Erasmus School of Economics

Erasmus University Rotterdam

Master Thesis: Econometrics & Management Science

Business Analytics & Quantitative Marketing

Forecast combination in high frequency data

Niels van der Slot

Student number: 545546

Supervisor: E. P. O'Neill

Co-reader: dr. A. Pick

August 27, 2021

Abstract

Previous studies showed that forecasting accuracy can be improved through combining forecasting methods. Research is mainly performed in low-frequency data. In this paper, the utility of combining forecasts is investigated for high-frequency data through daily, hourly and 15-min interval data. Our analysis showed that all investigated combination methods outperformed selecting a single forecaster. The two best performing combination methods were Inverse Weighting and a meta-learning framework, FFORMA. In Inverse Weighting, weights for the forecasters are based on their relative performance in a validation set. The FFORMA framework is based on machine learning where the model learns to relate features of a time series to the optimal set of weights. Three extensions of the FFORMA framework are investigated. Firstly, the computation time improved significantly and the accuracy slightly by using LightGBM as the machine learner. Secondly, we introduce a new feature to the FFORMA framework that accounts for nested seasonality, which can occur in high-frequency data. Finally, the rankings of validation errors are introduced as features, which do not improve the forecasting performance. Using data from multiple sectors in the FFORMA framework had a varying effect on the performance. Furthermore, the forecasts from the combination methods were again combined to increase the forecasting accuracy. Our analysis showed that forecasting accuracy can be improved by creating a weighted combination of forecasts obtained from different forecast combination methods.

Contents

1 Introduction

Forecasting is a long-lasting exploration that exists ever since humans started planning. Last decades, forecasting has gotten more exact in nature and a vast body of literature regarding forecasting methods has accumulated. Makridakis & Hibon (2000), the organizers of the international M forecasting competitions, stated the value in accurate forecasting clearly and concisely:

> Forecasting accuracy is a critical factor for, among other things, reducing costs and providing better customer service. Yet the knowledge and experience available for improving such accuracy for specific situations is not always utilized. The consequence is actual and/or opportunity losses, sometimes of considerable magnitude.

The year 2020 has proven the crucial role of forecasting once again. Due to the COVID-19 pandemic and its consequences, measures against an uncontrolled spread have to be taken quickly and efficiently. Many countries in Europe are taking measures to prevent hospitals from over flooding while trying to keep the economic costs of these measures as low as possible. This means that as decisions are made based on expected COVID cases, accurate forecasts are a vital part of the process. Outside of governmental decisions, accurate forecasting reduces waste, ensures efficient allocation of resources, and reduces labor costs.

The digitization of the 21st century has meant a stark increase in data collection. Automatic sensors and scanning devices now provide businesses with high-frequency data in various sectors. This data is valuable in all sorts of businesses, as labor resources and business processes can be optimized and customer flow modeled. Also, when looking at the food and hospitality industry many resources go bad quickly. The United Nations environment programme estimates that the amount of food wasted is 17 percent of the food available (United Nations Environment Programme, 2021). Accurate forecasts would be beneficial for the bottom-line of the food and hospitality industry, as well as the environment. Because of the increased availability of high-frequency data, it is possible to optimize many processes that operate on a higher frequency. The need for accurate high-frequency forecasting methods is therefore growing. The term high-frequency data is not a clearly defined concept, it is used as a relative term. Most often however, as in this paper, high-frequency data is defined as data that is generated at daily or smaller intervals (Andersen, 2000).

The existing knowledge on the topic of (combining) forecasts in high-frequency data is insufficient because of the lack of publicly available data sources of high-frequency data. This is not to say that high-frequency forecasting is an untouched domain. With Quinyx, a workforce optimization company, it is seen that clients from various industries

have already been building demand forecasting models with hourly demand. Workforce optimization is a topic from which many organizations can benefit, and is based around high-frequency forecasting. To investigate high-frequency forecasting, data is made available by Quinyx' clients, which come from the finance, education and hospitality sector. The data will be anonymized for the purpose of this paper. Time frequencies included are daily, hourly and 15-min interval data. Examples of time series used are visitor counts and products sold. At Quinyx it is seen that in practice data collection is limited to univariate data. The time series used in this paper are therefore univariate, covariates are not available.

Because of the growing need for accurate high-frequency forecasts, an investigation is done whether the results in low-frequency data carry over to high-frequency data. Compared to low-frequency data, high-frequency data contains an extra challenge in the form of nested seasonality. For sub-daily frequencies a daily, weekly and yearly pattern can exist. The literature on high-frequency forecasting might be limited, the research into low-frequency data is extensive. Since the seminal work of Bates & Granger (1969) on combined forecasts, combining forecasts has been considered a successful alternative to individual forecasting methods in low-frequency data. Through investigating the (in low-frequency data) well-performing methods in high-frequency data, the value of the combination methods is assessed. This addresses the question: does forecast combination increase forecasting performance in high-frequency data?

An important finding in combination forecasting is that, in practice, complex combinations of forecasting methods do not outperform simple combination schemes. It is found that a simple average of forecasts is often more accurate than optimized weights (Smith & Wallis, 2009). This phenomenon is also known as the 'forecast combination puzzle'. In a recent forecasting competition based on 100.000 time series, the M4 competition, simple combinations proved valuable again (Makridakis & Spiliotis, 2018), taking 3rd and 4th place. This is why three simple combination methods are considered in this paper. The best performing combination method in the M4 competition was a novel, complex approach to selecting weights: the FFORMA framework. In this complex approach, weights are estimated through a meta-learning approach using machine learning. Learning from a large number of time series, the machine-learning model selects the optimal weights, given the characteristics of a time series. The above-mentioned methods proved to be accurate when applied to low-frequency data. We perform an analysis of whether these methods achieve similar results in high-frequency data.

The absolute forecasting horizon generally decreases as the frequency of the time series increases. That is, for hourly data it is normal to make forecasts for a forecasting horizon of two weeks, whereas with quarterly data forecasts are made for a year. Also, the values available increase for higher frequencies, increasing computation time. Therefore, computation time becomes an increasingly important aspect of forecasting for higher

frequencies. Currently the gradient boosting method XGBoost is used in the FFORMA framework. LightGBM has shown to be a computationally inexpensive gradient boosting method. In this paper, an extension of the FFORMA framework is presented such that LightGBM is used as machine learner instead of XGBoost. The following question is investigated: does LightGBM decrease computation time while achieving similar or better performance?

Three additional analyses are performed in the FFORMA framework. Firstly, the ranking of the validation errors are introduced as features, giving insight to which forecasting methods perform well on a time series. Secondly, the *month* feature is introduced to capture yearly seasonality. Thirdly, an investigation is done on whether it is useful to include data from other sectors to train the model.

Finally, an extension on forecast combination is presented through repeated combination. In a second combination layer, a weighted forecast is produced out of the combination methods to increase the forecasting accuracy further. This means that it is not necessary to choose between combination methods, but the strengths of each combination method is used through repeated combination. The question addressed is: does repeating the combination process increase forecasting performance?

The results of this study suggest that all combination methods outperform selecting a single forecaster. In the best performing simple combination method, weights are selected according to historical accuracy. The FFORMA framework achieves the highest accuracy of all combination methods investigated. The FFORMA framework is extended through the use of LightGBM as machine learner, which decreases computation time and increases performance. Regarding repeated combination, the results indicate that the performance indeed increases when introducing a second combination layer. This finding is promising and it would be interesting to see if the result can be replicated in low-frequency data.

This paper is structured as follows: existing methods and relevant information are discussed in section 2. The error measure and combination methods used will be explained in-depth in section 3, as well as the tests to determine significant differences between the methods. The methods will then be applied to 2504 time series which include daily, hourly and 15-min interval data described in section 4. Results are stated in section 5 and discussed in section 6.

2 Literature

The seminal work on combining forecasts was provided by Bates & Granger (1969). Forecasting passenger miles flown, Bates and Granger concluded that combining two forecasts can lead to a smaller Mean-Squared-Error. In their study, the optimal weights for combining forecasts were estimated through minimizing sample variance. The initial work of Bates and Granger 'led to an explosion in the number of articles on the combination of forecasts' as stated by Clemen (1989) in his review on 20 years of combining forecasts. Although the idea of optimizing weights through minimizing variance is countered (Clemen, 1989; Stock & Watson, 2004), the conclusion that forecasting accuracy can be improved through combining forecasts is not. Hibon (2005) shows the straightforward value of combining forecasts. When evaluating all forecasting methods and combination methods, the best individual forecaster performs as well as the best combination method. But since it is not known which single forecasting method will perform best before-hand, this lacks practical value. When model selection is done to select the best single method, combination methods consistently outperform single forecasting methods. Since the introduction of forecast combination by Bates & Granger (1969), many schemes have been proposed to combine forecasts. The most prevalent and relevant combination methods are described below. Combination methods can be divided into two camps. Simple combination methods and combination methods where weights are optimized.

Frequently used combination methods that have optimized weighting schemes include Least Squares and Bayesian Model Averaging. With Least Squares, the actual values are regressed on the forecasts of different methods. Then, out of the least-squares estimates weights are produced. Most popular to minimize over is the Mallows criterion, since the Mallows criterion is asymptotically equal to the squared error (Hansen, 2007). The weights that minimize the Mallows criterion therefore also minimize the squared error. Even though some studies suggest that least-squares methods are superior (Holmen, 1987; Wan et al., 2010), a larger body of literature, containing both theoretical and empirical results, counter this finding (Clemen, 1989; Diebold & Lopez, 1996; Newbold & Harvey, 2002; Stock & Watson, 2004). Furthermore, it is possible to combine forecasting methods using Bayes' rule. Bayes' rule is used to update the probability of a model to be the true model as information becomes available. When using Bayes' rule for model selection, the model with the highest posterior probability is chosen. When combining forecasts, forecasting methods are weighted according to the posterior odds (Bordley, 1982; Koop & Potter, 2004). Although a Bayesian approach to forecast combination improves forecasting performance over a single forecasting method, it does not consistently outperform other combination methods (Hsiao & Wan, 2014). These are not the only complex combination methods in the literature. Frequentist model averaging (Kapetanios et al., 2005), (Bayesian) stacking (Yao et al., 2018; Breiman, 1996) and many others exist. There is

little empirical evidence that these consistently outperform simple combination methods however, and are therefore not considered.

In a literature review on combining forecasts, Clemen (1989) concluded that an equal weighting of individual forecasts is often most accurate, as opposed to complex (optimized) weighting schemes. Genre (2013) found similar results when comparing complex combining methods in macro-economic variables forecasting. The complex combining methods tried to estimate the optimal weights. They found that no complex method was consistently more accurate than the simple average, and recommended equal weighting as the headline indicator in the European Central Bank Survey of Professional Forecasters. The empirical finding that complex weighting schemes do not consistently outperform simple combinations is also known as 'the forecast combination puzzle' coined by Stock & Watson (2004). This is why three simple combination methods are used in this paper.

In their Monte Carlo study, Smith & Wallis (2009) provided an explanation of the forecast combination puzzle. The assumption is often made that the weights are known. In reality, the weights need to be estimated and contain a finite-sample error. Smith & Wallis showed empirically that the effect of the error of the estimated weights is the cause of optimized weights not performing well against simple weights. They therefore recommend ignoring the forecast error covariance when estimating weights for forecast combination. Claeskens (2016) contributed with a theoretical explanation for the results of Smith & Wallis. They state that the weights are estimated under the assumption that weights are fixed. While the weights are actually, through estimation, random. The forecast combination is therefore biased and its variance will be larger than in the simple weighted case.

In the most recent M forecasting competition, simple combination methods proved their value once again. The M competitions, named after its organizer Makridakis, are held roughly once a decade. The objective is to 'learn how to improve the forecasting accuracy, and how such learning can be applied to advance the theory and practice of forecasting' - Makridakis & Spiliotis (2018). The benefit is therefore not providing new advanced theory, but practical/actionable knowledge. The M competitions have been stated to have changed the landscape of forecasting research. In the M4 competition held in 2018 combination methods were successful being 6 of the 7 most accurate methods based on 100.000 time series. The value in simple combinations was seen as 3^{rd} and 4^{th} place made use of a weighting based on the inverse of the error and a simple average (of selected models) respectively. An M5 competition was organized in 2020, here covariates were introduced to increase forecasting accuracy. In this paper univariate time series are considered, the results of the M5 competition are therefore not discussed.

Jaganathan & Prakash (2020) achieved 4th place building on the guiding principles described in Principles of Forecasting: A Handbook for Researchers and practitioners (Armstrong, 2001). The evidence-based guiding principles were as follows: Use forecasting

methods that differ. If possible, use at least 5 methods. When using 5 or more forecasting methods, the combined forecast might be sensitive to extreme values. Therefore a trimmed mean or median is worth investigating. Jaganathan & Prakash (2020) explored the use of mean, trimmed mean and median of forecasts. The forecasting methods used were hand-picked and differed per frequency. Based on the performance of a hold-out sample the median operator was used in their submission. Because of this performance, a median operator is used in this paper. Outside of the M4 competition, a mean is recommended frequently in literature (Wallis, 2010; Stock & Watson, 2004).

