relatively high number of lagged variables as predictors in the models. However, we will also use the HAR model construction and build based on this idea machine learning models.
Poon and Granger (2003) further, describe that forecast horizon and data frequency play a significant role in predicting volatility, specifically that forecast horizon depends on data frequency. Many studies shown that for forecasts over large horizons, forecast error is larger for data with higher frequency. This resulted in simple methods, such as moving average models, being superior to model-based approaches, such as GARCH models. Overall, the higher the frequency of the data, the shorter the forecast horizon should be and vice versa. We take this into account in the emprical analysis by varying the forecast horizons between three months two years.
Two further essential aspects that must be considered when forecasting volatility are the cross-validation to estimate the test error of the model, as well as the choice of a metric for comparing models that takes into account possible asymmetries. We look at these two factors in more detail below.

5.1. Time Series Cross Validation

According to James et al. (2013), cross-validation is generally a resampling procedure used to estimate the test error of a model. The test error describes the average error of the model that results from the prediction of a new observation that was not used for training. The test error can be easily determined if test data are available. This is often not the case, since usually only one data set is available. To use this data set to estimate the test error, there are several methods, one being the cross-validation method. The idea of cross-validation is to hold out a subset of the data which is not used for training the models but for prediction. This general idea of cross-validation can be implemented in a number of different ways, with the best known and most widely used being K-fold cross-validation.

K-Fold Cross-Validation

In K-fold cross-validation, data $\mathcal{S}$ are randomly split into K parts of approximately equal size. Then $K-1$ parts of the data are used for training the model and the K-th part is used for forecasting and calculating prediction error $\mathcal{R}$ associated with a loss function ℓ. This procedure is repeated for $k = 1, \ldots K$ and prediction errors $\mathcal{R}^{(1)}, \mathcal{R}^{(2)}, \ldots, \mathcal{R}^{(k)}$ can be determined for each of the K parts. The test error from the K-fold cross-validation is then determined as the average over all K forecast errors

$$CV_{(K)} = \frac{1}{K} \sum_{k=1}^{K} \mathcal{R}^{(k)}. \tag{5.1}$$

Figure 5.1 schematically represents the K-fold cross-validation.

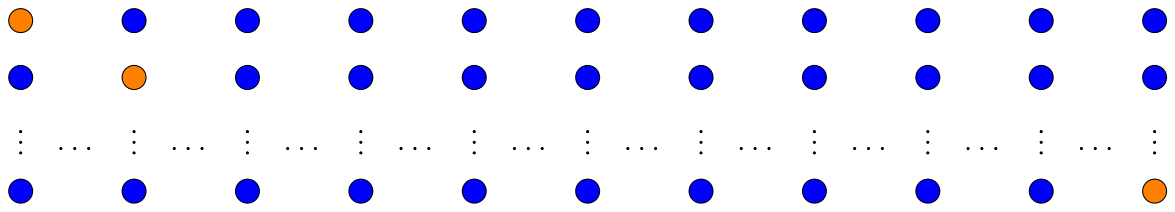


Figure 5.1.: $K = 11$-fold cross-validation. The blue dots represent $K-1 = 10$ data sets for training and the orange dots represent the respective test data set. Each line represents one iteration, from here $K = 11$ total iterations. See Hastie et al. (2009) and Hyndman and Athanasopoulos (2018).

From Figure 5.1, it seems that K-fold cross-validation does not appear to be appropriate for dependent data such as time series. This impression is confirmed when looking at the first row or iteration in this figure. Here one would try to forecast the past with future data. Described a bit more technically and following Bergmeir et al. (2018), this means that K-fold cross-validation would remove K randomly selected parts of time series $\{x_t\}$ to use as test data. This approach would destroy the time dependency of the series and therefore does not seem to be permissible, since the errors from the training data would correlate with the errors of the test data.

Based on the described problem the result obtained by Bergmeir et al. (2018) in their work on K-fold cross-validation for time series models is surprising. The authors show that if a purely autoregressive modeling is chosen and the errors of the models are uncorrelated, K-fold cross-validation can and should be used in the context of time series. This result is supported by a proof and an experiment. The proof presented by Bergmeir et al. (2018) is based on an $AR(p)$ model, where only the purely autoregressive structure of the model is crucial. Thus, it is possible to extend the proof to other models as well. In their experiment, the authors compare different cross-validation procedures such as 5-fold cross-validation, leave-one-out cross-validation, rolling-origin validation and others in terms of one-day-ahead out-of-sample prediction. In doing so, the authors consider various scenarios, seeking to determine how each cross-validation procedure performs when models are close to the true data-generating process and also when models are misspecified. It turns out that K-fold cross-validation gives the best estimates of the test error if the models are properly specified. If the models are misspecified, rolling-origin validation seems to give slightly better results.

It is not clear whether the models in the present work should be evaluated with K-fold cross-validation. This is because not all models like the GARCH model follow a strict autoregressive structure. It is possible to represent the GARCH model as an ARCH process as in Equation (3.2), which only use past squared returns to describe the conditional variance at the current time and is therefore purely autoregressive, but with an infinite number of delayed variables. Overall, the work of Bergmeir et al. (2018) is worth mentioning because of its astonishing results but should not be considered as the only evaluation option. In the econometric literature, out-of-sample prediction performance evaluation of models is usually performed using a procedure called *rolling-evaluation*. We discuss one such procedure in the following.

Rolling-Origin Validation

As Hyndman and Athanasopoulos (2018) and Tashman (2000) describe, the general idea of rolling-origin validation is that the entire time series of length T is first divided into a training data set of length t and a test data set $T - t$. The split is made so that all training data are temporally prior to the test data in order to exclude the possibility of predicting the past with future data. In addition, it must be taken into account that the training data is sufficiently large to obtain reliable forecasts. This first split is the first iteration of a general iterative process which is performed multiple times. In the first iteration, the full t training data are used to obtain a prediction of the $t + 1$ value. In the second step, the training data set is increased or updated by one observation and, accordingly, has a length of $t+1$. On this $t+1$ training data, the model is refitted and a forecast for the $t + 2$-th value is performed. This iterative procedure is performed until $T - 1$ training data is available and the T-th value is predicted. A visual depiction of the one-step-ahead rolling-origin validation is given in the figure below.

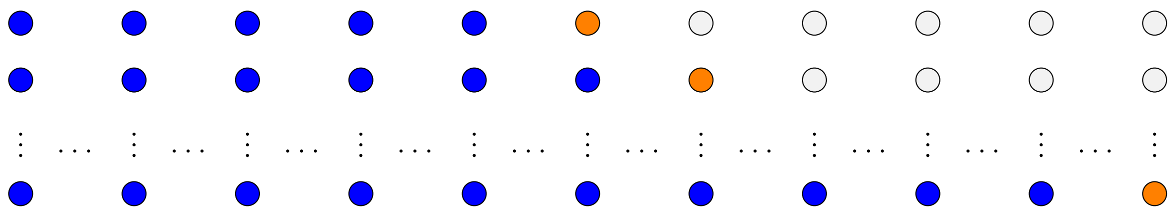


Figure 5.2.: Blue dots represent the training data and orange dots the test data respectively. The light gray dots represent the test data that are not used for the evaluation. See Hyndman and Athanasopoulos (2018).

Figure 5.2 shows a time series with $T = 11$ observations as an example. Here, the data is first divided into training data with an initial length of $t = 5$ observations and a test data set with a length of $T - t = 6$ observations. The first five observations are used for training in the first iteration and only the sixth observation is used for prediction. In the second iteration, the sixth observation is added or updated to the training data set and the prediction is performed for the seventh observation. This procedure is performed until the training data contains $T - 1 = 10$ observations and the $T = 11$ observation is predicted. For each iteration, the prediction error $\mathcal{R}$ can be determined and, similar to K-fold cross-validation, the test error of the rolling-origin validation can be determined by the average over all test errors of the iterations.
This validation approach can also be generalized to a direct h-steps-ahead forecasting scheme. According to Ben Taieb and Hyndman (2012), the direct forecasting strategy uses different models for each of the different forecast horizons h. Thus, for a h-step-ahead prediction of r_t, a model is trained using only lagged variables $(r_{t-h}, r_{t-h-1}, \dots)$ and therefore a direct h-step-ahead forecast can be generated according to

$$r_t = f(r_{t-h}, r_{t-h-1}, \dots). \tag{5.2}$$

Ben Taieb and Hyndman (2012) note that this forecasting strategy produces unbiased forecasts under the condition that function f is flexible enough. Moreover, if function f is close to the true function sought, the mean square error of the direct h-step forecast is smaller than that of the recursive h-step forecast. However, since a separate model must be fitted for each forecast horizon h, this approach is more computationally intensive than the recursive strategy. For a more detailed discussion of the different forecasting strategies for time series, we refer the reader to Ben Taieb and Hyndman (2012), Chevillon and Hendry (2005), and Chevillon (2007). Overall, the generalization does not change the described iterative process of rolling-origin cross-validation. Even in the generalized approach, the data is split into training data of length t and test data of length $T-t$. In the first iteration, the model is fitted on the t training data and the prediction is made for $t+h$ based on the data available until time t. In the second iteration, the training dataset is updated by one observation so that it now has length $t+1$. The model is then fitted on this data set and the prediction is performed for time point $t+(h+1)$ based on the data available until time $t+1$. This procedure is shown in the figure below for a $h=2$ step-ahead validation.

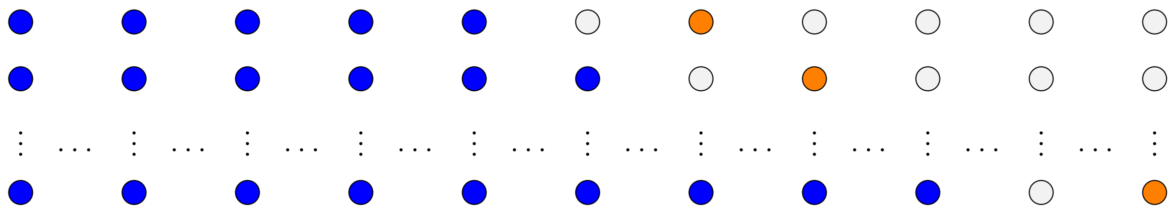


Figure 5.3.: Blue dots represent the training data and orange dots the test data respectively. The light gray dots represent the test data that are not used for the evaluation. See Hyndman and Athanasopoulos (2018)

5.2. Performance Measures

To evaluate the forecast performance from the cross-validation procedures, the forecast error on the test data set must be determined using statistical loss functions. As already described several times, volatility is a latent variable that has to be approximated thereby introducing potential noise. Thus, the choice of a volatility proxy also plays an essential role in the choice of loss functions for the evaluation and comparison of different models because biases may arise from the selection of a noisy proxy. According to Patton (2011), it is still possible to find suitable loss functions which can handle potentially noisy volatility proxies. For this purpose, he defines robust loss functions. These are loss functions whose model ranking with respect to the expected loss remains the same, regardless of whether the true conditional variance or an unbiased volatility proxy is used for the evaluation.

In the literature on volatility forecasting, there are many loss functions with different properties which are used for forecast evaluation and model comparison. An overview of the various loss functions commonly used is also given by Patton (2011). Most of the loss functions considered in his paper do not satisfy the definition of robustness. In fact, it turns out that out of nine different commonly used loss functions, only the mean squared error (MSE) and the quasi-likelihood loss functions (QLIKE) satisfy these requirements. Based on these results, we use the MSE and QLIKE loss functions in this paper, and, for comparison, the non-robust loss function, the mean absolute error (MAE). In addition, we use the linear exponential loss function (LINEX) proposed by Poon and Granger (2003). Following Patton (2011) and Poon and Granger (2003), MSE, MAE, QLIKE, and LINEX are defined as

$$MSE(\sigma^2, \hat{f}) = \frac{1}{T-t} \sum_{i=T-t}^{T} (\sigma_i^2 - \hat{f}_i)^2 \tag{5.3}$$

$$MAE(\sigma^2, \hat{f}) = \frac{1}{T-t} \sum_{i=T-t}^{t} \mid \sigma_i^2 - \hat{f}_i \mid \tag{5.4}$$

$$QLIKE(\sigma^2, \hat{f}) = \frac{1}{T-t} \sum_{i=T-t}^{T} \left(\frac{\sigma_i^2}{\hat{f}_i} - \ln(\frac{\sigma_i^2}{\hat{f}_i}) - 1 \right) \tag{5.5}$$

$$LINEX(\sigma^2, \hat{f}) = \frac{1}{T-t} \sum_{i=T-t}^{T} \left[\exp(-a(\hat{f}_i - \sigma_i^2)) + a(\hat{f}_i - \sigma_i^2) - 1 \right] \tag{5.6}$$

where σ_i^2 describes a volatility proxy and $\hat{f}_i$ the volatility forecast. The choice of different loss functions is due to not only the robustness criterion but also because specific properties of the loss functions play an important role in the choice of the evaluation criterion. MSE and MAE are symmetric loss functions because they weight overestimates and underestimates identically. This means that changing the order of σ^2, $\hat{f}_i$ in the functions does not matter for the loss and the penalty of an overestimation $\hat{f}_i = \sigma_i^2 + \delta$ is the same as for an underestimation $\hat{f}_i = \sigma_i^2 - \delta$. However, the penalization of incorrect forecasts differs between the two loss functions, as the mean square error penalizes incorrect forecasts more heavily than the mean absolute error does. The symmetric property of the MSE and MAE follow directly from their mathematical definitions, but it is useful to put these properties into the present context. In this paper, the objective is to forecast the volatility of the DAX. Therefore, the loss functions should describe economic uncertainties in the financial markets. In the present context, large forecast errors in volatility forecasts are disadvantageous for several reasons. First, it is conceivable that large forecast errors can lead to sub-optimal investment decisions or that the maximum expected loss of the capital under risk is wrongly estimated. Second, it is conceivable that an incorrect estimate of volatility affects market capitalization, since the value of a company can be derived from expected earnings, which are affected by volatilities. From these reasons, it can be seen that an underestimation of volatility in particular entails special risks and must be taken into consideration. MSE and MAE do not account for this special risk of underestimating volatility because they are symmetric loss functions penalizing underestimation in the same way as overestimation. However, the MSE seems to be more appropriate compared to the MAE since the former penalizes large forecast errors more heavily.

