A comparison of AdaBoost algorithms for time series forecast combination

Journal:	*International Journal of Forecasting*
Manuscript ID	INTFOR_1504178.R1
Manuscript Type:	Original Article
Keyword:	Forecasting, Time series, Boosting, Ensemble, Model combination, Neural networks
Abstract:	Recently, combination algorithms from machine learning classification have been extended to time series regression, most notably seven variants of the popular AdaBoost algorithm. Despite their theoretical promise their empirical accuracy in forecasting has not yet been assessed, either against each other or against any established approaches of forecast combination, model selection, or statistical benchmark algorithms. Also, none of the algorithms have been assessed on a representative set of empirical data, using only few synthetic time series. We remedy this omission by conducting a rigorous empirical evaluation using a representative set of 111 industry time series and a valid and reliable experimental design. We develop a full-factorial design over derived Boosting meta-parameters, creating 42 novel Boosting variants, and create a further 47 novel Boosting variants using research insights from forecast combination. Experiments show that only few Boosting meta-parameters increase accuracy, while meta-parameters derived from forecast combination research outperform others.

1. Introduction

In the 45 years since the seminal work on combining forecasts (Bates and Granger 1969), research on the combination of time series predictions has consistently been shown to improve accuracy over selecting a single 'best' forecast (Bunn 1988; Elliott, Granger, and Timmermann 2006). Extensive research has assessed how to best determine combination weights of multiple algorithms, e.g. applying forecast error variance minimization (Newbold and Granger 1974; Min and Zellner 1993), regression (Granger and Ramanathan 1984; Macdonald and Marsh 1994), or Bayesian probability theory (Bunn 1975; Bordley 1982; Diebold and Pauly 1990), with the simple arithmetic mean regularly proving a tough benchmark for more sophisticated schemes (Elliott, Granger, and Timmermann 2006). The empirical performance of forecast combination has been substantiated in various experiments (see e.g. Aksu and Gunter 1992; Gunter 1992; Stock and Watson 2004; Clements and Hendry 2007; Jose and Winkler 2008), in prominent empirical case studies in operational research (see e.g. Chan, Kingsman, and Wong 1999) and finance (Zhang 2007; Leung, Daouk, and Chen 2001), and in representative forecasting competitions such as the M3 (Makridakis and Hibon 2000) and NN3 (Crone, Hibon and Nikolopoulos, 2011), providing guidelines on forecast combination based on empirical evidence (de Menezes, Bunn, and Taylor 2000; Riedel and Gabrys 2009).

More recently, research on combining in machine learning has extended the popular classification algorithm AdaBoost by Freund and Schapire (1997) to regression on time series data, typically altering one or more of its components to create a set of seven distinct Boosting algorithms for forecasting (e.g., Shrestha and Solomatine 2006; Assaad, Bone, and Cardot 2008; de Souza et al. 2010) described in more detail in Section 3.8. Boosting algorithms adaptively perturb, reweight and resample training data to create diverse models of the same algorithm, iteratively focusing on harder to learn examples, thereby achieving diverse predictions which are combined in a dynamic and adaptive manner (Freund, Schapire, and Abe 1999). As Boosting combinations differ substantially from traditional forecast combination approaches, which

combine predictions from different algorithms all estimated on the same data, they promise a novel approach to improve accuracy from forecast combinations. However, while Boosting is widely accepted as a benchmark in cross-sectional predictive classification based on its preeminent performance in a number of empirical evaluations (Breiman 1998; Rokach 2009), the empirical accuracy of its variants in time series regression has not been systematically assessed. Although each of the seven Boosting variants by different authors proposes a different alteration of the original AdaBoost algorithm for Regression, their relative accuracy has not been compared against each other. Furthermore, none of the Boosting variants have been compared to the established approaches of conventional forecast combination, individual model selection, or aggregate selection of employing a simple statistical benchmarks such as the Naïve, Exponential Smoothing or ARIMA, making an assessment of their relative merit impossible. Furthermore, none of the algorithms have been assessed on a representative dataset of empirical data, typically evaluating only 1-3 synthetic time series which do not allow a generalisation of their reliability on empirical industry data. In the absence of an objective assessment of their empirical accuracy, the efficacy of this novel approach to forecast combination for applications in Operational Research remains unclear.

This paper seeks to remedy this omission by providing a rigorous empirical evaluation on empirical industry data, providing a three-fold contribution: (I) We provide the first empirical assessment of the seven existing AdaBoost variants, using a representative set of 111 empirical time series from the benchmark dataset of the NN3 competition, and a valid and reliable experimental design assessing multiple-step ahead forecasts across multiple rolling origins. (II) In order to identify the relative merit of each of the seven algorithms' innovations we decompose Boosting into six archetypical meta-parameters that determine the behaviour of all AdaBoost variants (the loss function and type, the loss update method, the stopping criteria, the combination method and the base model) and develop a full factorial design of all meta-parameter alterations, thereby creating 42 novel Boosting variants not previously considered, to

guide future choices and studies on meta-parameters. From the decomposition we note that all Boosting variants have ignored important insights from forecasting research, including the use of simple averages in combining predictions, avoiding higher-order loss functions, and the bias of MAPE based relative error estimation, which could potentially limit their adequacy and empirical performance. Consequently, (III) we create a further 47 novel Boosting variants with meta-parameters motivated from research in traditional forecast combination. In evaluating all against a set of representative benchmark algorithms, we identify a novel Best Combination (BC) boosting algorithm based on new meta-parameters which improves accuracy significantly.