In their Monte Carlo study on estimating weights, Smith & Wallis (2009) recommend weighting forecasting methods according to the inverse of the mean squared forecasting error. Pawlikowski & Chorowska (2020) followed on this result and took 3rd place in the M4 competition through a similar weighting process. The general procedure is as follows. Given a pool of methods, forecast errors are calculated based on a validation set using a rolling window evaluation. The weights are set according to the inverse of the validation forecast errors. Thus, different from equal weighting described above, forecasters with a relatively high historical accuracy will also have a large weight in the combined forecast.

The best performing combination method in the M4 competition was a novel approach to weighting forecasts. Montero-Manso et al. (2020) proposed feature-based forecast model averaging (FFORMA). The method follows on the framework for forecast-model selection with meta-learning by Talagala (2018). Since selecting an appropriate method can be challenging and time-consuming, Talagala (2018) propose a meta-learning model to select the best forecasting method. In the Feature-based FORecast Model Selection (FFORMS) model, Talagala (2018) uses a Random Forest to predict the best forecasting method corresponding to a specific set of features. The idea is that the features of a time series give valuable information as to which forecasting method performs best. Montero-Manso et al. (2020) follows on this work by using meta-learning to select the best combination of weights in a weighted forecast combination. For the purpose of a combined forecast, XGboost is used so that the objective function can be customized to incorporate the combined forecast. This complex approach to weighting proved more accurate than all submitted simple combinations.

The literature on combination methods is vast, and it can be time consuming to find the best combination method. It is also possible to make use of multiple combination methods. An example of this is multi-level stacking. In multi-level stacking there are multiple layers, with multiple base learners in a layer. Singh et al. (2020) apply multi-level stacking to load forecasting in the electricity grid. The output of a layer is used as input for the next layer. In the second layer, the output of methods of the first layer are combined and in the third layer the combination methods are combined. This idea of repeated combination, although in a different form, is applied in this paper.

Many studies into the best (combined) forecasting method have been conducted for

low and mixed-frequency data. Research in high-frequency data however, remains limited. Research is limited because there are few publicly available high-frequency data sources. Therefore, until now, high-frequency forecasting in literature was mainly restricted to the energy sector through the Global Energy Forecasting Competition (GEFCOM), where forecasts in energy demand are evaluated (Hong, 2020). In GEFCOM covariates are used, which differs from the data in this paper. Combination methods were used, as 3rd place made use of a simple average of forecasting methods (Smyl & Hua, 2019). A challenge that is unique to high-frequency data is a complex seasonal pattern in the form of nested seasonality or non-integer seasonality. For sub-daily frequencies a daily, weekly and even yearly pattern can exist (Dharmawardane et al., 2021). Non-integer seasonality is seen in the yearly pattern, as the frequency can be 52 or 53 weeks. The forecasting methods used in this paper do not handle the complexity of multiple seasonalities well, as few forecasting methods do (Naim et al., 2018).

An important aspect of evaluating forecasting methods is the performance measure. Many measures have been proposed, Hyndman & Koehler (2006) provide a summary of popular measures and their drawbacks. Many performance measures are either scale-dependent or handle 0 values poorly. This is undesirable when comparing forecasting methods across many time series. An alternative in scaling errors is to divide the error by the error of a standard forecasting method as done in the Mean Absolute Scaled Error (MASE). Hyndman & Koehler recommend that MASE becomes the standard when comparing forecasting accuracy's. The MASE was also used in the M4 forecasting competition, and has been shown to have favourable mathematical properties (Franses, 2016). Because of these results, the MASE is used in this paper to compare performances across methods.

3 Methodology

This section describes the methods that are used to make a comparison of combination techniques in high-frequency forecasting. In the literature, it is repeatedly found that simple combinations outperform complex optimized weighting schemes (see section 2). The recommendation is made to use either a (trimmed) mean, median, or weighting based on the inverse of the forecasting error (Clemen, 1989; Stock & Watson, 2004; Smith & Wallis, 2009). This recommendation held up in the M4 competition where these methods performed well. These three simple combination methods will therefore be investigated. Furthermore, the novel FFORMA framework proved the most accurate combination method in the M4 forecasting competition and will therefore also be used on our high-frequency data set. Least Squares and Bayesian Model Averaging will not be used because of the theoretical and empirical evidence of their inferiority compared to other combination methods (Hsiao & Wan, 2014; Stock & Watson, 2004). As a benchmark, the combination methods will be compared to the most accurate single forecasting method as determined by the validation set. This comparison will display the value of combining forecasts in high-frequency data.

3.1 Performance measure

In this paper, forecasting methods are compared across many time series. Popular measures such as the Mean Squared Error are in a branch of error measures where the error is scale-dependent. While this is valid when comparing methods on a single time series, it is not useful when comparing methods across time series with different scales. Tackling this problem, forecasting measures are designed based on percentage errors. The symmetric Mean Absolute Percentage Error (Makridakis, 1993) (sMAPE) is a popular scale-independent forecast measure used in the M3 and M4 forecasting competitions. Percentage errors have the disadvantage of being infinite when Y_t as the percentage error is calculated through $p_t = 100 * \frac{e_t}{Y_T}$. Even if there are no 0 values, values close to 0 will have a disproportionate impact. The data used in this paper contains 0 values, the sMAPE is therefore not suitable.

The Mean Absolute Scaled Error contains (MASE) is scale-independent and handles zero/low values well. Because of these favorable properties, the MASE is used to compare forecasting methods across time series. The scaling in the MASE is done through dividing the forecast error e_t by the in-sample mean absolute error of the seasonal naive forecast method. Seasonal naive forecasting is setting the forecast to be equal to the last observed value of the same season. For hourly data, this means that the forecast will be the value at the same hour the previous day. The scaled error is given by equation 1. Where n is the amount of points in-sample, and m is the frequency.

$$q_t = \frac{\hat{Y}_t - Y_t}{\frac{1}{n-m}\sum_{t=m+1}^{n} |Y_t - Y_{t-1}|} \tag{1}$$

The MASE is the mean of the absolute value of q_t, seen in equation 2, where h is the horizon for which forecasts are made.

$$MASE = \frac{1}{h}\sum_{t=n+1}^{n+h} |q_t| \tag{2}$$

3.2 Forecasting methods

The pool of forecasting methods consists of 9 methods as specified below. These methods are selected because of their empirical value in demand forecasting at Quinyx. The forecasting methods are described more in-depth in the Appendix, section A.3.

- Gradient boosting regressor
- Lasso
- Linear regression (OLS)
- Xgboost regressor
- Arima
- Holt Winters
- (Trend) Exponential Smoothing
- Moving Average
- Random forest auto regressor

In high-frequency forecasting, the absolute forecasting period is small relative to low-frequency forecasts. i.e., with quarterly data forecasts are made for a year whereas with hourly data the forecasting period can be two weeks or even a day. This means that forecasts are made more frequently for high-frequency data than for low-frequency data. Also, time series are longer in terms of the number of observations. Comparing 15 min-interval data to hourly data, there are already 4 times as many observations. Since there is more training data available, methods take longer to train. If a rolling window approach is used to determine weights, there are more potential windows that can be evaluated. Lastly, if multiple forecasting horizons are considered, the number of possible horizons increase with high-frequency data. Concluding, compared to low-frequency data, with high-frequency data the computation time increases as well as the frequency at which forecasts are made.

A short computation time is always desirable but is increasingly significant when the frequency at which forecasts should be made goes up. Computation time becomes increasingly significant and computationally expensive methods like LSTM neural networks are less desirable and are therefore not included in the forecasting pool. Also, many variations

of the mentioned forecasting methods exist. However, the value of combining lies in the fact that the various methods extract different information out of the data to generate forecasts (Armstrong, 2001). Small variations of the forecasting methods are therefore not included in the pool of methods. Many forecasting methods exist and the search for the best pool of forecasting methods can be extensive. The goal of this research is to explore the value (if any) in combination forecasts relative to single forecasting methods. For this purpose the pool of methods is sufficient.

3.3 Simple weighting

In the simple weighting model either a mean or median of the forecasting methods is used to compute the forecast. The forecasting pool consists of 9 forecasting methods. Some of the methods might forecast relatively extreme values. Taking the median of each forecast in the forecasting horizon controls for extreme values. It is also possible that it is a better strategy to use a mean operator on the best x forecasters, based on a validation set. In reality, it is not known what number of forecasters should be included. In the validation set, the performance is assessed when using the x best forecasters, varying x. Based on these results, the best value of x is determined and used when combining forecasts in the test set.

3.4 Inverse weighting

Based on successful results by Pawlikowski & Chorowska (2020) a weighting scheme based on the performance in the validation set will be used. The procedure is as follows. Forecasts are done on the validation set for each method in the forecasting pool. The MASE is then calculated for each forecasting method. The MASE is then transformed with an operator as described below. The weights for each forecasting method are based on the transformed values. The transformation that determines the weights is seen in equation 3. The weight is calculated through dividing the transformed value by the sum of all transformed values, so that the weights sum to one.

$$f_{inv}(e) = \frac{1}{|e| + \epsilon}, \tag{3}$$

In theory, it is possible that the MASE is 0. To avoid dividing by 0, $\epsilon = 0.0001$ is added to the denominator. In practice the MASE has a value upwards of 0.3, ϵ has no significant impact on the relative weights.

Including all forecasting methods in the forecasting pool might not prove beneficial to the overall forecasting accuracy. With a large pool of forecasting methods, the combined accuracy might only benefit from a specific amount of forecasts. The number of forecasting methods is optimized in the following manner. The top x number of methods are included

in the forecast average. The number of methods x is varied and the mean MASE across time series is calculated to determine the optimal value of x. In the inverse weighted combination method, the weights are based on previous forecasting errors. Tuning of the hyperparameters therefore requires an extra validation set. The extra validation set is used to calculate the errors, which are used to produce weights for the validation set. With the validation set, the optimal amount of forecasters x to include is determined. Then with the validation set and the optimal amount of forecasters x, forecasting errors are calculated to determine the weights for the test set.

Weights are now based on the forecasting errors in the validation set. This can introduce some randomness concerning the weights. A forecaster performing well on a single validation set, does not necessarily mean it usually performs well. In a rolling window approach the errors are calculated over multiple windows, the mean of these errors is computed and used to produce weights. The rolling window approach helps to mitigate the randomness in the errors, and thus the weights. The rolling window approach is computationally expensive however and is therefore not applied in this paper.

3.5 FFORMA

3.5.1 Meta-learning model

Following the work of Montero-Manso et al. (2020) as described in section 2, forecasts are combined via the FFORMA framework. The objective of the FFORMA framework is to derive a set of weights for a set of forecasting methods given the features of a time-series. To determine which weights correspond to a certain time-series, a meta-learning approach is taken. The FFORMA approach consists of two phases, as seen in algorithm 1.

In *phase 1* the meta-learning model is trained. To train the meta-learning model, many time series are needed, the reference set. In this paper, the reference set consists of all available time series. Reason for this is that forecasts are not made for new time series, but rather for new observations for each time series. The split into training and validation data is across time, not across time series. Forecasts are made for each forecasting method out of the forecasting pool and a set of forecasting errors $\mathbf{L}_i$ is computed for each time-series i. Features are extracted out of the training set (e.g. seasonality, trend, auto-correlations) and used as input for the meta-learning model. A full list of features is given in the Appendix. In this paper, the gradient boosting framework LightGBM is used to produce a combined forecast, given a set of features $\mathbf{f}_i$. LightGBM is used because the objective function is customizable and can be tailored to optimize the overall combined forecast. Through minimizing the average combined forecasting error, the meta-learning model is trained. In the original use of the FFORMA model XGBoost was used as the gradient boosting framework. As an extension of the FFORMA model LightGBM is now used. LightGBM is faster than XGBoost while achieving comparable accuracy (Ke et al., 2017). LightGBM and XGBoost are discussed more thoroughly in section 3.5.2.

In *phase 2* the meta-learning model is used to produce forecasts. Features are now calculated over all data except for the data on which the forecasts are evaluated. The features are then used as input for the trained meta-learning model. The meta-learning model gives a set of weights as output, which is then used to create the combined forecast. All 2504 time series are used to train the meta-learning model. The underlying assumption is that, given the reference set, the features of a new time series give insight to the data generating process and with it the ability to create an accurate combined forecast.