To account for the risk of underestimating volatility, asymmetric loss functions such as QLIKE and LINEX are more appropriate. By definition, the QLIKE loss function penalizes an underestimation of volatility more than an overestimation. The LINEX loss function has an additional parameter $a > 0$ that controls the symmetry or asymmetry of the function. In the limiting case for $a \to 0$, it even holds that $LINEX \to MSE$ and the LINEX loss function is nearly symmetric[1]. Thus, the parameter a controls the strength of the penalization and for large $a > 0$, overestimates are penalized approximately linearly while underestimates are penalized exponentially. This also justifies the naming of the LINEX loss function. When using these asymmetric loss functions, however, possible weaknesses should also be considered. The QLIKE and LINEX loss functions are only partial and imperfect solutions to control for risks of underestimation. This is because the loss functions penalize underestimates especially strongly, partly even stronger than MSE does. However, the advantage that these risk functions penalize underestimation so strongly can also be a disadvantage. This becomes clear, for example, if we look at two competing models and their volatility forecasts. Both models have a bias of identical magnitude, with one model having a positive bias (overestimation) and the other model having a negative bias (underestimation). In this case, the QLIKE and LINEX functions would favor the model with positive bias since overestimates are penalized less than underestimates. The reasons why underestimation should be particularly avoided support the use of these loss functions, of course, but the weaknesses should also be considered and the QLIKE and LINEX loss functions should be considered together with other loss functions. Nevertheless, the asymmetric loss functions can be used to assess which of the competing models is less likely to underestimate volatility and therefore better suited for forecasting shocks and highly volatile periods.

In addition to the loss functions mentioned, we also use the squared correlation between the actual values of the test data and the predicted values. The use of squared correla-

[1]To show that $LINEX \to MSE$ holds, we have to use the Taylor series at point zero, called the Maclaurin series for the exponential function which is given by $\exp\{a(\hat{f} - \sigma^2)\} = \sum_{i=0}^{\infty} a^i \frac{(\hat{f}-\sigma^2)^i}{i!}$. Plugging this expression into the LINEX loss function and taking the limit for $a \to 0$ proves the statement.

tion is equivalent to the R^2 of a regression of the observed quantities on the predicted ones. This procedure is called *Mincer-Zarnowitz Regression* and, according to Patton (2011), leads to a robust ranking with respect to a noisy volatility proxy of different models.

6. Empirial Results and Discussion

In this section, we consider the empirical analysis. Before we analyze the forecasting performance of the models and draw comparisons, we discuss the general procedure in section 6.1. We first focus on the partitioning and structuring of the data, as well as the calibration of the models' hyperparameters. In section 6.2, we present and discuss the results of the primary analysis. In doing so, in addition to comparing the forecasting performance of the individual models relative to the GARCH(1,1) benchmark model, we also consider the impact of the different modeling approaches and volatility proxies. Furthermore, we look at the forecasting performance of the models for different forecast horizons.

6.1. Empirical Setup

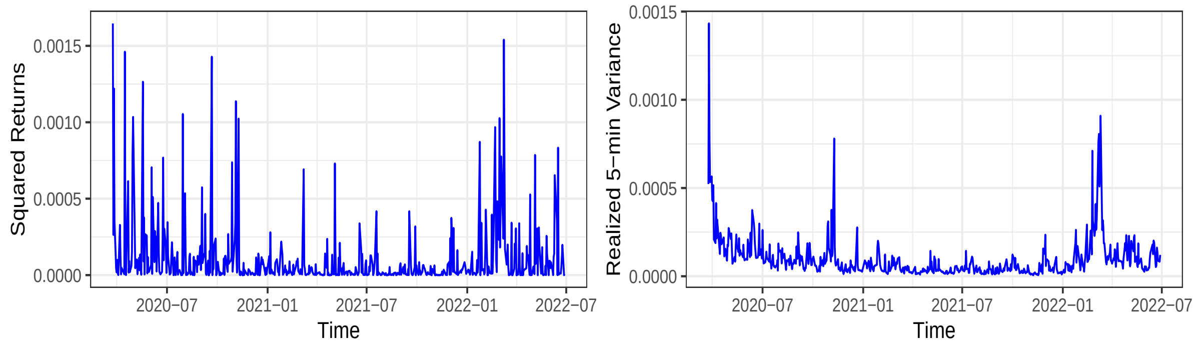
Data Splitting and Hyperparameter Tuning

As described in the introduction to this chapter, we first describe the general procedure before moving on to the analysis of forecast performance. The reason for this is two-fold. First, the procedure should be as transparent as possible and, second, some interesting results occur during these steps. It is necessary to split the whole data set to compare the forecasting performance of classical econometric models with machine learning models. This involves fitting the models on the training data and running the forecasts on test data. In addition, it must be taken into account that the machine learning models have hyperparameters that must be optimized to prevent overfitting and to obtain the best possible performance. Therefore, we need another data set, the validation data set. Overall, we divide the total data into training data, which is 80% of the total data, and a validation and test data set, each with 10% of the total data.
To calibrate the hyperparameters of the machine learning models, we use the *grid search* approach. Here we specify different values for the respective hyperparameters and consider all possible combinations. Models with different hyperparameter combinations are fitted on the training data and evaluated on the validation data using an rolling-origin

cross-validation approach. A hyperparameter combination is considered optimal if it has the smallest error on the validation data among all hyperparameter combinations considered.[1] Optimization is performed for the regression trees (TREE), random forests (RF), and support vector regression (SVR) for a total of six different models, as we consider two different modeling approaches. First, we consider purely autoregressive modeling with squared returns as features, where the models are referred to as ARCH-TREE, ARCH-RF, and ARCH-SVR below. Second, we use the theory of the Heterogeneous Market Hypothesis which underlies the HAR-RV model and use the same features from this model in the machine learning models. These models will be referred to as HAR-TREE, HAR-RF, and HAR-SVR in the following.
In the optimization process, a total of 210 ARCH-TREEs and HAR-TREEs as well as 10.080 ARCH-RFs and 9.072 HAR-RFs were evaluated and the best parameter combinations are chosen. The results of the hyperparameter optimization of the SVR model are particularly interesting. Here we evaluate 27 different ARCH-SVR and 120 HAR-SVR. It is shown in the HAR-SVR model that the optimal parameter combination includes $\epsilon = 0$. This means that all errors are penalized and that the number of support vectors is possibly equal to the number of observations. Although a high number of support vectors can often be an indication of overfitting the data, our finding here is the result of the grid search with many other SVR models allowing an error tolerance of $\epsilon > 0$. Thus, it is quite possible that the high number of support vectors is an expression of the high complexity and difficulty of learning the volatility. Another interesting observation is that the HAR-SVR model with $\epsilon = 0$ is similar to a leas absolute deviation regression with L_2 regularization. This suggests that regularization approaches are generally suitable for modeling and forecasting volatilities, which seems to be an interesting starting point for further investigation. However, in the present work, we use the empirical results from the grid search and use the optimal hyperparameter combination of the HAR-SVR including $\epsilon = 0$.
In addition to the tuning process, it is important tounderstand the structure of the test data, especially for time series. We therefore look at the test data for both volatility proxies that we want to forecast. Figure 6.1 shows the test data set of squared returns and realized variance. The last 10% of the total data is used as test data, which corresponds to about two trading years. From the time series of squared returns in Figure 6.1a, we can see that it has significantly more movements than that of realized variance

[1]As described in the section on random forests, these offer the advantage of out-of-bag data, which is used for grid search.

in Figure 6.1b. These stronger movements of squared returns were also noted in Section 2.3 and can be attributed to the theoretical conclusion that squared returns are a noisy volatility proxy.



(a) DAX volatility of test data measured with daily squared returns

(b) DAX volatility of test data measured with realized 5-min variance

Figure 6.1.: DAX volatility of the test data measured on daily squared returns and intraday 5-min squared returns.

This observation has several implications for modeling, forecasting, and evaluation. First, it is conceivable that autoregressive models that use only squared returns as features have problems correctly identifying structures in volatility data and making accurate forecasts because the data contain too much noise. However, models that use realized variance as features are also not able to achieve good forecast performance because squared returns have significantly more jumps that the models learn from the input. Overall, when squared returns are used as autoregressive features and as a volatility proxy, it is likely that the models will tend to underestimate and, thus, forecast performance, which is based on various loss functions that severely penalize underestimation, may appear to be inadequate. Overall, the higher noise component and associated stronger motions may cause forecast performance of the models to be underestimated. In comparison, the realized variance series in Figure 6.1b show a much smoother curve with less extreme swings. It is, therefore, reasonable to expect that the models produce significantly better forecasting results when evaluated with realized variance than with squared returns.

6.2. Results and Discussion

In this section we present the results of the empirical analysis, where in Section 6.2.1 we evaluate the results of the primary analysis for one-step, three-step, and five-step predictions obtained from the rolling-origin cross-validation procedure with full test data. In Section 6.2.2, we perform a robustness check by varying the length of test data. We consider forecast horizons of 66 days (three months), 132 days (six months), 264 days (one year), and a total test data length of 570 days (about 2 years).

6.2.1. Evaluation of the Forecasts

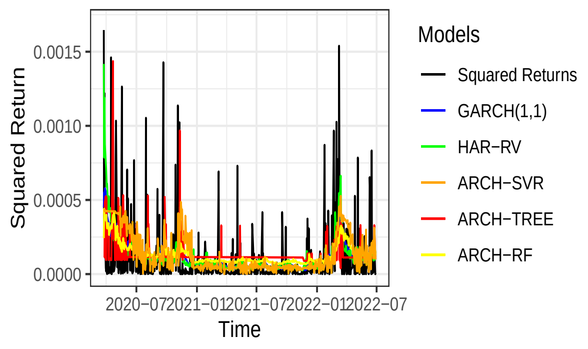
One-step-ahead predictions

We now analyze the one-step-ahead forecast performance of the various models. First, we consider the current values approximated by squared returns and realized variance and compare them to the forecasts of the respective models in Figure 6.2.

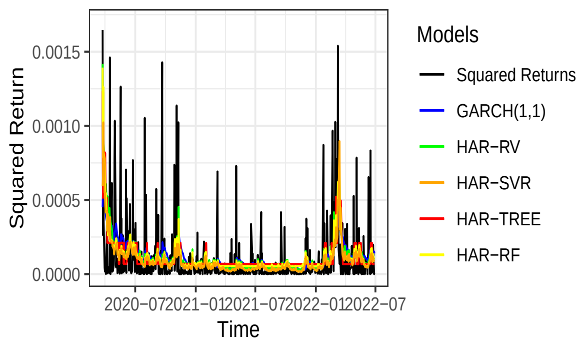
Overall, Figure 6.2 shows that the model types considered cannot forecast squared returns as accurately as realized variance. Figures 6.2a and 6.2b show in particular that many extreme expressions are poorly predicted. In comparison, realized variance and also its extreme expressions can be predicted much better, as seen in Figures 6.2c and 6.2d. As a result, the prediction performance of each model is significantly better when evaluated with realized variance than with squared returns. Figure 6.2 also shows that regardless of which volatility proxy is used, HAR-type models perform better than ARCH-type models. When the different models and the different volatility proxies are included in the analysis, the best forecasting performances are obtained for realized variance and HAR-type models. To support this initial graphical analysis, we consider the evaluation of the forecasting models using the performance measures described.

Table 6.1 presents the evaluation of the models using the described loss functions relative to the GARCH(1,1) model and use squared returns as a volatility proxy. In addition, the out-of-sample R^2 is given, which is not relative to the GARCH(1,1) model.

