Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Quantitative Finance

A boosting method for time-varying density forecasting of volatility

Hidde Baron (433911)

Supervisor:	O. Kleen
Second assessor:	A. Tetereva
Date final version:	1st May 2024

Abstract

This study proposes a novel cluster natural gradient boosting (NGboost) approach for forecasting the conditional density of stock return volatility. The NGboost algorithm uses natural gradients for parameter optimization, creating a multi-parameter optimization that enables predicting distributions. Employing an optimization algorithm, clusters are formed to minimize the Continuous Ranked Probability Score (CRPS) for 29 selected S&P 500 stocks. The cluster NGboost is used in two configurations: directly forecasting realized volatility (cluster NGboost) and forecasting the distribution of the residuals of a logarithmic HAR model (cluster ResNGboost). Both use a lognormal distribution due to its optimal performance for the validation period. My empirical analysis compares these models to their singular and pooled counterparts, as well as to a log HAR model with normality assumptions on its innovations. Results from the out-of-sample period (January 2016 to December 2020) demonstrate that both cluster NGboost models excel. However, using a moving window approach presents challenges during the 2020 Corona crisis. Whereas using an expanding window, which includes previous crisis data, shows greater resilience. For the expanding window, the cluster models show superior forecasting results in terms of mean, distribution, and quantile forecasting. This research underscores the potential of NGboost as a direct or overlay forecasting model for realized volatility.

Contents

1 Introduction and literature review

-

1.1 Introduction

Forecasting stock return volatility is an important task in financial economics because it supports effective risk management and strategic asset allocation. This study focuses on forecasting the conditional distribution of stock return volatility. Within the field of volatility research, the primary emphasis has been placed on the prediction of a specific mean of future volatility, as shown by the work of Poon and Granger (2003) in their study on forecasting volatility. A comprehensive literature review is conducted to address the research that has been done on predicting the density of volatility and to show where my research fits within the existing literature. According to Hothorn, Kneib and Bühlmann (2014), as stated in their paper, "the ultimate goal of regression analysis is to obtain information about the conditional distribution of a response given a set of explanatory variables"(p. 3).

Utilizing a distributional forecast for volatility is advantageous because it provides supplementary insights regarding the spectrum of forthcoming volatility levels and their corresponding probabilities. The portion of the distribution located on the right-hand side is of particular relevance to traders who are exposed to volatility risk, as noted by Žikeš and Baruník (2015). Knowing the predicted conditional distribution can help enhance the accuracy of option pricing and portfolio optimization. The predicted conditional forecast can facilitate more informed decision-making on resource allocation. For traders in volatility swaps, knowing the anticipated volatility distribution can aid in determining the desirability of a particular strike variance level. Understanding the distribution of volatility allows traders to use appropriate pricing models that account for the skewness, kurtosis, and other moments of the distribution rather than simply relying on the mean or variance.

The importance of modeling the distribution of volatility is confirmed by Corradi, Distaso and Swanson (2011), who states: "Indeed, since shortly after its inception in 1993, when the VIX, an index of implied volatility, was created for the Chicago Board Options Exchange, a plethora of volatility-based derivative products has been engineered, including variance and covariance swaps, overshooters, and upcrossers and downcrossers (see, e.g., Carr and Lee (2003)).Given the development of this new class of financial instruments, it is of interest to construct conditional (predictive) volatility densities, rather than just point forecasts thereof" (p. 1496).

This paper aims to examine the feasibility of predicting the conditional density of stock return volatility using natural gradient boosting (NGboost). My paper has two primary objectives: first, it provides insights into the potential of using the NGboost to directly forecast the conditional density of the realized volatility. Secondly, the objective is to show the potential of using the NGboost as an overlay model. The NGboost is used on the residuals of a

mean forecasting model (RESNGboost); in this way, it forecasts a distribution for the errors of that distribution. Essentially improving the forecast by providing a distributional forecast for the errors. In my research, I use an ResNGboost for the heterogeneous autoregressive realized volatility (HAR-RV) model proposed by Corsi (2009). My main research question is as follows: *Can the NGboost provide more accurate density forecasts for one-day-ahead realized volatility than heterogeneous autoregressive realized volatility models?*

My research will focus on realized volatility; this is an estimate of the volatility, as volatility itself is a latent variable. Duan et al. (2020) created the NGboost, a boosting algorithm based on gradient boosting that is extended to be able to provide a density forecast for the target variable. Instead of predicting a single value, it estimates the parameters of a distribution, such as the mean and variance for a normal distribution. My paper is the first to use NGboost for realized volatility forecasting. The concept of natural gradients forms the foundation of NGBoost. NGBoost uses natural gradients instead of ordinary gradients to optimize the parameters of a probability distribution. In Riemannian space, the generalized natural gradient represents the direction of the steepest ascent. The natural gradient is opposed to the standard gradient reparametrization invariant. To be able to forecast multi-parameter probability distributions, the NGboost uses multiparameter boosting. In Section 2.3, I provided a thorough explanation of the NGboost.

Three essential components comprise NGBoost: a scoring rule, a base learner, and parametric probability distributions. This allows users to modify the algorithm to their specific requirements, including the selection of the base learner (e.g., decision trees), probability distribution parameters (e.g., normal, lognormal), and scoring rule (e.g., logarithmic score). The NGboost algorithm is very suitable for my research as it is flexible and easily adaptable. In my literature review, I discuss this in detail, discussing past performances of the NGBoost in other fields of research as well as other boosting methods for realized volatility.

I extend the NGboost by pooling the data of stocks together, creating a pooled NGboost. This procedure is based on Bollerslev, Hood, Huss and Pedersen (2018); in their paper, by utilizing similarities in realized volatility patterns among stocks, they are able to produce improved out-of-sample forecasts of realized volatility. According to Kleen and Tetereva (2022), while pooled estimators may have some bias due to heterogeneity, they also yield significant efficiency improvements. The paper by Oh and Patton (2023) uses a cluster optimization algorithm to form a multifactor copula model. They show that their estimated cluster assignments significantly outperform clusters based on SIC codes. Inspired by their cluster optimization, I extend the NGboost by using a cluster optimization routine. After optimization, I pool the data for each cluster and estimate an NGboost model. I employ a static version of this optimization routine; however, the optimization can be further extended to be dynamic. To gain understanding of why specific clusters form, I use the data set of Gu, Kelly and Xiu (2020), which includes 94 stock characteristics. To this set, I add the realized volatility, the root mean squared error of the logaritmic HAR, and the first two digits of the Standard Industrial Classification Code.

In my empirical research, the NGBoost and ResNGboost models are compared to HAR models as introduced by Corsi, Mittnik, Pigorsch and Pigorsch (2008). This research uses 29 stocks from the S%P 500 and also includes the S&P500 index. I use two proper scoring rules

to compare the distributional forecasts of the different models. The lognormal distribution is selected for all NGboost specifications based on results obtained for the validation period (January 2012 to December 2015). This study uses two different estimation windows: a moving window containing 5 years of training data, and an expanding window containing data as of January 2007. All NGboost specifications suffer when a moving window is used to forecast the Corona crisis year 2020. Using an expanding window, however, yields very promising results for the one-day-ahead density forecasts for the out-of-sample period January 2016 to December 2020. The expanding window includes training data from the 2008 global economic crisis, whereas the moving window does not. The difference in performance suggests that the NGboost specifications benefit substantially from including crisis training data.

For the expanding window, in terms of continuous ranked probability score (CRPS), the cluster NGboost and cluster ResNGboost significantly outperform a log HAR model with a normal assumption on the innovation term for a minimum of 80 percent of stocks. This is tested with the Giacomini-White test, as introduced by Giacomini and White (2006). The model confidence set procedure by Hansen, Lunde and Nason (2011) shows that both the cluster specifications show a much higher model confidence set inclusion rate (MCSR) for both the CRPS as well as the logaritmic score (LogS). In terms of the squared error loss (SE), both cluster models have a lower average loss (AL) while outperforming the benchmark significantly in 50 percent of cases. Quantile forecasts are compared using an asymmetric piecewise linear scoring function (APL), as proposed by Gneiting (2011a). For the 99% quantile, both cluster ngboost specifications show a lower average loss; however, the MCSR is on par with the benchmark model. Furthermore, both the NGboost and ResNGboost cluster specifications outperform their individual and pooled counterparts, with one of the two cluster models always showing the highest MCSR and lowest AL for the CRPS, LogS, SE, and APL.

In a deepening analysis of the AL conditional on the daily and monthly realized volatility, this study finds that for both cluster models, the performance in terms of CRPS improves whenever realized volatility is higher in the past month. This indicates that the cluster models outperform other models during periods of economic turbulence. For the 99% quantiles, it shows that the MCSR of the benchmark model is high due to the high variation in the differences between the APL scoring functions of the cluster models and the benchmark. Using the accumulated local effects (ALE), this paper provides insights into the importance of different features for the cluster models. ALE and variable importance plots are presented for the forecasted scale and shape of the lognormal distribution.

The rest of this paper is structured as follows: in Section 1.2, the literature regarding distributional forecasting of volatility is discussed, as well as the performance of the NGBoost in other fields of research. In the literature section, I address the place of this work within the existing literature as well as my addition to the literature. Next, in Section 2, I first discuss my used benchmark models, and after that, I thoroughly discuss the NGBoost algorithm. Next, pooled and cluster specifications are discussed in Sections 2.3.2 and 2.3.3. In Section 2.4, I conclude my methodology section by discussing the different metrics used to evaluate the models' performance. I show my empirical research in Section 3; first, the data and features are presented in Section 3.1. Afterwards, the model choices and cluster optimization results are presented in

Section 3.2. Section 3.3 presents the main findings of the empirical research. In Section 4.1, I discuss the limitations of this study and suggestions for future research. Section 4.2 presents the concluding remarks.

1.2 Literature

Historically, standard econometric models such as ARCH, GARCH, and a variety of other GARCH-type models have been frequently used to forecast volatility. These models posit that past returns or latent variables drive volatility, providing valuable insights into its time-varying nature. A good review of these models and the history of volatility forecasting is written by Poon and Granger (2003).

Recently, machine learning algorithms like Random Forests, Support Vector Regression, and Neural Networks have become more popular because they can handle nonlinear relationships and make accurate predictions. These models are sometimes used in combination with more traditional models, like the heterogeneous autoregressive model (HAR) of Corsi (2009). An example of this is the research by Kleen and Tetereva (2022), where a random forest is combined with a HAR model.

In my paper, the main objective is to forecast the conditional distribution of volatility $P(y \mid x)$. In the next part, I discuss models that are used for distributional forecasting, assume a distribution for the predicted volatility, or are well-fit to predict the distribution of volatility.

1.2.1 Volatility of volatility

Among the first models to account for volatility of volatility are the stochastic volatility models Poon and Granger (2003). Stochastic volatility (SV) models assume that the volatility of an asset is a random process that can be described by a specific stochastic differential equation (SDE). This SDE defines how volatility behaves over time, including its distribution at any given point. The Heston model is still a popular stochastic volatility model, as found by Heston (1993).

In terms of HAR and GARCH-specified models, the volatility of realized volatility is explored by Corsi et al. (2008). They observe clustering in the errors of volatility models, such as the HAR and ARIFMA models. In addition, they discover that the errors have right skewness and heavy tails. Using estimators based on the asymptotic distribution theory derived from the work of Barndorff-Nielsen and Shephard (2002), they demonstrate this behavior not only for the errors of the models mentioned above but also for the volatility of realized volatility. Moreover, they reach this conclusion by employing various methods to calculate the realized quarticity. A result that supports these observations is found by Barndorff-Nielsen and Shephard (2005), who found that when plotting the confident intervals of the measurement error of realized volatility, this is also time-varying. We can conclude that the conditional distribution of realized volatility will also fluctuate over time, given the time-varying nature of the realized volatility measurement errors and the observed volatility. The volatility of volatility is essential for pricing options. D. Huang, Schlag, Shaliastovich and Thimme (2019) demonstrates that market participants

pay a premium not only to hedge against volatility but also to hedge against the volatility of volatility. This demonstrates the significance of modeling the volatility distribution.

Corsi et al. (2008) models the volatility of the realized volatility using a HAR-GARCH model, with the innovation term of the realized volatility following a conditional density with time-varying variance. That is, the one-step-ahead distribution of the realized volatility is dependent on the forecasted conditional density of the realized volatility's innovation term; however, the mean of this conditional density is assumed to be zero. As the GARCH implementation does not account for the non-normality of the innovation terms, a normal inverse Gaussian (NIG) distribution is also employed. My use of the NGboost as an overlay model for the HAR can be seen as a replacement for the GARCH equation. The NGboost is also forecasting the conditional density of the innovation term; however, the mean of the conditional density is not assumed to be zero. The NGboost allows all of a distribution's parameters to be time-varying.

The paper byTodorova (2015) employs the same terminology as Corsi et al. (2008). The paper examines the forecasts for the realized volatility of industrial metals futures contracts. In addition, he employs a specification of Corsi and Renò (2012) that accounts for leverage effects; the model is referred to as the HAR-L-GARCH model. The study demonstrates that the GARCH coefficients have a large and positive effect, validating the clustering of realized volatility, and that the GARCH coefficients are highly time-varying, validating the time-varying character of volatility of realized volatility. These findings indicate that, despite both models accounting for the stylized facts of volatility (short-term leverage effects and volatility clustering), the conventional HAR model cannot consistently outperform in terms of mean forecasts. Z. Huang, Liu and Wang (2016) also specifies a realized HAR-GARCH model, but with different parameters. The model of Z. Huang et al. (2016) is similar in name, but it does not use a GARCH model to describe the errors of a HAR model. Instead, it employs a GARCH model with a HAR specification for volatility. The model is not (yet) extended to model the volatility of volatility. The model of Z. Huang et al. (2016) does not account for time-varying volatility and is primarily an extension of the Realized GARCH model by Hansen, Huang and Shek (2012). A model that does build on the Realized GARCH ofHansen et al. (2012) and accounts for the time-varying volatility of volatility is the one by Catania and Proietti (2020). In their study, they employ a bivariate model for log-realized volatility and market returns. Their model employs a student-t distribution. They demonstrate clearly that their model outperforms, in terms of continuous ranked probability score, the HAR-GARCH specification by Corsi et al. (2008). A distinction between my research and theirs is that my models forecast a conditional distribution for the realized volatility instead of the log realized volatility. I discuss their results shortly in Section 4.1, mostly to address future research recommendations.

1.2.2 Microstructure noise

The review by McAleer and Medeiros (2008) shows the asymptotic distribution of the realized variance, demonstrating that it has a N(0,1) distribution under the assumption that there is no micro-structure noise. Several factors, such as price discreteness, bid-ask bounce, infrequent trading, and asynchronous trading, can cause micro-structure noise. The paper by McAleer and Medeiros (2008) also illustrates the severe problems that can arise as a result of having

microstructure noise in realized volatility estimates. It does so by discussing a number of papers that present methods for addressing microstructure noise. Every realized measure of volatility contains microstructure noise due to measurement errors. According to Liu, Patton and Sheppard (2015), the 5 minute realized volatility is very difficult to surpass; therefore, this will suffice for my research. The paper by Corradi, Distaso and Swanson (2009) proposes a non-parametric model to forecast the conditional density of realized volatility. Microstructure noise generated by the realized volatility estimators forms the basis of the model. Corradi et al. (2011) expands on this and employs the same methods, extending it by allowing for jumps, and concludes that spillover between markets has an impact on the conditional distribution of a market. Building on the paper by Corradi et al. (2011), the paper by Mukherjee and Swanson (2021) employs the non-parametric Nadarya-Watson kernel density type estimators introduced by Corradi et al. (2011) to compare the relative usefulness of density forecasts versus point forecasts. They find that density forecasts can enhance the profitability of their trading strategies.

The measurement error in the RV has an impact on the modeling process, as the measurement error in RV_{t-1} can have an impact on the coefficients of, for example, the HAR model. Through their HARQ model, Bollerslev, Patton and Quaedvlieg (2016) attempt to address micro-structure noise by permitting the parameters of the HAR model to vary with the level of realized quarticity, which is taken as an estimate of the measurement error. They find that their model outperforms their benchmarks, and they discover a negative effect, indicating that a high realized quarticity or measurement error decreases the persistence of the past in the present. As a result, I use the realized quarticity as a feature for my NGBoost models. However, as Cipollini, Gallo and Otranto (2021) point out, the realized quarticity may not be very important if the innovation term in the HARQ model is allowed to have a time-varying variance set by a garch equation similar to that in Corsi et al. (2008). Furthermore, with their Markov Switching AMEM model, which has multiplicative error structures and return regime-specific dynamics, they include a model that can provide conditional distributional forecasts. However, they do not test the performance of their model in terms of density forecasting. According to Cipollini et al. (2021), their model accounts for heteroskedasticity, and they state their slow-moving average level model is measurement error-robust. The paper by Caporin, Rossi and De Magistris (2017), similarly to (Cipollini et al., 2021), extends the MEM frame of Engle (2002), creating an AHAR-MEMλ_j model. Essentially, they extend the MEM model with an asymetric HAR and a time-varying jump component. With the use of four assumptions, they provide a conditional density forecast of the realized volatility and state that they are able to accurately forecast the 99% quantile.

Buccheri and Corsi (2021) remedies three problems they discover using a Kalman filter technique. The first is measurement error caused by microstructure noise. The second is that the original HAR model doesn't have any time-varying coefficients, which, as shown by Kleen and Tetereva (2022), can be very helpful if fixed. The third is the heteroskedasticity in the error terms. All of the aforementioned models of Bollerslev et al. (2016),Cipollini et al. (2021).Buccheri and Corsi (2021) are designed to make mean predictions, and the models are not tested for density forecast in their papers. However, an important takeaway is that their models improve by accounting for microstructure noise, heteroskedasticity, and non-linearity. As I will be employing a boosting algorithm on the residuals of a HAR, I am attempting to rectify the HAR model.

Doing so, the goal is to take care of the three problems stated by Buccheri and Corsi (2021). The NGboost is able to exploid non-linear links in the data and the realized quarticity is used as a feature to address the measurement error. Furthermore, the NGboost allows the variance of the distribution of the error term to be time-varying. In further research, NGboost model can be used as an overlay model for the model of Buccheri and Corsi (2021), as their model already attempts to adapt to the HAR's problems and outperforms many other decent models.

1.2.3 Quantile regression, bayesian, bootstrap and DCS

A model called the heterogeneous quantile autoregressive model is used by Žikeš and Baruník (2015) to predict volatility quantiles. Forecasting each quantile of the distribution would provide a kernel for the realized volatility, although this does not provide a direct forecast of the entire parametric distribution. They argue that their model accurately captures the dynamics of the corresponding conditional distributions. Their research also addresses the fact that volatility of volatility effect is evident in their data and that the volatility of volatility increases as volatility increases. In addition, they use upside and downside volatility as explanatory variables and discover that downside volatility predominates.

Both Nonejad (2017) and D. Li, Clements and Drovandi (2021) propose a Bayesian-type model that accounts for the uncertainty of the model's parameters. Nonejad (2017) demonstrates that they outperform a straightforward AR model with time-varying volatility of volatility and the assumption of normality.

The three papers, Trucíos and Hotta (2016), Trucíos, Hotta and Ruiz (2017), and Trucíos, Hotta and Ruiz (2018), use a bootstrap method instead of assuming a distribution for the errors and try to forecast the distribution by mapping the uncertainty given by the estimated parameters.

The study by A. Harvey and Palumbo (2023) demonstrates that a dynamic condition score model based on the theory of A. C. Harvey (2022) outperforms the model of Caporin et al. (2017) in terms of predictive likelihood. They use a Generalized Beta of the Second Kind (GB2) distribution for their model. I leave the implementation of a GB-2 distribution for NGBoost for further research.

The different papers give a broad overview of the various models used and the relevance of modeling the conditional distribution of (realized) volatility. My paper fits right into the literature, as it is the first to forecast the conditional distribution of volatility using a boosting method like the NGboost. I further extend the RV density forecasting literature by using the NGboost as an overlay model for the HAR model. In addition to this, I create a pooled and cluster NGboost model, which is also new to the literature. In the next part, I discuss the performance of the NGBoost in other fields.

NGboost

Despite the fact that boosting algorithms are not yet used for forecasting the conditional distribution of the realized volatility, there are a few papers that yield good results for forecasting the mean realized volatility. For example, (Teller, Pigorsch & Pigorsch, 2022) uses Extreme Gradient Boosting to forecast the realized volatility and shows that it significantly outperforms

both the HAR and Long Short-Term Memory models. The paper by Wu and Wang (2021) uses a LightGBM for forecasting RV, which essentially is an improved / adapted XGboost model, the LightGBM is introduced by Ke et al. (2017). They show that the LightGBM outperforms an XGboost model and a support vector machine model in terms of root mean squared error of the realized volatility forecasts.

In other areas, NGboost has shown promising performance. Chakraborty, Elhegazy, Elzarka and Gutierrez (2020) utilizes NGboost for estimating construction expenses. In geological sciences, Kavzoglu and Teke (2022) utilizes NGboost to predict potential landslides. In the discipline of meteorology, NGboost is used to predict the expected temperature, as done by Peng, Zhi, Ji, Ji and Tian (2020). Y. Li, Wang and Wu (2020) utilizes the NGboost to provide the anticipated wind power. The paper by Yan, Fan and Zhang (2023) makes probabilistic predictions of NOx emissions from gas turbine combustors using NGboost.