Concluding, there are two phases in the FFORMA model as seen in algorithm 1. Firstly, the phase in which the meta-learner is trained. This is a computationally expensive process. For each time series, features have to be extracted and individual forecasts are done to compute forecasting errors. Thereafter, weights are optimized through minimizing the overall combined forecasting error. Secondly, the online phase. The FFORMA model is now trained. Given a new forecasting period, features are extracted from the historical data and used as input in the model. As output, weights are given to create a combined forecast. This process is visualized in figure 1. The feature list is the full list used by

Montero-Manso et al. (2020), extended with the *month* and *time-interval* feature. Also, in this paper an attempt is made to improve the FFORMA framework through adding the ranking of validation errors as features. The idea is that the validation errors give information about which forecasters perform well on a time series. This is similar to Inverse Weighting, where the weights are determined through the validation errors.

Algorithm 1 : FFORMA Model

Phase 1: Compute meta-data and train the meta-learner

Set of time-series $(x_1, x_2, ..., x_N)$

F: set of functions to calculate features $\mathbf{f}_i$ for time-series i

M: set of forecasting methods to calculate forecasts $\mathbf{m}_i$ for time-series i

1: **for** i = 1, ..., N **do**
2: Split x_i into a training and test period
3: Extract set of features $\mathbf{f_i}$ out of the training set
4: Generate forecasts for each forecasting method over the test period
5: Calculate forecasting error L_{ij} for each forecasting method j over the test period
6: **end for**

Train the FFORMA model through minimizing:

$$argmin_w \sum_{i=1}^{N} \sum_{j=1}^{M} w(\mathbf{f_i})_j L_{ij}, \qquad \text{with} \qquad \sum_{j=1}^{M} w(\mathbf{f}_i)_j = 1$$

Phase 2: Use the meta-learner to forecast

1: **for** new time-series x_{new} **do**
2: Calculate features $\mathbf{f_{new}}$ by applying F
3: Use the meta-learner to create a vector of weights **w**
4: Compute the forecasts of each forecasting method m in the pool
5: Combine the individual forecasts using **w** to generate final forecasts
6: **end for**

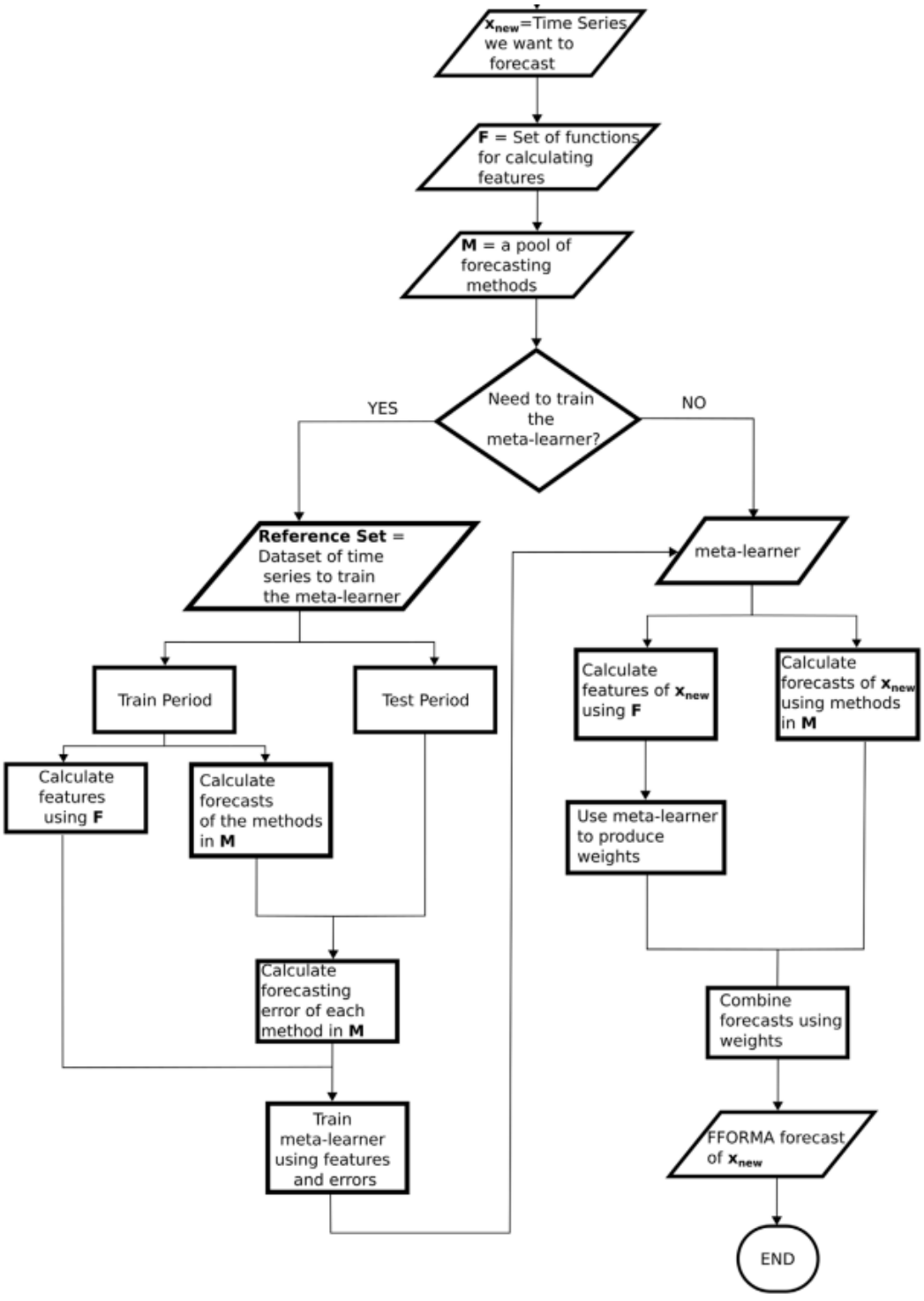


Figure 1: Visual representation of the FFORMA model (Montero-Manso et al., 2020)

3.5.2 Gradient boosting framework

In the described FFORMA model, the weights are optimized through the LightGBM gradient boosting framework (Ke et al., 2017). This differs from the original paper of Montero-Manso et al. (2020) where XGBoost (Chen & Guestrin, 2016) was used. Gradient boosting is a machine learning technique which creates a prediction model in the form of an ensemble of weak prediction models. In the case of LightGBM and XGBoost the weak learners are decision trees. Gradient boosting can be seen as an iterative gradient descent algorithm. For LightGBM this means that we have a model of decision trees, and in each step a leaf will be fitted to the residual of the model. In this manner the new leaf attempts to correct the error of the model. Leaves are added to the decision trees in an iterative process. XGBoost differs from LightGBM because it uses level wise growth of decision trees instead of leaf wise growth (Ke et al., 2017). This different manner of growth can lead to LightGBM outperforming XGBoost (Ma et al., 2018; Zhang & Gong, 2020). It can also lead to over-fitting however. A max depth is introduced and early stopping is used to prevent this. Also, LightGBM differentiates from XGBoost through using two novel techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Building (EFB). With GOSS only data instances with a large gradient and a random sample of small gradients are used to estimate information gain. This allows LightGBM to find the most influential cuts quickly. Through EFB mutually exclusive features are bundled to reduce the number of features. Both GOSS and EFB help reduce computing time.

In the FFORMA framework, the LightGBM algorithm takes as input the features $\mathbf{f}_i$ of a time series and produces values $p(\mathbf{f_i})_j$ for each forecasting method j. The softmax transformation seen in equation 4 is used to transform the values into probabilities. The softmax transformation helps prevent overfitting since it ensures the weights sum to one. The resulting error function is given in equation 5. A probability is given for each forecasting method, making the combined forecast more interpretable. The probabilities give insight as to which forecasting methods are important for a time series.

$$w(\mathbf{f}_i)_j = \frac{e^{p(\mathbf{f}_i)_j}}{\sum_{j=1}^{M} e^{p(\mathbf{f}_i)_j}} \tag{4}$$

$$L(\mathbf{f}_i, x_i) = \sum_{j=1}^{M} w(\mathbf{f}_i)_j \star MASE_j \tag{5}$$

A vital advantage of LightGBM (and XGBoost) is that the objective function is customizable. It is possible to tailor the objective function so that it contains the combined forecasting error. LightGBM minimizes the average combined forecasting error as given by our objective function in equation 6. If the objective function of a machine learning model is not customizable, the error is minimized through a standard measure. This usually means that the amount of times the most accurate forecaster is selected is optimized.

This is different from optimizing the combined forecast, which is the actual goal. For this reason, other machine learning methods are not considered. The objective function now contains the MASE, it is also possible to minimize over other error measures if desired.

$$argmin_w L = \sum_{i}^{N} L(\mathbf{f}_i, x_i) \tag{6}$$

In order for the gradient boosting to converge, both the gradient and Hessian are computed for LightGBM. The gradient of the loss function is given by equation 7. The Hessian can be approximated by equation 8 to avoid numerical problems in boosting.

$$G_{ij} = \frac{\delta L_i}{\delta p(\mathbf{f}_i)_j} = w_{ij}(L_{ij} - L_i) \tag{7}$$

$$H_{ij} = \frac{\delta G_i}{\delta p(\mathbf{f}_i)_j} \approx \hat{H}_i = w_i(L_i(1 - w_i) - G_i) \tag{8}$$

The LightGBM algorithm contains many hyperparameters. Tuning them via a grid or random search can be time-consuming. The hyperparameters are therefore optimized through Bayesian optimization using the validation set. In bayesian optimization, hyperparameters are optimized while taking previously learned information into account.

3.6 Method comparison

To test whether there are significant differences between two forecasting methods the Diebold-Mariano test is used frequently. However, this test is only used to compare forecasts on a single time series. This test is therefore not suitable, as it is desired to compare forecasting methods across many time series. Therefore, following the statistical test as done by Koning et al. (2005), an overall test is done to determine any difference between the combination methods.

3.6.1 Overall test

To compare the methods across many time series, the Friedman test is used (Friedman, 1937, 1939). The Friedman test is non-parametric and similar to the parametric ANOVA. The null hypopaper of the Friedman test is seen in equation 9 and the alternative hypopaper in equation 10, where τ_j is the method effect. If any method effect τ_j (method j) is smaller than any other method effect, the accuracy of method j will in general be better than the accuracy of any other method. See Koning et al. (2005) for a detailed derivation of the method effect τ_j.

$$H_0 : \tau_1 = \tau_2 = ...\tau_k \tag{9}$$

$$H_1 : \tau_1, \tau_2, ..., \tau_k \quad \text{not all equal} \tag{10}$$

If H_0 holds, then rankings $R_{n1}, R_{n2}, ..., R_{nk}$ are obtained by ranking i.i.d. random variables $V_{n1+\tau_1}, V_{n2+\tau_2}, ..., V_{n+\tau_k}$ for each time series n. If H_0 is rejected however, then the random variables $V_{n1+\tau_1}, V_{n2+\tau_2}, ..., V_{n+\tau_k}$ are independent, but may differ in location with respect to each other. The Friedman test is based on test statistic seen in equation 11.

$$S = \frac{12N}{K(K+1)} \sum_{k=1}^{K} (\bar{R}_k - \frac{K+1}{2})^2 \tag{11}$$

Under the null hypopaper, S converges in distribution to a Chi-square random variable with *K - 1* degrees of freedom, as N tends to infinity.

3.6.2 Multiple comparison with the best

If the Null hypopaper of the Friedman test is rejected, an investigation can be made into which components are rejected. For this case McDonald & Thompson (1967, 1972) developed a multiple-comparisons procedure. The component hypotheses are formed as seen in 12.

$$H_0, k_1, k_2 : \tau_{k_1} = \tau_{k_2} \text{ with } k_1 = 1, 2, .., k_2 - 1 \text{ and } k_2 = 1, 2, .., K \tag{12}$$

Each component H_{0,k_1,k_2} is rejected if and only if $|\bar{R_{k1}} - \bar{R_{k2}}| \leq r_{\alpha,K,N}$ where $r_{\alpha,K,N}$ is chosen to make the experiment-wise error rate equal to α. For large N, $r_{\alpha,K,N}$ is approximated by equation 13, where $q_{\alpha,K}$ is the upper α percentile point of the range of K independent standard normal variables (Hollander & Wolfe, 1999). Values of $q_{\alpha,K}$ can be found in Table 1 in Harter (1960).

$$r_{\alpha,K,N} \approx q_{\alpha,K} \sqrt{\frac{K(K+1)}{12N}} \tag{13}$$

The results are visualized with a plot. For method k an interval is taken of length $r_{\alpha,K,N}$, centered at R_K. If intervals of two methods do not overlap, H_0 is rejected.

3.7 Repeated combination

Through combining forecasting methods, the overall forecasting error can be reduced. An investigation is done whether repeating this process can reduce the forecasting error further. This would mean that the strengths of all combination methods are used, and it is not needed to select a single best combination method. The following methods are compared for combining forecast combinations. Firstly, a Random Forest classifier is used to select the best combination method. Features of the time series as described in section

3.5, are used as input for the classifier. The dependent variable is a categorical variable indicating the method with the smallest error. As the features represent the nature of a time series, the classifier will identify the strength of each combination method. Secondly, all previously discussed combination methods will be used to again combine the forecasts. That is, using Inverse Weighting, Median, Average and FFORMA weights will be produced to create a combined forecast once more. The outcome of the classifier and the combination methods are compared. The process of repeated combination is visualized in figure 2.