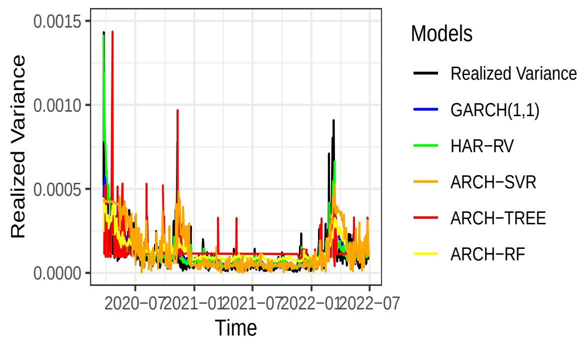
Various results can be seen from the table. First, all ARCH-type machine learning models appear to be inferior to the GARCH(1,1) model, since all loss functions are greater than one for those models and, thus, greater than that of the benchmark model. As described at the beginning of this chapter, this is due to the high complexity and the low signal-to-noise ratio in squared return data. In contrast, all HAR-type machine learning models perform better than the GARCH(1,1) baseline model. This finding allows us to draw some interesting conclusions. The better performance of the models, reflects



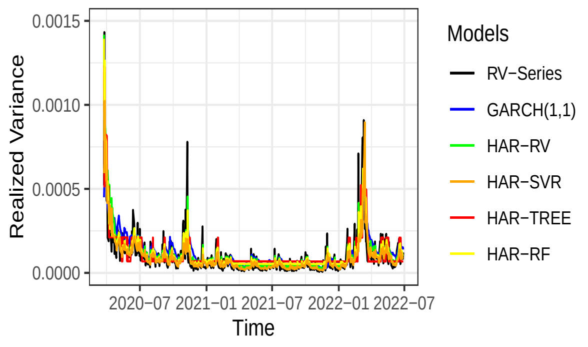
(a) ARCH-type model predictions with squared return volatility proxy



(b) HAR-type model predictions with squared return volatility proxy



(c) ARCH-type model predictions with realized variance volatility proxy



(d) HAR-type model predictions with realized variance volatility proxy

Figure 6.2.: One-step-ahead predictions from rolling-origin cross validation for HAR-type and ARCH-type models. The first row of the panel shows the evaluation with the squared return volatility proxy and the second row the realized variance.

the significantly higher signal-to-noise ratio of realized variance compared to squared returns, which allows the models to obtain more actionable information from the data. Moreover, since the main difference between daily squared returns and realized variance is sampling frequency, it seems that high frequency data can significantly increase the forecasting performance of the models. Another aspect that can be derived from these results is that economic theory, which is the basis for the choice of HAR features, also has an impact on forecast performance. Therefore, it seems reasonable to use theory as a basis when constructing the models and applying them. This is especially true for the use of machine learning models, which are based on a data driven approach using mathematical optimization. Thus, rather than arbitrarily using obvious features for forecasting, it may be advantageous to select features for the models based on economic

Table 6.1.: One-step-ahead out-of-sample relative loss functions with squared return volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.0000000	1.0000000	1.0000000	0.0921794
HAR-RV	0.9433166	0.9263867	0.9533064	0.9432779	0.1619839
ARCH-TREE	2.8923213	1.2451736	1.1138064	2.8883200	0.0005169
HAR-TREE	0.9780221	0.9633501	1.0250176	0.9779871	0.1345477
ARCH-RF	1.0385940	1.0769980	1.0368481	1.0386092	0.0581089
HAR-RF	0.9297015	0.9188890	0.9722340	0.9296777	0.1667894
ARCH-SVR	1.1405302	1.1194445	1.1316190	1.1404561	0.0701698
HAR-SVR	0.9433943	0.8626342	0.9671994	0.9433967	0.1437387
Best Model	HAR-RF	HAR-SVR	HAR-RV	HAR-RF	HAR-RF

The table reports the one-step-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and use squared returns as a volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

theory.

Let us now take a closer look at the individual models from Table 6.1. We first analyze the results of the loss functions MSE and MAE as well as the out-of-sample R^2, which do not account for volatility underestimation. Here we find that according to MSE and R^2 the HAR-RF performs best, and MAE prefer the HAR-SVR. All these three evaluation criteria attest that machine learning models produce forecasting performance superior compared to that of classical models. However, it should be noted that the performance of the HAR-RV model is only slightly worse than that of the superior machine learning models. This good performance of the HAR-RV model is special because it has the simplest structure compared to all other models considered. Next in the analysis, we include the two loss functions that explicitly account for underestimation. We find that the HAR-RV model performs best according to the QLIKE loss function while the LINEX loss function prefers the HAR-RF. This shows that while the simple HAR-RV model performs well, the HAR-RF performs best overall, as it outperforms the other models on three of the five evaluation criteria (i.e., MSE, LINEX, and R2). However, the differences between the HAR-RV, HAR-RF and HAR-SVR models are marginal whereas

the mean forecast performance of the three models are significantly better than that of the GARCH(1,1) benchmark model.
The previous results from Table 6.1 use daily squared returns as a volatility proxy, which is a noisy proxy. We now turn to realized variance as a volatility proxy to check whether it yields similar results. Table 6.2 gives the one-step-ahead forecasts of all models considered relative to the GARCH(1,1) benchmark model, using realized variance as a volatility proxy. The results from Table 6.1 and Table 6.2 coincide that all ARCH-type machine learning models are inferior to the GARCH(1,1) model, but all other results in table 6.2 are starkly different. Although the out-of-sample R^2 values for the ARCH-type models are better and all the models can significantly reduce their loss function values compared to the previous table, the loss functions show that the models now perform even worse compared to the benachmark model. This is because the forecast performance of GARCH(1,1) has improved significantly and more strongly compared to the ARCH-type models using realized variance as a volatility proxy. Compared to values when using squared returns as a proxy the values of the loss functions of the GARCH(1,1) decreased for MSE by 82.17%, for MAE by 54.50%, for QLIKE by 86.47%, and for LINEX by 82.17%, respectively.
This significant reduction in the values of the loss functions for the benchmark model also has an impact on the competing models HAR-type models. In Table 6.1, HAR-RV, HAR-RF and HAR-SVR were superior to the benchmark model for all evaluation criteria. However, this is no longer true using the realized variance as a volatility proxy. In Table 6.2, an ambiguous picture emerges. The loss functions that do not account for underestimation display different results as MSE indicates that both HAR-RV and HAR-RF are inferior to GARCH(1,1), but MAE attests superiority of both models over the benchmark model. The comparison between HAR-RV and HAR-RF also shows a contrasting picture between the two loss functions. According to the MSE, HAR-RF is superior whereas the MAE prefers the HAR-RV model. A similar picture develops from the QLIKE and LINEX loss functions, which penalize underestimation. According to the QLIKE function, HAR-RV and HAR-RF both outperform the benchmark model, with HAR-RV model performing better than HAR-RF. The LINEX loss function, on the other hand, indicates the opposite, certifying that both models perform worse than the benchmark model. These results are due to the GARCH(1,1) model having a larger reduction in the values of MSE and LINEX compared to HAR-RV and HAR-RF, whereas the reductions are smaller for MAE and QLIKE loss functions. The discussions provided in Chapter 5 provide further theoretical explanations for these results. Since the LINEX

Table 6.2.: One-step-ahead out-of-sample relative loss functions with realized variance volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.0000000	1.0000000	1.0000000	0.5145068
HAR-RV	1.1382622	0.9226583	0.8064079	1.1381519	0.5208179
ARCH-TREE	11.8837402	1.7952916	2.0362305	11.8629296	0.0054381
HAR-TREE	1.2970916	1.0811412	1.1872757	1.2970207	0.4461177
ARCH-RF	1.3986991	1.2805308	1.4099873	1.3987172	0.3542689
HAR-RF	1.0685640	0.9085725	0.8715917	1.0684612	0.5322668
ARCH-SVR	2.2566581	1.5191301	5.3875931	2.2564963	0.2349250
HAR-SVR	0.9149897	0.7737198	0.8082095	0.9149865	0.5464619
Best Model	HAR-SVR	HAR-SVR	HAR-RV	HAR-SVR	HAR-SVR

The table reports the one-step-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and used the realized variance as volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

function is a more complex version of the MSE, both loss functions behave similarly to a reduction. More surprising are the results on the described robustness of the MSE and QLIKE loss functions according to Patton (2011). According to this theory, the robust loss functions should lead to the same model preference regardless of the choice of volatility proxy but this is only true for QLIKE but not for MSE.

Despite the ambiguous results, a clear picture emerges with respect to the HAR-SVR model. This model performs best on all evaluation criteria except for the QLIKE statistic. The values of the loss functions for HAR-SVR drop the most. Compared to values when using squared returns as a proxy the values of the loss functions of HAR-SVR decreased for MSE and LINEX by 82.71%, for MAE 59.1%, and for QLIKE 88.69%, respectively. Moreover, it is interesting to note that HAR-RF was the best performing model from Table 6.1, although it performed only marginally better than the HAR-SVR. Overall, this shows that the HAR-SVR model seems to be particularly suitable for one-step-ahead volatility forecasting.

It should be briefly noted at this point that the HAR-SVR model was used here with $\epsilon = 0$, as it was considered optimal based on the grid search. The empirical results now

confirm the assumption that the model does not overfit the data, but that there is a high underlying complexity. Furthermore, the good forecasting results support the conjecture that regularization approaches can improve the forecasting of volatility, as HAR-SVR with $\epsilon = 0$ is similar to a mean absolute deviation regression with L_2 penalization. Thus, a possible further research approach could be to optimize and benchmark the HAR-RV model with mean absolute deviations and regularization.
The previous results from Table 6.2 and 6.1 are all based on rolling-origin cross-validation with one-step-ahead forecasts. Next, we consider forecasting for multiple time steps because, in addition to the practical relevance of multistep forecasts, we also want to check whether the previous results are preserved and what influence the multi step forecast has on the model choice.

Three-step-ahead predictions

We first consider the $h = 3$ multistep forecast in Table 6.3 with squared returns as a volatility proxy. Here the ARCH-type machine learning models show superior performance compared to HAR-type machine learning methods.

Table 6.3.: Three-steps-ahead out-of-sample relative loss functions with squared return volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.000000	1.0000000	1.0000000	0.0828083
HAR-RV	1.0144227	1.212505	0.5975491	1.0142682	0.1069744
ARCH-TREE	2.7237657	1.477453	0.6698557	2.7198232	0.0000010
HAR-TREE	1.1137196	1.238442	0.6196070	1.1135439	0.0700342
ARCH-RF	0.9752723	1.289243	0.6233780	0.9752163	0.0630731
HAR-RF	1.0463854	1.200716	0.6064805	1.0462348	0.0860393
ARCH-SVR	1.0975261	1.329749	2.1529380	1.0973901	0.0599594
HAR-SVR	0.9840365	1.089762	0.6129426	0.9839441	0.0995796
Best Model	ARCH-RF	GARCH(1,1)	HAR-RV	ARCH-RF	HAR-RV

The table reports the three-steps-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and used the squared returns as volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

These results are in contrast with the results from the $h = 1$ step-ahead predictions, where the HAR-type machine learning models in particular show superior prediction performance. In this context, however, it is interesting to note that the simple HAR-RV model shows good prediction performance. This is surprising, since obviously the HAR-type features also seem to be suitable for direct three-step prediction, but the machine learning models cannot reflect this. The only overlap in the results between the one-step and three-steps evaluations of forecasting performance is that random forest performs well as a machine learning model. Nonetheless, the arguments from the one-step-ahead forecast evaluation from Tables 6.1 and 6.2 that higher signal-to-noise ratio and economic theory can improve forecast performance cannot be confirmed by the results from Table 6.3. Therefore, it cannot be determined from the previous three-steps forecasting analysis whether ARCH-type or HAR-type models are appropriate for three-step forecasting. Rather, the question arises as to why the HAR-type machine learning models are apparently unable to use the information that the HAR-RV model uses to achieve good forecasting performance. One possible reason may be the noisy volatility proxy. To investigate this conjecture, we consider five-minute realized variance as the volatility proxy in the evaluation.

Table 6.4 presents the findings and shows that the forecast performance results differ significantly from those in Table 6.3. It shows a reversed picture in that all ARCH-type machine learning models are inferior to the GARCH(1,1) benchmark model and the HAR-type models perform significantly better. The HAR-SVR model performs best for four out of five evaluation criteria. These results are very similar to the results from Table 6.2, and, therefore, appear to be more consistent compared to the conclusions based on the evaluation with the squared returns proxy. Thus, it is justified to assume that the arguments given to explain the superior forecast performance of the HAR-type models are valid, since a consistent picture emerges when using realized variance to evaluate one-step-ahead and three-steps-ahead forecasts. The use of realized variance as a volatility proxy leads to a disproportionate reduction in loss function values of the HAR-type models for the three-step-ahead forecasts. The results are particularly interesting with respect to the QLIKE and LINEX functions, which account for the underestimation of volatility. Here we clearly see that, on average, the HAR-type models significantly underestimate volatility less for the three-steps-ahead forecasts than the benchmark model.