Mitrentsis and Lens (2022a) utilizes NGBoost to provide short-term solar power forecasts. Additionally, they use Shapley additive explanation (SHAP) values to test for features that have almost no effect; removing these features enhances the performance of the forecasts. Shen, Qin, Zhou and Liu (2022) employs NGboost to predict the runoff of a river; the hyperparameters are tuned using the tree-structured parzen estimator (TPE). In each of the papers listed above, NGboost outperformed the benchmark models. In addition, most of these publications only used NGboost's default parameters. As the NGBoost produces very promising results in other fields, I use it to forecast the distribution of realized volatility.

2 Methodology

As mentioned in the Introduction Section 1.1, I use the realized volatility as an estimate of the latent variable volatility. To be more clear, I am interested in the distribution of the integrated volatility, which is defined as the square root of the integrated variance (IV),

$$IV_t = \int_0^t \sigma^2(u)\, du.$$

However, the integrated variance is not directly observable. The daily realized variance is defined as the sum of intraday log returns, and the daily realized volatility (RV^d) is defined as the square root of the realized variance,

$$RV_t^d = \sqrt{\sum_{j=1}^{M} r_{t-(j-1)\Delta}^2}. \tag{2.1}$$

M is the number of intraday observations, with the sampling frequency defined as $\Delta = 1/M$, and the $r_{t,i}$ is defined as the log returns between two time periods with exactly one step Δ between them. The realized variance is observable, and as $M \to \infty$, Andersen and Bollerslev (1998) shows that the realized variance is a consistent estimator of the IV. However, data limitations constrain the value of M. I use a 5-minute sampling frequency, as Liu et al. (2015) shows that this is very hard to beat. But by doing so, Barndorff-Nielsen and Shephard (2002) shows that

the $RV_t = IV_t + \eta_t$, with η_t being the estimation error. The estimation error has a zero mean but a heteroskedastic variance; see Cipollini et al. (2021).

In my paper, I predict the distribution of the RV_t conditional on the information set I_{t-1}, given as $P(RV_t|I_{t-1})$. The variance of the η is known to increase with the realized volatility; see Bollerslev et al. (2016). The NGboost models try to encompass, by allowing all parameters of the parametric distribution to be time-varying and specifically the variance to be heteroskedastic, both the heteroskedestic variance of η as well as the uncertainty of the estimate of $E(RV_t|I_{t-1})$. The uncertainty of the estimate of $E(RV_t|I_{t-1})$ is due to possible misspecification, measurement errors in I_{t-1}, and estimation uncertainty of coefficients.

In section 2.1, I describe the HAR model that I use to forecast the realized volatility, of which the residuals are the input for the NGboost algorithm described in 2.3. The benchmarks I use are described in 2.2. In Section 2.4, I describe all metrics and tests used to compare my models, as well as the methods used to gain deeper insights into their performance.

2.1 Heterogenous autoregressive model for realized volatility

The heterogenous autoregressive (HAR) model was first used by (Corsi, 2009). The model uses different AR-type models of additive volatility estimates with different horizons. The foundation of the model is the Heterogeneous Market Hypothesis as presented by Muller et al. (1995). It tries to explain the observation of a strong positive correlation between volatility and market presence by making the assumption that the market is composed of heterogeneous traders. These traders can be divided into groups according to their trading horizons. These horizons are taken to be daily, weekly, and monthly, with each horizon creating its own volatility. Hence, the HAR model as used by Corsi (2009) is given as:

$$RV_{t+1}^{(d)} = c + \beta^{(d)} RV_t^{(d)} + \beta^{(w)} RV_t^{(w)} + \beta^{(m)} RV_t^{(m)} + \epsilon_{t+1}^{(d)}. \tag{2.2}$$

With the weekly and monthly realized volatility defined as $RV^w = RV_{t-5:t-2}$, $RV^m = RV_{t-22:t-6}$. Correspondingly, the parameters for the daily, weekly, and monthly RV are given by the vector $\boldsymbol{\beta} = (\beta_d, \beta_w, \beta_m)^\top$. Furthermore, in the regression, c is a constant, and $\epsilon^{(d)}$ is an innovation term. The HAR model can also be used in logarithmic form. Andersen, Bollerslev and Diebold (2007) introduces this for the realized variance; however, I use a log transformation on the realized volatility directly, and the formula is given as:

$$\log RV_{t+1}^{(d)} = c + \beta^{(d)} \log RV_t^{(d)} + \beta^{(w)} \log RV_t^{(w)} + \beta^{(m)} \log RV_t^{(m)} + \eta_{t+1}^{(d)}. \tag{2.3}$$

Similar to Equation 2.2, the parameter vector $\boldsymbol{\beta}$ is defined, but now the parameters correspond to the log of daily, weekly, and monthly realized volatility. Similar to the HAR specification, c is a constant, and there is an innovation term, $\eta^{(d)}$.

Both specifications are used as benchmarks; this is discussed in Section 2.2. These residuals of these two HAR specifications are both used as input for the ResNGboost; this is discussed in detail in Section 2.3.

2.2 Benchmarks

In total, there are seven models discussed in this section. The first 5 models are included in my paper as benchmarks; the last two are merely included to compare results with the paper of Corsi et al. (2008).The first two benchmarks I use are the HAR and Log HAR, as given in Equations 2.2 and 2.3. Under the previously given specification, these two models only generate point forecasts. To enable these models to produce a one-day-ahead distributional forecasts, an assumption on the distribution of the innovation term is made. For the HAR model in Equation 2.2, I assume that the innovation term $\epsilon^{(d)}$ has a normal distribution with a mean zero and a standard deviation equal to the empirical standard deviation of the residual of the estimation window. Until the model is re-estimated, this leads to a forecasted series of normal distributions with a varying mean and a homoskedastic variance. Similarly, for the log HAR model in 2.3, I assume for $\eta^{(d)}$ a normal distribution with a zero mean and standard deviation equal to the empirical standard deviation of the residuals of the estimation window. However, because of the log transformation, the resulting distribution for the realized volatility is log normal. Reversing the log transformation of the estimated normal distribution for the log realized volatility yields a log normal distribution for the realized volatility.

2.2.1 HAR-GARCH models

Now, based on the work by Corsi et al. (2008), I extend the HAR model with a GARCH model for the innovation term. Corsi et al. (2008) is the first to do so and adds the GARCH component to account for the volatility clustering in realized volatility. The HAR-GARCH model is defined as follows[1]:

$$y_t = c + \beta_d y_{t-1} + \beta_w y_{t-5:t-2} + \beta_m y_{t-22:t-6} + \sqrt{h_t}\varepsilon_t, \tag{2.4}$$

$$h_t = \omega + \sum_{j=1}^{q} \beta_j^2 (\sqrt{h_{t-j}}\varepsilon_{t-j})^2 + \sum_{j=1}^{p} \beta_j h_{t-j}, \tag{2.5}$$

$$\varepsilon_t | I_{t-1} \sim (0, 1). \tag{2.6}$$

Where I_{t-1} denotes the information set up to time t-1, y can be the RV, Log RV, or LOG RVar. My third benchmark model is a HAR-GARCH (1,1) model on the realized volatility with normality assumptions on the innovation ε_t (HAR-GARCH-N). I use a HAR-GARCH (1,1) model on the LOG RV with the assumption that the innovation term is normal as a fourth benchmark model. This gives me a lognormal distribution for the predicted realized volatility (Log HAR-GARCH-N). The last model that I use as a benchmark is the HAR-GARCH (1,1) on realized volatility, assuming a standardised normal inverse Gaussian (NIG) distribution for the innovation term (HAR-GARCH-NIG). The NIG distribution is a rather flexible distribution and is used in finance to model asymmetric behaviour and heavy tails in financial data; see further explanation in Section 2.3.1. The first to use the NIG distribution in a financial application was Barndorff-Nielsen (1997). In Section 2.3.1, I discuss the NIG distribution in more detail. I am

[1] I used the Rugarch package to estimate these models in my empirical research, see Ghalanos (2020)

not including a Log HAR-GARCH-NIG model on RV, which assumes a NIG distribution for the innovation term. There is no analytical solution for reversing the log transformation in terms of the NIG distribution, to the best of my knowledge. To be more clear, the log realized volatility would be NIG distributed; however, to the best of my knowledge, the CDF of the distribution of the realized volatility has no direct analytical solution. Using a change of variables method for the pdf of log realized volatility, you could get the pdf for the realized volatility. Afterwards, using numerical integration or the Markov Chain Monte Carlo method, you could get an estimate of the CDF of the distribution for the realized volatility. As this is not the main goal or model of my research but a benchmark, this is out of the scope of my research.

To be more comparable to Corsi et al. (2008), I do include their two top-performing models in my paper. The first of the two models is the log HAR-GARCH model on the realized variance with a normality assumption on the innovation term (Log HAR-GARCH-N(Rvar)). Second, I include the log HAR-GARCH on the realized variance, assuming a standard NIG distribution for the innovation term (Log HAR-GARCH-NIG(Rvar)). Reverting the resulting distribution back to a distribution for the realized volatility would need a change of variables, and to the best of my knowledge, this does not yield an analytical solution for the CDF. The described earlier methods could be used to get an estimate of the CDF; however, this is excluded from my research.

2.3 NGboost

Understanding gradient boosting

Prior to exploring NGBoost, an understanding of gradient boosting is needed, as introduced by Friedman (2001). It is a supervised ensemble machine learning technique that combines weak learners. It builds a model in stages, creating a series of base learners, which are commonly shallow decision trees. Each tree improves upon the previous one. The idea is to start with a simple initial model. At each boosting iteration, the algorithm calculates the negative gradient of the loss function with respect to the current model's predictions for each data point in the training set. These gradients are referred to as pseudo-residuals, which provide guidance on how to modify your predictions to minimize the loss by indicating the direction and magnitude of the steepest descent in the loss landscape. The new decision tree then uses these pseudo-residuals as target variables for training. This tree does not predict the actual target variable but instead tries to predict the pseudo-residuals. The forecast of the current boosting iteration for the actual target variable is then given by combining the forecast of the last boosting step and the prediction for the pseudo-residues times a learning rate. This learning rate is normally set low to prevent overfitting. Generally, gradient boosting attempts to forecast a single value, hence the introduction of NGboost, which attempts to forecast parametric distributions.

NGboost

The NGboost, as developed by Duan et al. (2020), is an algorithm that expands the functionalities of gradient boosting by incorporating probabilistic predictions. The NGboost addresses the shortcomings of conventional gradient-boosting methods by utilizing the concept of natural

gradients. Similar to gradient boosting, the NGboost needs a base learner. This can be any base learner. I will use a decision tree, which is also used by Duan et al. (2020)[2]. Decision trees are flexible and able to capture complex relationships. The NGboost forecasts a parametric distribution, which is different from the previously discussed gradient boosting. In Section 2.3.1, my distributional choice is discussed. Instead of a loss function, a scoring rule is needed, as those are able to compare different estimated probability distributions.As described by Gneiting and Raftery (2007), a proper scoring rule is a metric used to assess the accuracy of probabilistic forecasts by assigning a numerical score based on the forecast probability and the actual outcome ω. It is defined as proper as it rewards the highest score in expectation of the true distribution Q of the actual outcomes, higher or equal to any other distribution P. It should satisfy the following:

$$E_{\omega\sim Q}[S(Q,\omega)] \leq E_{\omega\sim Q}[S(P,\omega)] \quad \text{for all } P, Q. \tag{2.7}$$

For my research, similar to Duan et al. (2020), I use the logarithmic score for the NGboost. Gneiting and Raftery (2007) argues it is a strictly proper scoring rule, as formula 2.7 is only equal when Q is equal to P. The logarithm score is given as $\text{LogS}(p,\omega) = \log p(\omega)$, where p is the probability density function. Minimizing the negative of LogS gives the maximum likelihood estimator. The NGboost can use all proper scoring rules; in the paper of Duan et al. (2020), they also introduce the continuous ranked probability score (CRPS). The use of the ordinary gradient becomes invalid when a distribution and scoring rule replace the point prediction and loss function. As proven by Duan et al. (2020), "The problem is that "distance" between two parameter values does not correspond to an appropriate "distance" between the distributions that those parameters identify"(p. 3). In other words, the ordinary gradient is not invariant to reparameterization. First, instead of distance, a divergence between scoring rules is defined as by Dawid and Musio (2014):

$$D_S(Q||P) = E_{\omega\sim Q}[S(P,\omega)] - E_{\omega\sim Q}[S(Q,\omega)],$$

As argued by Dawid and Musio (2014), "locally, $D(P, P + dP)$ defines a Riemannian Metric on P" (p. 171). This means that as an infinitely small step is taken for P, the divergence creates a statistical manifold where each point is a probability distribution, and there is a Riemannian Metric that can be used to scale the gradient in this statistical manifold. This is where the natural gradient comes in. The natural gradient is the steepest ascent in this statistical manifold and invariant to reparametrization; the general form is introduced by Duan et al. (2020) as,

$$\widetilde{\nabla} S(\theta,\omega) \propto \lim_{\epsilon\to 0} \operatorname*{arg\,max}_{d:D_S(P_\theta||P_{\theta+d})=\epsilon} S(\theta+d,\omega)$$

which if solved gives,

$$\widetilde{\nabla} S(\theta,\omega) \propto \mathcal{I}_S(\theta)^{-1}\nabla S(\theta,\omega).$$

$\mathcal{I}_S(\theta)$ is the Riemannian Metric. Using this definition, Duan et al. (2020) creates a generalized natural gradient, which can be used for all proper scoring rules. As I use LogS, which induces

[2]For implementation of the NGboost models, parts of the code by Duan et al. (2020) are used, the code can be found in the github repository: https://github.com/stanfordmlgroup/ngboost.

a Kullback-Leiber divergence (Dawid, 2007), the natural gradient is given as:

$$\widetilde{\nabla}\text{LogS}(\theta, \omega) \propto \mathcal{I}_{\text{LogS}}(\theta)^{-1} \nabla \text{LogS}(\theta, \omega),$$

The Kullback-Leiber divergence is the original statistical manifold for which the natural gradient is defined by Martens (2020). For this divergence, the Riemannian Metric $\mathcal{I}_{\text{LogS}}(\theta)$ is given by the Fischer information from ω on P_θ:

$$\mathcal{I}_{\text{LogS}}(\theta) = E_{\omega \sim P_\theta} \left[\nabla_\theta \mathcal{L}(\theta, \omega) \nabla_\theta \mathcal{L}(\theta, \omega)^\top \right].$$

One of the primary obstacles encountered when utilizing gradient boosting for probabilistic forecasting is the parallel optimization of numerous parameters that define the probability distribution. NGBoost effectively tackles this obstacle by employing a multi-parameter boosting methodology. A conventional gradient boosting method, when used for distributional forecasting, has the tendency to optimize all parameters simultaneously. NGBoost optimizes each parameter independently. This methodology guarantees that proper attention is given to every parameter, thereby stopping the algorithm from running into suboptimal solutions. The natural gradient assures that the parameters approach convergence at a similar rate in each iteration, regardless of differences in means, variances, and distances from the initial marginal distribution. Furthermore, all parameters remain subject to a uniform scaling factor $\rho^{(m)}$.

NGboost algorithm

In the paper by citeAngboosthoofd, the NGboost algorithm is described in detail. I outline the key components, and Algorithm 1 provides a pseudocode. The NGboost training algorithm is very similar to gradient boosting; first, an initial prediction θ^0 is made. Note that θ contains as many values as the parameters in the chosen parametric distribution. Next, each boosting iteration, m, uses the natural gradient as a target variable to train a new decision tree. It is important to note that each parameter in θ gets its own decision tree in each boosting round. These trees are negatively scaled by a common scaling factor $-\rho^{(m)}$, which is the same for all parameters. This is done to ensure the score improves most efficiently. Line 16 of Algorithm 1 then updates the parameter vector. Before adding the already scaled decision trees to the old parameter vector, the decision trees are further scaled by a small negative learning rate η. The output of the NGboost for a new x_{1+1} is a conditional distribution from which the parameters are obtained by adding M decision trees trained in Algorithm 1. The decision trees are scaled similarly to those in the training process. It is important to note that all parameters of the chosen parametric distribution have their own sequence of decision trees. The final forecast for the distribution's parameters, θ, is given as follows:

$$\theta_{t+1} = \theta^{(0)} - \eta \sum_{m=1}^{M} \rho^{(m)} \cdot dt^{(m)}(x_{t+1}).$$

Where $dt^{(m)}$ contains a different decision tree for all parameters, and uniform scaling factor $rho^{(m)}$ is a singular value.

Algorithm 1 Natural Gradient Boosting (NGBoost) Algorithm

Input:
Training data $\{(x_i, y_i)\}_{i=1}^n$, features x_i and target variables y_i
Base learner: Decision Tree $(dt)_l$,parametric distribution with parameters θ, proper scoring rule: $LogS$
Number of boosting iterations M, Learning rate η
Output:
A probabilistic model that estimates the distribution of y given x, by using scalings and base learners $\{(\rho^{(m)}, dt^{(m)})\}_{m=1}^M$.
Initialize distribution parameters θ_0 by $\arg\min_\theta \sum_{i=1}^n LogS(\theta, y_i)$
for $t = 1$ to M **do**
    for i=1 to n **do**
        Compute natural gradient $g_i^m = \mathcal{I}_{\text{LogS}}(\theta_i^{m-1})^{-1} \nabla \text{LogS}(\theta_i^{m-1}, y_i)$
    end for
    train decision tree dt^m with target variable g_i^m and features x_i
    Calculate scaling factor $\rho^{(m)}$ to ensure score improves most efficiently
    $\rho^{(m)} \leftarrow \arg\min_\rho \sum_{i=1}^n LogS\left(\theta^{(m-1)} - \rho \cdot (dt)^{(m)}(x_i), y_i\right)$
    for i = 1 to n **do**
        $\theta_i^{(m)} \leftarrow \theta_i^{(m-1)} - \eta\left(\rho^{(m)} \cdot f^{(m)}(x_i)\right)$
    end for
end for

NGboost target variable

In my research, the NGboost algorithm serves two purposes. First, I apply the NGboost directly to the realized volatility. Using three different parametric distributions, I generate one-day-ahead distributional forecasts for the RV. Second, I use the in-sample residuals η of a HAR model as the target variable. In this way, I predict the one-day-ahead conditional distribution for the errors of a HAR model. Combining this with the deterministic forecast for the RV of a HAR model, $\widetilde{RV_{t+1}}^{(har)}$, gives again a one-day-ahead conditional distributional forecast for the RV. You can extend this modelling approach to any RV forecasting model, using the NGboost as an overlay model to correct the point forecast and estimate the distribution using the residuals from the main model. The predicted conditional distribution is given as:

$$P\left(RV_{t+1} \mid I_T\right) = \widetilde{RV_{t+1}}^{(har)} + P_{\theta_{t+1}}\left(\epsilon_{t+1} \mid I_T\right)$$

Throughout my paper, when the NGboost is used directly on RV, I refer to it as the NGboost. When I train the NGBoost using residuals from a HAR model, I refer to it as the ResNGBoost. I use both the residuals of the HAR defined in Function 2.2 and the residuals of the log HAR defined in Function 2.3. The idea of using the NGboost as an overlay model is based on the paper by Almeida, Fan, Freire and Tang (2023), in which they successfully use a machine learning method to improve predictions of the implied volatility surfaces.

2.3.1 Distributions NGboost

The goal of my research is to forecast a parametric distribution for the RV using the NGboost. In my choice of distribution, I stay close to Corsi et al. (2008). Similar to Corsi et al. (2008), I

use the normal, lognormal, and normal inverse gaussian (NIG) distributions. By staying close to their distributional choice, the performance of the NGboost can be compared to the benchmarks I selected. By using the same distributions, the difference in modeling can solely explain the difference in performance. Both the normal and lognormal distributions are well known. The normal is a symmetric distribution, whereas the log normal distribution is positively skewed, allowing for longer right tails. According to Corsi et al. (2008), the NIG distribution is "flexible and able to reproduce a range of symmetric and asymmetric distributions" (p. 63). This is due to the fact that the NIG distribution is defined by four parameters; the density function is given as:

$$f(x;\alpha,\beta,\mu,\delta) = \frac{\alpha}{\pi}\frac{K_1(\alpha\delta\sqrt{1+(\frac{x-\mu}{\delta})^2})}{\sqrt{1+(\frac{x-\mu}{\delta})^2}}\exp\left\{\delta(\sqrt{\alpha^2-\beta^2}+\beta(\frac{(x-\mu)}{\delta}))\right\}, \tag{2.8}$$

K_1 is the modified Bessel function of the second kind of order 1, μ is the location parameter, $\beta \in [-\alpha, \alpha]$ is the asymmetry parameter, $\alpha > 0$ is the tail heaviness parameter, and δ is the shape parameter. Due to α and β, the NIG distribution is able to produce skewed distributions with heavier tails.

In my empirical research, discussed in Section 3, I use two approximations for the NIG distribution. Firstly, to the best of my knowledge, there is no analytical solution for Fischer information available. Instead, I use approximation using the Monte Carlo method as implemented by 2.3. Secondly, the NIG distribution does not have an analytical solution for the gradient of LogS; this is due to the very complicated probability distribution function as given in Equation 2.8. A numerical approximation method is used to approximate this gradient.

Assuming a normal or NIG distribution for the realized volatility, there is a problem with misspecification, which is not discussed by Corsi et al. (2008). Because the realized volatility is strictly positive, assuming a distribution that allows for negative tails leads to a certain misspecification. How severe this misspecification is depends on the proportion of the tail of the distribution that is negative. The log normal distribution does not suffer from misspecification as it does not allow for negative tails.