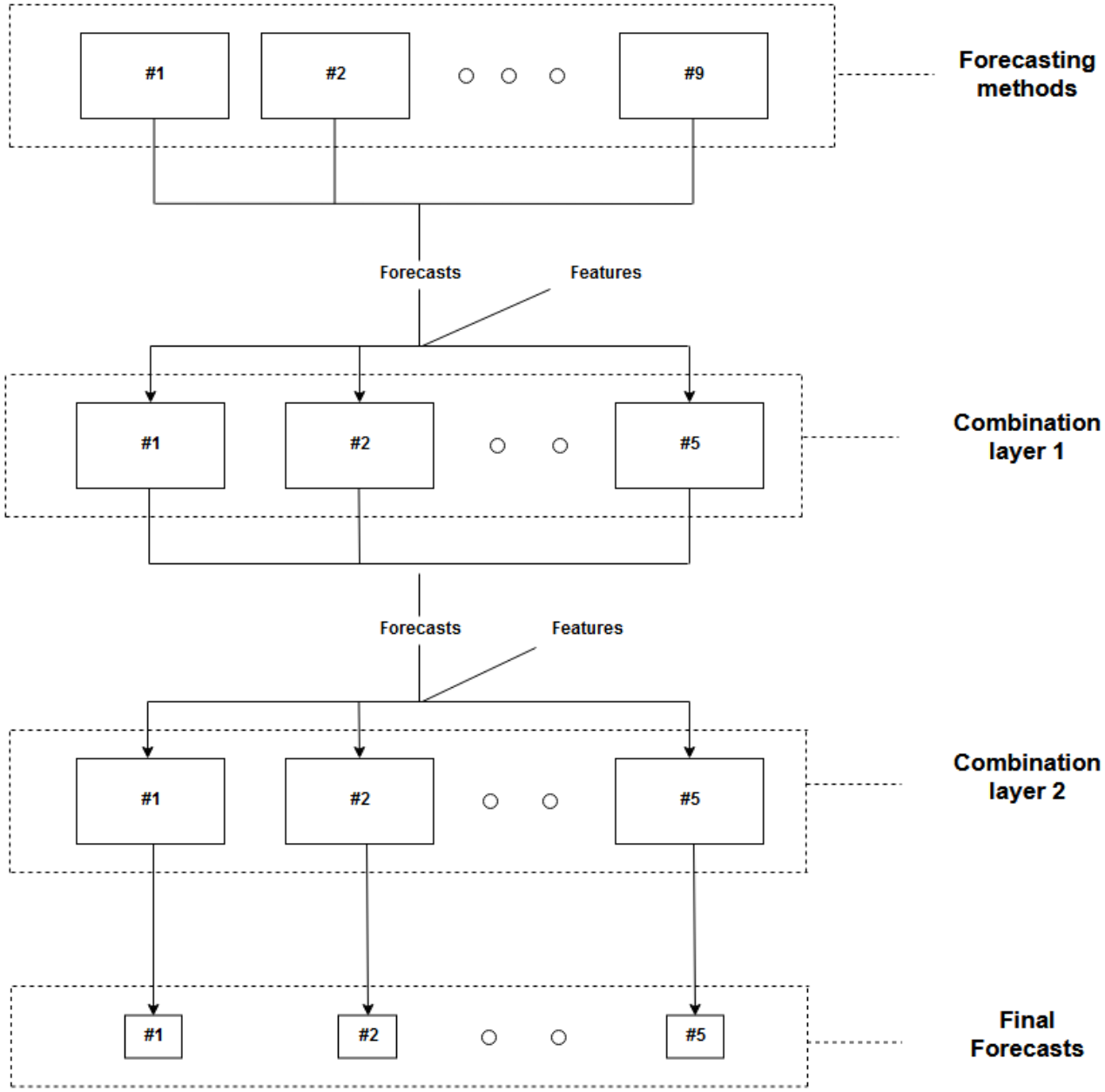


Figure 2: Flowchart of the repeated combination scheme

4 Data

The data set contains time series from the educational, financial and hospitality sectors. The accuracy of specific methods may differ across frequencies in low-frequency data. The proposed methods are therefore investigated for daily, hourly and 15-minute interval data. Examples of time series are the number of burgers sold or the number of visitors in a time period. Co-variates can be used to increase forecasting accuracy, but are not available in the data. All time series are univariate. Through digitization more data is being gathered. However, within the clients providing the data for this paper, the data gathering process remains limited to univariate data. A total of 2504 time series are included in the data set. Because of the nature of the data there are 0 values present in the time series. The minimum length of the train set of a time series is 2 months. Time-series shorter than 2 months will cause some of the more complex forecasting methods to become inaccurate. Due to the COVID-19 pandemic the time series is unpredictable after March 2020. Stores were closed for long stretches at a time. When they did open, closing times were irregular and rules regarding the amount of customers allowed varied from day to day. All time series used therefore only contain pre-Covid data. Also, some 0 values in the time series are caused by closing times. These are removed so that the same amount of values are present for every day and week. For more insight into the number of values used, the mean and quartiles of the number of observations per time series are seen in table 1. The means, especially for hourly data, are high compared to the median. This points towards a skewed distribution in the length of the time series.

Table 1: Length of time series per time interval

	Mean	**1st quartile**	**2nd quartile**	**3rd quartile**	**4th quartile**
Daily	472	74	333	444	577
Hourly	9778	516	4246	5670	7960
15-min interval	35264	1352	12645	22263	42554

The KPSS test is a test to determine if a time series is stationary around a deterministic trend. Stationarity means that the data generating process of a time series does not change over time, and thus neither the statistical properties of the time series. Regarding the FFORMA framework, it is essential for non-stationary time series to recalculate the features of a time series given a new window, since the features vary. The null hypothesis of the KPSS test is that of a time series being stationary against the alternative of the time series containing a unit root. In table the percentages of time series that are stationary is seen (H_0 not rejected). When time intervals are smaller, a smaller percentage of time series are seen to be stationary.

Table 2: Percentage of stationary time series as evaluated by the KPSS test

	$\alpha = 0.05$	$\alpha = 0.01$
Daily	41.1 %	54.9 %
Hourly	28.7 %	40.0 %
15-min interval	6.4 %	10.8 %

With this large pool of time series, the universal applicability of the proposed methods will be tested. The forecasting horizon h will consist of 4 weeks for daily data and 2 weeks for hourly and 15-minute interval data. These horizons are chosen as they are often used for forecasting demand. Although the forecasting horizons are the same across time series, the amount of values within the forecasting horizon is not. The data is retrieved from various sectors which have different opening times. i.e., a hospital has daily data for every day of the week, while some retail stores close on Sunday. This heterogeneity is also seen in hourly and 15 min interval data. The error measure as described in the section 3.1 accommodates this heterogeneity across time-series through a mean operator.

The forecasting accuracy of certain methods is affected by the specific time of year the forecast will be done. For example, retail data for October will be more similar to September than data of January will be to December. To make a fair comparison, the time series will end at various points in the year to account for variability. For both Inverse Weighting and FFORMA an extra validation set is needed for hyperparameter optimization. The time series will therefore be split into four parts: a training set, a validation and extra validation set and a test set as seen in figure 3.

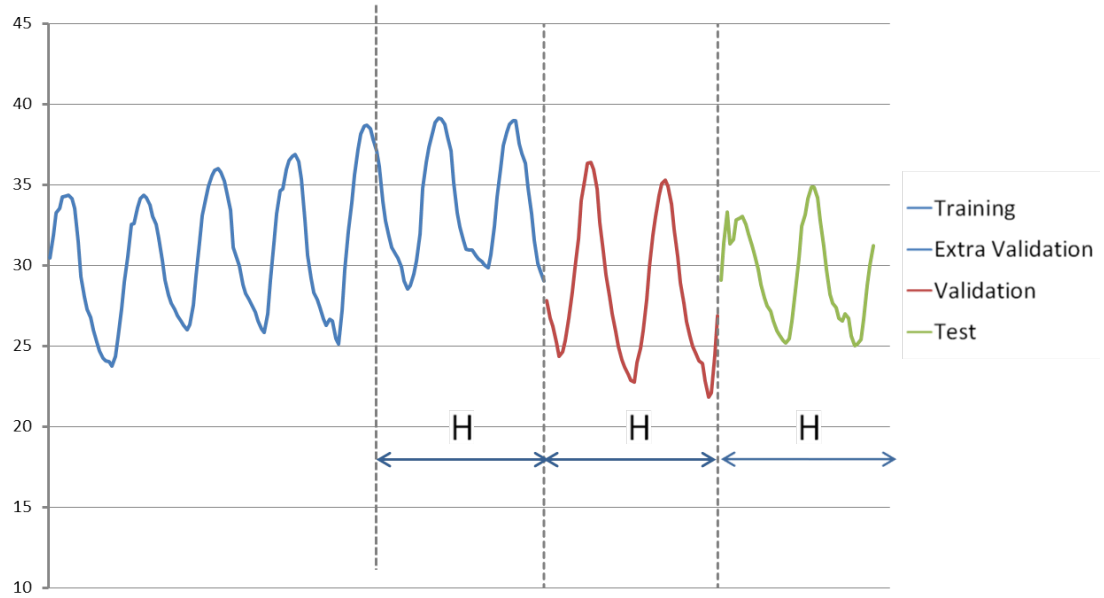


Figure 3: Time-series are split into four parts: train, extra validation, validation and test.

5 Results

The results in this section are based on 2504 time series as described in section 4.

5.1 Hyperparameter selection

The number of forecasting methods to include in the ensembles is optimized based on results in the extra validation set, for a maximum of 9 forecasting methods. Through calculating the MASE of each forecaster in the extra validation set, a ranking of forecasters is made according to their MASE. Then the x amount of forecasters with the lowest MASE are included in the combination method on the test set. In table 3 the MASE score is seen for a varying number of forecasters.

Table 3: MASE in the validation set of a Simple Average, Inverse Weighting and a Median operator

Method \ Amount forecasters included	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Simple Average	0.819	0.807	0.803	**0.801**	0.802	0.806	0.809	0.811
Inverse weighting	0.819	0.807	0.802	0.799	0.799	0.799	**0.798**	0.799
Median operator	0.819	0.808	0.800	0.796	**0.794**	0.796	0.797	0.798

According to the results in table 3, for the Simple Average, Inverse Weighting and Median operator, 5, 8 and 6 forecasters respectively achieve the best MASE. These amounts of forecasters are then used for the test set. Hyperparameter optimization for the LightGBM and XGboost model is done through bayesian optimization.

5.2 Comparison of combination methods

The results for the test set (including the FFORMA model) are seen in table 4. To assess the benefit of combining forecasts, the MASE is also stated when only one forecasting method is taken (Best Forecaster). With the Best Forecaster, the forecaster with the lowest MASE in the validation set is used to forecast for the test set.

Table 4: Mean Errors per Forecasting method

	Best Forecaster	**Simple Average**	**Inverse Weighting**	**Median operator**	**FFORMA**
MASE	0.902	0.830	0.799	0.814	**0.795**
MSE	12170	8428	6544	7046	**5608**
MAPE	55.28	50.77	48.77	49.20	**47.62**
MAE	28.64	24.11	22.32	22.86	**21.14**
R2	0.07	0.24	0.33	0.28	**0.35**

In table 4 it is seen that combining forecasts outperforms selecting the best forecaster. The FFORMA model has the lowest MASE out of all Ensembles. Forecasts are made for three different time intervals, daily, hourly and 15-min interval data. The time horizon for daily data is 4 weeks, for hourly and 15-min interval data the time horizon is 2 weeks. Characteristics differ for these intervals. An extra seasonality is present in hourly and 15-min interval data, the time of day. Further, 15-min interval data is spikier because of its shorter interval. To investigate the performance of Ensembles for forecasting for different time intervals, scores per time interval are seen in table 5.

Table 5: Mean MASE per time interval

Interval	**Length**	**Best Forecaster**	**Simple Average**	**Inverse Weighting**	**Median operator**	**FFORMA**
Day	1244	0.851	0.744	0.683	0.698	**0.658**
Hour	1110	0.928	0.878	**0.859**	0.877	0.864
15-Min	150	1.550	1.418	**1.378**	1.390	1.401

For daily values the FFORMA framework performs best, for hourly and 15-min intervals the inverse weighting performs best. It is seen that, on average, combination methods outperform taking the best single forecaster for all time intervals. The mean MASE increases as the frequency increases. This is expected because for daily forecasts, the seasonal naïve the forecast is equal to the observed value the week before. For hourly and 15-min interval data, the seasonal naïve is equal to the observed value the day before. Therefore, the seasonal naïve has a bigger advantage for sub-daily data than for daily data relative to the forecasts. Compared to the the seasonal naïve forecasts, the combination methods perform better for hourly data than for 15-min interval data. Thus, for 15-min interval data, the in-sample seasonal naive forecasts perform better, the combination forecasts perform worse or both. This could indicate that 15-min interval data contains more variability in the daily pattern from week to week.