Table 6.4.: Three-steps-ahead out-of-sample relative loss functions with realized variance volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.0000000	1.0000000	1.0000000	0.3480741
HAR-RV	0.9400281	1.1146282	0.2223225	0.9398359	0.4731413
ARCH-TREE	7.4811659	1.7439025	0.4204816	7.4672517	0.0037198
HAR-TREE	0.9787893	1.1378694	0.2547988	0.9785689	0.4753799
ARCH-RF	1.0274074	1.3046950	0.3098271	1.0273585	0.2560791
HAR-RF	0.8333681	1.0294754	0.2182369	0.8332023	0.5058117
ARCH-SVR	1.5277932	1.5321372	2.8790590	1.5275718	0.2030051
HAR-SVR	0.8149053	0.9161168	0.2177746	0.8147805	0.4502370
Best Model	HAR-SVR	HAR-SVR	HAR-SVR	HAR-SVR	HAR-RF

The table reports the three-steps-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and used the realized variance as volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

Five-step-ahead predictions

As a final forecasting step, we consider the five-steps-ahead forecast of volatility. We begin by evaluating model performance using the various loss functions and squared returns as a volatility proxy. Table 6.5 presents the corresponding results. Three of the five evaluation criteria indicate that a machine learning model is superior, and the other two criteria prefer the HAR-RV model and the GARCH(1,1) model once. However, it is notable that the ARCH-type random forest model performs particularly well and the results are generally very similar to the results from Table 6.3. Thus, when evaluating the models with squared returns as a volatility proxy, it is not evident whether ARCH-type or HAR-type features are better suited to five-step forecasting. Looking at MSE and MAE, it is unclear whether ARCH-RF or the GARCH(1,1) benchmark model performs better.

Turning to the results from the QLIKE and LINEX loss functions, which account for the dangers of underestimating volatility, we see that a machine learning model is superior, although the overall picture is mixed.

The QLIKE statistic indicates that, on average, the machine learning models do not

Table 6.5.: Five-steps-ahead out-of-sample relative loss functions with squared return volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.000000	1.0000000	1.0000000	0.0672432
HAR-RV	1.1144723	1.238475	0.6227791	1.1142619	0.0871241
ARCH-TREE	5.7165652	1.645766	0.6861945	5.7040704	0.0001981
HAR-TREE	1.0586127	1.199247	0.6362155	1.0584805	0.0774196
ARCH-RF	0.9804679	1.270656	0.6449600	0.9804222	0.0538403
HAR-RF	1.0815563	1.207418	0.6223103	1.0813710	0.0798813
ARCH-SVR	1.1000776	1.310149	0.7620742	1.0999535	0.0533527
HAR-SVR	1.1529672	1.110769	0.6402901	1.1527977	0.0484506
Best Model	ARCH-RF	GARCH(1,1)	HAR-RF	ARCH-RF	HAR-RV

The table reports the five-steps-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and used the squared returns as volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

underestimate volatility as much as the benchmark model, and in particular the good performance of the simple HAR-RV model must also be mentioned here. In contrast, the LINEX loss function indicates that of all the models considered, only the ARCH-RF model performs better than the benchmark model. Note that this result should be considered against the background that the LINEX loss function approaches the MSE loss function in the limiting case. On the whole, the results from Table 6.5 show a mixed picture, just as the evaluations using squared returns as a proxy did earlier. Therefore, in comparison we look at the evaluation of forecast performance using realized variance and see if, as before, this gives a more consistent picture of performance of the different models.

Table 6.6 shows the five-steps-ahead forecasts of the models with realized variance as a volatility proxy. Here we see that all superior models are machine learning models with HAR-type features. This is inconsistent with the previous results of the five-steps-ahead predictions using squared returns as a volatility proxy, they did not provide a clear picture of which features are suitable for prediction and whether machine learning models are superior or inferior. Nevertheless, the current results are consistent with

Table 6.6.: Five-steps-ahead out-of-sample relative loss functions with realized variance volatility proxy

	MSE	MAE	QLIKE	LINEX	R^2
GARCH(1,1)	1.0000000	1.0000000	1.0000000	1.0000000	0.2918195
HAR-RV	1.2421562	1.2241697	0.2626402	1.2418737	0.4225153
ARCH-TREE	18.7553299	2.0176781	0.4356751	18.7102896	0.0041078
HAR-TREE	0.9304624	1.1259193	0.2674193	0.9302774	0.4529715
ARCH-RF	1.0759533	1.3264220	0.3373528	1.0759131	0.2102532
HAR-RF	1.0907330	1.1478352	0.2529850	1.0905268	0.4157510
ARCH-SVR	1.4795954	1.5454435	0.5809830	1.4793872	0.2032242
HAR-SVR	1.0336128	0.9768818	0.2550898	1.0334172	0.4117495
Best Model	HAR-TREE	HAR-SVR	HAR-RF	HAR-TREE	HAR-TREE

The table reports the five-steps-ahead out-of-sample forecasts of the different models. The loss functions are relative to the GARCH(1,1) model and used the realized variance as volatility proxy. The last column describes the out-of-sample R^2 and is not specified relative to the GARCH(1,1) model.

the evaluations of one-step-ahead and three-steps-ahead predictions that used realized variance as a proxy. From Tables 6.2, 6.4, and 6.6, it appears that HAR-type features are the better choice compared to squared returns and that out of the 15 best models, 14 are HAR-type machine learning models.

The results differ, however, with respect to the best model. While HAR-SVR or HAR-RF were preferred in the one-step-ahead and three-step-ahead evaluations with realized variance, now the simple HAR-TREE is superior in the five-steps-ahead evaluation. This seems counterintuitive at first since the random forest model is an ensemble of tree models and the bias of a tree within the random forest equals the bias of the random forest. Yet, as Hastie et al. (2009) note, the trees within the random forest are restricted by the bootstrap and the number of random chosen regressors for each split m_{try}. Therefore, it is possible that an unpruned tree that is used outside the random forest may have a lower bias. With respect to the present case, we considered block lengths of the moving-block bootstrap procedure as a tuning parameter and identified an optimal value of five. This means that in bootstrap resampling, blocks of length five are each assembled to generate a bootstrap sample on which a tree is fitted for the random forest. The block lengths thus correspond exactly to the step size of the forecast and it is, therefore, conceivable that the correlation structure from the moving block bootstrap is not sufficient to perform such a forecast with reasonable accuracy. Moreover, there are only three HAR-type features in total, of which only two are used from the tuning process for randomly choosing features at the nodes. Since the bias increases with decreasing m_{try}, this may also have an impact on the prediction performance. Thus, a possible improvement could be to increase the block length and to include additional features in the model.

Discussion and Classification of Results

We now classify and discuss the results so far. The overall picture is mixed when analyzing the results from the one-step-ahead, three-steps-ahead, and five-steps-ahead evaluations using squared returns as a volatility proxy. Although the analysis shows that of the 15 model preferences emerging from the evaluation, a total of 11 machine learning models are indicated as superior and, of these, seven HAR-type machine learning models and four ARCH-type models are preferred. Moreover, of the total 15 superior models, nine are HAR-type models and only four are ARCH-type models. Thus, the evaluation with squared returns show that HAR-type features appear to be better suited for volatility forecasting than the ARCH-type features.
The picture becomes less straightforward when we look more closely at the individual model preferences and assume that the model considered superior is the one most often preferred by the loss functions. The results show that for the one-step-ahead forecast evaluated with squared returns, HAR-RF performs best. For the three-steps-ahead forecasts, no clear decision can be made, as both the ARCH-RF model and the simple HAR-RV model are each preferred twice by different loss functions. For the five-steps-ahead forecast, on the other hand, the ARCH-RF model is considered superior. We note that even though the ARCH-RF is preferred by the MSE and LINEX loss functions for the three-steps-ahead and five-steps-ahead forecasts, this can be problematic in model selection because, the LINEX function converges to the MSE in the limiting case of $a \to 0$. Examining the loss functions using squared returns across all prediction steps, we notice that ARCH-RF and HAR-RV models are each preferred twice by the evaluation criteria. Thus, no distinct picture emerges as to which model is superior across all forecast horizons. The ambiguous results from the evaluation using squared returns as a volatility proxies only show that the HAR-type models are preferred over the ARCH-

type models and that the random forest model appears to be the most appropriate for one-step-ahead forecasting. Further conclusions cannot be drawn from the results. One possible reason for the mixed results may be that squared returns are a noisy volatility proxy.

In the empirical analysis, we therefore also considered realized variance as a volatility proxy for all three forecasting steps. The empirical analysis with realized variance shows a consistent picture. Of the 15 evaluations, machine learning models are superior in 14 cases and the simple HAR-RV model only once for all forecast steps and all loss functions considered. Furthermore, all superior models use HAR-type features, suggesting that these features are better suited for volatility forecasting than ARCH-type features. Looking more closely at the individual model evaluation results, we find that for the one-step-ahead and three-step-ahead forecasts, the HAR-SVR model performs best. In both cases, the model is preferred by four out of five loss functions. Only for five-steps-ahead forecasts does HAR-TREE perform best. Based on the loss functions across all forecast horizons, the HAR-SVR model appears to be the most appropriate.

Generally, evaluating the models across the different forecast steps using realized variance shows a more consistent picture than model evaluation using squared returns as a volatility proxy. However, there are also overlaps in the results. Model performance evaluation using both volatility proxies show that in both cases machine learning models are superior to the benchmark model and also to the simple HAR-RV model. In addition, both analyses show that the HAR-type features tend to be better at producing accurate volatility forecasts. Despite these similarities, the evaluations differ significantly when it comes to choosing the best model. In the evaluations, out of a total of 30 model preferences from the loss functions, only three times are the same models considered optimal across all forecast horizons. The observation that the loss functions with the two volatility proxies arrive at very different model preferences is particularly surprising for the MSE and QLIKE functions. We choose the MSE and the QLIKE loss functions as evaluation criteria because they fulfill the robustness property according to Patton (2011) and, thus, should lead to the same model choice for the two different volatility proxies. In the case of the LINEX loss function, one would also expect very similar model preference results due to its similarity to the MSE. Despite this, we find that for the three forecast steps considered, the QLIKE function generates an identical model preference twice for both volatility proxies and the MAE comes to an identical assessment once. All other criteria prefer different models.

The question now arises as to which results are more reliable. To answer this question, we are helped by the results from Section 2.3, where we examined the properties of volatility proxies and showed that squared returns is a noisy proxy. In addition, we also examined the work of Hansen and Lunde (2006a), in which the authors point out that model comparison and choice based on squared returns is very likely to favor an inferior model. Therefore, the authors argue for the use of realized variance as an alternative volatility proxy. The results of by Liu et al. (2015), which we considered in detail in Section 2.3, also argue in favor of realized variance as a volatility proxy. The authors showed that among 400 different volatility proxies there is hardly any evidence that one of the proxies systematically outperforms the realized variance on a five-minute basis. Overall, therefore, evaluation results based on realized variance should be preferred in model selection. Based on our results, we argue that the HAR-type features and models are better suited than the ARCH-type ones in accurately forecasting volatility. The HAR-type features differ from the ARCH-type features in two major ways. First, realized variance is calculated based on high frequency data and, second, economic theory is used to determine HAR-type features. Since the HAR-type machine learning models are convincing in volatility forecasting, it seems that high frequency data and economic theory not only bring advantages to classical statistical methods like the HAR-RV model, but machine learning models can also use information in the features and economic theory for forecasting financial market volatility. The results also show that the HAR-SVR model is particularly suitable for one-step-ahead and three-steps-ahead forecasts. It is interesting to note that the HAR-SVR model is fitted with $\epsilon = 0$ and is therefore similar to a mean absolute deviation regression with L_2 regularization. It would be interesting for future research to use this result and fit the HAR-RV model as a mean absolute deviation regression with L_2 penalization. Further, the results show that the simple HAR-TREE performs best for five-steps-ahead forecasts. These results seem contradictory at first, but are consistent with the findings of many studies that simpler models often perform better for large multistep forecasts.[2]

6.2.2. Varying the Forecast Horizon

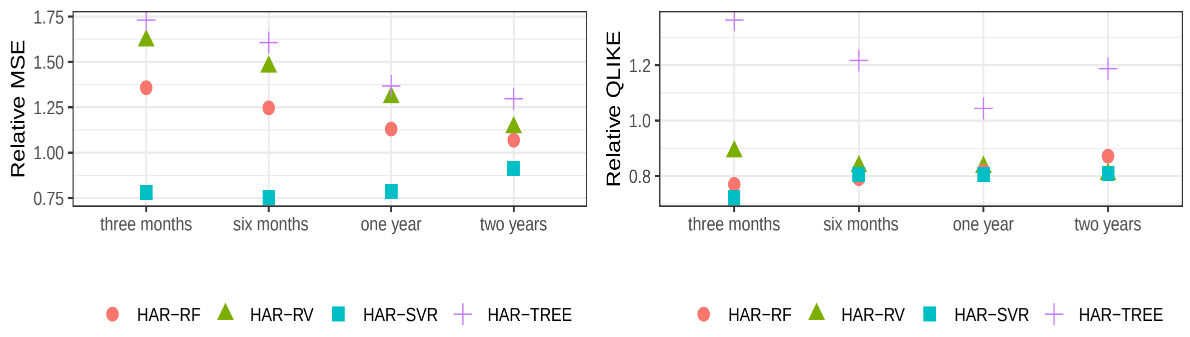
In the primary analysis in the previous section, we considered the one-step-ahead, three-steps-ahead, and five-steps-ahead forecasts on the whole test data, which corresponds to a time horizon of 570 days. Now, to check whether the results change or stay the same, we vary length of the test data set or forecast horizon for all h-step-ahead forecasts with $h \in$

[2]See Poon and Granger (2003).