2.3.2 Pooled NGboost

Extending the NGBoost on singular stocks, I use the NGboost on a pooled dataset (pooled NGboost). As before, I use both the pooled Ngboost directly on the realized volatility and the pooled NGboost on the residuals of a HAR model. Pooling data from stocks together to forecast realized volatility is also done by Bollerslev et al. (2018), who show that by utilizing similarities in realized volatility patterns among stocks, they are able to produce improved out-of-sample forecasts of realized volatility. According to Kleen and Tetereva (2022), while pooled estimators may have some bias due to heterogeneity, they also yield significant efficiency improvements. Bollerslev et al. (2018) utilizes a centering method for most of its models. For my pooled NGboost, this is not possible for all distributions, as the lognormal distribution cannot contain negative values, so training would fail when the data is centered. Therefore, in this study, the pooled, uncentered realized volatility estimates of all stocks are used. For my pooled ResNGboost I use the pooled residuals of the individual estimated (log) HAR models.

2.3.3 Cluster NGBoost

Based on the paper by Oh and Patton (2023), I introduce a cluster NGBoost algorithm. Rather than grouping all stocks together, as is the case with the pooled NGboost, I create clusters and train a separate NGboost model for each cluster. In the paper of Oh and Patton (2023), they use a cluster optimization algorithm to form a multifactor copula model. They do so by modifying a k-means clustering algorithm to suit their needs. Inspired by their algorithm, I create an algorithm to form optimal clusters that minimize the average continuous ranked probability score (CRPS) across all stocks. The explanation about the CRPS can be found in Section 2.4. Algorithm 2 presents a pseudocode of the optimization algorithm. First of all, a suitable number of clusters Z and optimization rounds K have to be chosen. In my research, as I work with 29 different stocks, as described in Section 3.1, so I use a Z between 2 and 29. Taking $Z = 1$ would represent the pooled NGboost. In my empirical research, I typically found a local minimum after a maximum of 15 optimization rounds; however, I set K = 50. This is done to ensure the algorithm only stops when reaching a local minimum and not due to reaching the maximum optimization rounds. It is important to note that the Cluster NGboost Algorithm ends up at a local minimum, which may be the global minimum. Increasing the number of runs of Algorithm 2 with varying Z increases the likelihood of finding or approaching the global minimum. There is, however, no mathematical proof that the global minimum is found. In my empirical research I run the algorithm 500 times for both cluster specifications. Another important detail is that Algorithm 2 is defined with just one training set and one validation set for each stock. During my empirical research, I modify the algorithm to allow for the estimation of multiple NGboosts and testing across multiple validation sets. To prevent clusters being optimized for only one training set and one validation set, the modified algorithm computes the average CRPS across all validation sets. The resulting clusters are more robust, as they are optimized over a longer time period. In the next enumeration, I explain how the standard optimization algorithm works.

1. **Initial Clustering and NGBoost Model Training**: An initial cluster assignment C_0 is generated in such a way that all clusters contain at least one stock. For every cluster z, the data of all stocks q is pooled together. This pooled data is then used to estimate a NGboost using Algorithm 1. For all stocks within a cluster, the CRPS($S_q z$) is calculated. The algorithm keeps track of the combined CRPS for all stocks for the current cluster assignment (C_{h-1}); this combined score is defined as $S_{C_{h-1}}$.

2. **Model Fit Assessment**: The calculated combined score is compared to the best combined score so far, which is given as $S_{C_{h-1-d}}$. Where d is the distance between the current iteration and the iteration that had the best score. If the combined score of this iteration is lower, d is set back to one, and the current cluster assignment C_{h-1} is the best cluster assignment so far. Step 3 is skipped whenever the current cluster assignment is not the best cluster assignment so far.

3. **Reassignment Consideration**: In this stage, the CRPS is utilized to determine whether an NGboost model from a different cluster provides a superior fit for a stock. This implies that there are as many check-ups as the number of stocks multiplied by the number of clusters minus one. These calculated scores are defined as $S_q u$. When a NGboost model

of another cluster offers a more suitable fit for a given stock, that stock is allocated to a list of potential re-allocations. If multiple clusters provide a more optimal fit, the cluster with the lowest CRPS for a given stock is selected. The list of potential reassignments is then sorted. Stocks that demonstrate the most substantial potential improvement are prioritized at the top of the list. This list is saved until a more optimal cluster assignment is found. This step corresponds to lines 20 to 34 in Algorithm 2.

4. **Reassignment Decision**: The sorted re-allocation list is used to re-allocate a number of stocks. The number of stocks that are allowed to be re-allocated depends on the difference between the current iteration and the iteration that was responsible for the most optimal cluster assignment. The stocks with the highest priority according to the list are reassigned in accordance with the permitted number of re-allocations. After reassignment, possible empty clusters are deleted before starting a new optimization round. The algorithm starts again at step 1, excluding the initial cluster assignment..

5. **Max re-allocations and Convergence**: This is a crucial part of the optimization algorithm. It ensures that the optimal cluster assignment's CRPS score only improves and cannot get worse. It achieves this by tracking the distance between the best cluster assignment and the current optimization round. All stocks in the list undergo reassignment if the current cluster assignment yields the lowest cumulative sum in terms of CRPS. If the discrepancy between the current iteration and the optimal iteration is one, ten assets are re-allocated. If the discrepancy is 2, only 5 are allowed to be reassigned. If it is 3, only two stocks are reassigned; if it is 4, only one stock is reassigned. If the discrepancy is 5, the algorithm stops; even re-allocating only the most prioritized stock does not result in a lower overall CRPS, concluding that a local optimum has been reached. In Algorithm 2, this step corresponds with lines 36 to 42.

There is an important remark about this optimization routine. I assume that reassigning the stock that has the highest priority does not provide a better overall score, then a local optimum has been reached. This holds true most of the time; however, in theory, there can be stock that has lower priority, which would improve the overall CRPS score when reassigned. This issue is not addressed, as this would severely use more computational power, and due to time limitations, it is not possible to address this in my research. This is an important limitation of the algorithm used and could be addressed in further research.

2.3.4 Optimizing NGboost

NGBoost, like the majority of machine learning algorithms, requires hyper-parameter optimization to obtain optimal results. I go over the parameters that can be optimized in the case of NGBoost; however, the optimization itself is not part of my research. The parameters and hyper-parameters to optimize are included for completeness. The learning rate η and number of boosting iterations are the most important ones; a lower learning rate is often used to prevent overfitting. When the database is large, a mini-batch fraction can ensure shorter run times, which also reduces overfitting. While the mini-batch fraction selects rows of the data, there is also column sampling, which can prevent overfitting as well as ensure diversity in the selection

Algorithm 2 Cluster NGBoost Algorithm

Input:
Stocks Q, which all have training data $\{(x_i, y_i)\}_{i=1}^{n}$. features x_i and target variables y_i.
Validation set $\{(x_i, y_i)\}_{i=n+1}^{t}$ for each q in Q
Number of clusters Z , maximal number of optimization rounds K, Scoring rule CRPS: S
Output: A probabilistic model for every cluster that estimates the distribution of y given x, by using scalings and base learners $\{(\rho^{(m)}, dt^{(m)})\}_{m=1}^{M}$.
Initialize random cluster assignments C_0 , so that Z clusters contain minimal 1 stock q.
Set distance between clusters d = 1 and total score Cluster ID -1 to $S_{C_{-1}} = \infty$
for $h = 1$ to K **do**
    for z in C_{h-1} **do**
        Pool training data of all $q \in z$
        use Algorithm 1 to estimate NGboost model NG_z using pooled training data.
        for $q \in z$ **do**
            Predict conditional distribution for all i in validation set of q using NG_z
            Calculate average CRPS over validation set: S_{qz}, save S_{qz}
            $S_{C_{h-1}} = S_{C_{h-1}} + S_{qz}$, keep track of total CRPS of Cluster ID C_{h-1}
        end for
    end for
    if $S_{C_{h-1}} < S_{C_{h-1-d}}$ **then**
        new best cluster found so: Set MaxAll = ∞ and Set d=0.
        for $q \in Q$ **do**
            Set dif$S_q = 0$, which represents best difference in Score
            for u in C_{h-1}, for which $q \notin u$ **do**
                Repeat line 13 and 14 with NG_u and get S_{qu}
                score diff for q amid original cluster and cluster u dif$S_{quz} = S_{qz}$ - S_{qu}
                if dif$S_{quz} >$ difS_q **then**
                    dif$S_q = difS_{quz}$, and save cluster allocation q to u, [q,u]
                end if
            end for
            if dif$S_q > 0$ **then**
                extend dataframe DF, which saves most optimal difS_q and [q,u] for stock q
            end if
        end for
        Sort DF in terms of difS_q from high to low.
        The sorted DF now contains all stocks that have lower S in another cluster u
    end if
    Important: the following If loop is repeated for the different max allocations in a
    a = [∞,10,5,2,1] , where a[1]= 10.
    if MaxAll = ∞ **then**
        From C_{h-1-d} re-allocate MaxAll q in DF, remove empty clusters ,this create Clusterization ID C_h. Set: MaxAll = $a[d]$ and d= d+1
    else
        Stop algorithm, local minimum is found, return C_{h-1-d}
    end if
end for
If algorithm did not stop, return C_{h-1-d} and return algorithm not optimized, use higher K.

of features used for the trees. Column sampling is the process of selecting a subset of all features in each boosting iteration. Now for my chosen base learner, there are also a couple of hyperparameters that can be optimized. The depth of the decision tree can be important. Next, for the tree, you could set the minimum number of samples required in a split and the minimum number of samples required in each split node. Cross-validation could optimize these parameters and hyper-parameters for optimal NGboost results. As a solution for this optimization, Shen et al. (2022) and Hu, Sun, Han, Hao and Pei (2022) use a tree-structured Parzen estimator. Next to the optimization of parameters and hyper-parameters, a paper by Mitrentsis and Lens (2022b) uses Shapley Additive Explanations to optimize the use of features by disregarding features that show low SHAP values and thereby have low influence in the training of the model.

2.4 Evaluation metrics

To compare the models, different metrics are used to evaluate the forecasts of the different models. I use the Continuous Ranked Probability Score (CRPS) and the logarithmic score (LogS) to assess the distributional forecasts. The Squared Error(SE) loss is used to evaluate point forecasting. In addition, I use an asymmetrical piecewise linear scoring rule (APL) to assess the performance of the quantile forecasts. To give more insights into how the NGboost forecasts depend on different features, I also include an accumulated local effects analysis.

2.4.1 CRPS, LogS, SE and APL

To assess and evaluate the forecast accuracy, I use the CRPS and the LogS. Both are proper scoring rules as defined by Gneiting and Raftery (2007). I use the LogS as this scoring rule is also used by Corsi et al. (2008). I use the CRPS because, according to Kleen (2023), the CRPS is gross-error insensitive. Furthermore, their research demonstrates that the CRPS not only exhibits the least sensitivity to observational errors in simulation, but also excels in a real-time example. It is seen that the CRPS is less sensitive to noisy proxies of realized volatility than the LogS. Due to the results of Kleen (2023), I use the LogS as a confirmation scoring rule for the results of the CRPS. The two scoring rules are given as:

$$LogS(f_{t+1|t}, y_{t+1}) = \log f_{t+1|t}(y_{t+1}),$$
$$CRPS(F, y_{t+1}) = \int_{-\infty}^{y_{t+1}} F(z)^2 dz + \int_{y_{t+1}}^{\infty} (1 - F(z))^2 dz.$$

Where $f_{t+1|t}$ is the predicted density, y_{t+1} is the realized value. While F(z) is the predicted cumulative distribution (CDF). I take the negative of the LogS to make it a negatively oriented scoring rule; throughout my paper, I still refer to this as the LogS.

Next, to assess the performance of the point forecasting, I use the SE loss. According to Patton (2011), this is a good loss function that assigns more weight to larger forecast errors. Corsi et al. (2008) also uses this loss function. It is given as

$$SE(\widehat{y}_{t+1|t}, y_{t+1}) = (y_{t+1} - \widehat{y}_{t+1|t})^2$$

Where y_{t+1} is still the realized value, whereas $\widehat{y}_{t+1}$ is the predicted value by the model.

To predict quantiles from a predicted CDF, you are essentially looking for the value x at which the CDF reaches a certain probability threshold. For a given probability τ, where $0 < \tau < 1$, the τ-th quantile q_τ is defined by the value for which the CDF equals τ:

$$F(q_\tau) = \tau.$$

In other words, q_τ is the point at which the proportion τ of the distribution lies below it. You can solve this equation for q_τ to find the quantile. In practice, this is achieved by inverting the CDF. If CDF is invertible, the τ-th quantile is given by:

$$q_\tau = F^{-1}(\tau)$$

However, whenever a CDF of parametric distributions is not invertible, numerical approximations are used.

To assess the performance of the forecasted quantile q_τ, I use an asymmetric piecewise linear scoring function as given by Gneiting (2011b), and according to Gneiting and Raftery (2007), it is a proper scoring rule for quantiles. This scoring function is defined as follows:

$$APL(\widehat{q}_{\tau,t+1|t}, y_{t+1}) = (1(\widehat{q}_{\tau,t+1|t} \geq y_{t+1}) - \tau) \times (\ \widehat{q}_{\tau,t+1|t} - y_{t+1}).$$

Where $\hat{q}_{\tau,t+1|t}$ is the predicted quantile. This scoring rule weighs the difference between the predicted quantile and the realization. Whenever $\hat{q}_{\tau,t+1|t}$ is smaller than y_{t+1}, the weight of the absolute difference is the value of τ, and if it is the other way around, the weight is $1 - \tau$. This results in a scoring rule for the 99 percent quantile, in which observations that fall above the predicted quantile carry a heavier weight.

2.4.2 Reporting the score and loss functions

As done by Kleen and Tetereva (2022), I calculate the average loss for every score/loss function S for every stock/index i for each model j across the out of sample period as,

$$L_i^j = \frac{1}{n} \sum_{t=0}^{n-1} S(\mathrm{RV}_{t+1}, \widehat{\mathrm{z}}_{t+1|t}).$$

Where the out-of-sample period exists of n number of observations and starts at t = 1. Furthermore, $\widehat{\mathrm{z}}_{t+1|t}$ is different for every score or loss function; it can be the predicted pdf, predicted CDF, predicted mean forecast $\widehat{\mathrm{RV}}_{t+1|t}$, or the predicted quantile $\widehat{q}_{\tau,t+1|t}$.

I use the formulas for the average loss ratio (AL) and the loss rate (LR) defined by Kleen and Tetereva (2022) as,

$$AL^j = \frac{1}{K} \sum_{i=1}^{K} \frac{L_i^j}{L_i^B},$$

$$LR^j = \frac{1}{K} \sum_{i=1}^{K} 1_{\{L_i^j < L_i^B\}}.$$

Where L_i^B is the average loss of the benchmark for stock i. The AL is calculated as the average loss across stocks relative to the benchmark. The LR quantifies the proportion of instances where model j provides better predictions compared to a benchmark model. It effectively measures the frequency with which model j achieves a lower loss than the benchmark across all evaluated stocks. In addition to these two, I add the significant loss rate (LR*), which reports the frequency with which a model achieves a significantly lower loss than the benchmark. To test for significance, I use the Giacomini-White test (GW) as introduced by Giacomini and White (2006).

However, the significant loss rate does not compare all models with each other. Therefore, to significantly test all models against each other, I also use the model confidence set MCS approach introduced by Hansen et al. (2011).

2.4.3 Model confidence set

The MCS is a statistical procedure developed to assess the relative performance of a collection of predictive models. This methodology aims to identify a set of superior models from a larger collection, guaranteeing a certain level of confidence in the selected set.

Hansen et al. (2011), the first to use MCS, denotes the set of all competing models by $\mathcal{M}$. For each stock i and each score/loss function S and each model j and l in $\mathcal{M}$, I define:

$$d_{i,t+1}^{j,l} = S(\mathrm{RV}_{t+1}, \widehat{\mathrm{z}}_{t+1|t}^{j}) - S(\mathrm{RV}_{t+1}, \widehat{\mathrm{z}}_{t+1|t}^{l}). \tag{2.9}$$

This is the difference in S for the models j and l, where $\widehat{\mathrm{z}}_{t+1|t}$ is the same as defined before. I compute the average loss difference per stock, $\bar{d}_i^{j,l}$. Note that originally, the MCS by Hansen et al. (2011) (2011) was used for loss functions as the SE and not for scoring rules. For the MCS to be used on scoring rules, it is important that the models are ranked in a similar way as Hansen et al. (2011). Their paper bases the ranking of model alternatives on their expected loss, with a preference for model j over model l for stock i if $\bar{d}_i^{j,l}$ is less than zero, indicating a lower expected loss for model j. So the scoring rules have to be changed in such a way that lower scores are preferred. For the LogS this means I use the negative LogS, to make sure lower scores are preferred. CRPS and APL already exhibit a negative orientation.

According to the paper by Hansen et al. (2011), you can now calculate the test statistic as follows:

$$t_i^{j,l} = \frac{\bar{d}_i^{j,l}}{\sqrt{\mathrm{Var}(d_i^{j,l})}} \quad \text{for} j, l \in \mathcal{M}. \tag{2.10}$$

As in detail described by Hansen et al. (2011) all these t-statistics together form the basis for the null hypothesis,

$$H_{0,\mathcal{M}} : E(d_{i,t+1}^{j,l}) = 0 \text{ for all } \; j, l \in \mathcal{M}.$$

The alternative hypothesis is that not all models perform equally. The MSC test statistic used to test this null hypothesis is given as follows:

$$T_{i,\mathcal{M}} = \max_{j \in \mathcal{M}} |t_i^{j,l}|. \tag{2.11}$$

If the null hypothesis is rejected, an elimination rule is used, and the worst-performing model is removed from $\mathcal{M}$. This is repeated until the MCS test statistic is not rejected; the remaining models in $\mathcal{M}$ are in de MCS for stock i denoted by $\mathcal{M}_{i,MCS}$. These models are included by the significance level α; I use $\alpha = 0.1$, similar to Kleen and Tetereva (2022) and (Liu et al., 2015). As discussed by Hansen et al. (2011), the asymptotic distribution of the MSC test static is non standard; similarly to Kleen and Tetereva (2022), this is approximated by block-bootstrapping using a block length equal to 22, and in my analyses, a 5000 bootstrap replication at each stage is sufficient. [3]. As used by Kleen and Tetereva (2022), I use an aggregate measure to report the results across stocks; it reports the proportion of stocks for which model j is included in the confidence set and is given as

$$MCSR^j = \frac{1}{K} \sum_{i=1}^{K} 1_{j \in \mathcal{M}_{i,MCS}}.$$

2.4.4 Accumlated local effects and variable importance

To open up the black box created by the use of the NGboost, I use a method called the accumaleted local effects (ALE). The technique is introduced by Apley and Zhu (2020). As described by Apley and Zhu (2020), this method is especially better to use than the partial dependence plot when features are strongly correlated. My features are strongly correlated; for example, the daily, weekly, and monthly realized volatility. Because there is a high autocorrelation in the realized volatility, these features will be strongly correlated. With the use of the accumulated effects, I am able to give insights on the effects of different features on the predicted parameters of the distribution. The ALE for a feature x_s is computed by integrating the expected partial derivative of the model with respect to x_s, conditioned on the value of x_s. The general formula for the ALE is defined as,

$$f_{s,ALE}(x_s) = \int_{z_{0,s}}^{x_s} E\left[\frac{\partial f(X_s, X_c)}{\partial X_s} | X_s = z_s\right] dz_s - \text{constant}.$$

Where $z_{0,s}$ is the minimum value of x_s, X_c include all features except for X_s, $\frac{\partial \hat{f}(X_s, X_c)}{\partial X_s}$ is the partial derivative of the model's prediction with respect to the feature x_s, indicating the local effect of a small change in x_s on the prediction. The model predictions in my research are the parameters of the used distribution. The integral accumulates these local effects from $z_{0,s}$ to x_s, providing a summary of how changes in x_s impact the model's prediction up to that point. The constant is subtracted to ensure the ALE plot is centered at zero, which makes it easier to interpret the ALE plot. To empirically calculate the ALE value for stock i and feature s, I use an approximation similar to the one used by Kleen and Tetereva (2022). However, instead of using equally spaced bins to create intervals for the features, I use the quantiles of the feature's distribution to create bins. First, I use 23 bins that all contain 4 percent of all the stock data for that specific feature. Next, as of the 92 percent quantile, the data is divided into an additional 8 bins that contain 1 percent of the data. In total, this creates 31 bins for every feature. The last

[3]For the MCS procedure, I use the Python package arch.bootstrap.MCS.
See: https://arch.readthedocs.io/en/latest/multiple-comparison/multiple-comparison-reference.html

8 percent is divided into smaller bins as some of my data is heavily skewed. Furthermore, it does not necessarily mean that the 4% bins or the 1% bins contain the same number of observations for a specific stock; the bins are created based on all the stock data and not separately for every stock. This method is used as it ensures the separate ALE plots can be aggregated across bins for the different stocks. The bins are defined as $(\tilde{z}_0, \tilde{z}_1], \ldots, (\tilde{z}_{L-1}, \tilde{z}_L]$. Similarly to Kleen and Tetereva (2022), the uncentered ALE approximation of the local parameter θ' is defined as:

$$\widehat{\widetilde{ALE}}_{s,i}^{\theta'}(z) = \sum_{l:\tilde{z}_l \leq z} \frac{1}{n_l} \sum_{t:z_{s,i,t} \in (\tilde{z}_{l-1}, \tilde{z}_l]} \left[\theta'(z_{(\tilde{z}_l;-s),i,t}) - \theta'(z_{(\tilde{z}_{l-1};-s),i,t}) \right],$$

$z_{.,i,t}$ is a vector that contains all feature values for stock i at time t, whereas $(z_{(\tilde{z}_l;-s),i,t})$ is the same vector, but the value of feature j is changed to the value of the bin $\tilde{z}_l$. To center the ALE function, the mean ALE across different bins is subtracted as follows:

$$\widehat{ALE}_s^{(\theta')}(x) = \widehat{\widetilde{ALE}}_s^{(\theta')}(x) - \frac{1}{L} \sum_{l=1}^{L} \widehat{\widetilde{ALE}}_s^{(\theta')}(x_l).$$

Additionally, as defined by Kleen and Tetereva (2022), I also report the variable important (VI) as the standard deviation of the ALEs per stock.

$$V_{s,i}^{(\theta')} = \sqrt{\frac{1}{T-1} \sum_{t=1}^{T} \left[ALE_{s,i}^{(\theta')}(z_{s,i,t}) \right]^2},$$

T is the total number of observations.