The mean of the error measure does not give complete information about the results of the methods. The list of errors for each combination method has a skewed, Poisson like distribution as seen in figure 4. All combination methods produce a similar distribution.

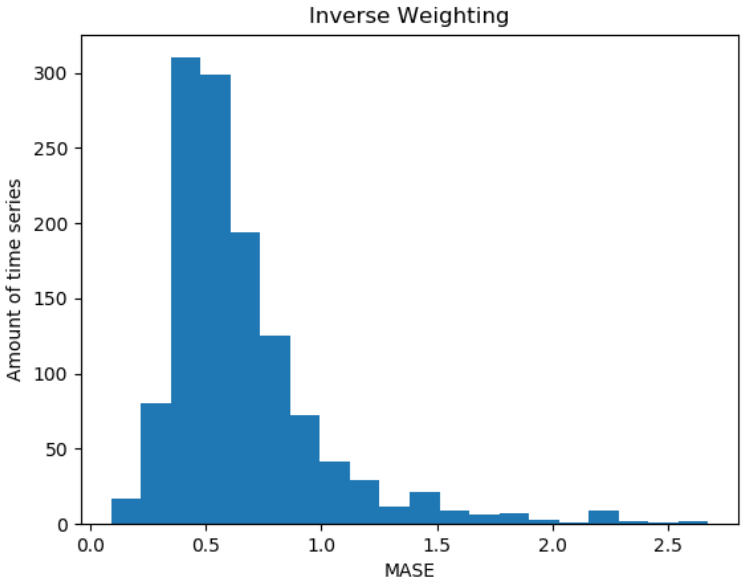
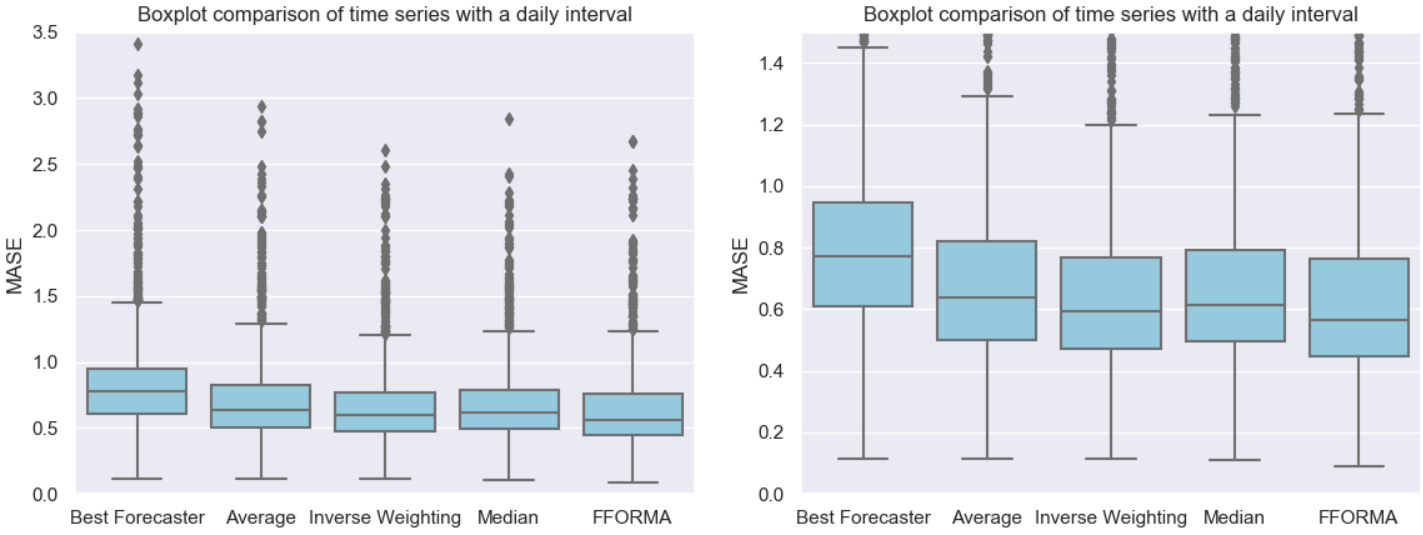


Figure 4: Distribution of the errors for Inverse Weighting.

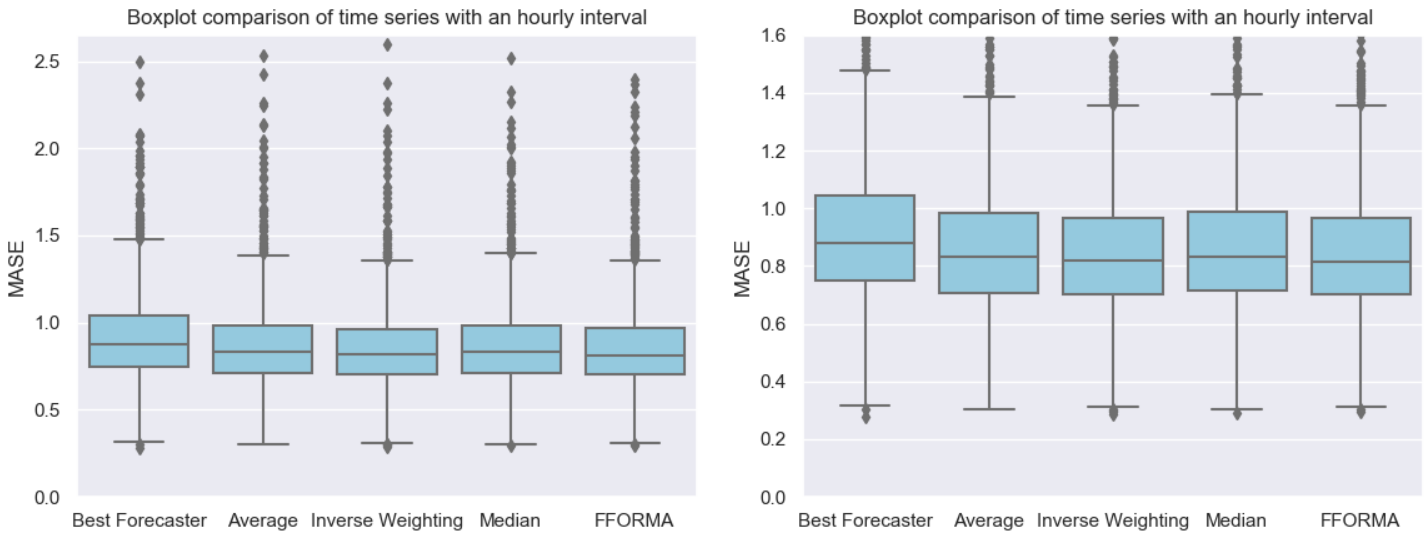
The mean is less informative with a skewed distribution, since a few high errors have a significant impact on the mean. Box-plots are more informative, where the median is reported as well as blocks of approximately 25% of the points. Box-plots for all time intervals are seen in figure 5, 6 and 7. Box-plots for the time intervals combined are seen in the Appendix.



(a) Complete box-plot

(b) Magnified box-plot

Figure 5: Box-plot comparison of methods for daily interval time-series



(a) Complete box-plot

(b) Magnified box-plot

Figure 6: Box-plot comparison of methods for hourly time series

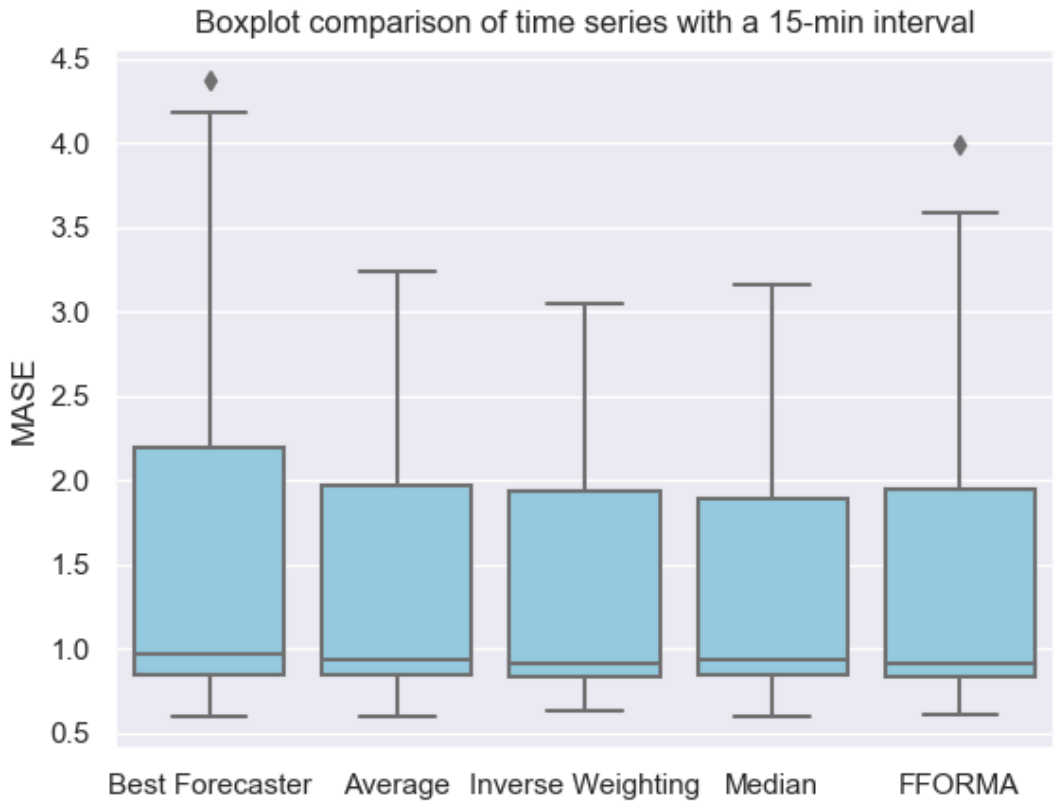


Figure 7: Box-plot comparison of methods for 15-min interval time series

In table 4 it was seen that combining forecasts outperformed taking the best forecaster based on a validation set. The same outcome is seen in figure 5 if the median is evaluated. The second quantile of both FFORMA and Inverse Weighting falls entirely below the second quantile of the non-combination method. The ranking of the combination methods in terms of the median is equal to the ranking according to the mean seen in table 4.

The test statistic for the Friedman test with $K = 5$ and $N = 2504$ is $S = 2933.74$. As stated in section 3.6.1, under the null hypothesis S converges in distribution to a Chi-squared distribution with $K - 1$ degrees of freedom. The critical value for $\alpha = 0.01$ is $\chi^2_4 = 13.277$. Therefore the null hypothesis $H_0 : \tau_1 = \tau_2 = ... = \tau_k$ is rejected, there are significant differences between the methods.

The forecasting methods are not equally accurate. A Multiple Comparison procedure is done to investigate which forecasting methods differ significantly. With $\alpha = 0.05$ and $K = 5$ $q_{\alpha,K} = 3.86$ (See Table 1 (Harter, 1960)) and $N = 2504$, $r_{\alpha,K,N} = 0.42$. For each method k, an interval is drawn with length $r_{\alpha,K,N}$ centered at $\bar{R}_k$ seen in figure 8. When intervals do not overlap, $H_{0,k_1,k_2} : \tau_{k_1} = \tau_{k_2}$ is rejected.