$\{1, 3, 5\}$. For this robustness check, we consider four horizons: three months (66 days), six months (132 days), one year (264 days), and about two years (570 days), specifying a trading month with 22 days. In addition, this approach allows us to determine the influence of forecast horizon on the model choice. Note that we use the results from the primary analysis and perform the variation of the forecast horizon only for the HAR-type models. Furthermore, we evaluate the forecast horizon variation results using only the MSE and the QLIKE loss functions.

One-step-ahead predictions and forecast horizon variations

We first consider the one-step-ahead forecasts of the HAR-type models relative to the GARCH benchmark model, as shown in Figure 6.3.
Figure 6.3a shows that, with respect to MSE for all forecast horizons, only the HAR-SVR model performs better than the benchmark model and that the differences between the models become smaller as the length of the test data increases. On the whole, it can be concluded that the HAR-SVR model is suitable for long and short forecast horizons for volatility forecasts. Moreover, the model ranking remains consistent across forecast horizons and essentially mirrors the results from the primary analysis.



(a) MSE relative to GARCH model over different horizons

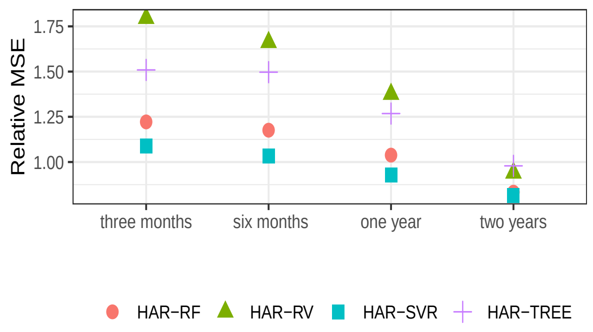
(b) QLIKE relative to GARCH model over different horizons

Figure 6.3.: The plots show the MSE and QLIKE loss functions relative to the GARCH model for one-step-ahead predictions over different lengths of test data. A point below one indicates that the model performs better than the GARCH benchmark model.

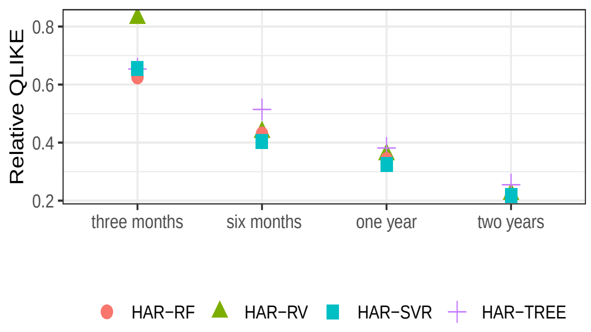
Figure 6.3b reflects the varying forecast horizons considering the underestimation of volatility. Here we see that over all forecast horizons only HAR-TREE performs worse than the benchmark model. On average, all other models underestimate volatility significantly less than the GARCH model. There are only minor differences in the value of the QLIKE statistic between HAR-RV, HAR-RF, and HAR-SVR across all forecast horizons and the results appear to be robust. All in all, these results confirm the primary analysis and support the argument that the HAR-SVR model is preferred for one-step-ahead forecasting over all test data lengths considered.

Three-step-ahead predictions and forecast horizon variations

Next, we consider the three-steps-ahead predictions with varying test data length in Figure 6.4. First, we notice that the two figures show similar behavior of the loss functions, as the values of these converge with increasing test data length. Yet, the structure in the two figures also indicates that the results may be less consistent.
Let us first look at the relative MSE in Figure 6.4a. Here we see that none of the models outperforms the benchmark model for the forecast horizons of three and six months. Starting from a forecast horizon of six months, as the forecast horizon increases, the performance of the HAR-SVR is almost identical to that of the GARCH and, for even longer horizons, the HAR-SVR performs still better. The remaining models, in contrast, only outperform the GARCH model for the longest forecast horizon of 570 days, with HAR-RF performing significantly better than the benchmark model and HAR-RV and HAR-TREE performing only marginally better.



(a) MSE relative to GARCH model over different horizons



(b) QLIKE relative to GARCH model over different horizons

Figure 6.4.: The plots show the MSE and QLIKE loss functions relative to the GARCH model for the three-steps-ahead predictions over different lengths of test data. A point below one indicates that the model performs better than the GARCH benchmark model.

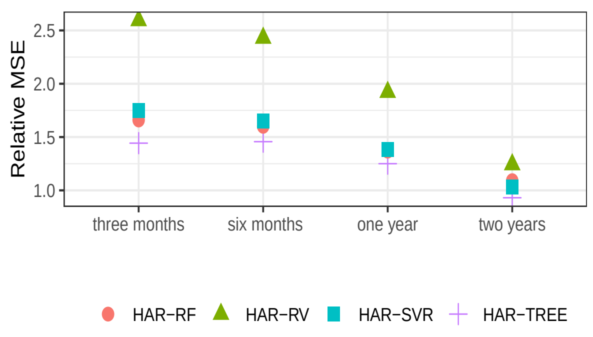
In Figure 6.4b, we consider the QLIKE loss function over the different forecast horizons

to account for the risks of underestimating volatility. Here we find that all the models considered perform better than the GARCH benchmark model over the varying time horizons. Here, HAR-SVR performs best and HAR-RV performs worst among the superior models. The loss function values for the different models are very similar across all time horizons and become smaller relative to the benchmark model as the horizon increases. This suggests that the GARCH model systematically underestimates volatility and, therefore, appears less well suited to forecast highly volatile phases.
In general, we find that MSE prefers the GARCH model for the three and six months forecast horizon in the three-steps-ahead analysis, and that the HAR-SVR model performs better from a test data length of greater than six months. When we account for the dangers of underestimating volatility by using the QLIKE function, we see that the HAR-SVR model is also preferred for shorter forecast horizons.

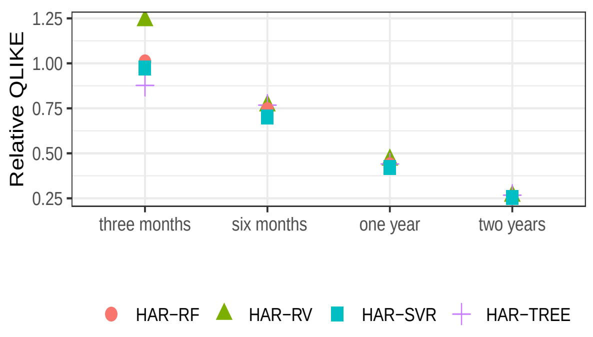
Five-step-ahead predictions and forecast horizon variations

Last, we consider the five-steps-ahead forecast performance of the models with varying forecast horizons in Figure 6.5. Here we see essentially the same structure as in the analysis of three-steps-ahead forecasts with varying forecast horizons, as the values of the loss functions converge with increasing test data length. However, the model preferences now differ significantly from the previous analysis. Figure 6.5a shows that none of the models considered outperforms the benchmark model up to a forecast horizon of two years. Despite the poor general performance over shorter horizons across all models, HAR-TREE performs relatively best, HAR-RV model performs relatively worst, and the performance of HAR-SVR and HAR-RF are almost identical. From a forecast horizon of two years, HAR-TREE then outperforms the benchmark model.
To make a statement about the general performance for the five-step-ahead forecasts and varying forecast horizon, we further consider the QLIKE statistic in Figure 6.5b which accounts for problems due to volatility underestimation. Again, we see that for an increasing forecast horizon, the values of the loss function converge. Interestingly, only HAR-TREE performs better than the GARCH benchmark model for a horizon of three months. All other models except HAR-RV show similar performance to the benchmark. For a forecast horizon of six months and longer, all models perform better than the benchmark model, although the differences between the models appear to be only marginal.
A clear statement on which model across varying forecast horizons for the five-steps-ahead forecasts is more difficult to assess in this case. This is because the GARCH

benchmark model for MSE performs better for three of the four horizons considered, but this is no longer true for the QLIKE statistics. Even so, it seems overall reasonable to choose the GARCH model for very short forecast horizons and direct five-step forecasts, as it performs better, on average, with respect to MSE and has a similar performance with respect to the QLIKE function. For medium and longer forecast horizons, on the other hand, HAR-TREE appears to be a better fit because, although the model's MSE is slightly worse for medium horizons, it avoids underestimation more often, on average, than the bechmark model. By and large, this interpretation of the results supports the findings of the primary analysis and shows a robust picture for medium and longer forecast horizons.



(a) MSE relative to GARCH model over different horizons



(b) QLIKE relative to GARCH model over different horizons

Figure 6.5.: The plots show the MSE and QLIKE loss functions relative to the GARCH model for the five-steps-ahead predictions over different lengths of test data. A point below one indicates that the model performs better than the GARCH benchmark model.

Discussion and Classification of Results

The robustness analysis in this subsection shows that for the one-step-ahead and three-steps-ahead forecasts, the HAR-SVR model performs best for all forecast horizons considered. In addition, the analysis shows that when the volatility underestimation is accounted for, HAR-SVR, HAR-RF, and HAR-RV perform significantly better than the benchmark model. This suggests that, on average, these models underestimate volatility less frequently than the GARCH model and are, therefore, better suited to forecast highly volatile phases. In general, the results from the robustness test confirm the results

from the primary analysis.
The findings from the five-steps-ahead forecast with different test data lengths show a somewhat less consistent picture than the results from the primary analysis. Here it turns out that for shorter forecast horizons, the GARCH benchmark model may be the most suitable model, as it performs significantly better for MSE than any of the other models and there is little difference in the QLIKE function. For intermediate and longer forecast horizons, on the other hand, HAR-TREE appears to be the most appropriate. For medium horizons, the MSE loss function of the HAR-TREE model is slightly higher than that of the benchmark model, but the model underestimates volatility less on average. For longer horizons, however, HAR-TREE performs better for both loss functions and should, therefore, be preferred.
Taken together, the analysis shows that there is a possible relationship between forecast horizon and multi-step direct forecasts. The results suggest that for one-step-ahead and three-step-ahead forecasts, the HAR-SVR model is generally superior. For five-step-ahead forecasts, on the other hand, the forecast horizon plays an important role which should be taken into consideration, because for shorter horizons the GARCH model is superior, whereas for intermediate and longer horizons HAR-TREE is superior.