The remainder of the paper is organised as follows: we begin with a structured literature review on Boosting for forecast combination identifying the research gaps and further motivating this study. Next, in Section 3 we describe and decompose the AdaBoost algorithm into its archetypical meta-parameters in order to understand and evaluate the resulting Boosting variants, and to derive properties of a Best Combination Boosting approach. Section 4 describes the experimental design and implementation of the empirical evaluation, including benchmark methods and specification of base models. This is followed by results on statistical significance of all meta-parameter combinations in Section 5, the relative forecast accuracy in comparison to other contenders of the NN3 competition in Section 6, and conclusions in Section 7.

2. A review of AdaBoost for forecast combination

In this study we consider the AdaBoost algorithm, the most well-known and widely applied Boosting algorithm in research and practice (Schapire 2003)[1]. Despite offering a distinct approach to combining predictions, and the success of this approach in predictive classification, only few studies in machine learning have extended AdaBoost to regression on time series data, and with almost no resonance in forecasting and econometrics domains. The first application of

[1] It should be noted that other boosting approaches exist based on the work of Friedman et al. (2000) and Friedman (2001) who linked Boosting to functional gradient descent optimization and stagewise additive function approximation. These approaches have had some success in time series forecasting e.g. the gradient boosting approach of Taieb and Hyndman (2014) which ranked 5th out of 105 participating teams in the Kaggle load forecasting competition. Other areas of application include finance (Audrino 2006; Audrino and Bühlmann 2009), economics (Wohlrabe and Buchen 2014) and production (Robinzonov, Tutz, and Hothorn 2012).

Boosting for regression on time series data is credited to Avnimelech and Intrator (1999), who applied a modification of the original AdaBoost algorithm in forecasting the laser data from the Santa Fe time series competition (Weigend and Gershenfeld 1993). Shrestha and Solomatine (2006) later developed the AdaBoost.RT algorithm also used to predict the Santa Fe time series data. Since then the limited research to date has focussed on what can only be described as marginal extensions of AdaBoost, typically altering one or more of the algorithm's components to create a set of Boosting variants for forecasting. These include a modification of AdaBoost by Goh, Lim, and Peh (2003) who predict drug dissolution profiles for developing drug dosage regimens, an application of Drucker's AdaBoost.R2 algorithm by Canestrelli et al. (2007) forecasting tide levels, and an application of AdaBoost.R2 by Assaad, Bone, and Cardot (2008) inaccurately describing it as a new algorithm in applying a recurrent neural network learner.

With all aforementioned research published in the machine learning literature, with a focus on neural network journals, many of these extensions have ignored important insight from forecasting research within the management sciences. For example, Adaboost.R2 (Drucker 1997), Modified AdaBoost (Goh, Lim, and Peh 2003), AdaBoost.RT (Shrestha and Solomatine 2006) and the Boosting algorithm of Assaad, Bone, and Cardot (2008), all developed exclusively for time series data, each apply a weighted median or mean to combine final predictions, although current research in forecasting concurs in the use of simple averages (see e.g. Winkler and Clemen 1992). Similarly, from early algorithms of Adaboost.R2 to more recent developments of Boosting (Shrestha and Solomatine 2006; Assaad, Bone, and Cardot 2008), many suggest the use of a squared or exponential loss function in guiding the Boosting algorithm on harder to learn examples, despite higher-order errors proven to bias results in forecasting research (see e.g. Tashman 2000). As a remedy Shrestha and Solomatine (2006) suggest AdaBoost.RT using a linear relative error similar in nature to Mean Absolute Percent Error (MAPE), despite MAPE's documented biases in asymmetry and problems with zero observations, and on low values (Hyndman and Koehler 2006). Without interaction with the

forecasting literature, Boosting variants have ignored proven best practices from forecasting research, such as the simple mean for combination, an absolute error loss function and a corresponding mean absolute error (MAE).

At the same time, the different Boosting variants have not been compared against each other, neither conceptually nor in an empirical evaluation of their predictive accuracy, which does not allow an assessment of their individual merits for different data conditions. For those few studies that attempt an empirical evaluation, the experimental designs are often found to be lacking in scientific rigor (see, e.g., the recommendations by Tashman, 2000): all studies forecast altogether less than five time series, assess accuracy only from a single time origin, use biased error metrics, and provide no assessment on the statistical significance of findings. More importantly, the performance of Boosting on time series is not compared to other competing algorithms. Only select studies compare to another machine learning approach (usually Bagging, i.e. the study by Avnimelech and Intrator (1999), Deng, Jin, and Zhong (2005) or Shrestha and Solomantine (2006)), but none compare the performance of Boosting to statistical benchmarks of forecasting combinations, simple model selection, or even basic statistical forecasting algorithms, which are those regularly employed in forecasting practice. Reflecting on the relevance of an empirical evaluation to a methods's efficacy, this signifies a significant gap in research which our study seeks to remedy.

To facilitate a comprehensive understanding of AdaBoost across disciplines, as well as allow a replication of the extensions and experiments, the algorithm is formally described in section 3, where it is decomposed into its archetypical components and meta-parameters.

3. Meta-parameters for Boosting

3.1 A generic AdaBoost algorithm

The AdaBoost algorithm is described below in general, with a detailed discussion of each meta-parameter choice in the following subsections.

Initially, from a time series, a dataset $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ is created consisting of

N pairs of observations $(\mathbf{x}_i, y_i)$, for $i = 1, \ldots, N$, where $\mathbf{x}_i$ is a vector comprised of lagged, autoregressive (AR) realisations $x_t, x_{t-1}, \ldots, x_{t-p+1}$ up to a lag length p to capture an AR(p)-process, and y_i is a future realisation of the same variable of interest, $y_i = x_{t+1}$. In the first iteration $k = 1$, each observation $(\mathbf{x}_i, y_i)$ in S is assigned weight $w_i = 1$, for all $i = 1, \ldots, N$, creating a uniform distribution. Based on its weight, each observation i is assigned a probability $p_i^k = w_i^k / \sum w_i^k$ of being included in the training set on iteration k. Each iteration k of AdaBoost then constructs a forecast model $f_k: X \rightarrow Y$ using a base-learner (i.e. a predictive algorithm, see Section 3.7) capable of learning such mapping, and calculates the loss suffered on predicting each observation L_i^k, with $L_i^k \in [0, 1]$, according to a choice of loss function (see Section 3.2) specifying the relative accuracy with which each observation i has been predicted.