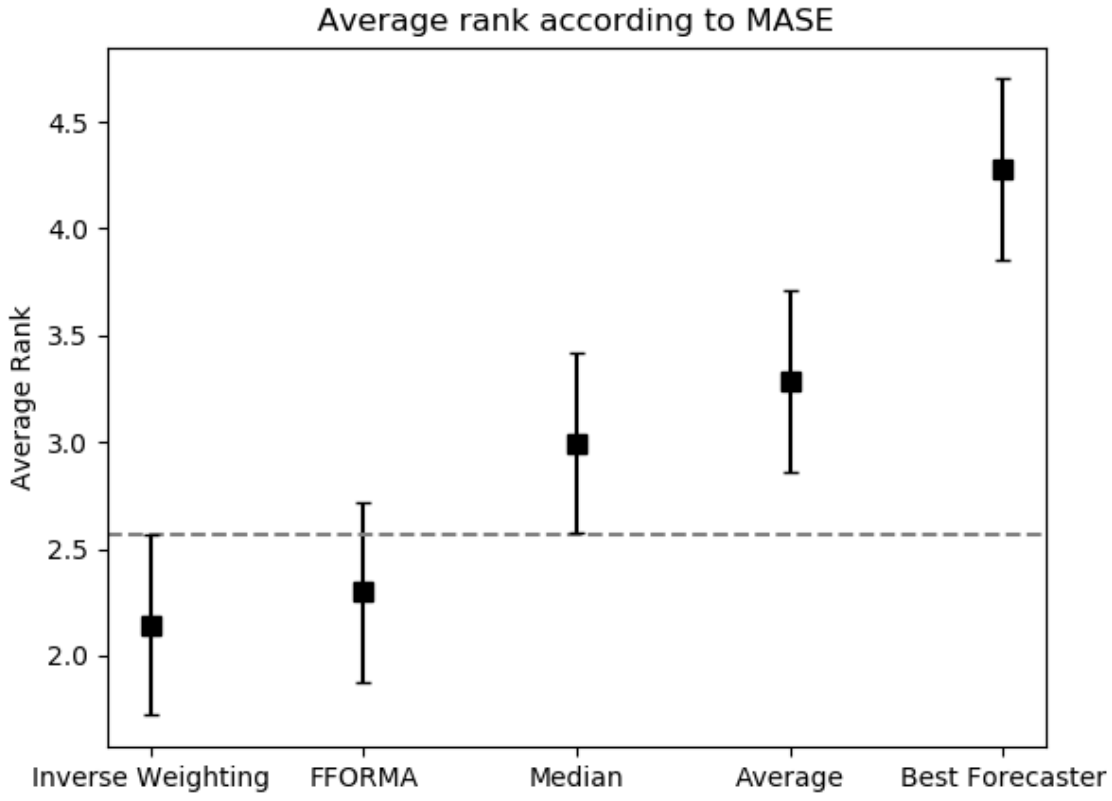


Figure 8: Average rank of 5 methods over 2504 series, MCB intervals.

In figure 8 we see that Inverse Weighting and FFORMA do not significantly differ, $H_{0,k_1,k_2} : \tau_{k_1} = \tau_{k_2}$ is not rejected. The Median, Average and Best Forecaster methods

perform significantly worse than Inverse weighting. For the simple average, this points to the fact that some of the forecasters included are quite inaccurate. The significant difference between the median combination and the Inverse Weighting combination was not seen in low-frequency data (Jaganathan & Prakash, 2020). It is possible that there is uneven distribution of forecasts around the true value. That is, there could be 6 forecasters that regularly over forecast and 2 that under forecast, resulting in a lower forecasting performance.

5.3 FFORMA

5.3.1 Model performance

As stated in section 2, originally XGboost was used as the learner for the FFORMA framework. However, as described in section 3.7, the Light Gradient Boosting Machine framework was used in this paper as the learner in the FFORMA model. To assess the usefulness of LightGBM, both LightGBM and XGBoost were applied in the FFORMA model. A comparison of forecasting errors is seen in table 6.

Table 6: Mean Errors per Forecasting method

	FFORMA LGBM	**FFORMA XGBoost**
MASE	**0.795**	0.797
MSE	**5608**	5727
MAPE	**47.62**	47.71
MAE	**21.14**	21.25
R2	**0.35**	0.34

Using LightGBM in the FFORMA framework provides a lower forecasting error. According to the Multiple Comparison with the best as seen in figure 10 in the appendix, the difference is not significant. The average computation time over different sets of time series of FFORMA with the LightGBM framework is 5.6 seconds, with XGBoost the computation time is 34.3 seconds. LightGBM improves forecasting performance while decreasing computation time by a factor 6.

5.3.2 Features

The light gradient boosting machine framework is used for the FFORMA model. In figure 9, feature importance of LightGBM is seen. The feature importance is represented by the number of times a feature is used in all trees. In the LightGBM framework a feature is used if it is capable of optimizing the objective function. Therefore, features with a higher count are more important for minimizing the sum of forecasting errors.

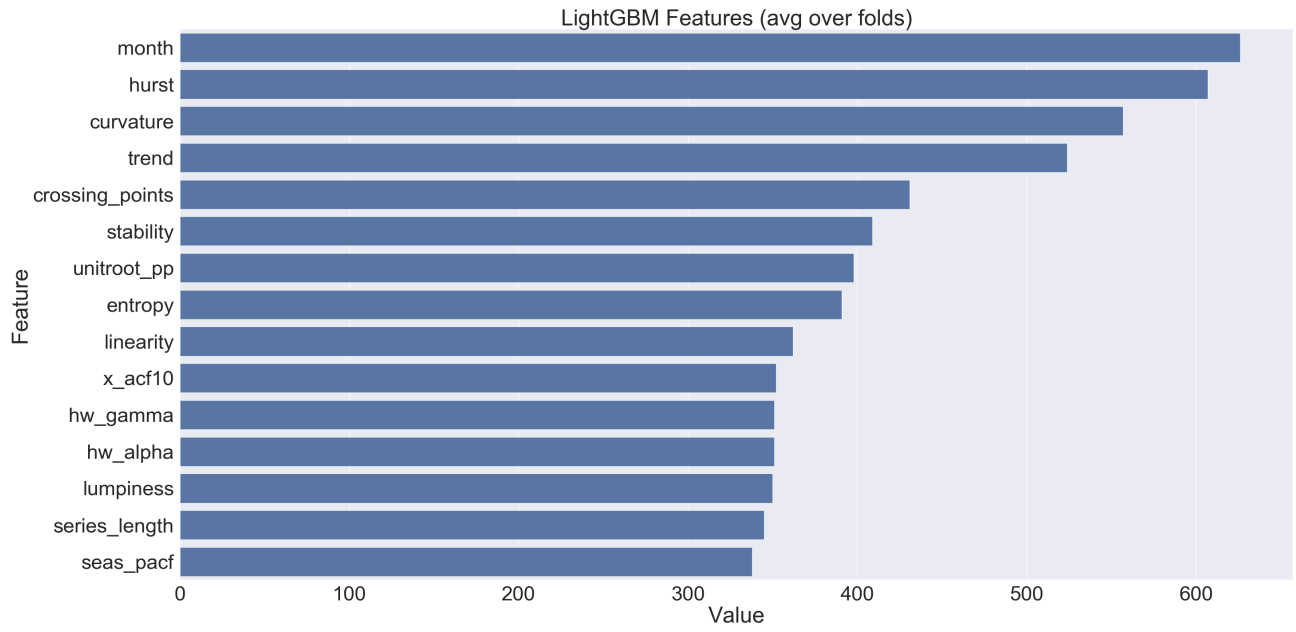


Figure 9: Feature importance of the 15 most important features of the Light Gradient Boosting Machine

The feature that described which month of the year was forecasted for, achieved the highest feature importance. This means that the feature *month* is most important in minimizing the total forecasting error. Each of the forecasting methods captures different characteristics of a time series to make forecasts. The feature *month* is used to relate a time series to specific forecasting methods, capturing yearly seasonality. It is recommended to use week of the year as feature in future use of the FFORMA framework, so that the yearly seasonality can be taken into account more accurately. The feature *hurst* is the hurst exponent which relates to the autocorrelation of a time series. It reports the tendency of a time series to either regress to the mean or to cluster in a direction. The *curvature* of a time series is based on the coefficients of an orthogonal quadratic regression. An orthogonal quadratic regression results in a smoother line compared to a quadratic regression. The *trend* relates to the trend using the STL decomposition.

As additional features the forecasting errors of the 8 individual forecasters in the validation set were introduced as features. With introducing these features, the performance decreased slightly to a MASE of 0.798, they are therefore not used in the final model.

5.3.3 Multi-sector data

To evaluate the value of using data across different sectors in the FFORMA framework, the errors are calculated with both all data and only the data of a specific sector.

Table 7: FFORMA results for data from the Educational sector using both all data and only customer-specific data

	Educational sector		
	All data	**Cust. spec. data**	**amount time-series**
Daily	0.851	0.852	372
Hourly	0.950	0.955	239
15-min	2.212	2.244	63

Table 8: FFORMA results for stores in the Financial sector using both all data and only customer-specific data

	Financial sector		
	All data	**Cust. spec. data**	**amount time-series**
Daily	0.537	0.538	719
Hourly	0.867	0.865	718

Table 9: FFORMA results for hospitality using both all data and only customer-specific data

	Hospitality sector		
	All data	**Cust. spec. data**	**amount time-series**
Daily	0.766	0.762	150
Hourly	0.734	0.728	149
15-min	0.858	0.854	94

In table 4, 5 and 6 it is seen that cross using data has a small positive effect in the educational sector, a varying effect on the financial sector and a negative effect on hospitality sector. It is expected that more data would have a positive effect on the model performance. These results indicate that time series of one sector do not provide valuable information for another sector.

5.4 Repeated combination

The combined forecasts are again used to produce a forecast through two processes. Firstly, a Random Forest is used as a classifier to select the best (combination) forecast.

The forecasters included in the classifier are the Best Forecaster, Simple Average, Inverse Weighting, Median and the FFORMA framework. Secondly, all combination methods discussed are used to create a weighted forecast out of the combined forecasts. For this purpose, the combination forecasts included are Simple Average, Inverse Weighting, Median and the FFORMA framework.

Table 10: Mean Errors per combined forecaster Forecasting method in the second combination process.

	Classifier	**Average**	**FFORMA**	**Inverse Weighting**	**Median**
Mean MASE	0.797	0.810	0.789	0.791	0.799
Standard deviation	0.421	0.402	0.393	0.382	0.401

The classifier performs slightly worse than FFORMA framework in the first combination layer. The Average and Median perform worse than both the FFORMA and Inverse Weighting in the first combination layer. However, both the classifier and the average of combinations don't differ significantly from the FFORMA framework as seen in figure 10 in the Appendix. Inverse Weighting and the FFORMA framework are seen to achieve a higher forecasting accuracy than any of the combination methods out of the first combination layer. The combination methods in the first layer therefore differ sufficiently such that combining them improves performance. The performance of the combination methods in the first layer differs significantly, so some of the forecasts included in the combination are relatively inaccurate. It is therefore expected that a simple average of combination methods is not the best combination method.

6 Conclusion

The empirical finding in the literature is that combining forecasts improves the forecasting accuracy over selecting a single forecasting method (Stock & Watson, 2004), this holds true in high-frequency data. For every time interval and all score measures, all combination schemes as described in section 3 outperform selecting a single forecasting method. The recommendation to combine forecasting methods in low-frequency data therefore also stands for high-frequency data.

The FFORMA framework had the lowest forecasting error on average as measured by the MASE, similar to the results in the M4 competition (Makridakis & Spiliotis, 2018). The mean, median and box plots were reported because of the skewed distribution as showed in figure 4. The FFORMA method has the lowest error in terms of mean and median. Therefore, the FFORMA framework should be used to achieve the lowest overall error. When evaluating the results per time interval, the FFORMA framework proved to be the most accurate for daily data, and the Inverse Weighting for both hourly and 15-min interval data. Our analysis showed that the difference in forecasting accuracy between the FFORMA framework and Inverse Weighting was not significant, similar to results in low-frequency data (Jaganathan & Prakash, 2020). The empirical finding that simple combinations outperform more elaborate, complex combination schemes known as the "forecast combination puzzle" (Stock & Watson, 2004) is not true for high-frequency data, according to the results of the FFORMA framework in this paper. A rolling window approach is recommended because previous studies showed that this improves the accuracy of Inverse Weighting. However, in our analysis it was not used because of the computation cost, as mentioned in section 4.

Different from results in low frequency data is the finding that both the median and simple average combination method had a significantly worse performance than the FFORMA framework and Inverse Weighting. In the FFORMA framework and Inverse Weighting, the weights are related to the validation errors. In the simple average this is not the case, this points to the fact that some forecasting methods perform consistently worse than others and should be weighted accordingly. Our analysis showed a significant difference between the median and the Inverse Weighting combination which was not seen in low-frequency data (Jaganathan & Prakash, 2020). A possible explanation for this is an uneven distribution around the actual values between forecasters who often over forecast and those who under forecast. That is, if 6 forecasters regularly over forecast and 2 under forecast, a median will not perform optimally.

Multiple analyses are performed into extensions of the FFORMA framework. In these analyses, a different learner (LightGBM) is used, new features are introduced and the value of using data from multiple sectors is explored. When LightGBM is used in the FFORMA framework instead of XGBoost, forecasting accuracy increases. This follows

the results in the literature (Ke et al., 2017; Ma et al., 2018; Daoud, 2019) that LightGBM decreases computation time while achieving comparable accuracy. Our analysis showed that LightGBM improves the computation speed by a factor 6 and increases forecasting accuracy slightly. Therefore, it is recommended to use LightGBM in the FFORMA framework. Only LightGBM and XGBoost were considered, as the objective function of these methods can be customized such that the MASE is minimized, which was the objective in this paper. It is however also possible to use other machine learning methods to determine the weights, such as neural networks (Prudêncio & Ludermir, 2006). It would be interesting to investigate whether a high forecasting performance is still achieved when using other machine learning methods which minimize over a different criterion.