7. Conclusion

The aim of this paper was to examine whether there is a theoretical and empirical basis to successfully use machine learning models in the time series domain for forecasting the volatility of the DAX. To achieve this goal, we first analyzed and presented the most important theoretical foundations. We started by studying the variable of interest, volatility. In the process, we saw that volatility is a latent variable that needs to be approximated and that there are various ways to do so. For this paper, we considered two of the best-known volatility proxies, daily squared returns and realized variance. The analysis of the proxies in Section 2 showed that the choice of volatility proxy has a significant impact on model evaluation and ranking. It was found that squared returns is a very noisy proxy that deviates from the true value of the variance by more than 50% in 74% of the time and, therefore, can lead to erroneous model comparisons and model rankings. The analysis of the second proxy, realized variance, showed that it has more desirable statistical properties than squared returns and that the empirical approximation performance calculated on a five-minute frequency has the best performance among many volatility proxies and leads to consistent model comparison and model ranking.
Another step toward the paper's goal was to answer the question of whether there is a theoretical basis for using machine learning models for time series to forecast volatility. First, we focused on the different data structure of time series and the data format for supervised machine learning models. We showed that the intuitive idea of putting a time series into such a format can be justified theoretically and that it is possible to use it for learning and forecasting with ML methods. We derived the theoretical justification for this from Takens' Embedding Theorem. This confirms that it is possible, under certain conditions, to reconstruct an equivalent state space containing the dynamics that generate the time series. This result, in turn, implies that there is a relation between a finite window of a time series and the state of the dynamic system generating the series. Taken together, this led us to put the time series into a supervised learning format.
In the second step, we addressed the question of whether there is a statistical learning theory for time series in general and, more specifically, for volatility processes that gives

us clues as to whether it is possible to derive learning guarantees from it and justify the use of machine learning models for volatility forecasting. We found that GARCH processes exhibit β-mixing properties and that learning guarantees exist for stationary and nonstationary cases. Based on these results, we argued that the use of machine learning in the context of time series is justified from a theoretical point of view. Since volatility processes such as the GARCH process have the required properties to obtain learning guarantees, we concluded that forecasting volatility with machine learning models seems quite promising.
To empirically test whether machine learning models are suitable for successfully performing volatility forecasting, we conducted a study comparing various regression trees, random forests, and support vector regressions to classical models such as the HAR-RV model and the GARCH model. The machine learning models were used in two variants with different features to test whether economic theory can be used to improve forecasts and to evaluate the impact of high frequency data on forecast performance. The first variant of machine learning models was the ARCH-type models, which used only the past squared returns to perform forecasts. The second variant, the HAR-type models, used high frequency data and the Heterogeneous Market Hypothesis to derive features based on realized variance using an economic theory.
In our study, we also considered other factors that are important for forecasting volatility. First, we examined one-step-ahead forecasts, as well as direct three-steps-ahead and five-steps-ahead forecasts of all models and varied the forecast horizon between three months, six months, one year, and two years. In addition, we used special loss functions to account for the risks of underestimating volatility. These loss functions also allowed us to compare the models in terms of their forecasting performance of highly volatile periods. We performed the evaluations of the models and the model comparison for all described factors for both volatility proxies.
The empirical analysis showed that the different volatility proxies indicate different superior models. The model evaluation with squared returns showed that the machine learning models performed better. In addition, while we found that the HAR-type models perform better than the ARCH-type models, a closer look at the model comparison with squared returns also showed that no specific model performed best across all forecast steps. For the one-step-ahead forecast, HAR-RF performed best, and for the five-steps-ahead forecast, ARCH-RF performs best. The three-steps-ahead forecast gave a mixed picture, as here ARCH-RF and HAR-RV were equally preferred by the loss functions. Overall, the loss functions together with the squared returns as a volatility

proxy most often showed the random forest model to be superior.
In contrast, the empirical evaluation with realized variance, showed more consistent results. Here, we found the machine learning models outperformed the GARCH and HAR-RV models across all forecast horizons. We also showed that the HAR-type models perform better than the ARCH-type models when using realized variance as the volatility proxy. In fact, a closer look at the individual forecast horizons showed that for the one-step-ahead and three-steps-ahead forecasts, the HAR-SVR model outperformed all other models. Only for the five-steps-ahead forecast did the HAR-TREE model perform better.
All in all, we inferred from the results of the empirical analysis that the HAR-type features perform better. This suggests that high frequency data and economic theory can be used to improve the prediction performance not only for classical models, but also for machine learning models. Since many of the arguments given support realized variance as a superior volatility proxy, we concluded that the HAR-SVR model performs best for short-term forecasts such as one-step-ahead and three-steps-ahead while the simpler HAR-TREE model tends to perform better for longer term forecasts.
To corroborate the results from the primary analysis, we performed a robustness check by looking at model performance for different forecast horizons. We used results from the primary analysis and evaluated the well-performing HAR-type models over different forecast horizons. The robustness check confirmed the results from the primary analysis, as HAR-SVR performed best for the one-step-ahead and three-steps-ahead forecasts over the different forecast horizons. Only for the five-steps-ahead forecast over the various forecast horizons does the picture differ somewhat from that in the primary analysis. The GARCH model was found to be superior for shorter forecast horizons and the HAR-TREE model performed better for medium and longer forecast horizons. Another result from the robustness analysis of the different forecast horizons showed that the preferred models, on average, underestimate volatility less than the GARCH benchmark model for all forecast steps considered and are therefore better suited to forecast highly volatile phases.

7.1. Limitations and Further Research

Volatility forecasting continues to be a difficult task. In this paper, we showed that it is possible to obtain a significant improvement in volatility forecasting over classical models using well-known and easy-to-implement machine learning models. However, we obtained these results only for the DAX and it is conceivable that these results do not hold in other markets. Therefore, it seems reasonable to check whether these results hold in other markets in future research. In doing so, markets comparable to the DAX should be considered to be able to compare the results.

In addition, we used only three different machine learning approaches for the volatility forecast in this paper. The use of these three machine learning models in our paper is because we had relatively little data available to apply other, possibly more complex methods. More complex methods often require significantly more data to obtain good performance from the models. Therefore, much more data should be collected to check if the performance of our models can be improved and if there are other methods which perform even better.

The results obtained in this paper are mainly focused on pure autoregressive modeling, which permits us to say that a relatively simple data structure still allows machine learning models to perform significantly better than the classical volatility models. This is especially true for HAR-type models whose features are generated based on economic theory. In future research, it would be interesting to explore if there are other financial market theories that would generate additional features to further improve forecasting performance. In general, it would be interesting to find out if the additional inclusion of features can further improve the results obtained here.

Another aspect that may provide new insights is the use of machine learning models to generate features. Here, a classification procedure could be used to predict the direction of volatility movement. This could be achieved, for example, through binary classification, by trying to forecast the change in return and additionally including this forecast as a feature in the volatility models.

A final interesting avenue for future investigation concerns the adaptation and improvement of existing methods to use machine learning methods more efficiently for time series in general. This idea goes back to the fact that in this work we used the moving block bootstrap instead of the classical bootstrap to be able to use the random forest model for time series. This method does not seem to be optimal, since it is only designed for stationary series, for example, and the series should have a smooth course. The adaptation or improvement of these procedures can possibly help to consider more complex time series and to optimize existing models for use.

A. Appendix

A.1. Example of Natural Filtration of an Stochastic Process

Let a probability space be given by $\Omega = \{1,2,3\}, \mathcal{A} = \mathcal{P}(\{1,2,3\}), \mathbb{P}(\omega)) = \frac{1}{3}$ for all $\omega \in \Omega$. Furthermore, let a continuous stochastic process $\{X(t)\}_{t\in\mathbb{T}}$ be defined by $X(t,\omega) = \max\{t-\omega, 0\}$. Then the natural filtration of the process is given as

$$\mathcal{I} = \begin{cases} \{\emptyset, \Omega\} & \text{for } t \in [0,1] \\ \{\emptyset, \Omega, \{1\}, \{2,3\}\} & \text{for } t \in (1,2] \\ \mathcal{P}(\{1,2,3\}) & \text{for } t > 2. \end{cases}$$

To check this, only the definitions of natural filtration and the measurability condition have to be used. So it must be shown that process $X(t)$ is $\mathcal{A}_t$-measurable for all t .
This is easily shown. For $t \in [0,1]$, $X(t) = 0$ holds, which becomes clear by substituting for all $\omega \in \Omega$. Thus $X^{-1}(t, \{0\}\}) = \Omega$. The smallest σ-algebra containing Ω and, for which $X(t)$ is measurable, is $\{\emptyset, \Omega\}$.
Similarly, for $t \in (1,2]$, we obtain $X(t) = \max\{t-1, 0\}$ for possible values of $X(t)$. Thus $X^{-1}(t, \{0\}) = \{1,2\}$ and $X^{-1}(t, \{t-1\}) = \{1\}$ holds. It follows that the smallest σ-algebra for which $X(t)$ is measurable and contains the pre-images is $\{\emptyset, \Omega, \{1\}, \{2,3\}\}$.
For $t > 2$, the range is $X(t) = \max\{t-1, t-2, 0\}$ and the smallest σ-algebra for which $X(t)$ is measurable must contain the sets $\{1\}, \{2\}$ and $\{3\}$. In the present case, this is $\mathcal{P}(\{1,2,3\})$.

A.2. Random Walk as Martingale

Let $\{X_t\}_{t\in\mathbb{T}}$ be a sequence of independent random variables with $\mathbb{P}(X_t = 1) = \frac{1}{2} = \mathbb{P}(X_t = -1)$ and $\mathcal{I}_t = \sigma(\{X_s\}_{s\leq t})$. Then $R_t = X_1 + \cdots + X_t$ is called a random walk. To show that $\{R_t\}$ is a martingale, $\mathbb{E}[R_{t+1}|\mathcal{I}_t] = R_t$ must hold. So we have

$$\mathbb{E}[R_{t+1}|\mathcal{I}_t] = \mathbb{E}[R_t + X_{t+1}|\mathcal{I}_t] = \mathbb{E}[R_t|\mathcal{I}_t] + \mathbb{E}[X_{t+1}|\mathcal{I}_t] = R_t.$$

The first equality follows from $R_{t+1} = X_1 + \dots X_t + X_{t+1} = R_t + X_{t+1}$. The second equality is due to the additivity of conditional expectation. The last equality follows from the fact that X_{t+1} is independent of $\mathcal{I}_t$ and therefore it holds $\mathbb{E}[X_{t+1}|\mathcal{I}_t] = \mathbb{E}[X_{t+1}] = 0$ and R_t is $\mathcal{I}_t$ measurable and, therefore, $\mathbb{E}[R_t|\mathcal{I}_t] = R_t$ from which it follows overall that random walk $\{R_t\}$ is a martingale.

A.3. Proof of Equation (2.25)

The proof presented here essentially goes back to Shreve et al. (2004) and Hassler (2007). We assume that the return process can be defined as an Ito integral of the form

$$r(0,t) = \int_0^t \sigma(s)\mathrm{d}W(s).$$

where $\{\sigma(s)\}$ is a simple stochastic process adapted to filtration $\{\mathcal{A}_t\}_{t\in\mathbb{T}}$ associated with Brownian motion $\{W(t)\}_{t\geq 0}$. This means that, for partitioning $P_n([0,t]) : 0 = s_o < s_1 < \cdots < s_n = t$, $\sigma(s)$ is constant on each subinterval $[s_k, s_{k+1})$ and is $\mathcal{A}_t$-measurable for $t \geq 0$. For an arbitrary subinterval $[s_k, s_{k+1})$ of partitioning Ito integral can then be defined by the Ito sum as

$$\mathbb{I}(t) = \sum_{j=0}^{k-1} \sigma(s_j)(W(s_{j+1}) - W(s_j)) + \sigma(s_k)(W(s) - W(s_k)).$$

Then the quadratic variation of the return process up to time t is given by

$$QV(0,t) = \int_0^t \sigma^2(u)\mathrm{d}u$$

Note: The reason why we give the proof for only a simple stochastic process is because, loosely speaking, the result is also valid for general stochastic processes. Similar to the construction of the Lebesgue integral, we can define the limit of a sequence of simple stochastic processes with some properties as an approximation of a general stochastic process. So $\lim_{n\to\infty} \sigma_n(s) = \sigma(s)$ holds and the results can thus be generalized. Moreover, the proof here is only meant to provide a basic understanding.

Proof:
We first consider the quadratic variation on arbitrary subinterval $[s_j, s_{j+1})$ from partitioning $P_n([0,t]) = \{s_0, s_1, \dots, s_n\}$ with $0 = s_0 < s_1 < \cdots < s_n = t$. We further decompose the subinterval $[s_j, s_{j+1})$ into m subintervals such that $[s_j, s_{j+1}) = [k_0, k_1) \cup [k_1, k_2) \cup \cdots \cup [k_{m-1}, k_m)$ with $s_j = k_0 < k_1 < \cdots < k_m = s_{j+1}$. Then, we can write the quadratic variation of the subinterval $[s_j, s_{j+1})$ with the help of the

definition of the Ito integral as

$$\begin{aligned}
QV_m(s_j, s_{j+1}) &= \sum_{i=0}^{m-1}[\mathbb{I}(k_{i+1}) - \mathbb{I}(k_i)]^2 \\
&= \sum_{i=0}^{m-1}[\sigma(s_j)(W(k_{i+1}) - W(k_i))]^2 \\
&= \sum_{i=0}^{m-1}\sigma^2(s_j)[W(k_{i+1}) - W(k_i)]^2 \\
&= \sigma^2(s_j)\sum_{i=0}^{m-1}[W(k_{i+1}) - W(k_i)]^2
\end{aligned}$$

Now, we have to consider the sum $\sum_{i=0}^{m-1}[W(k_{i+1}) - W(k_i)]^2$ which is the quadratic variation of the standard Brownian motion over interval $[s_j, s_{j+1})$. As Hassler (2007) describes, for the quadratic variation of a standard Brownian motion over the interval $[s_j, s_{j+1})$, it holds that $QV_m(s_j, s_j+1) \xrightarrow{m.s} (s_{j+1} - s_j) = QV(s_j, s_{j+1})$. We will now prove this fact, but only for convergence in probability, i.e., $QV_m(s_j, s_{j+1}) \xrightarrow{\mathbb{P}} (s_{j+1} - s_j) = QV(s_j, s_{j+1})$. For this purpose, we first consider the expected value of the quadratic variation, which is determined as

$$\begin{aligned}
\mathbb{E}[QV_m(s_j, s_{j+1})] &= \sum_{i=0}^{m-1}\mathbb{E}[(W(k_{i+1}) - W(k_i))^2] \\
&= \sum_{i=0}^{m-1} var[(W(k_{i+1}) - W(k_i))] \\
&= \sum_{i=0}^{m-1}(k_{i+1} - k_i) \\
&= k_m - k_0 \\
&= s_{j+1} - s_j.
\end{aligned}$$