3 Empirical analyses

3.1 Data

My data set contains a range of variables for 29 out of the 186 permanent constituents of the S&P 500 index and the S&P 500 itself[1]. The data set spans from January 1, 2007 to December 30, 2020. The explicit stocks included are: Microsoft (MSFT), Coca Cola (KO), Exxon Mobil (XOM), General Electric (GE), IBM (IBM), Chevron (CVX), Apple (AAPL), Raytheon (RTX), Procter & Gamble (PG), Catterpillar (CAT), Boeing (BA), Dow Chemical[2] (DOW), Pfizer (PFE), Johnson & Johnson (JNJ), 3M (MMM), Merck (MRK), Disney (DIS), Mcdonals (MCD), JPMOrgan Chase (JPM), Walmart (WMT), Nike (NKE), American Express (AXP), Intel (INTC), Travellers (TRV), Verizon (VZ), Home Depot (HD), Cisco (CSCO), Goldman Sachs (GS), UnitedHealth (UNH) [3]. All data is processed high-frequency intraday data. [4] As

[1] S&P 500 index 1 minute open,close, high, and low price data is provided by O. Kleen

[2] Dow Chemical is only included in the validation set, as this stock in my data set is only available until 2017.

[3] These are the exact same stocks as Catania and Proietti (2020) used in their paper, I refer to their results in Section 4.1

[4] The data for specific stocks is provided by O. Kleen,the data set contains daily values for RV, RV+, RV-, RQ,Ret.

can be found in the paper of Kleen and Tetereva (2022), the data is cleaned similarly to Bollerslev et al. (2019) and Bollerslev et al. (2022). The data is created by merging daily CRSP data with NYSE Trade and Quote (TAQ) intraday data. The daily CRSP data files yield daily open and close prices, while the NYSE TAQ provides all other intraday transaction data. The linking tables of Wharton Research Data Services (WRDS) merge the two data sets. The intraday data is cleaned according to Barndorff-Nielsen and Shephard (2002); only trades from the exchange are included that are referenced in the daily CRSP data. Alongside the lagged RV daily, weekly, and monthly, as outlined in Section 2.1 for the HAR model, I incorporate the realized quarticity as a feature for my NGboost models. I use the realized quarticity as a feature as I intend to attempt to construct the model while considering measurement error, as in detail described in the literature section 1.2. The following formulation of realized quarticity for stock i is employed:

$$RQ_{i,t}^{d} = \frac{M}{3}\sum_{j=1}^{M}(r_{i,t-(j-1)\Delta})^4,$$

M and Δ are defined as in Equation 2.1. Similar to the comprehensive HARQ-F model proposed by Bollerslev et al. (2016), which incorporates lagged daily, weekly, and monthly realized quarticity, I utilize the lagged daily, weekly, and monthly realized quarticity ($RQ^{w} = RQ_{t-5:t-2}$, $RQ^{m} = RQ_{t-22:t-6}$). However, based on the findings of Bollerslev et al. (2016), it is expected that the weekly and monthly realized quarticity will be less influential than the daily realized quarticity. In my research, I also include a leverage effect component; I use lagged returns in the same way that I use lagged realized volatility. Corsi and Renò (2012) includes the leverage effect by including a positive and negative component of the returns in the forecasting equation. In contrast, the tree algorithm will effectively address this issue by performing splits at the most optimal locations. The leverage effect will be captured consequently through the utilization of the daily, weekly, and monthly returns ($Ret^{d} = \sum_{j=1}^{M} Ret_{t-(j-1)}$, $Ret^{w} = Ret_{t-5:t-2}$, and $Ret^{m} = Ret_{t-22:t-6}$). Additionally, Kambouroudis, McMillan and Tsakou (2021) supports including lagged returns as features to capture the leverage effect.

The final daily stock-specific features considered are the semi-variances that were initially introduced by Barndorff-Nielsen, Kinnebrock and Shephard (2008), who demonstrates the significant effect of semi-variances on volatility forecasting. The semivariances for positive and negative returns, in that specific order, are defined as follows:

$$RV_{i,t}^{+} = \sum_{j=1}^{M} r_{i,t-(j-1)\Delta}^{2} I_{\{r_{i,t-(j-1)\Delta}>0\}}, \quad RV_{i,t}^{-} = \sum_{j=1}^{M} r_{i,t-(j-1)\Delta}^{2} I_{\{r_{i,t-(j-1)\Delta}<0\}}$$

.

3.1.1 Market features

In this section, I describe every feature that has similar values across stocks. Starting off, I use the CBOE volatility index (VIX), an indicator of US stock volatility that measures the market volatility of the S&P500. Z.-C. Li, Xie, Zeng, Wang and Zhang (2023) demonstrate that using the VIX for volatility (density) forecasting outperforms every other uncertainty measure they

examine. Kambouroudis et al. (2021) confirms this role of the VIX.

I also consider the Cboe VVIX Index, an index that measures the volatility of the VIX. This feature holds considerable potential significance as it relates to the volatility of volatility, a subject that is important in my research, seeing the influence volatility on volatility has on the distribution of the realized volatility, as described in Section 1.2. I have restricted my research to the years 2007 through 2020 due to the limited accessibility of VVIX data, only available as of 2007. Due to the findings of Christie (1982) and Alam and Uddin (2009), which indicate that the yield curve is associated with stock variances, I incorporate this characteristic as well; more specifically, I will consider the yield curve 10–2 years (T10Y2Y).

Some other uncertainty measures that can have influence, according to Z.-C. Li et al. (2023), that I take into account are the financial stress index (FSI), the geopolitical risk index (GPR), and the infectious disease equity market volatility index (IDEMV). Y. Li, Liang, Ma and Wang (2020) finds that the IDEMV has significant influence during the COVID 2020 period.

Additionally, the risk aversion index (RAI), as used by Dai and Chang (2021), is considered. The final feature considered is the daily high frequency 5-min realized volatility of the Dow Jones index (DJI)[5], as according to Zhang, Wang and Ma (2021), the usage of other indices can have a positive influence on forecasting results due to the volatility spillover effect. The summary statistics for each feature are included in Table 3.1[6].

Table 3.1: Average full sample summary statistics

	Mean	Median	Std	Min	Max	Skew	Kurt
Panel A: Stock specific features							
RV^d	1.249	1.006	0.888	0.255	20.569	4.321	37.116
RV^w	1.249	1.016	0.822	0.325	15.057	3.763	25.558
RV^m	1.248	1.027	0.762	0.423	9.260	3.318	18.388
RV^-	1.178	0.497	3.223	0.019	191.373	17.945	615.977
RV^+	1.200	0.511	3.393	0.019	203.565	17.586	606.330
$Rkurt^d$	113.484	1.944	3579.978	0.006	649230.858	106.621	14765.360
$Rkurt^w$	113.292	2.923	2030.710	0.026	163110.498	48.504	2863.424
$Rkurt^m$	113.251	4.202	1149.231	0.065	43572.529	22.219	575.968
Ret^d	0.011	0.029	1.481	-22.047	25.818	-0.073	15.597
Ret^w	0.011	0.030	0.704	-9.273	8.048	-0.290	13.340
Ret^m	0.011	0.022	0.347	-4.099	3.834	-0.672	13.287
Panel B: Market features							
IDEMV	2.116	0.310	7.716	0.000	112.930	6.243	52.489
DJI	1.204	0.431	3.319	0.019	86.240	11.079	193.497
GPR	94.015	88.994	38.815	7.063	413.456	1.295	7.484
T10Y2Y	1.377	1.420	0.837	-0.150	2.910	-0.006	1.858
FSI	0.347	-1.120	4.954	-5.334	29.320	2.505	10.891
RAI	3.232	2.846	1.685	2.425	32.711	9.000	115.695
VIX	19.983	17.060	9.756	9.140	82.690	2.368	10.606
VVIX	91.576	88.920	16.601	59.740	207.590	1.543	7.341

Notes: This table reports the average summary of statistics across all stocks in my analyses. For the features used, the cross-sectional mean, median, standard deviation, minimum, maximum, and skewness are reported. The sample used spans from January 2007 to December 2020.

[5]Data used from the Oxford-Man Institute

[6]A similar table for the S&P 500 is presented in the appendix; see Table A.1.

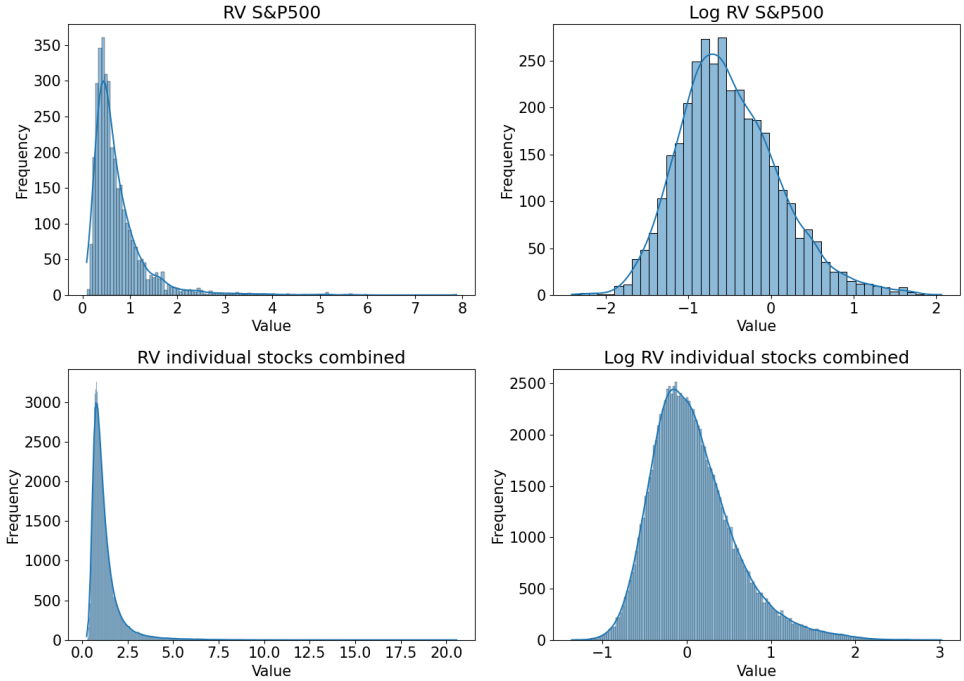
3.1.2 Descriptive statistics

Figure 3.1 shows the kernel distributions of realized volatility and log realized volatility for both the S&P500 and of the 29 individual stocks taken together. The log of the realized volatility typically shows a greater amount of similarity to a normal distribution than the realized volatility itself; in the figure, the right-hand-sided plots look more like a normal distribution. The statistics presented in Table 3.2 for the log RV of the S&P500, where the kurtosis approaches three, provide support for this observation. However, still in the figure, it can be seen that the right-hand side has slightly bigger tails, which is due to the positive skew in Table 3.2.

Realized volatility across all stocks combined mirrors this behavior. Upon taking the log of the realized volatility, the kurtosis approaches three and the skewness approaches zero. However, as illustrated in Figure 3.1 and described in Table 3.2, there continues to be a significant right-tail elongation. This is due to the kurtosis still being high (4.386) and the skewness being considerably higher than 0 (0.923).

Figure A.1 shows the realized volatility over time for individual equities and the SP 500 index. Recognizing significant market events, such as the global economic crisis of 2008/2009 and the coronavirus crisis that emerged at the beginning of 2020. Furthermore, an obvious trend is the considerably lowered realized volatility of the S&P 500 when compared to individual stocks; this can be linked to the implementation of a diversified portfolio. Because of its composition, which consists of 500 unique stocks, the S&P500 functions as a well-diversified portfolio that, in most cases, exhibits a lower level of realized volatility in comparison to its individual components.

Figure 3.1: Histogram with empirical density for realized volatility and log realized volatility



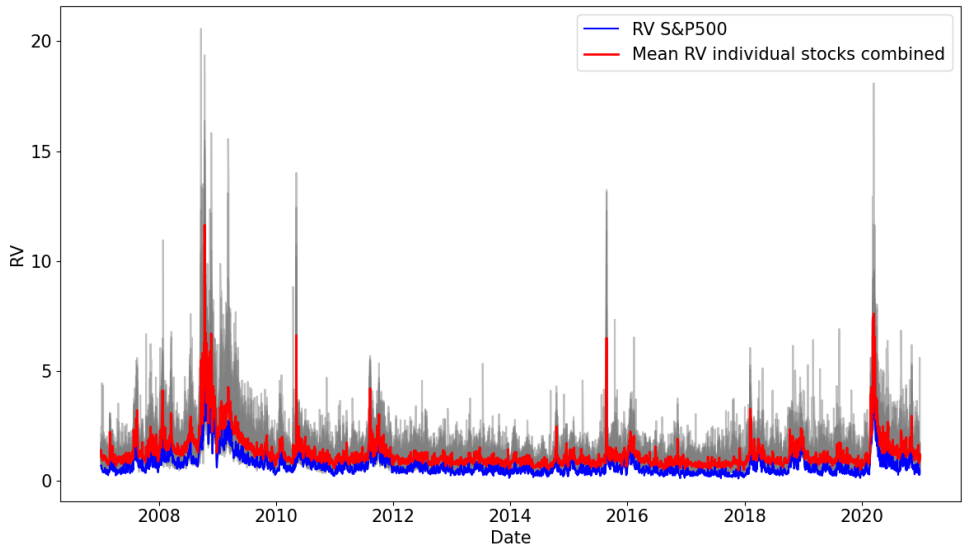
Notes: This figure shows empirical densities for the RV and logaritmic RV. The two top plots show this for S&P500. This is shown in the two bottom plots for the combined RV and log RV of all stock data.

Table 3.2: Average full sample summary statistics for RV and LOG RV

	Mean	Median	Std	Min	Max	Skew	Kurt
RV S&P500	0.744	0.566	0.609	0.089	7.878	3.624	23.622
Log RV S&P500	-0.502	-0.570	0.604	-2.416	2.064	0.568	3.545
RV stocks combined	1.249	1.007	0.888	0.255	20.569	4.321	37.118
Log RV stocks combined	0.077	0.007	0.495	-1.367	3.024	0.923	4.386

Notes: This table reports the average summary of statistics across all stocks in my analyses and for the S&P500. For the realized volatility and log realized volatility, the (cross-sectional) mean, median, standard deviation, minimum, maximum, and skewness are reported. The sample used spans from January 2007 to December 2020.

Figure 3.2: Realized volatility plotted for each stock and SP&500(2007-2020)



Notes: This figure shows the realized volatility for the S&P 500 (blue), the RV of individual stocks (gray), and the mean RV across the individual stocks (red). The sample used spans from January 2007 to December 2020.

3.2 Estimation and model choices

Due to the computational intensity of the models discussed in this paper, it is not feasible to run each model for every stock within the time constraints. Instead, I select models considering their performance on the realized volatility of the S&P 500. In this way, I select the best benchmark from the ones presented in Section 2.2. Similarly, I determine the best distribution for both the individual NGboost and ResNGboost by forecasting the realized volatility of the S&P 500. It is important to mention that I employ two distinct out-of-sample periods. The initial one, which spans from 2012 to 2015, is utilized to select the best models and optimize both cluster NGBoost specifications.This sample is referred to throughout the remainder of this paper as the validation set. The subsequent out-of-sample period, which runs from 2016 to 2020, is referred to as the out-of-sample period. This represents the real out-of-sample period, as no model is selected or optimized for this window. It is important to note that my research is limited by the assumption that the most effective models for the S&P 500 index are also the most effective models for each individual stock.

I employ two different estimation windows. I employ a moving window, as used by Kleen and

Tetereva (2022). Five years of training data are subsequently utilized to develop a model that provides one-day-ahead forecasts over the course of one year. Re-estimation of the model occurs annually. Second, I utilize an expanding window, as described by Corsi et al. (2008), again the model is re-estimated annually. The two methods are not compared directly; rather, for robustness, they are both executed to determine whether the order of models remains constant across different estimation windows. In order to optimize the cluster models and select a suitable benchmark and distributions for the NGboost models, I exclusively utilize the moving window estimation.

The distribution employed for the cluster models is identical to that of the individual NGboost and ResNGBoost, respectively. The subsequent sections discuss the results of the benchmark selection process, as well as the distributional decisions made for the individual NGboost and ResNGboost. The CRPS calculation for these two sections is an approximation based on 100,000 draws from the forecasted pdf. This approach is necessary due to the absence of an analytical solution for the CDF of the NIG distribution. Up to three decimals, these approximations are comparable to the theoretical CRPS for the lognormal and normal distributions. I employ the theoretical CRPS in the remainder of my research. In these two sections I also report the root mean squared instead to address the performance based on the squared error. Following these sections, an analysis of the optimized clusters for both the NGboost and the ResNGBoost is given.

Although it would be beneficial for NGboost models to independently optimize the parameters and hyper-parameters for each model and each stock, I restrict myself from doing so. In order to ensure a reasonable comparison among models, I maintain the parameters and hyper-parameters constant. By doing so, the difference between individual NGboost and cluster NGboost will be solely explainable by clusterization. I use 500 boosting iterations, with a learning rate of 0.01. My base learner is a decision tree regressor using Friedman's MSE. The tree has a maximum depth of 3. The NGBoost uses the LogScore as its scoring rule throughout the entire paper. The results in this paper primarily show the potential of the NGBoost for forecasting realized volatility, as optimizing these and other hyper-parameters leaves room for improvement for the NGBoost models.

Selecting a suitable benchmark

In Section 2.2, I outline seven models. I use the method mentioned in Section 3.2 to determine an appropriate benchmark. The results for the various benchmarks are displayed in Table 3.3. The models are compared to the log HAR-GARCH model on realized volatility. The bottom two models are only shown to be compatible with Corsi et al. (2008); they are not taken into consideration as potential benchmarks as discussed in Section 2.2. It is noteworthy that log transformations improve the logaritimic score (LogS); similar results were found by Corsi et al. (2008).The CRPS validates this by showing a decrease in values following a log transformation. Both LOG HAR and LOG HAR-GARCH, with assumed normal distributions for the error terms, perform well. I select the LOG-HAR model as a benchmark due to its simplicity and the fact that the two models do not differ significantly from one another.

The results are in good agreement with Corsi et al. (2008) in terms of root mean squared

error (RMSE); the differences between the models are not significant and small. The results for LogS are also equivalent; the models that rely directly on realized volatility, such as HAR and HAR-GARCH, perform the worst. The results are slightly better when a HAR-GARCH with an assumed NIG distribution is included (HAR-GARCH-NIG), but log transformations really make a difference, much like in the paper of Corsi et al. (2008).

The fact that the Log HAR-GARCH on the realized variance (log HAR-GARCH(RVar) in absolute terms performs better than the Log HAR-GARCH-NIG(Rvar) is noteworthy. This difference to Corsi et al. (2008) could be due to the usage of different forecasting periods; in the original paper, the difference between the two models is also small, but slightly in favor of the Log HAR-GARCH-NIG(Rvar). The computation of the LogS for these models is similar to the paper of Corsi et al. (2008) [7].

Table 3.3: Benchmark models one-day-ahead forecasting evaluation for RV of S&P500 validation set: 2012 to 2015

Model	RMSE	LogS	CRPS
HAR	1.001	7.204*	1.185*
HAR-GARCH	**0.999**	1.4986*	1.050*
HAR-GARCH-NIG	1.005	1.161*	1.022*
Log HAR	0.9998	1.006	**0.996**
Log HAR-GARCH	1.000	1.000	1.000
Log HAR-GARCH(RVar)	1.002	**0.989**	-
Log HAR-GARCH-NIG(RVar)	1.030	0.998	-

Notes:For the different models discussed in Section 2.2, the root mean squared error, logarithmic score, and continuous ranked probability score are presented. The values present the ratio between the score or loss of the specific model and the score or loss of the log HAR-GARCH. A value below 1 indicates better performance for that specific loss or score than the log HAR-GARCH. The * represents a significant difference at a 5% level based on the Giacomini-White test.

Selecting suitable distribution for the NGboost and ResNGboost

Table 3.5 displays the results of the NGboost models on realized volatility for the three distributions discussed in Section 2.3.1. Table 3.4 displays the results for the ResNGboost with the same distributions. I use two approaches for the ResNGboost. First, I use NGboost on the residuals of a HAR model, as given in Equation 2.2. For this ResNGboost, I use the normal and NIG distributions, as the lognormal distribution cannot include negative values. Secondly, I use the residuals of the log HAR model specified in Equation 2.3. For this ResNGboost, I use a normal distribution, and reversing the log transformation results in a lognormal distribution.