Month, hurst, curvature and trend are the features most important in minimizing the objective function of the FFORMA framework, the feature importance is reported in figure 9. The feature *month* is an addition to the literature and was added because of the finding that in practice the characteristics of time series vary from month to month. The finding that the characteristics of time series differ over time is in line with the results from the KPSS test, which showed that more than 60 % of the time series in this paper are non-stationary. For example, December can produce spikier data because of the holidays and promotions than other months of the year. Using this feature can also be seen as a new manner of handling nested seasonality. Nested seasonality is an extra challenge that is present in high-frequency data. For example, hourly data contains daily, weekly and yearly seasonality. Forecasters are usually unable to catch the complex structure of multiple seasonalities. In the FFORMA framework seasonality is handled through the feature *month*, where the month is related to the forecasting methods. To capture yearly seasonality more accurately, the week of the year should be included as feature. Weekly seasonality can be captured similarly, although the model should be altered slightly. Currently in the FFORMA framework, the LightGBM is trained through relating the features to the MASE of the forecasting horizon, which is 2 or 4 weeks. This way, day of the week can not be added as feature. Instead the features should be related to the forecast errors of a single day to incorporate day of the week as feature.

As another extension to the FFORMA model, the ranking of the forecasting errors of the validation set are added as features. Including this extra information, the FFORMA model performed worse. The performance in the training set does improve. This is an indication of over-fitting. The Machine Learning model finds a relation between the errors and the features that is not in fact there. Early stopping is used but has not prevented over-fitting. The ranking of the forecasting errors of the validation set do not have additional predictive value for determining the weights for the test set, and are not recommended to be used in the FFORMA model.

As seen in table 7, 8 and 9, the value in using data from multiple sectors is small, varying between a positive and negative impact on the performance. A reason for this

result could be that the data varies significantly between sectors. It is not possible to learn from the other time series because many of the features are not in the same range across sectors. It would be interesting to divide the sectors further into sub-groups and investigate if this has an impact on the performance. It is possible that including only a specific part of the data is enough to achieve maximum performance. This way, computation time can be reduced. It is recommended to vary the sample size used in the FFORMA framework. It might be beneficial to only train on data similar to the time series we forecast for. The data should be grouped according to its features and only similar data should be included. The required similarity should be varied, such that the relation between the data and performance is made explicit. If the relation is known, it can be leveraged in the data gathering process. A rolling window approach can be used in the FFORMA model to generate more data about similar time series. This could potentially increase accuracy in the FFORMA model.

There is an extensive body of literature on how to combine forecasts. However, we did not find literature on repeating the combination process to increase forecasting performance in the manner proposed in this paper. In this paper an attempt is made to repeat the combination process with the combined forecasts. That is, the forecasts of the combination methods are again used to create a combined forecast. Firstly, a Random Forest is used as a classifier to identify the best Ensemble for a time series given its features. The labels to be classified are the relative ranks of the Ensembles. As seen in figure 10, the performance of the Random Forest comes close to the performance of the FFORMA framework in the first combination layer. Secondly, the FFORMA framework, Inverse Weighting, Median and Average are again used to produce weights for the combined forecasts. Both the Average and Median of the combined forecasts perform worse than the best performing combination methods of the first combination layer. This indicates that some of the forecasts included in the combination are quite inaccurate. Within the combination methods, the Average and Median had the lowest performance already. Taking this into account, an average or median might not be the best way to combine the forecasts for the methods used. Using the FFORMA framework or Inverse Weighting to produce weights for the combined forecasts does increase performance further. It means that the benefit of combining forecasts did not converge after a single combination process. This result can have a great impact on the forecasting community. If the combination methods are used to produce a weighted forecast through Inverse Weighting or the FFORMA framework, the strengths of each combination method is utilized. The search into the best combination method (in the first layer) would become obsolete. We recommend to perform multiple investigations into the approach of repeated combination, since including more combination methods in the first layer and/or second layer could improve the final accuracy further. Firstly, in this paper only combination methods that perform well on a large number of time series were considered. However, if a combination method

performs well only on a small number of time series, the method can be weighted accordingly in the second combination layer. Then, even though a combination method does not perform well on average, it could still add to the final performance. Secondly, only the combination methods that are known to perform well in the first combination layer were considered for the second combination layer. Other combination methods should be investigated, as they might prove valuable in the second combination layer. Finally, it is possible that the forecasting performance did not yet converge, such that an extra combination layer would increase the performance further.

It is not known if this result holds for low-frequency data, and should therefore be investigated. Repeating the combination process might be a practical answer to the long-lasting search of the best combination method: simply use all (well-performing) combination methods. It is also recommended to investigate repeated combination in methods that make use of covariates.

References

Andersen, T. G. (2000). Some Reflections on Analysis of High-Frequency Data. *Journal of Business Economic Statistics*, *18*(2), 146. doi: 10.2307/1392552

Armstrong, J. S. (2001). Principles of forecasting: a handbook for researchers and practitioners. Springer Science and Business Media.

Bates, J. M., & Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Society*, *20*(4), 451-468. doi: http://dx.doi.org/10.2307/3008764

Bordley, R. F. (1982). The Combination of Forecasts: A Bayesian Approach. *The Journal of the Operational Research Society*, *33*(2), 171. doi: 10.2307/2581301

Breiman, L. (1996). Stacked regressions. *Machine Learning*, *24*(1), 49–64. doi: 10.1007/bf00117832

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR*, *abs/1603.02754*. Retrieved from `http://arxiv.org/abs/1603.02754`

Claeskens, M. J. R. V. A. L. . W. W., G. (2016). The forecast combination puzzle: A simple theoretical explanation. *Oxford Bulletin of Economics and Statistics*, *32*(3), 754-762. doi: https://doi.org/10.1016/j.ijforecast.2015.12.005

Clemen, R. T. (1989). Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting*, *5*(4), 559-583. doi: https://doi.org/10.1016/0169-2070(89)90012-5

Daoud, E. A. (2019). Comparison between xgboost, lightgbm and catboost using a home credit dataset. *International Journal of Computer and Information Engineering*, *13*(1), 6 - 10. Retrieved from `https://publications.waset.org/vol/145`

Dharmawardane, C., Sillanpää, V., & Holmström, J. (2021). High-frequency forecasting for grocery point-of-sales: intervention in practice and theoretical implications for operational design. *Operations Management Research*. doi: 10.1007/s12063-020-00176-7

Diebold, F. X., & Lopez, J. A. (1996). 8 forecast evaluation and combination. In *Statistical methods in finance* (Vol. 14, p. 241-268). Elsevier. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0169716196140104` doi: https://doi.org/10.1016/S0169-7161(96)14010-4

Franses, P. (2016). A note on the Mean Absolute Scaled Error. *International Journal of Forecasting*, *32*(1), 20–22. doi: 10.1016/j.ijforecast.2015.03.008

Friedman, M. (1937). The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, *32*(200), 675–701. doi: 10.1080/01621459.1937.10503522

Friedman, M. (1939). A Correction: The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, *34*(205), 109. doi: 10.2307/2279169

Garza, G. K., F. (2020). *tsfeatures: Calculates various features from time series data. python package version 0.2.0.*

Genre, K. G. M. A. . T. A., V. (2013). Combining expert forecasts: Can anything beat the simple average? *International Journal of Forecasting*, *29*(1), 108-121. doi: https://doi.org/10.1016/j.ijforecast.2012.06.004

Hansen, B. E. (2007). Least Squares Model Averaging. *Econometrica*, *75*(4), 1175–1189. doi: 10.1111/j.1468-0262.2007.00785.x

Harter, H. L. (1960). Tables of Range and Studentized Range. *The Annals of Mathematical Statistics*, *31*(4), 1122–1147. doi: 10.1214/aoms/1177705684

Hibon, . E. T., M. (2005). To combine or not to combine: selecting among forecasts and their combinations. *International Journal of Forecasting*, *21*(1), 15-24. doi: https://doi.org/10.1016/j.ijforecast.2004.05.002

Hollander, M., & Wolfe, D. (1999). *Nonparametric Statistical Methods* (2nd Edition ed.). Hoboken, NJ, Verenigde Staten: Wiley.

Holmen, J. S. (1987). A note on the value of combining short-term earnings forecasts. *International Journal of Forecasting*, *3*(2), 239–243. doi: 10.1016/0169-2070(87)90005-7

Hong, T. (2020). Forecasting with high frequency data: M4 competition and beyond. *International Journal of Forecasting*, *36*(1), 191–194. doi: 10.1016/j.ijforecast.2019.03.013

Hsiao, C., & Wan, S. K. (2014). Is there an optimal forecast combination? *Journal of Econometrics*, *178*, 294–309. doi: 10.1016/j.jeconom.2013.11.003

Hyndman, R., & Koehler, A. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. doi: 10.1016/j.ijforecast.2006.03.001

Jaganathan, S., & Prakash, P. (2020). A combination-based forecasting method for the M4-competition. *International Journal of Forecasting*, *36*(1), 98–104. doi: 10.1016/j.ijforecast.2019.03.030

Kapetanios, G., Labhard, V., & Price, S. G. (2005). Forecasting Using Bayesian and Information Theoretic Model Averaging: An Application to UK Inflation. *SSRN Electronic Journal*. doi: 10.2139/ssrn.824726

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Nips.*

Koning, A. J., Franses, P. H., Hibon, M., & Stekler, H. (2005). The M3 competition: Statistical tests of the results. *International Journal of Forecasting*, *21*(3), 397–409. doi: 10.1016/j.ijforecast.2004.10.003

Koop, G., & Potter, S. (2004). Forecasting in dynamic factor models using Bayesian model averaging. *The Econometrics Journal*, *7*(2), 550–565. doi: 10.1111/j.1368-423x.2004.00143.x

Ma, X., Sha, J., Wang, D., Yu, Y., Yang, Q., & Niu, X. (2018). Study on a prediction of P2P network loan default based on the machine learning LightGBM and XGboost algorithms according to different high dimensional data cleaning. *Electronic Commerce Research and Applications*, *31*, 24–39. doi: 10.1016/j.elerap.2018.08.002

Makridakis, S. (1993). Accuracy measures: theoretical and practical concerns. *International Journal of Forecasting*, *9*(4), 527–529. doi: 10.1016/0169-2070(93)90079-3

Makridakis, S., & Hibon, M. (2000). The m3-competition: results, conclusions and implications. *International Journal of Forecasting*, *16*(4), 451–476. doi: 10.1016/s0169-2070(00)00057-1

Makridakis, S., & Spiliotis, E. (2018). The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802-808. doi: https://doi.org/10.1016/j.ijforecast.2018.06.001

McDonald, B. J., & Thompson, W. A. (1967). Rank Sum Multiple Comparisons in One- and Two-Way assifications. *Biometrika*, *54*(3/4), 487. doi: 10.2307/2335040

McDonald, B. J., & Thompson, W. A. (1972). Rank sum multiple comparisons in one- and two-way classifications. *Biometrika*, *59*(3), 699–699. doi: 10.1093/biomet/59.3.699

Montero-Manso, P., Athanasopoulos, G., Hyndman, R., & Talagala, T. (2020). FFORMA: Feature-based forecast model averaging. *International Journal of Forecasting*, *36*(1), 86–92. doi: 10.1016/j.ijforecast.2019.02.011

Naim, I., Mahara, T., & Idrisi, A. R. (2018). Effective Short-Term Forecasting for Daily Time Series with Complex Seasonal Patterns. *Procedia Computer Science*, *132*, 1832–1841. doi: 10.1016/j.procs.2018.05.136

Newbold, P., & Harvey, D. (2002). Forecast combination and encompassing. In M. P. Clements & D. F. Hendry (Eds.), *A companion to economic forecasting* (p. 268-283). Oxford: Blackwell.

Pawlikowski, M., & Chorowska, A. (2020). Weighted ensemble of statistical models. *International Journal of Forecasting*, *36*(1), 93–97. doi: 10.1016/j.ijforecast.2019.03.019

Prudêncio, R., & Ludermir, T. (2006). A machine learning approach to define weights for linear combination of forecasts. In S. D. Kollias, A. Stafylopatis, W. Duch, & E. Oja (Eds.), *Artificial neural networks – icann 2006* (pp. 274–283). Berlin, Heidelberg: Springer Berlin Heidelberg.