The average loss $\bar{L}_k$ for model k over all observations i is then calculated as the weighted sum of the probabilities and their respective losses (see Section 3.1):

$$L_k = \sum_{i=1}^{N} L_i^k p_i^k \tag{1}$$

Based on this average model loss, a measure of predictive accuracy β_k of the resulting forecast model is calculated. How this is calculated differs across Boosting variants, and is related to the stopping criteria (see Section 3.3). Next, the weights of observation i for the following iteration $k + 1$ are updated depending on its prior weight w_i^k, its current observation loss L_i^k, and the model loss β_k:

$$w_i^{k+1} = w_i^k \beta_k^{\left(1 - L_i^k\right)} \tag{2}$$

A large β_k gives more weight to poorly predicted observations, forcing the underlying learning algorithm to focus on poorly predicted observations. Finally, the predictions $f_k(\mathbf{x}_i)$ of the k models are combined using weights based on β_k (see Section 3.4).

Consequently, we identify six meta-parameters that determine the behaviour of all Boosting variants (Rokach 2009; Bühlmann and Yu 2010): the loss function and type, the loss update method, the stopping criteria, the combination method and the base model. We describe each meta-parameter in a unified notation, review existing literature to highlight the impact and interaction of these meta-parameters, and contrast their choices with research findings on best practice from the forecast combination literature to suggest improved meta-parameters

3.2 Loss function

The loss function controls the magnitude of the relative error attributed to a predicted observation, and thereby the weight update of that observation for the next iteration k+1. We first consider the threshold-based loss function of Shrestha and Solomatine (2006) based on the absolute relative error (*ARE*):

$$ARE_i^k = \left|\frac{f_k(\mathbf{x}_i)-y_i}{y_i}\right|. \tag{3}$$

If the relative error of an observation is above a preset error threshold ϕ, the forecast is said to be poor; conversely, it is classed as a good forecast. The resulting loss L_i^k for observation i is then given by the threshold function:

$$L_i^k = \begin{cases} 1, & ARE_i^k > \phi \\ 0, & otherwise \end{cases} \tag{4}$$

with two classes of errors for poor or good forecasts respectively. In contrast, the proposal of AdaBoost.R2 by Drucker (1997) utilises the absolute percentage error (APE) as follows:

$$APE_i^k = \frac{|f_k(\mathbf{x}_i) - y_i|}{D} \tag{5}$$

where $D_k = sup_i(|f_k(\mathbf{x}_i) - y_i|)$ is the maximal prediction error over all y_i such that $APE_i^k \in [0, 1]$. Based on APE the loss L_i^k for observation i is given by the non-threshold function:

$$L_i^k = \begin{cases} APE_i^k, & f_k(\mathbf{x}_i) \neq y_i \\ 0, & f_k(\mathbf{x}_i) = y_i \end{cases} \quad (6)$$

The impact of loss functions on time series forecast accuracy, and interactions with other meta-parameters has been largely ignored, with the only evaluation of the linear, squared and exponential loss of AdaBoost.R2 on time series data by Assaad et al. (2008). Considering the loss function type, the threshold loss function in Eq. (4) was only evaluated against a non-threshold loss function by Shrestha and Solomatine (2006), without further investigation in other studies. In this study we extend the functional form of the threshold and non-threshold loss functions in Eq. (4) and Eq. (6) to a number of possible loss functions based on Minkowski-r-metrics for linear, squared and higher order exponential loss functions for *APE* (non-threshold) and *ARE* (threshold based). This facilitates the comparison of different loss functions, e.g. the squared *APE* loss used in Drucker (1997),

$$APE_i^k = \frac{|f_k(\mathbf{x}_i) - y_i|^2}{D^2} \quad (7)$$

to a threshold squared *ARE* loss function not previously assessed for its performance:

$$ARE_i^k = \frac{(f_k(\mathbf{x}_i) - y_i)^2}{{y_i}^2}. \quad (8)$$

From a forecasting perspective, the formulation and choice of loss functions seems to ignore established research findings in time series prediction. Relative errors as suggested in AdaBoost.RT are similar in nature to MAPE, despite its documented biases in asymmetry of over- versus underforecasting, and issues with zero values (see, e.g., the discussion ensuing the M3 competition, e.g., in Makridakis et al. 1993). The potential problems are notable with observations close to zero, where small absolute errors will lead to large yet irrelevant percentage errors, misguiding the attention of Boosting unto small valued observations.

Similarly, the suggestion of a squared or even higher-order error loss ignores proven biases of such error metrics in forecasting research (see e.g., Tashman 2000), which would

emphasise training on outliers with higher squared error contribution. The formulation of the non-threshold loss function might further induce problems for non-stationary time series, where errors from higher levels of the series are compared to those of lower levels. On the other hand, the benefit of downscaling metric errors to binary classification errors using a threshold based logic loses a lot of information contained in the error distances of how far easy and hard to predict instances are separated. As a result of these questionable formulations, a systematic evaluation is missing to assess loss functions in Boosting, i.e. linear, squared and exponential, and the loss function type, threshold against non-threshold, on their relative merit in forecasting.