Further, we determine the variance of the quadratic variation by

$$\begin{aligned}
var[QV_m(s_j, s_{j+1})] &= \sum_{i=0}^{m-1} var[(W(k_{i+1}) - W(k_i))^2] \\
&= \sum_{i=0}^{m-1} \{\mathbb{E}[(W(k_{i+1}) - W(k_i))^4] - \mathbb{E}[(W(k_{i+1}) - W(k_i))^2]^2\} \\
&= \sum_{i=0}^{m-1} \{3(k_{i+1} - k_i)^2 - (k_{i+1} - k_i)^2\} \\
&= 2\sum_{i=0}^{m-1} (k_{i+1} - k_i)^2 \\
&\leq 2 \max_{0 \leq i \leq m-1} [(k_{i+1} - k_i)] \sum_{i=0}^{m-1} (k_{i+1} - k_i) \\
&= 2 \max_{0 \leq i \leq m-1} [(k_{i+1} - k_i)](s_{j+1} - s_j) \\
&= \overset{m\to\infty}{\longrightarrow} 0.
\end{aligned}$$

Here, the first equality uses independence of the increments of the Brownian motion, the second equality follows from the variance shift, and the third equality comes from the fact that kurtosis of a normal distributed random variable with zero expectation equals to $3\sigma^2$, which in the present case can be translated to $3(k_{i+1} - k_i)^2$. The last equality is valid because for $m \to \infty$ the partitioning of the subintervals becomes finer and the lengths of the intervals become smaller so that in the limiting case the maximum step size approaches zero. Now we can use Chebyshev's inequality and it follows that

$$\begin{aligned}
\mathbb{P}(|QV_m(s_j, s_{j+1}) - (s_{j+1} - s_j)| > \epsilon) &\leq \frac{2 \max_{0 \leq i \leq m-1} [(k_{i+1} - k_i)](s_{j+1} - s_j)}{\epsilon^2} \\
&= \overset{m\to\infty}{\longrightarrow} 0
\end{aligned}$$

which means that the quadratic variation of the standard Brownian motion converges in probability to $(s_{j+1} - s_j)$, i.e. $QV_m(s_j, s_{j+1}) \xrightarrow{\mathbb{P}} (s_{j+1} - s_j)$. Using this result, we get for $m \to \infty$

$$\begin{aligned}
QV(s_j, s_{j+1}) &= \sigma^2(s_j)(s_{j+1} - s_j) \\
&= \int_{s_j}^{s_{j+1}} \sigma^2(u) \mathrm{d}u.
\end{aligned}$$

Now, we can apply the same procedure over all n subintervals of $[0, t]$, i.e., add up the quadratic variation just determined and get

$$\begin{aligned} QV(0,t) &= \sum_{j=0}^{n-1} \int_{s_j}^{s_{j+1}} \sigma^2(u)\mathrm{d}u \\ &= \int_0^t \sigma^2(u)\mathrm{d}u. \end{aligned} \qquad \square$$

A.4. Derivation of Equation (3.9) for a GARCH(1,1)

We now consider a stationary GARCH(1,1) model, which is the most widely used specification of these models. This can be written as

$$\sigma_t^2 = \alpha_0 + \alpha_1 r_{t-1}^2 + \beta_1 \sigma_{t-1}^2.$$

Iterating backwards and using $\sigma_{t-1}^2 = \alpha_0 + \alpha_1 r_{t-2}^2 + \beta_1 \sigma_{t-2}^2$, we get

$$\begin{aligned}\sigma_t^2 &= \alpha_0 + \alpha_1 r_{t-1}^2 + \beta_1 \sigma_{t-1}^2 \\ &= \alpha_0 + \alpha_1 r_{t-1}^2 + \beta_1(\alpha_0 + \alpha_1 r_{t-2}^2 + \beta_1 \sigma_{t-2}^2) \\ &= \alpha_0(1+\beta_1) + \alpha_1 r_{t-1}^2 + \beta_1 \alpha_1 r_{t-2}^2 + \beta_1^2 \sigma_{t-2}^2.\end{aligned}$$

We can iterate further back and for $\sigma_{t-2}^2 = \alpha_0 + \alpha_1 r_{t-3}^2 + \beta_1 \sigma_{t-3}^2$, we obtain

$$\begin{aligned}\sigma_t^2 &= \alpha_0(1+\beta_1+\beta_1^2) + \alpha_1 r_{t-1}^2 + \beta_1 \alpha_1 r_{t-2}^2 + \beta_1^2 \alpha_1 r_{t-3}^2 + \beta_1^3 \sigma_{t-3}^2 \\ &= \alpha_0(1+\beta_1+\beta_1^2) + \alpha_1(r_{t-1}^2 + \beta_1 r_{t-2}^2 + \beta_1^2 r_{t-3}^2) + \beta_1^2 \sigma_{t-3}^2\end{aligned}$$

We can further perform the backward iteration for σ_{t-i}^2 for $i > 2$ and thus obtain

$$\begin{aligned}\sigma_t^2 &= \alpha_0(1+\beta_1+\beta_1^2+\beta_1^3+\dots) + \alpha_1 \sum_{i=1}^{\infty} \beta_1^{i-1} r_{t-i}^2 \\ &= \alpha_0 \sum_{i=0}^{\infty} \beta_1^i + \alpha_1 \sum_{i=1}^{\infty} \beta_1^{i-1} r_{t-i}^2 \\ &= \frac{\alpha_0}{1-\beta_1} + \alpha_1 \sum_{i=1}^{\infty} \beta_1^{i-1} r_{t-i}^2.\end{aligned}$$

From the assumption of stationarity with $\alpha_1 + \beta_1 < 1$, it follows that the geometric series $\sum_{i=0}^{\infty} \beta_1^i$ converges to $\frac{1}{1-\beta_1}$. This gives us an ARCH(∞) representation of the GARCH(1,1) process. In the general notation from equation (3.9), we have $\nu_0 = \frac{\alpha_0}{1-\beta_1}$, and $\nu_i = \alpha_1 \beta_1^{i-1}$.

A.5. Takens' Embedding Theorem

To describe Takens' theorem graphically we follow the idea of Jemwa (2003) and the description of Garcia (2022). We consider the *Henon* map and show that the representation (4.1) is admissible. This map describes a dynamical system in discrete time which exhibits chaotic behavior. The Henon map can be described by the equations

$$\begin{aligned} x_t &= 1 - a x_{t-1}^2 + y_{t-1} \\ y_{t-1} &= b x_{t-1}. \end{aligned}$$

Based on these equations, we uniformly sample a sequence of values to generate a Henon time series which can be seen in the following figure.

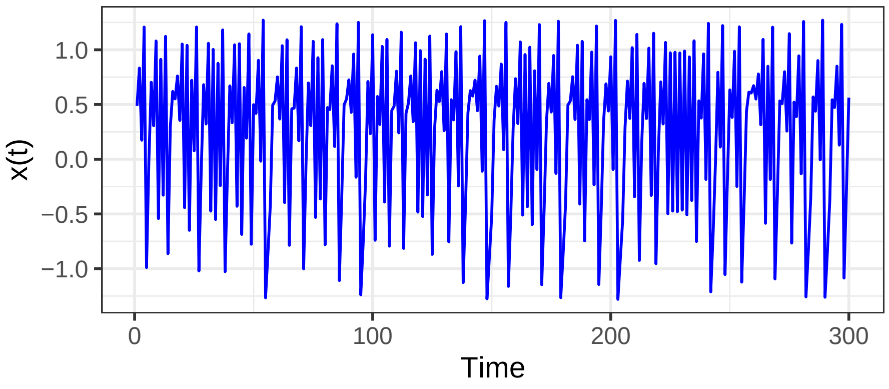


Figure A.1.: Henon Series with coefficients $a = 1.4$ and $b = 0.3$

This figure shows a time series with data in a familiar format. This data can now be converted into a matrix form with embedding vectors from Takens' theorem.

$$\mathbf{x_t} = \begin{pmatrix} 0.487 \\ 0.833 \\ 0.174 \\ 1.208 \\ -0.990 \\ -0.010 \\ \vdots \end{pmatrix} \quad \Rightarrow \quad \mathbf{X} = \begin{pmatrix} 0.174 & 0.833 & 0.487 \\ 1.208 & 0.174 & 0.833 \\ -0.990 & 1.208 & 0.174 \\ -0.010 & -0.990 & 1.208 \\ \vdots & \vdots & \vdots \end{pmatrix}.$$

Each row vector of matrix **X** corresponds to an embedding vector. Here it should be noted that in representation (4.1), the first column of the matrix corresponds to the vector of labels. However, in this example, the observed time series is projected into a three-dimensional reconstructed state. This space can be represented graphically by plotting lagged variables on the axes, which reveals the dynamics generating the series. This can be seen in the following figure.

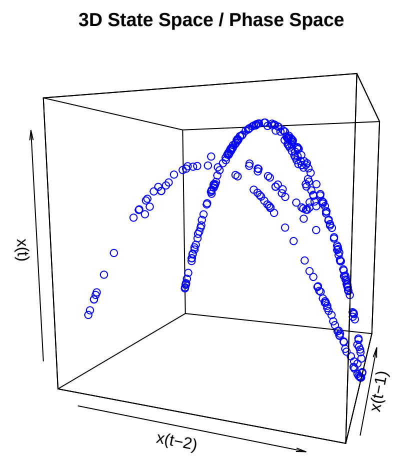


Figure A.2.: Dynamics of the Henon system in the embedded space

Overall, it appears that this approach supports the restructuring of a time series into a supervised learning data format.

A.6. Proof of Equation (4.18)

We now want to show that the variance of the bagged predictor $\hat{f}^B_{Bag}$ for identically distributed trees and prediction functions that are positively correlated with each other is exactly equal to the expression from equation (4.18)

$$var(\hat{f}^B_{Bag}(\mathbf{x}))= \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

This representation of the variance is from the book by Hastie et al. (2009) Equation 15.1 and is also asked to proof there as an exercise. Since this book is a standard in the statistical machine learning literature, there are probably many different approaches to derive this equation.

Proof:

We first assume that all for all $i \in \{1, \dots, B\}$ prediction functions are identically distributed with $\hat{f}^*_i \sim \mathcal{D}(\mu, \sigma^2)$ and there is a positive pairwise correlation between the prediction functions $\hat{f}^*_i$ and $\hat{f}^*_j$ with correlation coefficient $\rho > 0$. Thus, with the help of the calculation rules of variance and covariance, it follows

$$\begin{aligned}
var(\hat{f}^B_{Bag}(\mathbf{x})) &= var\left(\frac{1}{B}\sum_{i=1}^{B}\hat{f}^*_i(\mathbf{x})\right) \\
&= \frac{1}{B^2}var\left(\sum_{i=1}^{B}\hat{f}^*_i(\mathbf{x})\right) \\
&= \frac{1}{B^2}cov\left(\sum_{i=1}^{B}\hat{f}^*_i(\mathbf{x}), \sum_{i=1}^{B}\hat{f}^*_i(\mathbf{x})\right) \\
&= \frac{1}{B^2}\left\{\sum_{i=1}^{B}var(\hat{f}^*_i(\mathbf{x})) + \sum_{i\neq j}cov(\hat{f}^*_i(\mathbf{x}), \hat{f}^*_j(\mathbf{x}))\right\} \\
&= \frac{1}{B^2}\left\{B\sigma^2 + \sum_{i\neq j}cov(\hat{f}^*_i(\mathbf{x}), \hat{f}^*_j(\mathbf{x}))\right\} \\
&= \frac{1}{B^2}\left\{B\sigma^2 + \sum_{i\neq j}\rho\sigma^2\right\} \\
&= \frac{1}{B^2}\left\{B\sigma^2 + (B^2 - B)\rho\sigma^2\right\} \\
&= \rho\sigma^2 + \frac{(1-\rho)}{B}\sigma^2 \quad \square
\end{aligned}$$

The first five equalities following from using the definitions and applying the variance and covariance rules, and the relationship between the two measures. Note that the sixth equality follows from the fact that $\rho = \frac{cov(\hat{f}_i^*(\mathbf{x}), \hat{f}_j^*(\mathbf{x}))}{\sigma^2}$. The seventh equality follows from the fact that there are exactly B^2 pairs of i, j where for B pairs $i = j$ holds. Accordingly there have to be $(B^2 - B)$ paris with $i \neq j$ and therefore we get $\sum_{i \neq j} = (B^2 - B) = B(B-1)$.