In the tables, I compare all distributions to the lognormal distribution. In both tables, it is evident that the log normal distribution performs best. For the Logs and CRPS, it shows that a lognormal distribution significantly outperforms all other distributions, based on the Giacomini White test with a significance level of 5%. This seems to confirm the observation in Section 3.1, where it became evident that after log transformation, the realized volatility seems to be more

[7] $LogS\left(\{f_{t|t-1}\}_{t=1}^{n}, \{RV_t\}_{t=1}^{n}\right) = n^{-1}\sum_{t=1}^{n}\log\left(\frac{2(f_{t|t-1}(\log RVar_t)}{RV_t}\right)$

normally distributed according to Table 3.2 and Figure 3.1. Furthermore, it is in agreement with the best-performing benchark model, which also assumes a lognormal distribution.

Table 3.4: ResNGboost models one-day-ahead forecasting evaluation for RV of S&P500 validation set: 2012 to 2015

Model	RMSE	LogS	CRPS
Lognormal	1	**1**	**1**
Normal	**0.986**	1.250*	1.050*
NIG	1.041*	1.169*	1.068*

Notes: Lognormal represents an NGboost with a normal distribution on the log HAR residuals. Norm and NIG represent an NGboost with a normal or NIG distribution on the residuals of the HAR. For the three models, the root mean squared error, logarithmic score, and continuous ranked probability score are presented. The values represent the ratio of the specific model's score or loss to the lognormal's score or loss. A value below 1 indicates a better performance than the lognormal distribution for that specific loss or score. The * represents a significant difference at a 5% level based on the Giacomini-White test.

Table 3.5: NGboost models one-day-ahead forecasting evaluation for RV of S&P500 validation set: 2012 to 2015

Model	RMSE	LogS	CRPS
Lognormal	**1**	**1**	**1**
Norm	1.037	1.223*	1.072*
NIG	1.001	1.404*	1.037*

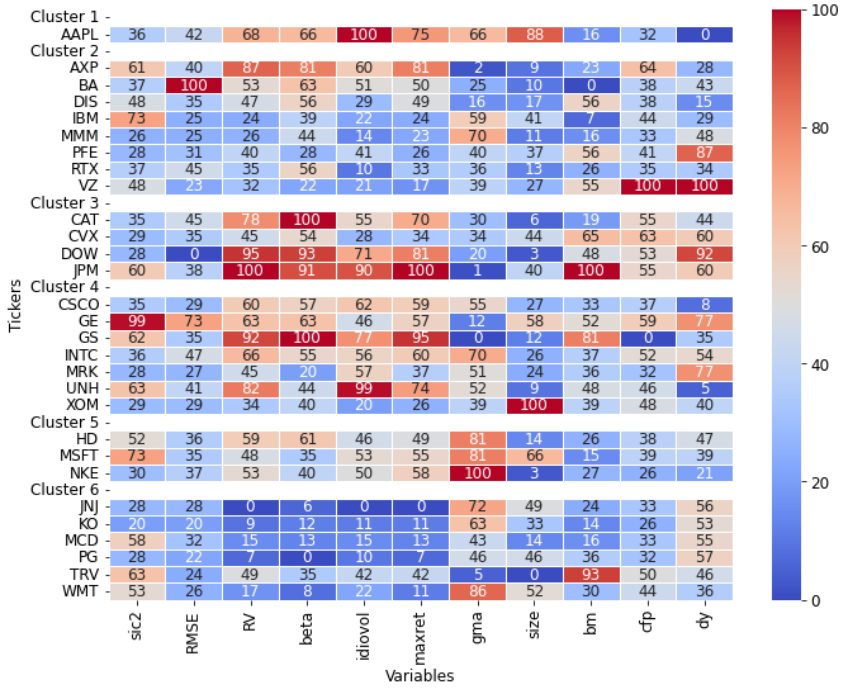
Notes: Lognormal, Norm, and NIG represent an NGboost with a Lognormal, Normal, or NIG distribution on the realized volatility. For the three models, the root mean squared error, logarithmic score, and continuous ranked probability score are presented. The values represent the ratio of the specific model's score or loss to the lognormal's score or loss. A value below 1 indicates a better performance than the lognormal for that specific loss or score. The * represents a significant difference at a 5% level based on the Giacomini-White test.

3.2.1 Pooled and cluster models

I use the lognormal distribution for both the pooled and cluster models, as previous results indicate it is the most suitable distribution. The cluster NGboost models are optimized following the method outlined in Section 2, only using the validation period. I no longer consider the S&P500, but instead use all 29 stocks. The optimized clusters for the cluster NGBoost can be found in Figure 3.3, and the optimized clusters for the cluster ResNGboost can be found in Figure 3.4. For both cluster specifications, using six clusters is optimal. To gain insight into why clusters form, I used the data set of Gu et al. (2020), who collected 94 stock characteristics. All these characteristics have monthly or yearly updated values. I calculated the average of these characteristics for the period from January 2007 to December 2015. This period encompasses the training and validation sets utilized for cluster optimization. Using heatmaps, I made a selection to see which characteristics could explain the clusters. Out of the 94, I selected six characteristics that provide some insights. These are the beta, idiosyncratic volatility (idiovol), maximum daily return (maxret), gross profitability (gma), cash flow to price ratio (cfp), and dividend to price ratio (dy). I also included size and book-to-market (bm) ratio, as those are two well-known characteristics. However, as can be seen in Figures 3.3 and 3.4, the size and bm do not seem to explain the formed clusters. To further analyze the formed clusters, I also include the mean realized volatility and the root mean squared error of the log HAR model

(RSME). The realized volatility is the target variable for the NGboost, hence the inclusion of the mean realized volatility, whereas the residuals of the HAR model are the target variable of the ResNGboost, hence the inclusion of the RMSE of the log HAR. Lastly, the first two digits of the Standard Industrial Classification Code (sic2) are included. In both Figures 3.3 and 3.4, the normalized data is given, except for sic2. The normalized data is used because it makes it easier to spot differences between clusters. All the normalized columns have a range from 0 to 100. For the same time period, I also estimate the contemporaneous correlation matrix of the realized volatility. However, the only interesting observation is that AAPL and DOW have the lowest mean correlation across stocks. They both have a mean correlation of 0.74, with the overall mean correlation across all stocks given as 0.83. For the interested reader, the contemporaneous correlation matrix RV from 2007 to 2015 is included in the appendix as Figure A.1.

Figure 3.3: Optimized clusters for cluster NGboost including heatmap with characteristics



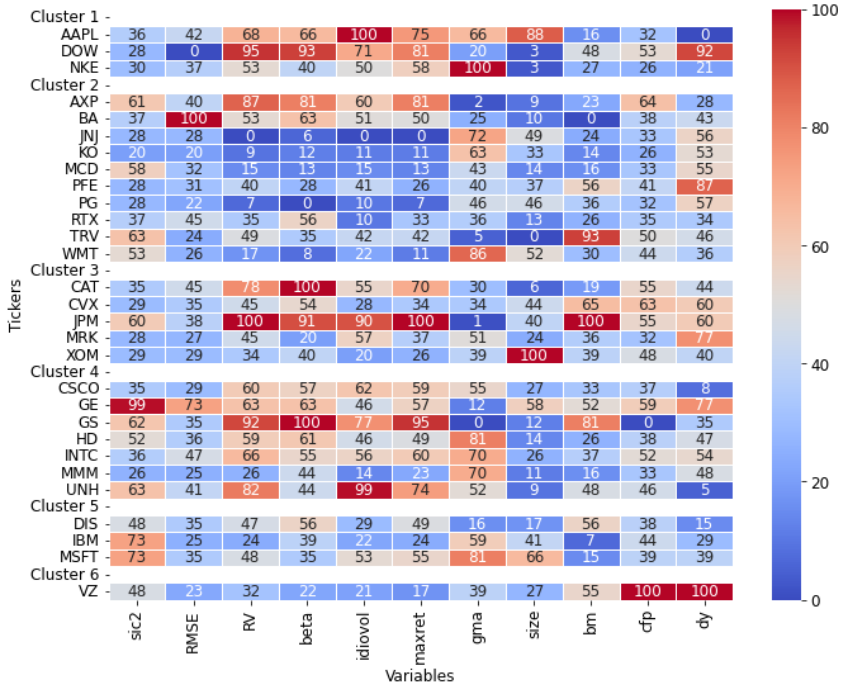
Notes: This figure shows the six optimized clusters for the cluster NGboost, with realized volatility as the target variable. The clusters are optimized using the algorithm described in Algorithm 2. Except for sic2, the heatmap represents normalized data for the stock characteristics. The highest value across stocks is labeled 100, and the lowest is labeled zero. Data was used from 2007 to 2015 for all the characteristics.

Cluster NGboost

In this part, I discuss the optimized clusters of the cluster NGboost as given in figure 3.3. Only AAPL is present in Cluster 1 of the NGboost cluster, possibly due to its low RV correlation with other assets. Cluster 2 appears to have a medium RV and a medium beta. This cluster also exhibits low to medium idovol and maxret values. In this cluster, AXP and BA seem to be a bit on the high end for all these characteristics. The stocks in cluster 3, with the exception of CVX, are on the high end of RV, beta, maxret, and idovol. The four stocks do have a similar cash flow-to-price ratio. Cluster 4 is a difficult cluster to explain, as across stocks, all

charasterics have medium-low to high values. Cluster 5 has a medium RV, beta, idovol, and maxret. Most importantly, the high values for gma indicate that cluster 5 appears to have high gross profitability. Cluster 6 is characterized by low RV, beta, idiovol, and maxret values. The only exception here is TRV. However, TRV has a medium dividend-to-price ratio, which is the case for every stock in this cluster. It is important to note that the SIC code does not explain how these stocks are clustered.

Figure 3.4: Optimized clusters for cluster ResNGboost including heatmap with characteristics



Notes: This figure shows the six optimized clusters for the cluster NGboost, with the residuals of the log HAR as the target variable. The clusters are optimized using the algorithm described in Algorithm 2. Except for sic2, the heatmap represents normalized data for the stock characteristics. The highest value across stocks is labeled 100, and the lowest is labeled zero. Data was used from 2007 to 2015 for all the characteristics.

Cluster ResNGboost

In this subsection, I discuss the clusters of the cluster ResNGboost, as given in Figure 3.4. Cluster 1 of the cluster ResNGboost contains both the AAPLE and the DOW, which are the two stocks with the lowest correlation across stocks for the RV. However, since this cluster also includes the NKE, we can best identify it as a cluster with a medium-high maxret and all the sic2 falling in the manufacturing category. Cluster 2 of the ResNGboost is a bit like Cluster 6 of the NGBoost; it has mostly medium- to low-values for RV, beta, idovol, and maxret. AXP and BA are exceptions. This can be explained by the observation that cluster 2 for the ResNGboost appears to primarily combine clusters 2 and 6 from the NGboost cluster. This cluster is best identified as a cluster with a low RMSE for the log HAR model. Cluster 3 is difficult to identify by its characteristics; however, it has four stocks with a SIC corresponding to manufacturing in them, and those stocks also have a relatively low RSME. JPM, on the other hand, has a very high RSME and SIC corresponding to finance, insurance, and real estate. The medium cash

flow to price ratio (cfp) is best suited to explain this cluster as a whole. Cluster 4 is identifiable as the medium beta cluster; the values are close together, as can be seen in Figure 3.4. The exception in this cluster is GS, which has a very high beta. Cluster 5 presents a challenge in identifying commonalities; however, it boasts a medium-low to low RSME. Crucially, the cfp values, which in this case are all around 40, serve as the best indicator of this cluster. Cluster 6 contains only VZ, which has the highest cfp and dividend-to-price ratio (dy).

3.3 Results

This section discusses the main results. Firstly, I discuss performance using the CRPS and NLL metrics. Next, I analyze the performance of the mean forecasts by examining the SE loss. Thereafter, I discuss the results for quantile forecasts using the asymmetric piecewise linear scoring function. Lastly, I give some insight into feature effects and their importance with the help of accumulated local effects.

3.3.1 CRPS and Logs Evaluation

As previously mentioned, I use both the CRPS and LogS as scoring rules to evaluate the performance of the forecasted conditional distribution for realized volatility. I consider them in tandem, particularly highlighting the LogS role as a confirmatory metric, as evident in Tables 3.6 and 3.7, where only the LR and MCSR are reported for the LogS. As discussed in Section 2.4, it is recognized that the CRPS generally provides more consistent model rankings for financial data. The next part discusses the distributional performance according to the CRPS, with the exception of a clear mention of the LogS.

Cluster Models' Superior Performance

Both cluster NGBoost and cluster ResNGboost models exhibit outstanding performance across both validation and out-of-sample periods, consistently surpassing the benchmark, individual NGBoost, and pooled NGBoost specifications for both estimation methods. The validation set results in Table 3.6 showcase cluster models achieving the lowest average loss (AL) and a high significant loss ratio's (LR*), with minimum values of 96.6%, indicating that the benchmark has been outperformed significantly for almost all stocks. Furthermore, the cluster models show the highest Model Confidence Set Rates (MCSR), with minimum inclusion rates of 79.3 percent and peaking at 96.6 percent. The LogS confirms this superiority, once again showcasing the highest MCSR across both estimation methods for the validation set.

This dominance extends into the test set results in Table 3.7, with the exception of panel A, which is discussed in the next subsection. Panels A and B report the results for the whole out-of-sample period, whereas Panels C and D exclude the 2020 coronacrisis. For the CRPS in panels B, C, and D, both cluster models show the lowest AL. Furthermore, the cluster models show again a high LR*, with a minimum across panels of 82.1 percent, indicating significant outperformance of the benchmark. Most importantly, the MCSR inclusion rate across panels is once again higher than any other model. Panel B, which displays the expanding window results for the entire out-of-sample period, includes the cluster NGboost model 100% of the time and shows the lowest

AL across panels with a value of 0.911, demonstrating its consistency and ability to perform well in turbulent markets. Comparing Panel D to Panel B, both of which use an expanding window, reveals an increase in MCSR inclusion rates, indicating consistent performance for both cluster models, even in the face of economic turbulence. This is also confirmed by the MCSR of both CRPS and LogS in Panels B and D, with Panel B showing higher inclusion rates than Panel D. A small note on this is that the benchmark's MCSR inclusion rate is higher in panel B than panel D, showing that it is more difficult to exclude the benchmark when forecasting for the entire out-of-sample period.

Estimation Method Impacts

As can be seen in Table 3.6 and Panels C and D of Table 3.7, whenever there are periods of low volatility, the results look very similar. There is a small difference in terms of the best-performing model for the different estimation methods. For moving windows in low volatility periods, this is the Cluster ResNGboost according to the MCSR inclusion rate; for expanding windows in the same period, this is the Cluster NGboost.

The discrepancy in model efficacy between moving and expanding estimation windows is pronounced, however, when looking at panels A and B of 3.7. In panel A, you can see that all NGboost models have high AL values and much lower LR and LR* values than in panels B, C, and D earlier, indicating poor performance. You see very poor performance, especially when looking at the values for AL, LR, and LR* for all the ResNGboost models. However, the MCSR shows an inclusion rate of nearly 1 for all models, indicating high performance variability that prevents their exclusion. Section 3.3.3 provides a more thorough explanation of why the MCSR could be high for a model when it has high variability in performance. Given the results in Panel C, where the three models perform very well, there has to be very poor performance for the Corona crisis year 2020. The combination of good performance before 2020 and bad performance in 2020 leads to high performance variability. The same argumentation, in a less severe sense, applies to the NGboost models. This, along with the fact that the inclusion of 2020 in the expanding window actually yields more promising results in terms of AL and MCSR, leads to the conclusion that the NGboost model underperforms in crisis situations when it lacks crisis training data. To be more precise, the expanding window includes the global economic crisis as training data, and this may be of the utmost importance for the NGboost model results displayed in Panel B.

Comparative analysis between pooled and individual NGBoost models reveals a nuanced relationship influenced by the estimation window selection; this holds for both the Ngboost and ResNGboost models. Pooled models tend to outperform individual ones with a moving window, whereas individual models come to the fore with an expanding window. This indicates that individual NGboosts flourish when given more data, whereas pooled models may not efficiently handle the increased data volume associated with expanding windows. Given these insights, investigating alternative NGBoost settings, such as a stochastic version or a higher number of boosting iterations, emerges as a possible solution for the pooled NGBoost; however, this needs further research. Moreover, note that these results are not totally supported by the LogS MCSR inclusion rates in Table 3.6 and Table 3.7. Only a small decrease in the inclusion rate for the

pooled NGboost models and a small increase for the individual NGboost models are observed.

NGboost vs ResNGboost

As indicated before, for the cluster models, the estimation methods seem to be important. However, the cluster NGboost seems to perform the most consistently across panels; still, the cluster ResNGboost falls not far behind, showing the potential for the NGboost to be an overlay model for any realized volatility mean forecasting model. Moreover, for both the individual and the pooled models, it is noticeable that in terms of MCSR, ResNGboost models are preferred across the different panels, with one exception in panel D of Table 3.7. In absolute terms, when looking at AL and LR, the pooled models perform on par with each other. For the individual specification, however, you can see that the ResNGboost models are preferred most of the time. These findings confirm NGBoost's potential added value as an overlay model.

Table 3.6: Forecasting evaluation of the validation period

Model	SE				CRPS				NLL	
	AL	LR	LR*	MCSR	AL	LR	LR*	MCSR	LR	MCSR
Panel A: Moving window period 2012-2015										
NG	0.915	0.862	0.000	0.345	1.024	0.276	0.069	0.103	0.000	0.034
Pooled NG	0.861	**1.000**	0.207	0.759	0.943	**1.000**	0.897	0.517	0.931	0.724
Cluster NG	0.824	**1.000**	0.276	**1.000**	0.931	**1.000**	0.966	0.897	0.931	0.828
Log HAR	1.000	-	-	0.483	1.000	-	-	0.000	-	0.103
ResNG	0.903	0.897	0.103	0.759	0.969	0.897	0.517	0.379	0.034	0.069
Pooled NG	0.875	0.966	**0.448**	0.966	0.938	**1.000**	**1.000**	0.724	**1.000**	0.862
Cluster NG	**0.815**	0.966	0.207	**1.000**	**0.926**	**1.000**	0.966	**0.966**	0.966	**0.931**
Panel B: Expanding window period 2012-2015										
NG	0.876	0.931	0.000	0.586	0.978	0.793	0.414	0.310	0.000	0.034
Pooled NG	0.862	**1.000**	0.379	0.793	0.937	**1.000**	0.966	0.448	**0.966**	0.586
Cluster NG	**0.821**	**1.000**	0.414	**1.000**	**0.920**	**1.000**	0.966	**0.966**	**0.966**	0.862
LOG_HAR	1.000	-	-	0.517	1.000	-	-	0.000	-	0.069
ResNG	0.852	**1.000**	0.379	0.862	0.937	**1.000**	**1.000**	0.483	0.655	0.069
Pooled ResNG	0.872	**1.000**	0.379	0.931	0.938	**1.000**	**1.000**	0.448	**0.966**	0.655
Cluster ResNG	0.829	**1.000**	**0.448**	0.966	0.924	**1.000**	**1.000**	0.793	**0.966**	**0.931**

Notes: For the squared error loss function (SE) and the continuous ranked probability score (CRPS), the average loss ratio (AL) relative to the benchmark (log HAR) (lower values are better), the proportion of instances one model is performing better than the benchmark model (LR) (higher is better), the proportion of instances one model is significantly performing better than the benchmark model (LR*) (higher is better), and the model confidence set inclusion rate (MSCR) across stocks (higher is better) are reported; see explanation in Section 2.4. For the LR*, a Giacomini-White test with a significant level of 5% is used. For the logaritmic score (LogS), only the LR and MCSR are reported. Individual NGboost models are found in the rows indicated as NG, pooled NGboost models are in the rows indicated as pooled NG, and cluster NGboost models are indicated as cluster NG. The results for log HAR can be found in rows labeled log HAR. Results for individual, pooled, and cluster ResNGboost can be found in rows labeled as ResNG, pooled ResNG, and cluster ResNG. The NG models use a lognormal distribution with the realized volatility as the target variable. The ResNG employs a normal distribution with the log HAR residuals as target variables. All models forecast the one-day-ahead density of RV. The out-of-sample period ranges from 2012 to 2015; throughout my paper, this period is referred to as the validation period. The AL, LR, LR*, and MCSR are calculated using 29 stocks as described in Section 3.1. The numbers in bold represent the best outcome for the different evaluation metrics in each panel.