3.3 Stopping Criteria

The variable β_k measures how confident we are in the predictions produced by model k and is given by (see, e.g., in AdaBoost.R2 by Drucker 1997):

$$\beta_k = \log \frac{1 - \bar{L}_k}{\bar{L}_k} \tag{9}$$

which requires an average loss $\bar{L}_k \leq 0.5$. For all $\bar{L}_k > 0.5$ the β_k turns negative, the weights w_i^k turn negative (see Eq. (2)) and the algorithm stops. In this case we describe the meta-parameter as being bounded, as used by most Boosting variants, e.g. AdaBoost.R2 (Drucker, 1997), Threshold-based Boosting (Avnimelech and Intrator, 1999), and Modified AdaBoost (Goh, Lim, and Peh 2003).

In contrast, AdaBoost.RT does not inherently impose a classification based "natural" stopping rule (when $\bar{L}_k$ exceeds 0.5), with β_k for model k calculated as follows:

$$\beta_k = \log \frac{1}{\bar{L}_k^n} \quad , \tag{10}$$

whereby n is the power coefficient, and $n = 1$ (linear $\bar{L}_k$), $n = 2$ (squared $\bar{L}_k^2$) or $n = 3$ (cubic $\bar{L}_k^3$). Based on this definition, it is possible to improve performance even further by creating a user-defined, theoretically infinite number of models, as was also shown by the BCC

Algorithm (de Souza et al., 2006). We refer to this as an unbounded stopping criterion. Shrestha and Solomatine (2006) observed that even when including models having $L > 0.5$, the combined prediction could still outperform the best individual prediction. Additionally, Quinlan (1996) showed empirically that in some cases, while it is possible to obtain better performance on training data with fewer Boosting iterations and $\bar{L}_k \leq 0.5$, the generalisation performance of AdaBoost was poorer than when more models were included.

As before, interpreting the loss bound stopping criteria from a forecasting perspective, a minimum average loss seems nonsensical. Note that in time series forecasting no theoretical definition of a 'weak' learner exists (which achieves a binary classification accuracy marginally larger than 50% on a balanced dataset), yet still the performance requirement for base models in Boosting for Regression relates to a stopping criterion taken from classification. In forecasting however, for interval scaled observations a measure of performance does not have a natural upper bound, as the performance of the predictive algorithm will be bound by the signal-to-noise ratio inherent in the data. Consequently we assess the choice of bounding average loss as the stopping criteria meta-parameter and evaluate two stopping criteria; training stops when $\bar{L}_k \leq 0.5$ (bounded) or training continues beyond $\bar{L}_k > 0.5$ but less than 1 (unbounded), as would be expected in a regression context.

3.4 Combination Method

The final model step combines the forecasts of the individual models using either a weighting method or a meta-combination method. The most widely used combination methods are the weighted median and the weighted average. AdaBoost.R2 and the Boosting algorithm of (Assaad, Bone, and Cardot 2008) combine the final forecast f_z using a weighted median of all models' predictions f_k:

$$f_c(\mathbf{x}) = inf\left\{y: \sum_{k:f_k(\mathbf{x})\leq y} \beta_k \geq \frac{1}{2}\sum_k \beta_k\right\} \quad (11)$$

In contrast, the final model combination in AdaBoost.RT (Shrestha and Solomantine, 2006), BCC Algorithm (de Souza et al., 2006) and Modified Adaboost (Goh et al., 2003) is estimated using the weighted mean:

$$f_c(\mathbf{x}) = \sum_k \beta_k f_k(\mathbf{x}) / \sum_k \beta_k \quad (12)$$

Similar to forecasting research, substantial research in machine learning has been invested to establish a combination method in Boosting, yet no consensus has been achieved. For example, Assaad et al. (2008) found that the weighted median combination gives better predictive performance than the weighted average. Bone et al. (2003) and Deng et al. (2005) both conclude that the weighted median is better than the weighted mean because it is less sensitive to outliers, while Avnimelech and Intrator (1999) find no statistically significant difference between the two. Surprisingly, Boosting research on time series has neglected simpler approaches of equally weighted combinations (Kourentzes et al. 2014), with only threshold-based Boosting (Avnimelech and Intrator, 1999) using a simple median for combination. The omission is deemed of particular importance, as the simple arithmetic mean has proven to be the most widely applied method in Forecasting making it a sensible benchmark. We propose to assess both the weighted and equal-weighted mean and medians for combination, across all other meta-parameters.

3.5 Loss calculation

In Boosting, the termination of training depends on the performance of the current model k being constructed, and not on the performance of the combined predictions. It is however possible that the training error of the combined model drops to zero before other termination conditions are reached, or before the training error of any single model drops to zero. It would

be reasonable to assume that further iterations in such situations will increase complexity, but not improve performance on the training set data. In such a situation it may be wise to terminate training, as we would already have the simplest and best in-sample combination of models. In this study, we propose a novel approach to calculate the loss at every iteration using the combined model output up to that iteration. We refer to this as the ensemble loss calculation, calculating the loss L_i^k of the combined model output at each iteration k, rather than the single model output created on that iteration (model loss calculation), and consider it as another meta-parameter to be assessed across all others.

3.6 Combination size

While the ensemble size has not been explicitly considered in past Boosting studies, non-threshold based estimation such as AdaBoost.R2 uses an unbounded stopping criterion, allowing the addition of many models for arbitrary ensemble sizes. As the size of an ensemble in forecasting as in machine learning is often considered to be of significance, we consider it another implicit meta-parameter and evaluate two competing methodologies for determining combination size.

The first is cross validation early stopping. Under this scheme, each time series is split into three disjoint datasets, a training, validation and test set. If the addition of more models to the combination increases the MSE on the validation set training stops; this is a type of early stopping to prevent overfitting. The second scheme uses a pre-specified number of 50 models to be included in the final combination, without the use of a validation set. Assessing final accuracy on a fixed number of observations withheld for testing, the additional observations become available for training. This number of 50 models was selected based both on common practices in literature, and observing the convergence of the in-sample error training on preliminary simulation of each of the 111 time series. The trade-off between this approach and cross validation early stopping is that while there is no validation set to assist in preventing overfitting, it provides more data for training, possibly allowing better learning and convergence

of the model to the underlying data generating process.