Singh, S., Yassine, A., & Benlamri, R. (2020). Internet of energy: Ensemble learning through multilevel stacking for load forecasting. In *2020 ieee intl conf on dependable, autonomic and secure computing, intl conf on pervasive intelligence and computing, intl conf on cloud and big data computing, intl conf on cyber science and technology congress (dasc/picom/cbdcom/cyberscitech)* (p. 658-664). doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00113

Smith, J., & Wallis, K. F. (2009). A simple explanation of the forecast combination puzzle. *Oxford Bulletin of Economics and Statistics*, *71*(3), 331-355. doi: https://doi.org/10.1111/j.1468-0084.2008.00541.x

Smyl, S., & Hua, N. G. (2019). Machine learning methods for GEFCom2017 probabilistic load forecasting. *International Journal of Forecasting*, *35*(4), 1424–1431. doi: 10.1016/j.ijforecast.2019.02.002

Stock, J. H., & Watson, M. W. (2004). Combination forecasts of output growth in a seven-country data set. *Journal of Forecasting*, *23*(6), 405-430. doi: https://doi.org/10.1002/for.928

Talagala, H. R. J. . A. G., T. S. (2018). "meta-learning how to forecast time series". *"Department of Econometrics and Business Statistics"*. Retrieved from `https://www.monash.edu/business/econometrics-and-business-statistics/research/publications/ebs/wp06-2018.pdf`

United Nations Environment Programme. (2021). *Food Waste Index Report 2021* (Tech. Rep.). Retrieved from `https://www.unep.org/resources/report/unep-food-waste-index-report-2021`

Wallis, K. F. (2010). Combining forecasts – forty years later. *Applied Financial Economics*, *21*(1-2), 33–41. doi: 10.1080/09603107.2011.523179

Wan, A. T., Zhang, X., & Zou, G. (2010). Least squares model averaging by Mallows criterion. *Journal of Econometrics*, *156*(2), 277–283. doi: 10.1016/j.jeconom.2009.10.030

Yao, Y., Vehtari, A., Simpson, D., & Gelman, A. (2018). Using Stacking to Average Bayesian Predictive Distributions (with Discussion). *Bayesian Analysis*, *13*(3). doi: 10.1214/17-ba1091

Zhang, D., & Gong, Y. (2020). The Comparison of LightGBM and XGBoost Coupling Factor Analysis and Prediagnosis of Acute Liver Failure. *IEEE Access*, *8*, 220990–221003. doi: 10.1109/access.2020.3042848

A Appendix

A.1 Features

	Feature	Description
1	length	length of time series
2	trend	strength of trend
3	seasonality	strength of seasonality
4	linearity	linearity
5	curvature	curvature
6	spikiness	spikiness
7	e_acf1	first ACF value of remainder series
8	e_acf10	sum of squares of first 10 ACF values of remainder series
9	stability	stability
10	lumpiness	lumpiness
11	entropy	spectral entropy
12	hurst	Hurst exponent
13	nonlinearity	nonlinearity
14	alpha	ETS(A, A, N) $\hat{\alpha}$
15	beta	ETS(A, A, N) $\hat{\beta}$
16	hwalpha	ETS(A, A, A) $\hat{\alpha}$
17	hwbeta	ETS(A, A, A) $\hat{\beta}$
18	hwgamma	ETS(A, A, A) $\hat{\gamma}$
19	ur_pp	test statistic based on Phillips-Perron test
20	ur_kpss	test statistic based on KPSS test
21	x_acf1	first ACF value of the original series
22	diff1_acf1	first ACF value of the differenced series
23	diff2_acf1	first ACF value of the twice-differenced series
24	x_acf10	sum of squares of first 10 ACF values of original series
25	diff1_acf10	sum of squares of first 10 ACF values of the differenced series
26	diff2_acf10	sum of squares of first 10 ACF values of the twice differenced series
27	seas_acf1	autocorrelation coefficent at first seasonal lag
28	diff2x_pacf5	sum of squares of first 5 PACF values of twice-differenced series
29	seas_pacf	partial autocorrelation coefficient at first seasonal lag
30	crossing_point	number of times the time series crosses the median
31	flat_spots	number of flat spots
32	nperiods	number of seasonal periods inthe series
33	seasonal_period	length of seasonal period
34	peak	strength of peak

35	trough	strength of trough
36	arch_acf	sum of squares of the first 12 autocorrelations of z^2
37	garch_acf	sum of squares of the first 12 autocorrelations of r^2
38	arch_r2	R^2 value of an AR model applied to z^2
39	garch_r2	R^2 value of an AR model applied to r^2
40	month	Month of the year
41	time interval	time interval (daily/hourly/15 min interval)

Table 11: Time-series features used as input in the FFORMA model. The features are calculated in python using the *tsfeatures* package (Garza, 2020).

A.2 Tables and figures

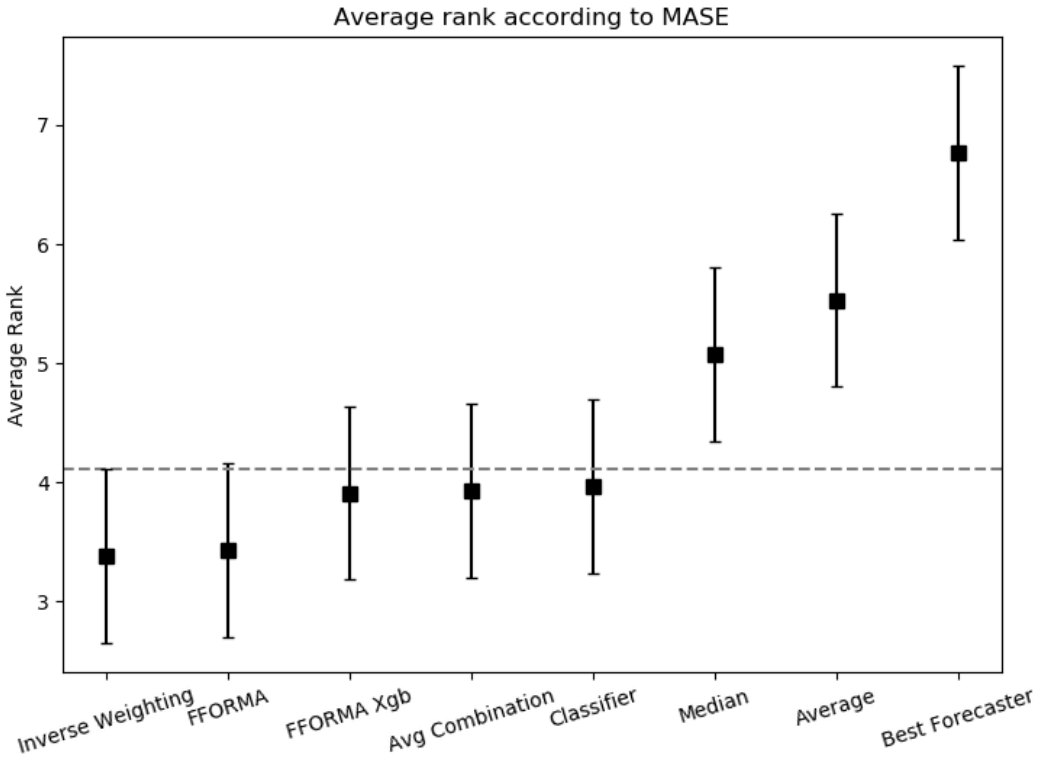
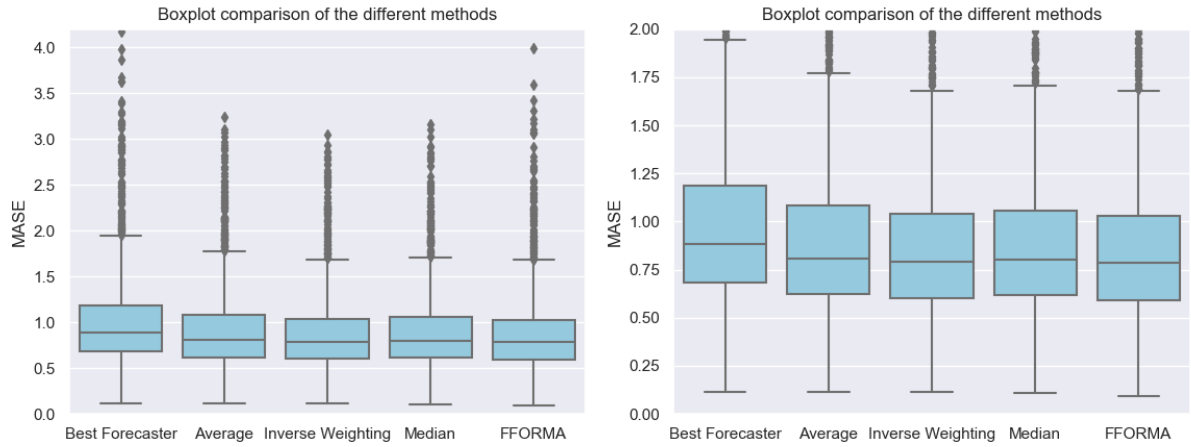


Figure 10: Multiple comparison with the best



(a) Complete box-plot

(b) Magnified box-plot

Figure 11: Box-plot comparison of methods for all time series

A.3 Forecasting methods

- **Gradient boosting regressor** - Forecaster that uses a gradient boosting regressor to determine the expected value. The base learner is a regression tree.
- **Lasso** - Forecaster that uses a linear model with L1 prior as regularizer to determine the expected value.
- **Linear regression (OLS)** - Forecaster that uses a Linear Regression model to determine the expected value based on (lagged) datetime derived features.
- **Xgboost regressor** - Forecaster that uses a XGBoost regressor to determine the expected value based on (lagged) datetime derived features. The base learner is a regression tree.
- **Seasonal Arima** - Seasonal Autoregressive Integrated Moving Average, an ARIMA model where additional seasonal terms are included.
- **Holt Winters** - Forecaster that uses Holt Winters principle to determine the expected value. The optimal smoothing factor α, optimal trend factor β and optimal seasonality factor γ estimated and used to determine the expected value.
- **(Trend) Exponential Smoothing** - Calculates the optimal smoothing factor α, and applies it to determine the expected value.
- **Moving Average** - Applies moving average principle to determine the expected value.
- **Random forest auto regressor** - Forecaster that uses a random forest regressor to determine the expected value based on (lagged) datetime derived features.

A.4 Code

LightGBM: Loss function

```
    def fforma_loss(self, predt, dtrain):
        '''
        Compute the loss of the FFORMA framework
        dtrain represents features, predt the forecast errors
        '''
        y = dtrain.get_label().astype(int)
        n_train = len(y)

        preds = np.reshape(predt,
                           self.contribution_to_error[y, :].shape,
                           order='F')
        #lightgbm uses margins!
        preds_transformed = softmax(preds, axis=1)
        weighted_avg_loss_func = (preds_transformed*self.
            contribution_to_error[y, :]).sum(axis=1)
        fforma_loss = weighted_avg_loss_func.mean()

        return 'FFORMA-loss', fforma_loss, False
```

LightGBM: Gradient and Hessian of the Loss function

```
    def fforma_objective(self, predt, dtrain):
        '''
        Compute value of objective function.
        dtrain represents features, predt the forecast errors
        '''
        y = dtrain.get_label().astype(int)
        n_train = len(y)
        preds = np.reshape(predt,
                           self.contribution_to_error[y, :].shape,
                           order='F')

        preds_transformed = softmax(preds, axis=1)

        weighted_avg_loss_func = (preds_transformed*self.
            contribution_to_error[y, :]).sum(axis=1).reshape((n_train, 1)
            )

        grad = preds_transformed*(self.contribution_to_error[y, :] -
            weighted_avg_loss_func)
        hess = self.contribution_to_error[y,:]*preds_transformed*(1.0-
            preds_transformed) - grad*preds_transformed
        print(len(grad.flatten('F')), len(hess.flatten('F')))
        return grad.flatten('F'), hess.flatten('F')
```

LightGBM: Train LightGBM using Cross-Validation

```
    def _train_lightgbm_cv(holdout_feats, best_models,
                           params, fobj, feval,
                           early_stopping_rounds,
                           verbose_eval, seed,
                           folds, train_model=True):
        params = copy.deepcopy(params)
        num_round = int(params.pop('n_estimators', 100))

        params['num_class'] = len(np.unique(best_models))


        indices = np.arange(holdout_feats.shape[0])
        dtrain = lgb.Dataset(data=holdout_feats, label=indices)

        gbm_model = lgb.cv(
            params=params,
            train_set=dtrain,
            fobj=fobj,
            num_boost_round=num_round,
            feval=feval,
            early_stopping_rounds=early_stopping_rounds,
            verbose_eval=verbose_eval,
            folds=folds(holdout_feats, best_models)
        )

        optimal_rounds = len(gbm_model[list(gbm_model.keys())[0]])
        best_performance = gbm_model[list(gbm_model.keys())[0]][-1]
        params['n_estimators'] = optimal_rounds

        optimal_gbm_model = _train_lightgbm(holdout_feats, best_models,
                                            params, fobj, feval,
                                            early_stopping_rounds,
                                            verbose_eval, seed)


        plotImp(model=optimal_gbm_model, X=holdout_feats)
        return optimal_gbm_model
```

Optimize hyperparameters LightGBM

```
    import lightgbm as lgb
    from bayes_opt import BayesianOptimization

    lgbBO = BayesianOptimization(fforma_loss, params, random_state=0)
```

```
lgbBO.maximize(init_points=init_round, n_iter=opt_round)
lgbBO.res['max']['max_params']
```