Table 3.7: Forecasting evaluation of the out-sample period

Model	SE				CRPS				LogS	
	AL	LR	LR*	MCSR	AL	LR	LR*	MCSR	LR	MCSR
Panel A: Moving window out-sample period 2016-2020										
NG	1.150	0.286	0.071	**1.000**	1.060	0.214	0.107	0.607	0.036	0.036
Pooled NG	1.108	0.250	0.000	**1.000**	1.003	0.571	0.214	0.786	**0.750**	**0.857**
Cluster NG	**0.972**	**0.643**	**0.179**	**1.000**	**0.983**	**0.679**	**0.357**	0.893	0.571	0.643
Log HAR	1.000	-	-	**1.000**	1.000	-	-	0.893	-	0.607
ResNG	4.960	0.214	0.000	**1.000**	1.196	0.179	0.107	**1.000**	0.000	0.036
Pooled ResNG	6.107	0.000	0.000	**1.000**	1.435	0.000	0.000	0.964	0.143	0.714
Cluster ResNG	6.699	0.000	0.000	**1.000**	1.287	0.000	0.000	0.964	0.500	0.607
Panel B: Expanding window out-sample period 2016-2020										
NG	0.854	0.786	0.321	0.964	0.955	0.714	0.536	0.750	0.286	0.071
Pooled NG	1.003	0.607	0.036	0.964	0.958	0.786	0.571	0.429	**0.929**	0.429
Cluster NG	0**.777**	**0.964**	**0.607**	**1.000**	**0.911**	**0.964**	0.821	**1.000**	0.857	**0.857**
Log HAR	1.000	-	-	0.964	1.000	-	-	0.357	-	0.143
ResNG	0.911	0.857	0.464	**1.000**	0.918	**0.964**	0.750	0.821	0.786	0.429
Pooled ResNG	1.202	0.179	0.000	**1.000**	0.971	0.786	0.321	0.571	**0.929**	0.393
Cluster ResNG	0.825	**0.964**	0.500	**1.000**	0.917	**0.964**	**0.857**	0.893	**0.929**	0.821
Panel C: Moving window out-sample period 2016-2019										
NG	0.939	0.679	0.393	0.643	0.994	0.500	0.393	0.393	0.000	0.000
Pooled NG	0.869	0.929	0.821	0.786	0.936	0.929	0.857	0.643	0.929	0.786
Cluster NG	0.867	0.929	0.857	**0.893**	0.932	0.929	0.821	0.821	0.786	0.714
Log HAR	1.000	-	-	0.250	1.000	-	-	0.036	-	0.143
ResNG	0.896	0.893	0.714	0.714	0.954	0.821	0.607	0.500	0.071	0.036
Pooled ResNG	**0.847**	**1.000**	0.929	**0.893**	0.921	**1.000**	**1.000**	0.857	**0.964**	**0.929**
Cluster ResNG	**0.847**	**1.000**	**0.964**	**0.893**	**0.920**	**1.000**	**1.000**	**0.893**	0.821	0.786
Panel D: Expanding window out-sample period 2016-2019										
NG	0.903	0.929	0.643	0.857	0.952	0.929	0.679	0.571	0.286	0.071
Pooled NG	0.878	0.893	0.750	0.643	0.934	0.929	0.893	0.464	0.929	0.643
Cluster NG	0.871	0.893	**0.857**	**0.929**	**0.923**	0.964	0.893	**0.929**	0.929	**0.786**
Log HAR	1.000	-	-	0.143	1.000	-	-	0.036	-	0.071
ResNG	0.901	0.893	0.821	0.821	0.932	0.929	0.893	0.607	0.821	0.321
Pooled ResNG	0.880	**0.964**	0.821	0.750	0.935	0.964	0.929	0.429	0.929	0.679
Cluster ResNG	**0.868**	0.929	**0.857**	0.893	0.924	**1.000**	**0.929**	0.786	**0.964**	0.750

Notes: For the squared error loss function (SE) and the continuous ranked probability score (CRPS), the average loss ratio (AL) relative to the benchmark (log HAR) (lower values are better), the proportion of instances one model is performing better than the benchmark model (LR) (higher is better), the proportion of instances one model is significantly performing better than the benchmark model (LR*) (higher is better), and the model confidence set inclusion rate (MSCR) across stocks (higher is better) are reported; see explanation in Section 2.4. For the LR*, a Giacomini-White test with a significant level of 5% is used. For the logaritmic score (LogS), only the LR and MCSR are reported. Individual NGboost models are found in the rows indicated as NG, pooled NGboost models are in the rows indicated as pooled NG, and cluster NGboost models are indicated as cluster NG. The results for log HAR can be found in rows labeled log HAR. Results for individual, pooled, and cluster ResNGboost can be found in rows labeled as ResNG, pooled ResNG, and cluster ResNG. The NG models use a lognormal distribution with realized volatility (RV) as the target variable. The ResNG employs a normal distribution with the residuals of the log HAR as target variables. All models forecast the one-day-ahead density of RV. The out-of-sample period ranges from 2016 to 2020. The AL, LR, LR*, and MCSR are calculated using 29 stocks as described in Section 3.1. The numbers in bold represent the best outcome for the different evaluation metrics in each panel.

3.3.2 Squared error loss

Validation Set and Non-Crisis Out-of-Sample Performance

The analysis of model performance for the squared error loss during the validation period, as depicted in Table 3.6, in conjunction with Panels C and D of Table 3.7, reveals a notable consistency in the efficacy of the NGboost models. Across all model variants, there is a recurring pattern of low average loss (AL) values in conjunction with high loss ratios (LR), indicating that the models consistently perform better in absolute terms than the benchmark under typical market conditions. This trend is pronounced in both the cluster model variants, which not only maintain the lowest values for AL but also exhibit LR* values close to 1 in panel C and D of table 3.7, indicating significant outperformance in the non-crisis years of the out-of-sample period.

The Model Confidence Set Rates (MCSR) further corroborates the robustness of these models. For both cluster models during the validation period, the MCSR inclusion rate is never below 96%, signifying a high degree of confidence in their performance. This high inclusion rate is sustained in the out-of-sample non-crisis periods (Panels C and D of Table 3.7), with a minimum rate of 89.3% for the cluster models, again reflecting their reliability. These combined metrics lead to the clear conclusion that the NGboost models, particularly the cluster variants, perform admirably in typical low-volatility market environments, showcasing their potential utility in mean forecasting scenarios.

Crisis Period Out-of-Sample Performance

Transitioning to an analysis of the crisis period represented in Panels A and B of Table 3.7, we observe a stark contrast in model performance for the moving window. During the crisis period, as evidenced by the out-of-sample moving window results in Panel A, all models exhibit elevated AL values. The NGboost models struggle to adapt to the volatile conditions, resulting in a pronounced underperformance. The MCSR reinforces this observation, with no models achieving significant outperformance over the benchmark. This is in agreement with the results for the CRPS.

In Panel B, which encompasses crisis years for the expanding window estimation, we see an improvement in absolute terms for the individual and cluster NGboost models compared to Panel A. Their lower AL values, even lower than the values in Panel D, suggest that they have resilience when faced with crisis data. The performance of the pooled models in terms of AL is a worse than in Panel B. But still, overall, these improvements seem to show the importance of including crisis data in the training data of the NGboost.

Despite the NGboost Cluster models significantly outperforming the benchmark 60% of the time, according to the LR* in panel B, the benchmark's inclusion rate in the MCSR is high at 96.4%. The high rate across all models in Panel B suggests significant performance variability within this period, which prevents the exclusion of any model from the set of competitive models. It underscores the notion that in times of crisis, predictive performance across models becomes highly variable, rendering any model, including the benchmark, indispensable within the model confidence set. This is confirmed by comparing the value of the test statistic for the MCS as

given in Equation 2.11. For Panel D, this test statistic gives higher values than for Panel B, resulting in a lower MCSR for the benchmark in Panel D and not in Panel B.

Table 3.8: Quantile forecast evaluation of the out-sample period

Model	1% quantile				99% quantile			
	AL	LR	LR*	MCSR	AL	LR	LR*	MCSR
Panel A: Moving window out-sample period 2016-2020								
NG	1.562	0.107	0.036	0.357	1.329	0.214	0.000	0.571
Pooled NG	1.045	**0.357**	**0.214**	0.821	**0.791**	**0.929**	**0.250**	0.964
Cluster NG	1.197	0.214	0.107	0.750	0.858	0.821	0.071	0.750
Log HAR	**1.000**	-	-	0.786	1.000	-	-	**1.000**
ResNG	1.853	0.000	0.000	0.750	1.081	0.286	0.000	0.393
Pooled ResNG	3.370	0.000	0.000	**1.000**	0.915	0.893	0.000	0.964
Cluster ResNG	1.238	0.250	0.107	0.893	1.011	0.500	0.000	0.857
Panel B: Expanding window out-sample period 2016-2020								
NG	1.181	0.250	0.036	0.893	1.090	0.536	0.000	0.714
Pooled NG	1.024	0.321	0.036	0.714	0.844	**0.964**	0.000	0.893
Cluster NG	1.001	**0.714**	0.179	**0.964**	0.841	0.893	**0.107**	0.964
Log HAR	1.000	-	-	0.643	1.000	-	-	**1.000**
ResNG	1.015	0.500	0.107	**0.964**	0.900	0.821	0.071	0.821
Pooled ResNG	1.011	0.429	0.071	0.643	0.843	**0.964**	0.000	0.964
Cluster ResNG	**0.988**	0.679	**0.357**	0.857	**0.826**	**0.964**	0.071	**1.000**

Notes: For the 1% quantile and 99% quantile using an asymmetric piecewise linear scoring function, the average loss ratio (AL) relative to the benchmark (log HAR) (lower values are better), the proportion of instances one model is performing better than the benchmark model (LR) (higher is better), the proportion of instances one model is significantly performing better than the benchmark model (LR*) (higher is better), and the model confidence set inclusion rate (MSCR) across stocks (higher is better) are reported; see explanation in Section 2.4. For the LR*, a Giacomini-White test with a significant level of 5% is used. Individual NGboost models are found in the rows indicated as NG, pooled NGboost models are in the rows indicated as pooled NG, and cluster NGboost models are indicated as cluster NG. The results for log HAR can be found in rows labeled log HAR. Results for individual, pooled, and cluster ResNGboost can be found in rows labeled as ResNG, pooled ResNG, and cluster ResNG. The NG models use a lognormal distribution with the realized volatility as the target variable. The ResNG uses a normal distribution with the residuals of the log HAR as target variables. All models forecast the one-day-ahead density of RV, in this table the quantiles of those forecasts are evaluated. The out-of-sample period ranges from 2016 to 2020; the AL, LR, LR*, and MCSR are calculated using 29 stocks as described in Section 3.1. The numbers in bold represent the best outcome for the different evaluation metrics in each panel.

3.3.3 Quantile evalution

Quantile Forecast Evaluation at 1%

Using the asymmetric piecewise linear scoring function, the 1% quantile forecast performance is accessed, and results are shown in Table 3.8. The difference in the Average Loss (AL) values between Panels A and B substantiates previous conclusions from the CRPS, LogS, and SE analyses: NGboost models generally underperform in crisis years in the absence of crisis training data.

Panel B, however, has better results, with AL values closer aligning with one for the 1% quantile, indicating a parity with the benchmark. The cluster models exhibit competent forecasting, matching the benchmark closely and performing better than the Log HAR model in absolute terms for a majority of instances, as suggested by the LR. The performance of the cluster models is further highlighted by their higher inclusion rate in the MCSR, with a 21% to 30% higher inclusion rate than the benchmark. This suggests that while they do not overwhelmingly outperform other models, they offer a consistent and reliable forecast for lower-tail risks. The individual ResNGboost is also noticeable; despite having a slightly higher AL and slightly lower LR and LR*, the MCS still includes this model 96.4% of the time.

Quantile Forecast Evaluation at 99%

The 99% quantile forecasts are particularly pertinent for portfolio managers and traders monitoring the risk of high volatility spikes. The analysis here draws attention to the nuanced details of model performance in predicting extreme upper-quantile events. In Panel A, it is noticeable that there are 3 models with an AL lower than 1, the pooled NGboost, cluster NGboost, and pooled ResNGboost. Especially the pooled models show, with a high LR and a high MCSR, that they can compete with the benchmark in forecasting higher tail risk without needing the crisis data to forecast a crisis period.

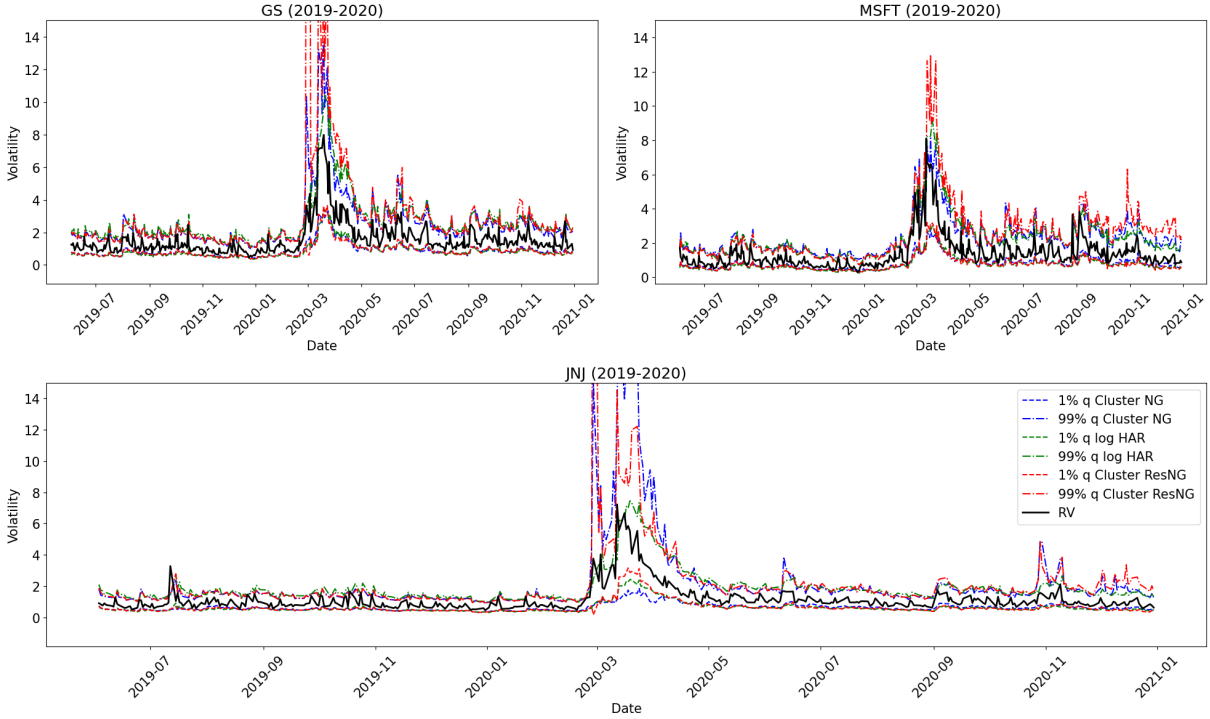
Shifting focus to Panel B, the landscape is markedly improved for the 99% quantile. Here, AL values for most models are below one, with the NGBoost Pooled and Cluster models in particular showing a distinct outperformance over the benchmark model, evident in at least 89.3% of cases, as indicated by the LR. Despite this, the LR* values hover close to zero, revealing that this outperformance is typically not statistically significant. The MCSR inclusion rate for the benchmark stands at a full 100%, demonstrating no significant difference between it and the cluster models, which are at least included 96.4% of the time. However, a closer look at the AL and LR metrics in tandem suggests that, in absolute terms, NGboost models generally incur a lower average loss than the benchmark across a broad spectrum of stocks. This nuanced perspective underscores the potential benefits of integrating NGboost models into risk management strategies, even if their edge over the benchmark is not often significant.

Comparing low AL for the cluster NGboost with high MCSR for the benchmark

For the 99% quantile forecasting, the high MCSR rate for the benchmark seems off when compared to the low values for AL for the cluster NGBoost. But, just like the difference between MCSR and AL in Panel B for the squared error loss, it is likely that this is the case because the differences in scores, as shown in Equation 2.9, vary a lot. When these differences in score vary a lot, the test statistics in 2.10 become lower due to a higher denominator. I inspected the individual test statistics for the models as given in Equation 2.10, and the log HAR has a lower test statistic on several occasions than when two different NGboost models are compared. This could explain why the MCS does not always include the cluster NG, even with a low AL, primarily in Panel A. In Panel B, the MCS for the cluster NGboost is high (96.4%). For the expanding window, Figure 3.5 shows the 1% and 95% predicted quantiles for a stock with low volatility (GS), a stock with median volatility (MSFT), and a stock with high volatility (JNJ).

In the short period shown prior to the 2020 coronavirus crisis, the model forecasts closely align, but the onset of the crisis accentuates the more conservative nature of the cluster models, particularly for the 99% quantile. This is well visible for the JNJ plot, and it shows this conservatism continuing after the March 2020 peak, implying a sustained cautious forecast approach from the cluster models in periods of high volatility. Whereas the log HAR model seems to underestimate the 99% quantile more often, causing a higher score on some occasions. The APL assigns a much higher weight to these kind of underestimations, as can be seen in Equation 2.4.1. This differences between the quantile forecasts indicate a high variability, for the period shown, in the differences vector as calculated in Equation 2.9. All together, this could explain the high model inclusion rate for the benchmark, while still having models like the cluster NGboost that perform better in absolute terms.

Figure 3.5: Quantile plots



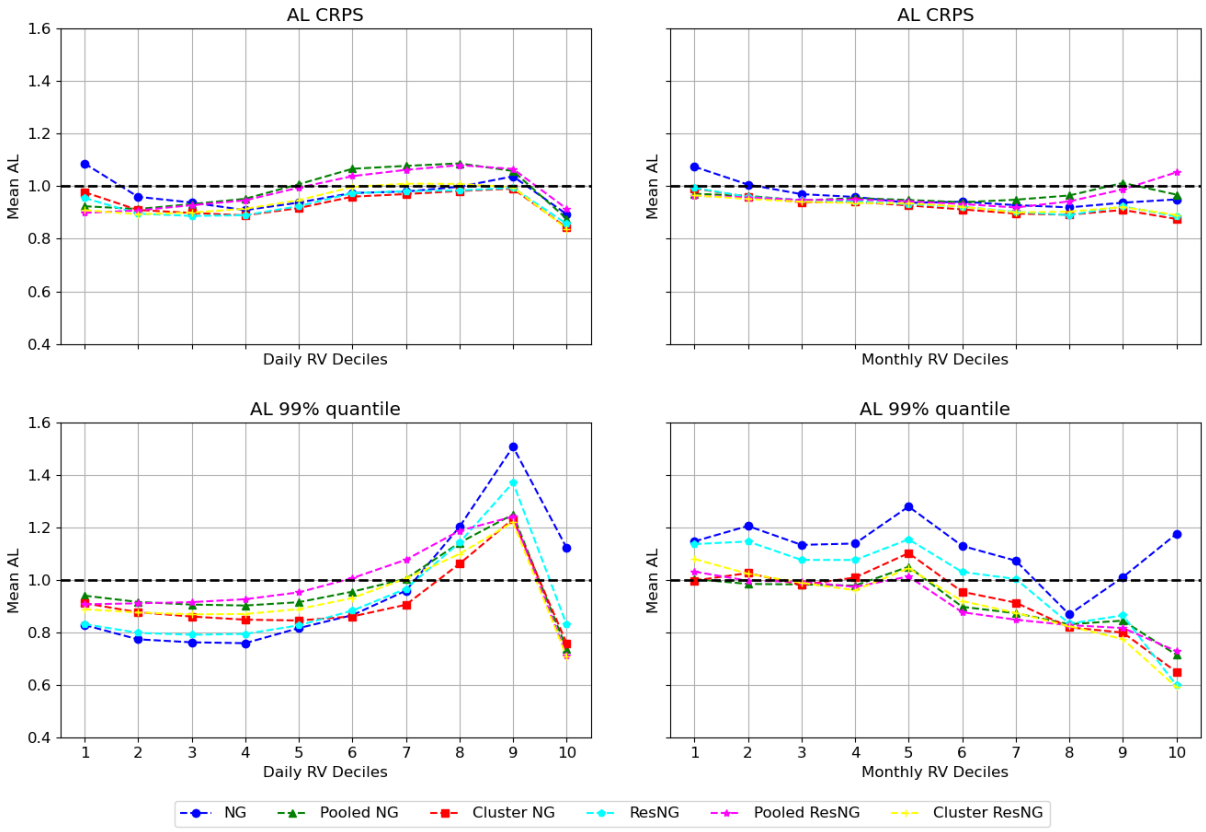
Notes: The plots show the 1% and 99% quantiles for the forecasted conditional distribution of the stocks Goldman Sachs (GS), Microsoft (MSFT), and Johnson & Johnson (JNJ). These three models represent an average RV that is high (GS), medium (MSFT), and low (JNJ); see Figure 3.3. The three models used to forecast the conditional distribution are the log HAR, the cluster NGboost with a lognormal distribution and target variable realized volatility, and the cluster ResNGboost with a normal distribution and target variable the residuals of the log HAR. The graphs of GS and JNJ are truncated to facilitate their readability.

3.3.4 Deepening analyses of AL results

In this section, I turn my attention to the conditional average loss (AL) for different volatility regimes for the 99% quantile forecasts using the APL scoring function and the distribution forecasts using the CRPS. In Figure 3.6, I categorize each stock's out-of-sample forecasts into deciles based on daily and monthly realized volatility at the time of the forecast. These deciles are created for each stock separately, and the AL is calculated for every decile. Therefore,

for every stock, each decile holds exactly the same number of observations. For this analysis, I exclusively used an expanding window, as it yields the most promising results. The plots show the AL conditionally on the value of the daily realized volatility (left) or monthly realized volatility (right) on the day of the forecast.

Figure 3.6: Expaning window AL results for daily and monthly realized volatility deciles



Notes: This figure reports the average loss (AL) measure across different volatility regimes. The AL is calculated conditionally on the value of the daily realized volatility (left) or monthly realized volatility (right) on the day that the distribution is forecasted. The out-of-sample observations are grouped into ten deciles for the daily and monthly RV. Every decile has the same number of observations for every stock. The AL is calculated across all stocks used for the out-of-sample period.