3.7 Base model

Although Boosting is a generic algorithm irrespective of the underlying base model used, which therefore does not constitute one of the archetypical meta-parameters, we briefly discuss it to consider it in our systematic empirical evaluation. The predictive algorithm underlying Boosting is traditionally required to be a weak learner (see also Section 3.3). Short of a formal performance criterion for a weak learner in forecasting, such as $L_t < 0.5$ in classification, Bühlmann and Yu (2010) interpret a weak learner as a learning algorithm with few degrees of freedom. The few degrees of freedom allow it to adaptively build up the complexity of the combined model, and also to remain sensitive to initial conditions creating diverse predictions from only small variations to the training data, e.g. a neural network or decision tree. If Boosting starts with a base model that is too strong, the training error can quickly rise even as early as the second iteration from overfitting, and the out-of-sample performance becomes very poor. Consequently, in classification, Decision Trees constitute the most widely boosted base models, e.g. in Drucker (1997) and Shresta and Solomatine (2006), often applying the popular classification and regression tree (CART) algorithm by Breiman (1984). Regression trees are powerful, effective and easily interpretable, and are able to automatically select relevant features. In forecasting however, CARTs have not been widely applied, and the studies on Boosting for time series regression have generally applied artificial neural networks. The popular Multilayer Perceptron (MLP), a class of universal approximators (Hornik 1991), can facilitate weak or strong learning through architectures of different complexity, setting the number of layers and hidden nodes accordingly. In the absence of an interpretation for what constitutes a weak or strong learner in time series prediction, studies might need to consider both CART and MLP as base models for their efficacy in Boosting. A description of the setup of both base learners and details on the parameterisation is given in Appendix A.

3.8 Meta-parameter summary

The set of archetypical meta-parameters we identified above, including the loss function and type, the loss update method, the stopping criteria, and the combination method are summarized in Table 1. For each of these, we consider previous realisations of meta-parameters from the seven Boosting variants, e.g. combination methods weighted mean and weighted median, and introduce novel meta-parameter realisations as motivated from forecasting research, e.g. equally weighted mean and median. Combining their respective realisations yields a framework of 96 possible combinations of meta-parameters for Boosting. Of the 96 possible algorithms, 42 novel Boosting variants come from combinations of existing meta-parameters of loss function and type, stopping criteria and combination method and are highlighted in italics, and 47 novel Boosting variants underlined in Table 1 are motivated from forecasting research and distinguished by the use of the ensemble loss update method.

Table 1: Framework of Boosting meta-parameters, specifying 96 different Boosting algorithms

		Mean				Weighted Mean				Median				Weighted Median			
		Bound		Unbound		Bound		Unbound		Bound		Unbound		Bound		Unbound	
		Ens	Mod	Ens	Mod	Ens	Mod	Ens	Mod	Ens	Mod	Ens	Mod	Ens	Mod	Ens	Mod
Threshold	Linear	#1	*#2*	#3	*#4*	#5	**#6**	#7	**#8**	#49	**#50**	#51	*#52*	#53	*#54*	#55	*#56*
	Square	#9	*#10*	#11	*#12*	#13	*#14*	#15	#16	#57	*#58*	#59	*#60*	#61	*#62*	#63	*#64*
	Expon.	#17	*#18*	#19	*#20*	#21	*#22*	#23	#24	#65	*#66*	#67	*#68*	#69	*#70*	#71	*#72*
Non-threshold	Linear	#25	*#26*	#27	**#28**	#29	*#30*	#31	#32	#73	*#74*	#75	*#76*	#77	**#78**	#79	*#80*
	Square	#33	*#34*	#35	*#36*	#37	*#38*	#39	#40	#81	*#82*	#83	*#84*	#85	**#86**	#87	*#88*
	Expon.	#41	*#42*	#43	*#44*	#45	*#46*	#47	**#48**	#89	*#90*	#91	*#92*	#93	**#94**	#95	*#96*

Notes: Numbers highlighted in boldface correspond to the seven existing boosting variants already introduced by prior research studies. Numbers in italics are the 42 novel Boosting variants derived from combinations of existing meta-parameters. Numbers underlined identify the 47 novel Boosting variants motivated from forecasting research. # = Algorithm Id; Mod = Model loss calculation; Ens = Ensemble loss calculation

The remaining 7 correspond to the seven previously proposed algorithms and are highlighted in boldface. These are the AdaBoost.R2 by Drucker (1997) and its variants labelled algorithms #78, #86 and #94 (as does Boosting by Assaad et al., 2008) implementing linear, squared and exponential loss with a non-threshold based bounded loss function and a weighted median combination. AdaBoost.RT, proposed by Shrestha and Solomantine (2006) corresponds to algorithm # 8 using the weighted mean, an unbounded average loss and a linear threshold loss function. Similarly, #6 corresponds to Modified AdaBoost (Goh et al., 2003), #48 to the BCC

Algorithm (de Souza et al., 2006), and #50 to the original Threshold-based Boosting Avnimelech and Intrator (1999), allocating all prior studies in Boosting within the framework.

The additional benefits of the meta-parameter framework are three-fold. First, in addition to the seven existing AdaBoost variants, the combination of all meta-parameters proposed individually as innovations facilitates the creation of 89 novel Boosting algorithms which have not previously been considered in literature. Second, a complete enumeration of all 96 existing and potentially viable Boosting algorithms in a balanced design allows not only a systematic assessment of their relative accuracy, but also an evaluation of the marginal contribution and statistical significance of each meta-parameter choice using a multifactorial ANOVA across all existing Boosting variants, plus any potential interactions. And third, combining those meta-parameter choices with the highest positive marginal contributions to predictive accuracy allows the proposal of a novel algorithm coined Best Combination Boosting (AdaBoost.BC, #28), which is determined from the results of the empirical evaluation experiment outlined below[2].