References

Andersen, T. G. and Bollerslev, T. (1998). Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International economic review*, pages 885–905.

Andersen, T. G., Bollerslev, T., Christoffersen, P. F., and Diebold, F. X. (2013). Financial risk measurement for financial risk management. In *Handbook of the Economics of Finance*, volume 2, pages 1127–1220. Elsevier.

Andersen, T. G., Bollerslev, T., Diebold, F. X., and Labys, P. (2001). The distribution of realized exchange rate volatility. *Journal of the American statistical association*, 96(453):42–55.

Andersen, T. G., Bollerslev, T., Diebold, F. X., and Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica*, 71(2):579–625.

Athey, S. and Imbens, G. (2019). Machine learning methods economists should know about. *arXiv preprint arXiv:1903.10075*.

Audrino, F. and Knaus, S. D. (2016). Lassoing the har model: A model selection perspective on realized volatility dynamics. *Econometric Reviews*, 35(8-10):1485–1521.

Awad, M. and Khanna, R. (2015). Support vector regression. In *Efficient learning machines*, pages 67–80. Springer.

Bachelier, L. (1900). Théorie de la spéculation. In *Annales scientifiques de l'École normale supérieure*, volume 17, pages 21–86.

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

Ben Taieb, S. and Hyndman, R. (2012). Recursive and direct multi-step forecasting: the best of both worlds. Technical report, Monash University, Department of Econometrics and Business Statistics.

Bergmeir, C., Hyndman, R. J., and Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120:70–83.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Blair, B. J., Poon, S.-H., and Taylor, S. J. (2010). Forecasting s&p 100 volatility: the incremental information content of implied volatilities and high-frequency index returns. In *Handbook of quantitative finance and risk management*, pages 1333–1344. Springer.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327.

Bontempi, G., Ben Taieb, S., and Borgne, Y.-A. L. (2012). Machine learning strategies for time series forecasting. In *European business intelligence summer school*, pages 62–77. Springer.

Bradley, R. C. (2005). Basic properties of strong mixing conditions. a survey and some open questions. *Probability surveys*, 2:107–144.

Breiman, L., C. A. (2003). Manual: Setting up, using and understanding random forests, v4.0. URL: https://www.stat.berkeley.edu/~breiman/Using_random_forests_v4.0.pdf, (22.08.2022), University of California, Berkeley.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brockwell, P. J. and Davis, R. A. (2002). *Introduction to time series and forecasting*. Springer.

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

Bühlmann, P. (2002). Bootstraps for time series. *Statistical science*, pages 52–72.

Carlstein, E. (1986). The use of subseries values for estimating the variance of a general statistic from a stationary sequence. *The annals of statistics*, pages 1171–1179.

Carr, P., Wu, L., and Zhang, Z. (2019). Using machine learning to predict realized variance. *arXiv preprint arXiv:1909.10035*.

Carrasco, M. and Chen, X. (2002). Mixing and moment properties of various GARCH and stochastic volatility models. *Econometric Theory*, 18(1):17–39.

Casdagli, M., Eubank, S., Farmer, J. D., and Gibson, J. (1991). State space reconstruction in the presence of noise. *Physica D: Nonlinear Phenomena*, 51(1-3):52–98.

Chevillon, G. (2007). Direct multi-step estimation and forecasting. *Journal of Economic Surveys*, 21(4):746–785.

Chevillon, G. and Hendry, D. F. (2005). Non-parametric direct multi-step estimation for forecasting economic processes. *International Journal of Forecasting*, 21(2):201–218.

Christensen, K., Siggaard, M., and Veliyev, B. (2021). A machine learning approach to volatility forecasting. *Available at SSRN*.

Clements, A. and Preve, D. P. (2021). A practical guide to harnessing the HAR volatility model. *Journal of Banking & Finance*, 133:106285.

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

Danielsson, J. (2011). *Financial risk forecasting: the theory and practice of forecasting market risk with implementation in R and Matlab*. John Wiley & Sons.

De Stefani, J., Caelen, O., Hattab, D., and Bontempi, G. (2017). Machine learning for multi-step ahead forecasting of volatility proxies. In *MIDAS@ PKDD/ECML*, pages 17–28.

Efron, B. and Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007.

Fama, E. F. (1965). The behavior of stock-market prices. *The journal of Business*, 38(1):34–105.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, 25(2):383–417.

Fryzlewicz, P. and Rao, S. S. (2011). Mixing properties of arch and time-varying arch processes. *Bernoulli*, 17(1):320–346.

Garcia, C. A. (2022). *nonlinearTseries: Nonlinear Time Series Analysis*. R package version 0.2.12.

Hall, P., Horowitz, J. L., and Jing, B.-Y. (1995). On blocking rules for the bootstrap with dependent data. *Biometrika*, 82(3):561–574.

Hansen, P. R. and Lunde, A. (2005). A forecast comparison of volatility models: does anything beat a GARCH (1, 1)? *Journal of applied econometrics*, 20(7):873–889.

Hansen, P. R. and Lunde, A. (2006a). Consistent ranking of volatility models. *Journal of Econometrics*, 131(1-2):97–121.

Hansen, P. R. and Lunde, A. (2006b). Realized variance and market microstructure noise. *Journal of Business & Economic Statistics*, 24(2):127–161.

Hansen, P. R. and Lunde, A. (2011). Forecasting volatility using high frequency data.

Hassler, U. (2007). *Stochastic integration and time series modeling: an introduction with applications from finance and econometrics*. Springer.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hautsch, N. (2011). *Econometrics of financial high-frequency data*. Springer Science & Business Media.

Heber, Gerd, A. L. N. S. and Sheppard, K. K. (2009). Oxford-man institute's realized library. Oxford-Man Institute, University of Oxford. Library Version: 0.3.

Held, C. (2018). Gabler business dictionary definition: German Stock Index (DAX). URL: https://wirtschaftslexikon.gabler.de/definition/deutscher-aktienindex-dax-31116/version-254682, (10.07.2022), Springer.

Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Israel, R., Kelly, B. T., and Moskowitz, T. J. (2020). Can Machines 'Learn' Finance? *Journal of Investment Management*.

Izzeldin, M., Hassan, M. K., Pappas, V., and Tsionas, M. (2019). Forecasting realised volatility using ARFIMA and HAR models. *Quantitative Finance*, 19(10):1627–1638.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Jemwa, G. T. (2003). Multivariate nonlinear time series analysis of dynamic process systems. Master's thesis, Stellenbosch: University of Stellenbosch.

Karandikar, R. and Vidyasagar, M. (2009). Probably approximately correct learning with beta-mixing input sequences. *submitted for publication*.

Karatzas, I. and Shreve, S. (2012). *Brownian motion and stochastic calculus*, volume 113. Springer Science & Business Media.

Kirchgässner, G., Wolters, J., and Hassler, U. (2012). *Introduction to modern time series analysis*. Springer Science & Business Media.

Kreiss, J.-P. and Lahiri, S. N. (2012). Bootstrap methods for time series. In *Handbook of statistics*, volume 30, pages 3–26. Elsevier.

Künsch, H. R. (1989). The jackknife and the bootstrap for general stationary observations. *The annals of Statistics*, pages 1217–1241.

Kuznetsov, V. and Mohri, M. (2014). Generalization bounds for time series prediction with non-stationary processes. In *International conference on algorithmic learning theory*, pages 260–274. Springer.

Kuznetsov, V. and Mohri, M. (2015). Learning theory and algorithms for forecasting non-stationary time series. *Advances in neural information processing systems*, 28.

Kuznetsov, V. and Mohri, M. (2018). Lecture notes on advanced machine learning: Time series prediction. URL: https://cims.nyu.edu/~mohri/amls/aml_time_series.pdf, (16.08.2022), New York University.

Lee, S.-W. and Hansen, B. E. (1994). Asymptotic theory for the GARCH (1, 1) quasi-maximum likelihood estimator. *Econometric theory*, 10(1):29–52.

Li, S. Z. and Tang, Y. (2021). Forecasting realized volatility: An automatic system using many features and many machine learning algorithms. *Available at SSRN 3776915.*

Lindner, A. M. (2009). Stationarity, mixing, distributional properties and moments of GARCH (p, q)–processes. In *Handbook of financial time series*, pages 43–69. Springer.

Liu, L. Y., Patton, A. J., and Sheppard, K. (2015). Does anything beat 5-minute RV? A comparison of realized measures across multiple asset classes. *Journal of Econometrics*, 187(1):293–311.

Lommers, K., El Harzli, O., and Kim, J. (2021). Confronting machine learning with financial research. *The Journal of Financial Data Science*, 3(3):67–96.

Lopez, J. A. (2001). Evaluating the predictive accuracy of volatility models. *Journal of forecasting*, 20(2):87–109.

López de Prado, M. (2019). Beyond econometrics: A roadmap towards financial machine learning. *Available at SSRN 3365282.*

Luong, C. and Dokuchaev, N. (2018). Forecasting of realised volatility with the random forests algorithm. *Journal of Risk and Financial Management*, 11(4):61.

Masini, R. P., Medeiros, M. C., and Mendes, E. F. (2021). Machine learning advances for time series forecasting. *Journal of Economic Surveys.*

McDonald, D. J., Shalizi, C. R., and Schervish, M. (2011). Generalization error bounds for stationary autoregressive models. *arXiv preprint arXiv:1103.0942.*

Mohri, M. and Muñoz Medina, A. (2012). New analysis and algorithm for learning with drifting distributions. In *International Conference on Algorithmic Learning Theory*, pages 124–138. Springer.

Mohri, M. and Rostamizadeh, A. (2007). Stability bounds for non-iid processes. *Advances in Neural Information Processing Systems*, 20.

Mohri, M. and Rostamizadeh, A. (2008). Rademacher complexity bounds for non-iid processes. *Advances in Neural Information Processing Systems*, 21.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of machine learning.* MIT press.

Müller, U. A., Dacorogna, M. M., Davé, R. D., Pictet, O. V., Olsen, R. B., and Ward, J. R. (1993). Fractals and intrinsic time: A challenge to econometricians. *Unpublished manuscript, Olsen & Associates, Zürich*, 130.

Patton, A., Politis, D. N., and White, H. (2009). Correction to “automatic block-length selection for the dependent bootstrap” by d. politis and h. white. *Econometric Reviews*, 28(4):372–375.

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

Peng, Y., Albuquerque, P. H. M., de Sá, J. M. C., Padula, A. J. A., and Montenegro, M. R. (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.

Politis, D. N. and White, H. (2004). Automatic block-length selection for the dependent bootstrap. *Econometric reviews*, 23(1):53–70.

Poon, S.-H. (2005). *A practical guide to forecasting financial market volatility.* John Wiley & Sons.

Poon, S.-H. and Granger, C. W. (2003). Forecasting volatility in financial markets: A review. *Journal of economic literature*, 41(2):478–539.

R Core Team (2022). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

Rahimikia, E. and Poon, S.-H. (2020). Machine learning for realised volatility forecasting. *Available at SSRN*, 3707796.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms.* Cambridge university press.

Shephard, N. and Sheppard, K. (2010). Realising the future: forecasting with high-frequency-based volatility (heavy) models. *Journal of Applied Econometrics*, 25(2):197–231.

Shreve, S. E. et al. (2004). *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer.

Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222.

Takens, F. (1981). Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pages 366–381. Springer.

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. *International journal of forecasting*, 16(4):437–450.

Tsay, R. S. (2005). *Analysis of financial time series*. John wiley & sons.

Vortelinos, D. I. (2017). Forecasting realized volatility: HAR against principal components combining, neural networks and GARCH. *Research in international business and finance*, 39:824–839.

Wang, C. S.-H., Bauwens, L., and Hsiao, C. (2013). Forecasting a long memory process subject to structural breaks. *Journal of Econometrics*, 177(2):171–184.

Webel, K. and Wied, D. (2016). *Stochastic Processes*. Springer.

Yu, B. (1994). Rates of convergence for empirical processes of stationary mixing sequences. *The Annals of Probability*, pages 94–116.

Zhang, C., Zhang, Y., Cucuringu, M., and Qian, Z. (2022). Volatility forecasting with machine learning and intraday commonality. *arXiv preprint arXiv:2202.08962*.

Zivot, E. (2009). Practical issues in the analysis of univariate GARCH models. In *Handbook of financial time series*, pages 113–155. Springer.

Zivot, E. (2011). Lecture notes on financial econometrics: Introduction to realized variance. URL: https://faculty.washington.edu/ezivot/econ589/econ512realizedvariance.pdf, (23.07.2022), University of Washington.

Declaration of Authenticity

The work contained in this thesis is original and has not been previously submitted for examination which has led to the award of a degree.

To the best of my knowledge and belief, this thesis contains no material previously published or written by another person except where due reference is made.

Dominik Bruckmeier