AL for 99% quantile across volatility regimes

In Figure 3.6, the two plots at the bottom represent the AL values for the APL scoring function for the forecasted 99% quantiles. The AL scores demonstrate the cluster models' superior performance in periods with the highest daily realized volatility. However, the high variability described in the previous section can be noticed; as for the 8th and 9th deciles, the cluster models perform substantially worse than the benchmark. This confirms further that this is the reason for the high MCSR of the benchmark. The lower variability between the NGboost model specifications is also visible, most models are moving closely together. When realized volatility is considered over a 22-day rolling period, the cluster models present AL values as low as 0.6 for the highest values of the monthly realized volatility. Meaning that the models are performing best when there is higher realized volatility in the past month before the forecasted observation. When the realized volatility has been low over the past month, the cluster models perform on

par with the benchmark.

AL for CRPS across volatility regimes

Switching to the CRPS performance, the daily decile analysis reveals that cluster models perform best across daily volatility levels (left upper plot). The cluster models consistently outperform the log HAR model across all daily volatility levels. And as before, during the mid-high and high-volatility segments, the performance of the cluster models is lowest compared to the other segments. With performance peaking and the highest segment of daily realized volatility. The monthly decile analysis (the right upper plot) clearly indicates that as monthly realized volatility increases, so does the relative AL performance of both cluster models. This is in agreement with the earlier observations that the overall AL for the cluster models is lower when the Corona crisis is included in the test-sample, as seen in Table 3.7. The Corona crisis causes higher realized volatility over longer periods (including higher monthly realized volatility). This is also confirmed by the different plots in Figure 3.5.

3.3.5 Accumulated local effects for the cluster NGboost

This section explores the impact of various features on the forecasting results of the cluster NGboost, using an expanded window. At the end of this section, I also briefly discuss the results for the cluster ResNGboost using an expanding window. Figure 3.7 illustrates the ALE for the VIX, RV^+, and daily returns, dissecting their impacts on the scale and shape parameters of the log-normal distribution forecasted by the cluster NGboost. The ALE plots are truncated on the right-hand side, as after a certain value of the specific features, the influence remains mostly constant.

For the lognormal distribution, the scale parameter is indicative of its central tendency; a higher scale implies a shift of the entire distribution toward greater values while maintaining the same percentage variability. On the other hand, the shape parameter governs the distribution's variability, while a higher value for the shape also indicates a more asymmetric and right-tailed distribution. So a higher shape means higher variability and a higher chance of right-sided extreme observations.

For the VIX, there is an upward trend visible in the ALE plot, demonstrating a correlation between the VIX levels and the scale parameter. This trend aligns with intuitive market behavior, where heightened VIX levels go along with increased volatility expectations for individual stocks. A more nuanced pattern emerges when the shape parameter is examined. Mid-range VIX values correspond with a small dip in the shape parameter, whereas extreme VIX values push the shape parameter higher, indicating a higher variance and chance on extreme observations.

The relationship between RV^- and the model's predictions is markedly pronounced; as the positive semi-variances ascend, so does the scale parameter of the model's predictions. This steady and sizeable increase suggests that the model anticipates a corresponding escalation in future volatility in response to the rise of negative semi-variances. As values increase, the shape parameter follows a similar pattern to the scale parameter. When the negative semi-variance gets higher, higher variability and a higher chance on the right extremes are expected, as the shape parameter gets a higher value. I decided to add the plots for RV^-; however, RV^+, RV^d, RV^w,

and RV^m display similar behavior in their plots. There is a difference in the level of influence for the scale and shape, which is confirmed by the variable importance plot in Figure 3.8. However, for all five realized measures, the pattern of higher scale and shape with increasing values of these features is similar. The combined explanation for this behavior is that the autocorrelation between lags of realized measures is proven to be very high, so whenever the value of one of these realized variance features is higher, a higher scale is to be expected. Further, as discussed in Section 1.2, the higher the value of realized volatility, the higher the measurement error, which increases the variance, so a larger shape is to be expected.

Regarding the risk aversion index, it is clear that this feature does not influence the scale too much. Except for when it gets very high, it then causes the scale to be higher. The fact that it does not influence the scale too much is also confirmed by the variable importance plot in Figure 3.8. For the shape, it is different. You see a far more dynamic display; when the risk aversion gets a bit higher, the shape gets lower, causing the distribution to have less variability. However, around 4, you can see that the shape starts to climb along with the increasing values of the risk aversion index. When it reaches its peak, it becomes clear that there is a very high discrepancy in how it influences the shape of the distribution for different stocks. Therefore, when risk aversion is high, it tends to positively influence the shape of the distribution for some stocks more than it does for other stocks.

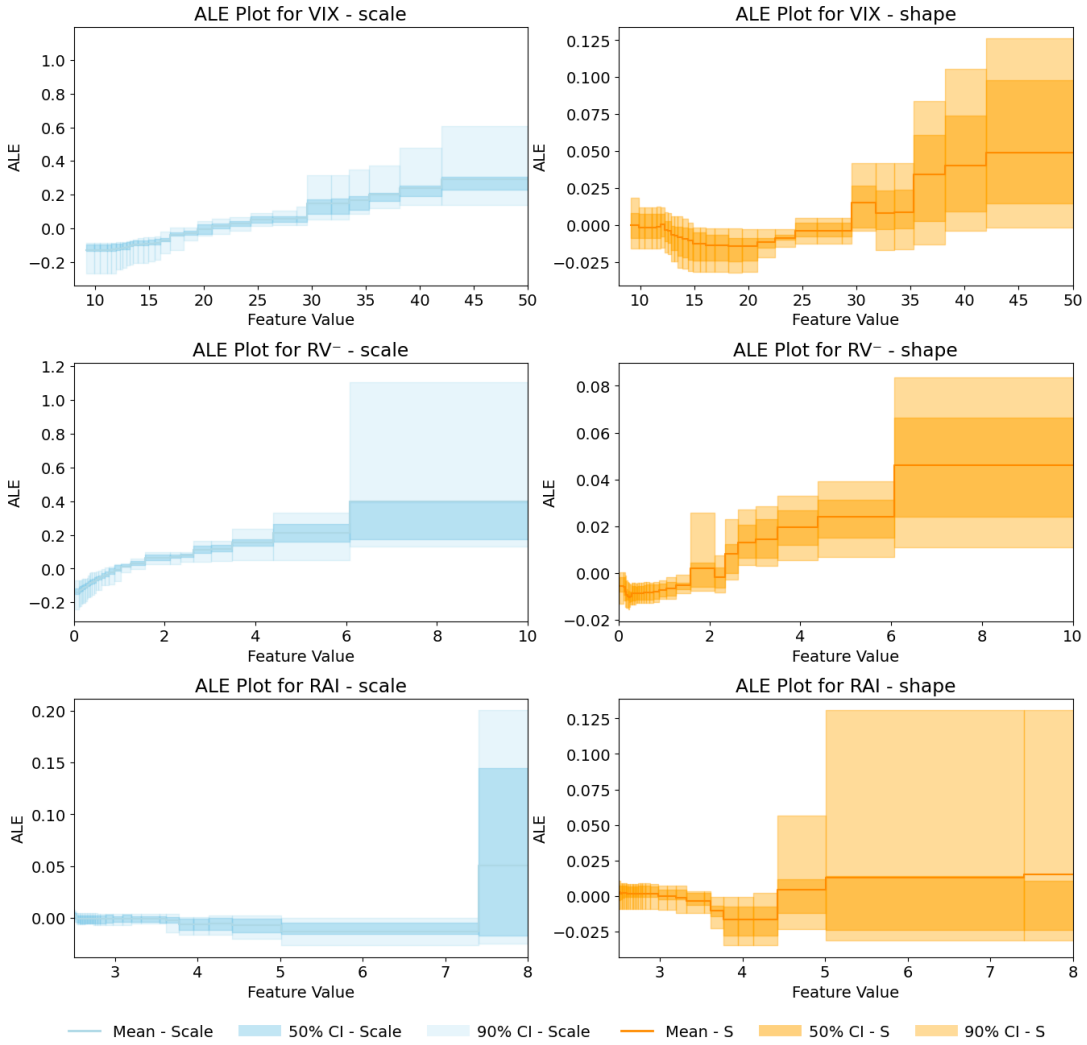
Variable importance

The scale parameter, which directly impacts the central tendency of the predicted log-normal distribution, shows that realized volatility (RV) measures, particularly RV^-, RV^+,RV^d,RV^w, stand out as the most influential predictors.The height of Ret^d shows that the model indicates a bit of a leverage effect. The VIX has a notable impact on the scale, but mostly it is the stock-specific realized measures that are important.

When examining the shape parameter, which influences the distribution's variability and asymmetry, the findings are slightly different. Realized volatility measures still feature prominently, yet the relative importance of these variables is less pronounced than for the scale parameter. Intriguingly, uncertainty measures like the FSI and VIX are more influential in shaping the distribution than in scaling it. This demonstrates that they contribute more significantly to the variance and skewness of predicted volatility than to the magnitude. Moreover, the IDEMV has a high influence on the shape parameter, which could be due to the Corona crisis being included in the out-of-sample period, as Y. Li, Liang et al. (2020) found a significant effect for the IDEMV for 2020. To formally conclude if this is the case, further analysis is required.

Interesting is the low influence of the realized quarticity on both scale and shape. I do not find the same high influence as found by Bollerslev et al. (2016); however, this is somewhat in line with the results of Cipollini et al. (2021) as discussed in Section 1.2. They find that when the variance is allowed to be heteroskedastic, the influence of realized quarticity is not necessarily significant. Of course, my NGboost models are not directly comparable to theirs in terms of feature interaction. Importantly, my models, similar to theirs, allow for heteroskedastic variance in the realized volatility.

Figure 3.7: the aggregated ALE plots for scale and shape (s)

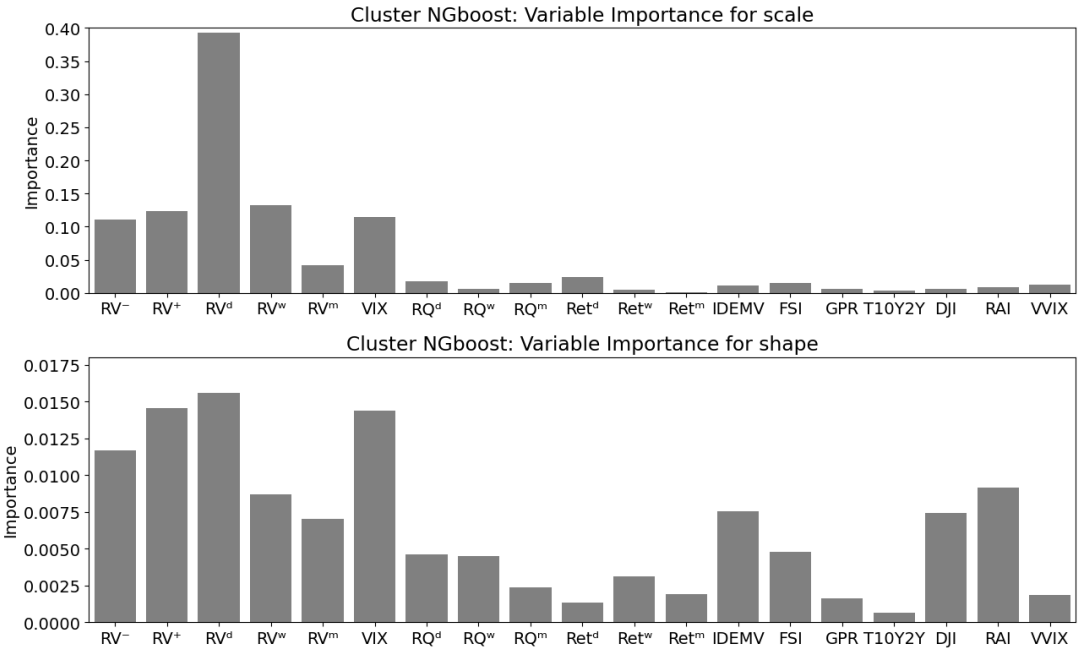


Notes: For the cluster NGboost model, this figure shows the ALE fan charts for the scale and shape parameters of the lognormal distribution. The individual calculated ALE values are aggregated into fan charts. The bands correspond to a 50% and 90% interval. Included is the mean ALE value across stocks. Note that the x-as is truncated for each feature separately; after the specific value, the fan chart does not change and stays constant.

ALE and VI for the cluster ResNGboost

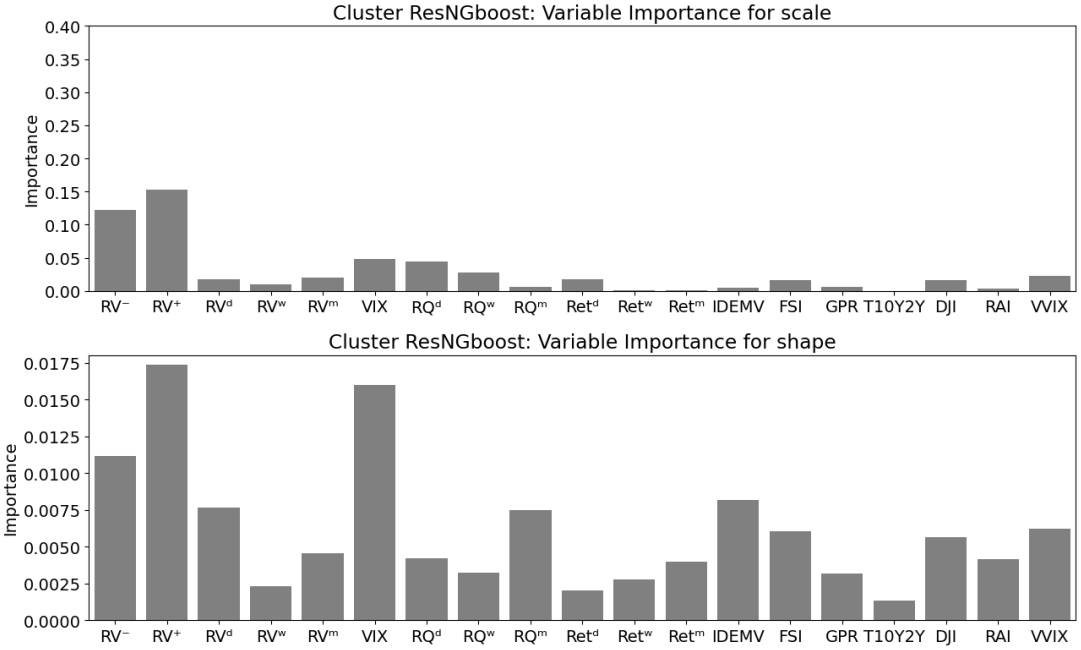
We find similar results for the ALE plots for the cluster ResNGboost; the level of influence for the VIX is slightly lower, but the movement of the plots remains exactly the same. For this reason, the appendix includes the ALE plots for the cluster ResNGboost in Figure A.2. However, the VI of the cluster ResNGBoost is more interesting. Because the NGboost model now only predicts the distribution of the residuals of the log HAR model, there are some variables that are less important for the scale. RV^d, RV^w, and RV^m are much less important for the scale; this is in line with expectations, as the mean of the realized variance is predicted by a log HAR model that incorporates those features, see Equation 2.3. For the shape, we see that there is a drop in VI for the same features; however, they remain important. The most important thing to notice is the increase in the VI of the VVIX. This volatility of volatility measure becomes more important when forecasting the shape parameter for the errors of a log HAR model.

Figure 3.8: Mean cluster NGboost variable importance across stocks



Notes: This figure shows the variable importance based on the accumulated local effects for the cluster NGboost model. For all features the variable importance is calculated as the mean variable importance across stocks.

Figure 3.9: Mean cluster ResNGboost variable importance across stocks



Notes: This figure shows the variable importance based on the accumulated local effects for the cluster ResNGboost model. For all features the variable importance is calculated as the mean variable importance across stocks.

4 Discussion and Conclusion

4.1 Discussion

Before providing the concluding remarks, I discuss the boundaries and limitations of my research. In this section, I also provide recommendations for further research. My main contribution to the literature is the use of cluster NGboost and ResNGboost for the conditional distribution of realized volatility. As discussed in Section 2.3.3, the cluster algorithm is limited in two ways. Firstly, as a boundary rather than a limitation, I solely utilize static cluster optimization. The algorithm can be expanded to be dynamic and form more optimal clusters for every new estimation sample. However, since optimizing the optimization algorithm takes approximately three weeks, it is advisable to utilize a computer with a higher computational power [1]. Secondly, when the re-allocation of the stock with the highest potential gain in CRPS fails to increase the overall CRPS, the algorithm assumes a local minimum. As discussed in Section 2.3.3, this does not have to hold true; in theory, there can be other stocks in the list of possible re-allocations that improve the overall CRPS when they are the only stock reassigned. However, adapting the algorithm in this way would severely slow down the optimization routine. The algorithm should incorporate this to increase the certainty of finding the global minimum. I suggest using a dynamic and adapted cluster optimization algorithm in future research. There is also the possibility of forming optimal data input for each stock separately, rather than clustering. To be more clear, in Figure 3.3, Apple is a cluster on its own; however, it could well be that the performance of Apple will increase when put in a cluster with other stocks. This does not happen in the current optimization routine, as the overall CRPS will drop; however, in further research, every stock could have its own optimal data input. This shifts the research from forming optimal clusters to forming optimal datasets for every stock.

For my research to be executable, I limited it to forecasting the conditional distribution of the realized volatility with a small selection of distributions that do not need a change of variables. However, given the results of my research, the potential of the cluster NGBoost reaches much further. As discussed in Section 1.2, Catania and Proietti (2020) forecast the distribution of the log realized volatility for the same stocks in a different time period. In terms of CRPS, their model, using a student-t distribution with two degrees of freedom, significantly outperforms the Log HAR-GARCH-NIG (Rvar) model of Corsi et al. (2008). They suggest conducting further research with a skewed student-t distribution. In further research, the cluster NGboost and ResNGboost could incorporate these distributions and forecast the conditional distribution of the log realized volatility. Instead of assuming two degrees of freedom as done by Catania and Proietti (2020), the NGboost is able to optimize all parameters of the used distribution.

One last important limitation of my research is that I did not optimize the parameters and

[1] I use a computer with a Processor 12th Gen Intel(R) Core(TM) i5-1235U, 1300 MHz, 10 core(s), 12 processor(s), and 8 GB of RAM.

hyperparameters for the NGboost specifications. Further research can optimize them to yield better results. Even without optimization, both cluster models show good results; however, optimizing these parameters is important for the reasons described in Section 2.3.4. Next, the NGboost can be extended with different base learners as well as using a different scoring rule. The CRPS can be used as a scoring rule, or a new innovative proper weighted scoring rule can be used, as described in the discussion paper by de Punder, Diks, Laeven and van Dijk (2023). With a weighted scoring rule, one could assign more weight to the right side of the distribution, which for risk managers is important.

4.2 Conclusion

In this study, I use a new cluster natural gradient boosting method to forecast the conditional distribution of the realized volatility for 29 stocks selected from the S&P500. An optimization algorithm optimizes the clusters to find those with the lowest overall CRPS when using a separate NGBoost for each cluster to provide forecasts for the validation period (January 2012 to December 2015). I use the new cluster NGboost in two ways: first, I apply it directly to the realized volatility (cluster NGBoost) to provide a direct forecast for the RV's conditional distribution. Secondly, I use a cluster NGboost on the residuals of a log HAR model (cluster ResNGboost). The lognormal distribution is used for both of these cluster models, as this distribution performs best for their singular NGboost counterparts for the S&P500 index in the validation period. As a suitable benchmark, I selected the log HAR model with normality assumptions on its innovations based on the results of forecasting the distribution of the realized volatility of the S&P500 index in the validation period. For both the cluster NGboost and cluster ResNGboost, it is optimal to use six clusters. I gain some understanding of the formation of clusters by examining the realized volatility, the RMSE of the log HAR, the beta, idiosyncratic volatility, maximum daily return, gross profitability, cash flow to price ratio, dividend to price ratio, and the first two digits of the Standard Industrial Classification Code. Different clusters have different values for these characteristics.

In my empirical research, these two cluster models are compared to their singular and pooled counterparts, as well as to a log HAR with normality assumptions on its innovations. For my out-of-sample period, containing data from January 2016 to December 2020, both cluster NGboost specifications excel. For all models, I use two different estimation windows: a moving window containing 5 years of training data, and an expanding window containing data as of January 2007. The NGboost models, particularly the cluster models, exhibit strong performance in the validation set and non-crisis out-of-sample periods for both distribution and mean forecast, demonstrating their potential to significantly outperform the benchmark under typical market conditions. However, the use of a moving window estimation presents challenges for these models during the 2020 Corona crisis year. The expanding window estimation method exhibits resilience, possibly due to the inclusion of previous crisis data. The expanding window incorporates the global economic crisis in its training data.

Using the expanding window estimation, both cluster models show the most accurate distributional forecasts over the whole out-of-sample period, with the lowest CRPS across stocks. Both models exhibit high model confidence set (MCS) inclusion rates for the CRPS and LogS,

significantly outperforming the benchmark. From a deepening average loss analysis for the CRPS, it is concluded that in terms of AL, the cluster performance increases relative to the benchmark when the monthly realized volatility increases. So, in turbulent times, the cluster models perform better relative to the benchmark. This extends to forecasting the 99% quantile of the distribution; conditional on the monthly realized volatility, the performance in terms of AL becomes better when realized volatility over the past month increases. However, the APL scoring function, which tests the quantiles, indicates that the cluster models do not significantly outperform the benchmark. This is explained by the high variation in the differences between the scoring functions of the models. On days with low- to medium-realized volatility and on days with the highest realized volatility, the cluster models are performing best. However, when the realized volatility is medium-high, the benchmark provides better quantile forecasts. For the APL scoring function, the cluster models have the lowest AL. They also have the lowest AL for squared error loss, showing the best performance in absolute terms of mean forecasting. According to the MCS for the SE loss, the models significantly outperform the benchmark when the corona year 2020 is excluded.