4. Empirical evaluation

4.1 Dataset

We assess all Boosting variants on the dataset of 111 empirical time series of industry sales from the NN3 forecasting competition (Crone, Hibon, and Nikolopoulos 2011). Created as a subset of renowned M3-competition, it has been established as a valid, reliable, and representative benchmark dataset for computational intelligence (CI) methods in multiple empirical studies. The dataset consists of a representative set of 50 long (L) and 50 short (S) monthly industry time series, with the shortest time series 68 months and the longest 144 months long. Furthermore, the long and short series contain an equal balance of 50 seasonal (S) and 50 non-seasonal patterns (NS), plus 11 time series that exhibit particular complexities in their time series patterns. The balanced design of the data allows an analysis of the accuracy of the Boosting algorithms across typical industrial data conditions, testing their relative

[2] The choice of base learner is not a constituting meta-parameter of AdaBoost; in evaluating two base learners, our experiments will effectively assess 192 boosting variants empirically.

performance across long vs. short and seasonal vs. non-seasonal and complex data patterns. Examples of six time series are shown in Figure 1.

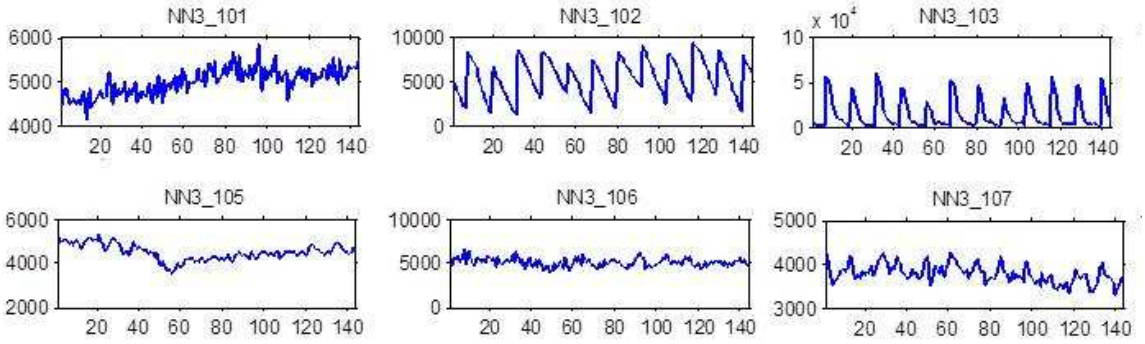


Figure 1 : Plot of 6 time series from the NN3 dataset.

4.2 Forecast performance evaluation

We assess the out-of-sample accuracy of all algorithms in withholding 18 observations for test data in line with the setup of the NN3 competition and to allow a point of comparison with competition results. For a fixed forecast horizon of 12-steps ahead, forecasts are computed from 6 time origins. Overall, this yields a total of 666 multiple-step-ahead out of sample predictions across multiple time origins, 72 per time series, and a total of 7992 predicted data points, which is deemed sufficient to ensure valid and reliable results (of course, albeit only for the assessed dataset). We use the symmetric mean absolute percent error (SMAPE), a comparatively unbiased, scale independent error measure that allows comparing accuracy across multiple time series of different levels. Despite selected criticism of SMAPE, our motivation is to apply the same error measure utilised in the NN3 and M3 competition to facilitate external comparisons with prior findings. For a given time series, the SMAPE is calculated from actuals X_t and forecasts F_t for all n observations indexed by t:

$$SMAPE = \frac{1}{n}\sum_{t=1}^{n}\left(\frac{|X_t - F_t|}{(|X_t| + |F_t|)/2}\right) \times 100 \quad . \tag{13}$$

For each method the SMAPE is calculated across all horizons from each forecasting origin t, averaged across the set of multiple origins per time series (see Tashman 2000), and ultimately

averaged across all time series for the training, validation and test data subsets. We test for statistically significant differences in the performance ranking of competing methods' forecast errors using the nonparametric Friedman test (Friedman 1937; Friedman 1940) and post-hoc Nemenyi test (Nemenyi 1962), providing further evidence on differences of meta-parameters and forecasting benchmarks methods. The tests impose no assumptions regarding the distributions of the errors, and are both based on the mean ranking of the forecast errors of each method over each forecast origin. The Friedman test outputs a p-value evaluating whether at least one of the methods is statistically different from the rest. The Nemenyi test outputs a critical distance based on the mean ranking of each method and which depends on the number of methods, sample size and significance level. This is used to assess whether the mean ranking of methods are significantly different, i.e. differ from each other by more than the critical distance.

In addition to test errors, which assess a true ex ante out-of-sample accuracy of Boosting meta-parameters, we also assess in-sample accuracy on the training and validation sets to assess the ability to approximate and learn the underlying data generating process and to identify potential overfitting. When fixed size combinations are used, the training set is equal to the remaining observations after removing the test data. When the combination size is determined through cross validation, the 14 observations preceding the 18 observations of the test set are used for validation, leaving the remainder for training. However, for the purpose of evaluating and comparing performance evenly across different forecasting methods and combination approaches, we always report forecast error on the 18 holdout observations as test set error, the preceding 14 observations as validation set error, and the remainder as training error[3].

4.3 The multifactorial ANOVA analysis

To quantify the impact and significance of each meta-parameter unto forecasting accuracy, we conduct a multifactorial analysis of variance (ANOVA) with extended multiple comparison tests of estimated marginal means on the out-of-sample forecast errors. The

[3] The validation set is 14 observations (=68-18-36) to allow a sufficient number of training observations, 36, for estimating seasonality in the shortest time series being 68 observations.

experimental setup follows a balanced factorial design, modelling each meta-parameter in Table 1 as a different factor treatment of equal cell sizes. Loss function, loss function type, loss calculation method, combination method, stopping criteria, base model and combination size method are modelled as fixed main effects to test whether the factor levels show different linear effects on the forecast error on the test dataset. In addition to test error, we assess the impact on the dependent variables of training and validation errors to control for potential in-sample overfitting. A meta-parameter is considered to have a relevant impact on forecast performance if it is consistently significant at a 0.01 level using Pillai's trace statistic[4] as measured across all datasets. Additionally, a meta-parameter must prove significant for the individual test set in order for it to be considered as having an impact on out-of-sample performance irrespective of the in sample data. We disregard a significant Box's test of equality of Covariance matrices and a significant Levene's test statistic for equality of error variances as we use a large dataset with equal cell sizes across all factor-level-combinations. In comparing the impact of different factor levels for each of the individual meta-parameter factors, we perform a set of post-hoc multi comparison tests using Tukey's post-hoc test accounting for unequal variances in the factor cells[5]. The test allows us to evaluate for a given factor, the positive or negative impact for each factor level on forecast accuracy. We estimate marginal means across training, validation and test dataset, and use $mm_{factorlevel}$ = {training; test} to represent the difference in the marginal mean forecast error between one factor level and another (ignoring validation errors). A positive value (impact) indicates a decrease in forecast error and vice versa.

4.4 Ensemble Benchmark methods

We assess the accuracy of boosting in relation to other ensemble benchmarks, and

[4] Pillai's trace statistic is one of four multivariate (hypothesis) tests of the significance. These tests replace the usual F value as the indicator of significance in a MANOVA test. Further details on the test can be found in several advanced statistical texts such as the one by Foster et al. (2005).

[5] Tukey's post-hoc test is used to assess more detailed interactions in the MANOVA. Smaller ANOVA models are built using meta-parameters found to significantly impact forecasting accuracy e.g. the loss function. The test compares all possible paired combinations of the factor levels e.g. linear and squared, squared and exponential, linear and exponential, and provides a mean difference between the performances of each factor level, providing a p-value to indicate whether the two differ significantly.

statistical forecasting methods. A popular predecessor to Boosting, Bagging (short for bootstrap and aggregating) is also based on the idea of forming multiple models estimated using different samples of the original training set, and then combining them (Breiman 1996). Recent studies have applied Bagging in a variety of forecasting applications with positive results (e.g., Watson 2005; Lee and Yang 2006; Lin and Zhu 2007; Hillebrand and Medeiros 2010). In this study we apply Bagging to base learners of MLP and CART, using the moving block bootstrap (Kunsch 1989) which samples the original time series while preserving the temporal covariance structure within each input vector. The number of bootstraps is set to 50, and each MLP is trained with early stopping (as described below).

In addition, we implement ensembles of MLP averaging over predictions from multiple initialisations of the same neural network (Hansen and Salamon 1990). For this study, we utilise 50 initialisations of the connecting weights of an identical MLP architecture, and average their 50 predictions for the complete forecasting horizon from each time origin. For both MLP ensembles and Bagging the fixed size of 50 forecasts were found to give the adequate results for both MLP and CART benchmarks. In addition, we also evaluated Ensembles of MLP where the size of the ensemble is determined through cross validation, estimating error improvements while adding additional base model. Furthermore, we compare all methods to simple model selection, where the base-learner with the lowest validation error is selected as a forecast, constituting only a single 'best' model's prediction.

5. Empirical results on Meta-parameter Choice

5.1 Significance of meta-parameter choice

We analyse the results across 96 Boosting variants using multifactorial ANOVA. Table 2 shows between-subject effects using Pillai's trace for all data (training, validation and test subset combined) and for each subset separately, both across all base models and for MLP and CART separately across all data subsets.

Table 2: Significance of meta-parameter main effects by dataset and base model using Pillai's trace.

Factors	Significance by dataset	Significance by method

	All	Train	Valid	Test	MLP	CART
Loss function	0.000**	0.033*	0.175	0.592	0.000**	0.140
Loss function type	0.000**	0.000**	0.000**	0.000**	0.000**	0.000**
Loss calculation	0.000**	0.000**	0.000**	0.000**	0.000**	0.000**
Stopping criteria	0.000**	0.000**	0.000**	0.000**	0.000**	0.000**
Combiner	0.000**	0.000**	0.000**	0.113	0.000**	0.000**
Base model	0.000**	0.000**	0.000**	0.000**	–	–
Combination size	0.000**	0.000**	0.000**	0.000**	0.000**	0.000**

* = Significant at the 0.05 level (2-tailed); ** = Highly significant at the 0.01 level (2-tailed).

Results confirm that the performance of Boosting methods is significantly dependent on the choice of all meta-parameters (at the 0.01 level) across all data (i.e. training, validation, and test combined) and most data subset individually: loss function type, loss calculation method, stopping criteria, base model and combination size determination method are all highly significant at the 0.01 level for all data subsets. The choice of loss function shows no significant impact on validation or test dataset performance, and the choice of combination method has no significant impact on the test set error. Upon closer examination by algorithm, all factors remain significant for MLPs, but not for CART, where the loss function shows no significant impact on forecast accuracy. The significance of all effects on training data versus selected insignificant effects on validation and test data could indicate overfitting, and warrants further analysis.

In addition to the results displayed in the table, all two-way interactions of the factors except those including {loss function} are significant at the 0.01 level across all data subsets, no three-way interactions with the exception of {loss function * loss function type * base model} have a significant impact on forecast accuracy, and only those four-way interactions involving {stopping criteria} prove significant with all other interactions insignificant. To account for the different data conditions, Table 3 shows the significance of each factor based on between-subject effects for the test data analysed by time series type.