For the cluster NGBoost with an expanding window, based on the accumulated local effects analyses, I find that an increasing VIX value, which corresponds to a higher volatility regime, indicates an increase in shape and scale for the lognormal distribution. For the RV^{-}, similar results are found. Interestingly, in terms of variable importance, I find the realized quarticity has little influence on the scale parameter and only a small influence on the shape parameter. This is in line with Cipollini et al. (2021), who states that if the variance is allowed to be heteroskedastic, the realized quarticity is less important. Furthermore, I observe that the scale parameter gives the highest VI to the various stock-specific realized volatility measures and VIX, while the shape parameter adds the Dow Jones index, risk aversion index, and infectious disease equity market volatility index to the list.

Overall, this research demonstrates the potential of NGBoost for forecasting the distribution of realized volatility. Without optimization of parameters and hyperparameters, the cluster models are able to perform best in absolute terms for quantile and mean forecasting. While they significantly outperform all other models based on the CRPS and LogS. The good performance of the different ResNGboost specifications also showcases the potential for the NGboost as an overlay model.

References

Alam, M. M. & Uddin, G. (2009). Relationship between interest rate and stock price: empirical evidence from developed and developing countries. *International Journal of Business and Management (ISSN 1833-3850)*, *4*(3), 43–51.

Almeida, C., Fan, J., Freire, G. & Tang, F. (2023). Can a machine correct option pricing models? *Journal of Business & Economic Statistics*, *41*(3), 995–1009.

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

Andersen, T. G., Bollerslev, T. & Diebold, F. X. (2007). Roughing it up: Including jump components in the measurement, modeling, and forecasting of return volatility. *The review of economics and statistics*, *89*(4), 701–720.

Apley, D. W. & Zhu, J. (2020). Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *82*(4), 1059–1086.

Barndorff-Nielsen, O. E. (1997). Processes of normal inverse gaussian type. *Finance and stochastics*, *2*, 41–68.

Barndorff-Nielsen, O. E., Kinnebrock, S. & Shephard, N. (2008). Measuring downside risk-realised semivariance. *CREATES Research Paper*(2008-42).

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

Barndorff-Nielsen, O. E. & Shephard, N. (2005). How accurate is the asymptotic approximation to the distribution of realized variance. *Identification and inference for econometric models. A Festschrift in honour of TJ Rothenberg*, 306–311.

Bollerslev, T., Hood, B., Huss, J. & Pedersen, L. H. (2018). Risk everywhere: Modeling and managing volatility. *The Review of Financial Studies*, *31*(7), 2729–2773.

Bollerslev, T., Patton, A. J. & Quaedvlieg, R. (2016). Exploiting the errors: A simple approach for improved volatility forecasting. *Journal of Econometrics*, *192*(1), 1–18.

Buccheri, G. & Corsi, F. (2021). Hark the shark: Realized volatility modeling with measurement errors and nonlinear dependencies. *Journal of Financial Econometrics*, *19*(4), 614–649.

Caporin, M., Rossi, E. & De Magistris, P. S. (2017). Chasing volatility: A persistent multiplicative error model with jumps. *Journal of econometrics*, *198*(1), 122–145.

Carr, P. & Lee, R. (2003). Trading autocorrelation. *Unpublished paper: Courant Institute, NYU*.

Catania, L. & Proietti, T. (2020). Forecasting volatility with time-varying leverage and volatility of volatility effects. *International Journal of Forecasting*, *36*(4), 1301–1317.

Chakraborty, D., Elhegazy, H., Elzarka, H. & Gutierrez, L. (2020). A novel construction cost prediction model using hybrid natural and light gradient boosting. *Advanced Engineering*

Informatics, *46*, 101201.

Christie, A. A. (1982). The stochastic behavior of common stock variances: Value, leverage and interest rate effects. *Journal of financial Economics*, *10*(4), 407–432.

Cipollini, F., Gallo, G. M. & Otranto, E. (2021). Realized volatility forecasting: Robustness to measurement errors. *International Journal of Forecasting*, *37*(1), 44–57.

Corradi, V., Distaso, W. & Swanson, N. R. (2009). Predictive density estimators for daily volatility based on the use of realized measures. *Journal of Econometrics*, *150*(2), 119–138.

Corradi, V., Distaso, W. & Swanson, N. R. (2011). Predictive inference for integrated volatility. *Journal of the American Statistical Association*, *106*(496), 1496–1512.

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

Corsi, F., Mittnik, S., Pigorsch, C. & Pigorsch, U. (2008). The volatility of realized volatility. *Econometric Reviews*, *27*(1-3), 46–78.

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

Dai, Z. & Chang, X. (2021). Forecasting stock market volatility: Can the risk aversion measure exert an important role? *The North American Journal of Economics and Finance*, *58*, 101510.

Dawid, A. P. (2007). The geometry of proper scoring rules. *Annals of the Institute of Statistical Mathematics*, *59*, 77–93.

Dawid, A. P. & Musio, M. (2014). Theory and applications of proper scoring rules. *Metron*, *72*(2), 169–183.

de Punder, R., Diks, C. G. H., Laeven, R. J. A. & van Dijk, D. (2023). *Localizing strictly proper scoring rules* (Discussion Paper No. TI 2023-084/III). Amsterdam and Rotterdam: Tinbergen Institute.

Duan, T., Anand, A., Ding, D. Y., Thai, K. K., Basu, S., Ng, A. & Schuler, A. (2020). Ngboost: Natural gradient boosting for probabilistic prediction. In *International conference on machine learning* (pp. 2690–2700).

Engle, R. (2002). New frontiers for arch models. *Journal of applied econometrics*, *17*(5), 425–446.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Ghalanos, A. (2020). Introduction to the rugarch package.(version 1.3-1). *Manuscript, http://cran. r-project. org/web/packages/rugarch. Accessed*, *11*.

Giacomini, R. & White, H. (2006). Tests of conditional predictive ability. *Econometrica*, *74*(6), 1545–1578.

Gneiting, T. (2011a). Making and evaluating point forecasts. *Journal of the American Statistical Association*, *106*(494), 746–762.

Gneiting, T. (2011b). Making and evaluating point forecasts. *Journal of the American Statistical Association*, *106*(494), 746–762.

Gneiting, T. & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, *102*(477), 359–378.

Gu, S., Kelly, B. & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33*(5), 2223–2273.

Hansen, P. R., Huang, Z. & Shek, H. H. (2012). Realized garch: a joint model for returns and realized measures of volatility. *Journal of Applied Econometrics*, *27*(6), 877–906.

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

Harvey, A. & Palumbo, D. (2023). Score-driven models for realized volatility. *Journal of Econometrics*.

Harvey, A. C. (2022). Score-driven time series models. *Annual Review of Statistics and Its Application*, *9*, 321–342.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies*, *6*(2), 327–343.

Hothorn, T., Kneib, T. & Bühlmann, P. (2014). Conditional transformation models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *76*(1), 3–27.

Hu, Y., Sun, Z., Han, Y., Hao, X. & Pei, L. (2022). Pavement skid resistance evaluation based on hyperopt-ngboost fusion model using non-contact measurement of 3d macrotexture data. *Arabian Journal for Science and Engineering*, 1–18.

Huang, D., Schlag, C., Shaliastovich, I. & Thimme, J. (2019). Volatility-of-volatility risk. *Journal of Financial and Quantitative Analysis*, *54*(6), 2423–2452.

Huang, Z., Liu, H. & Wang, T. (2016). Modeling long memory volatility using realized measures of volatility: A realized har garch model. *Economic Modelling*, *52*, 812–821.

Kambouroudis, D. S., McMillan, D. G. & Tsakou, K. (2021). Forecasting realized volatility: The role of implied volatility, leverage effect, overnight returns, and volatility of realized volatility. *Journal of Futures Markets*, *41*(10), 1618–1639.

Kavzoglu, T. & Teke, A. (2022). Predictive performances of ensemble machine learning algorithms in landslide susceptibility mapping using random forest, extreme gradient boosting (xgboost) and natural gradient boosting (ngboost). *Arabian Journal for Science and Engineering*, *47*(6), 7367–7385.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

Kleen, O. (2023). Scaling and measurement error sensitivity of scoring rules for distribution forecasts. *Available at SSRN 3476461*.

Kleen, O. & Tetereva, A. (2022). A forest full of risk forecasts for managing volatility. *Available at SSRN*.

Li, D., Clements, A. & Drovandi, C. (2021). Efficient bayesian estimation for garch-type models via sequential monte carlo. *Econometrics and Statistics*, *19*, 22–46.

Li, Y., Liang, C., Ma, F. & Wang, J. (2020). The role of the idemv in predicting european stock market volatility during the covid-19 pandemic. *Finance research letters*, *36*, 101749.

Li, Y., Wang, Y. & Wu, B. (2020). Short-term direct probability prediction model of wind

power based on improved natural gradient boosting. *Energies*, *13*(18), 4629.

Li, Z.-C., Xie, C., Zeng, Z.-J., Wang, G.-J. & Zhang, T. (2023). Forecasting global stock market volatilities in an uncertain world. *International Review of Financial Analysis*, *85*, 102463.

Liu, L. Y., Patton, A. J. & 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.

Martens, J. (2020). New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, *21*(146), 1–76.

McAleer, M. & Medeiros, M. C. (2008). Realized volatility: A review. *Econometric reviews*, *27*(1-3), 10–45.

Mitrentsis, G. & Lens, H. (2022a). An interpretable probabilistic model for short-term solar power forecasting using natural gradient boosting. *Applied Energy*, *309*, 118473.

Mitrentsis, G. & Lens, H. (2022b). An interpretable probabilistic model for short-term solar power forecasting using natural gradient boosting. *Applied Energy*, *309*, 118473.

Mukherjee, A. & Swanson, N. R. (2021). Evidence on the importance of volatility density forecasting for financial risk management. *Available at SSRN 3964200*.

Muller, U. A., Dacorogna, M. M., Dave, R. D., Pictet, O. V., Olsen, R. B. & Ward, J. R. (1995). Fractals and intrinsic time – a challenge to econometricians.

Nonejad, N. (2017). Modeling and forecasting aggregate stock market volatility in unstable environments using mixture innovation regressions. *Journal of Forecasting*, *36*(6), 718–740.

Oh, D. H. & Patton, A. J. (2023). Dynamic factor copula models with estimated cluster assignments. *Journal of Econometrics*.

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

Peng, T., Zhi, X., Ji, Y., Ji, L. & Tian, Y. (2020). Prediction skill of extended range 2-m maximum air temperature probabilistic forecasts using machine learning post-processing methods. *Atmosphere*, *11*(8), 823.

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

Shen, K., Qin, H., Zhou, J. & Liu, G. (2022). Runoff probability prediction model based on natural gradient boosting with tree-structured parzen estimator optimization. *Water*, *14*(4), 545.

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

Todorova, N. (2015). The course of realized volatility in the lme non-ferrous metal market. *Economic Modelling*, *51*, 1–12.

Trucíos, C. & Hotta, L. K. (2016). Bootstrap prediction in univariate volatility models with leverage effect. *Mathematics and Computers in Simulation*, *120*, 91–103.

Trucíos, C., Hotta, L. K. & Ruiz, E. (2017). Robust bootstrap forecast densities for garch returns and volatilities. *Journal of Statistical Computation and Simulation*, *87*(16), 3152–3174.

Trucíos, C., Hotta, L. K. & Ruiz, E. (2018). Robust bootstrap densities for dynamic conditional

correlations: implications for portfolio selection and value-at-risk. *Journal of Statistical Computation and Simulation*, *88*(10), 1976–2000.

Wu, Y. & Wang, Q. (2021). Lightgbm based optiver realized volatility prediction. In *2021 ieee international conference on computer science, artificial intelligence and electronic engineering (csaiee)* (pp. 227–230).

Yan, P., Fan, W. & Zhang, R. (2023). Predicting the nox emissions of low heat value gas rich-quench-lean combustor via three integrated learning algorithms with bayesian optimization. *Energy*, *273*, 127227.

Zhang, Y., Wang, Y. & Ma, F. (2021). Forecasting us stock market volatility: How to use international volatility information. *Journal of Forecasting*, *40*(5), 733–768.

Žikeš, F. & Baruník, J. (2015). Semi-parametric conditional quantile models for financial returns and realized volatility. *Journal of Financial Econometrics*, *14*(1), 185–226.

A Appendix

Table A.1: S&P 500 summary statistics

	Mean	Median	Std	Min	Max	Skew	Kurt
Panel A: Stock specific variables							
RV^d	0.744	0.566	0.609	0.089	7.878	3.624	23.622
RV^w	0.744	0.573	0.570	0.160	5.363	3.320	18.677
RV^m	0.743	0.571	0.525	0.198	4.200	3.028	15.050
rv_minus	0.461	0.156	1.229	0.003	23.684	9.122	119.911
rv_plus	0.463	0.157	1.362	0.005	38.456	11.840	228.833
lagRQ1	1.502	0.832	2.664	-1.017	38.002	5.670	55.125
lagRQ2to5	1.505	1.133	1.418	-0.612	11.137	2.342	11.428
lagRQ6to22	1.508	1.350	0.791	-0.067	4.879	0.911	3.846
lagreturns1	0.005	0.051	0.975	-8.504	7.324	-0.452	12.442
lagreturns2to5	0.005	0.032	0.438	-3.945	3.054	-0.634	11.113
lagreturns6to22	0.005	0.025	0.201	-1.521	1.367	-0.735	8.795
Panel B: non-stock specific variables							
daily_infect_emv	2.116	0.310	7.716	0.000	112.930	6.243	52.489
DOWjones_RV_lag1	1.204	0.431	3.319	0.019	86.240	11.079	193.497
GPRD_lag1	94.015	88.994	38.815	7.063	413.456	1.295	7.484
T10Y2Y_lag1	1.377	1.420	0.837	-0.150	2.910	-0.006	1.858
OFR FSI_lag1	0.347	-1.120	4.954	-5.334	29.320	2.505	10.891
RiskAversion_lag1	3.232	2.846	1.685	2.425	32.711	9.000	115.695
vix	19.983	17.060	9.756	9.140	82.690	2.368	10.606
VVIX_lag1	91.576	88.920	16.601	59.740	207.590	1.543	7.341

Notes: This table reports the average summary of statistics for the S&P500. For the features used, the mean, median, standard deviation, minimum, maximum, and skewness are reported. The sample used spans from January 2007 to December 2020.

Figure A.1: Contemporaneous correlations Matrix RV 2007-2015 Heatmap

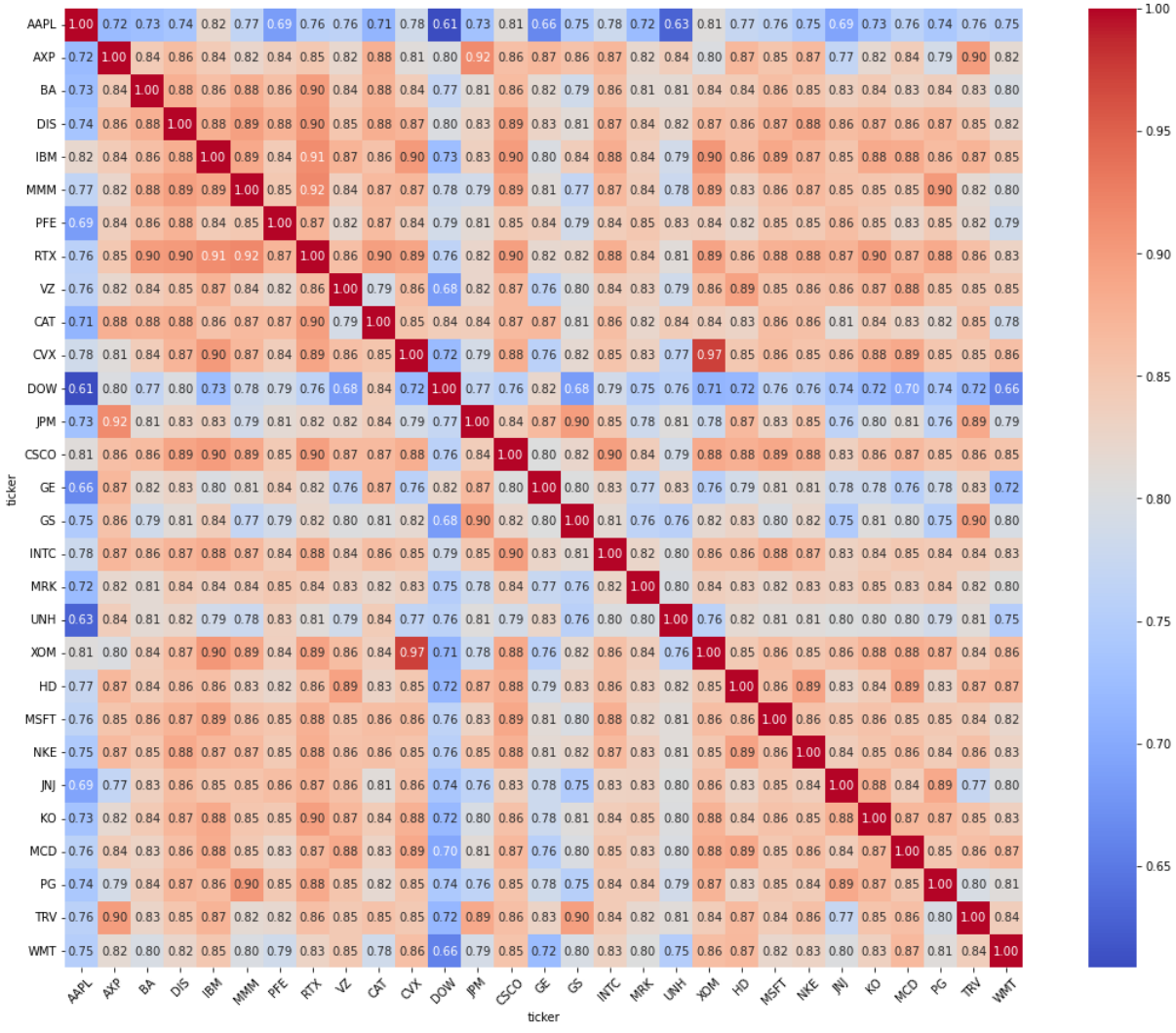
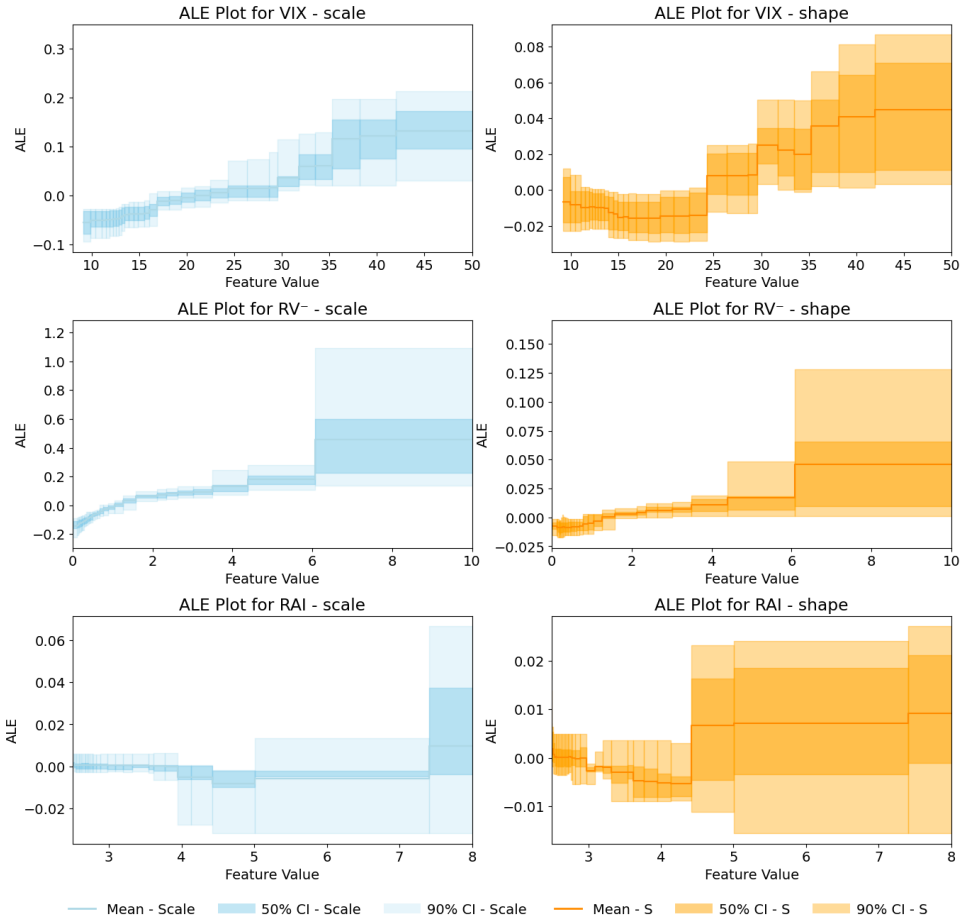


Figure A.2: ALE ResNGBoost



Notes: For the cluster ResNGboost model, this figure shows the ALE fan charts for the scale and shape parameters of the lognormal distribution. The individual calculated ALE values are aggregated into fan charts. The bands correspond to a 50% and 90% interval. Included is the mean ALE value across stocks. Note that the x-as is truncated for each feature separately; after the specific value, the fan chart does not change and stays constant.