Table 3: Significance of meta-parameter factor main effects by time series type for test dataset.

Factors	Significance by time series type					
	SS	SNS	LS	LNS	COMPLEX	ALL
Loss function	0.550	0.001 **	0.752	0.997	0.030 *	0.592
Loss function type	0.000 **	0.000 **	0.000 **	0.000 **	0.000 **	0.000 **
Loss update	0.058	0.000 **	0.000 **	0.073	0.000 **	0.000 **
Stopping criteria	0.000 **	0.000 **	0.000 **	0.000 **	0.000 **	0.000 **
Combiner	0.168	0.000 **	0.156	0.008 **	0.000 **	0.113
Base model	0.738	0.000 **	0.000 **	0.000 **	0.000 **	0.000 **
Combination size	0.000 **	0.000 **	0.003 **	0.000 **	0.000 **	0.000 **

* = Significant at the 0.05 level (2-tailed).; ** = Highly significant at the 0.01 level (2-tailed).

Loss function type, stopping criteria and combination size are highly significant across each of the five time series data conditions. However, all other factors appear insignificant on one or two data conditions. More factors prove insignificant for shorter (seasonal) time series than longer ones, which often prove particularly challenging to build models. However, all factors show a significant influence of test errors on short and non-seasonal time series (SNS, i.e. a simple pattern with only level, noise and autoregressive patterns), and complex time series (i.e. the most complicated patterns), allowing no inference between factors and the difficulty of patterns. More generally, this signifies the interaction of meta-parameters with time series patterns, pointing to the importance of analysis across different data conditions. These results of statistical significance are of relevance, as they signify that all meta-parameters identified in this study have a significant impact on the forecast accuracy of Boosting, across data conditions and base learners, but this varies. Not only does this justify our research question and experimental design, but equally questions the validity of studies which arbitrarily combine meta-parameters of Boosting (or other) algorithms without a systematic evaluation of their partial contributions.

Surprisingly, the insignificant effect of the Combiner and Loss function in Boosting on test error performance stands in contrast to the majority of forecast combination literature. A more detailed analysis of each factor treatment, i.e. the choice of Combiners including mean, median, weighted mean and weighted median, might reveal possible explanations of the lack of influences due to the similarity of selected approaches, which is conducted further below.

5.2 Accuracy of meta-parameter choice

Having identified the significance of the main effects, Table 4 summarises the SMAPE errors for all meta-parameters found significant averaged across all time series and data subsets.

Table 4: Average SMAPE across all time series and across all significant main factor levels.

Stopping	Model	Size	Loss Update	Training		Validation		Test	
				Loss function type					
				Non Threshold	Thres -hold	Non Threshold	Thres -hold	Non Threshold	Thres -hold
CART	Bound	Cross	Ens	9.00%	9.75%	16.78%	18.28%	18.95%	20.29%
		Valid	Mod	8.58%	9.40%	16.47%	18.02%	18.55%	20.05%
		Fixed	Ens	8.92%	10.06%	15.48%	17.51%	17.47%	19.69%
			Mod	8.15%	9.56%	14.69%	17.25%	16.81%	19.06%

	Unbound	Cross	Ens	8.89%	9.12%	16.60%	17.04%	18.58%	18.90%
		Valid	Mod	8.56%	8.63%	16.42%	16.60%	18.29%	18.38%
		Fixed	Ens	8.73%	9.02%	15.23%	16.33%	16.78%	17.49%
			Mod	**8.03%**	8.36%	15.20%	16.31%	16.69%	17.04%
MLP	Bound	Cross	Ens	13.14%	13.20%	14.04%	14.33%	17.15%	17.56%
		Valid	Mod	12.84%	12.98%	13.84%	14.32%	16.98%	17.57%
		Fixed	Ens	13.61%	14.17%	9.32%	9.53%	16.71%	17.64%
			Mod	13.01%	13.79%	9.44%	9.61%	16.39%	17.63%
	Unbound	Cross	Ens	13.76%	12.95%	14.79%	13.80%	17.69%	17.07%
		Valid	Mod	13.38%	12.73%	14.28%	13.72%	17.40%	17.03%
		Fixed	Ens	15.16%	13.36%	11.33%	**8.71%**	17.84%	16.45%
			Mod	13.63%	12.91%	10.32%	8.84%	16.78%	**16.36%**

Note: The method having the lowest forecast error in each of the training, validation and test datasets is highlighted in boldface. The method with the lowest forecast error in each dataset for each of MLP and CART are underlined.
Ens = ensemble loss calculation, Mod= model loss calculation
Cross Valid = Cross validation early stopping, Fixed = Fixed size model combination

Results show that the lowest overall test error is obtained using an unbound stopping criteria, with a fixed combination size and a model calculated loss using a threshold based loss function together with a MLP base model (SMAPE of 16.36%). The same meta-parameter combination, with the exception of using a non-threshold based loss function, is ranked second (with an SMAPE of 16.39%). Averaging errors across the main factors, it is apparent that the CART base model consistently achieve lower training errors than the MLPs (8.92% vs. 13.41%) regardless of the choice of other factors, but vice versa on validation (16.51% vs. 11.89%) and test set (18.31% vs. 17.14%). This indicates substantial in-sample overfitting by CART compared to MLPs, a novel insight for Boosting on time series data, which in contrast to results obtained in Bagging (Breiman 1996; Bauer and Kohavi 1999). In addition we noted that errors of CART are more sensitive to other meta-parameter choices in comparison to MLPs, however with only very small impacts from different factor treatments. As no interaction effects exist between the base model of CART or MLPs with other meta-parameters (see Figure 2), the results are consistent, clearly identifying CART as an inferior base model to MLPs for Boosting across all data conditions of the time series in this dataset.

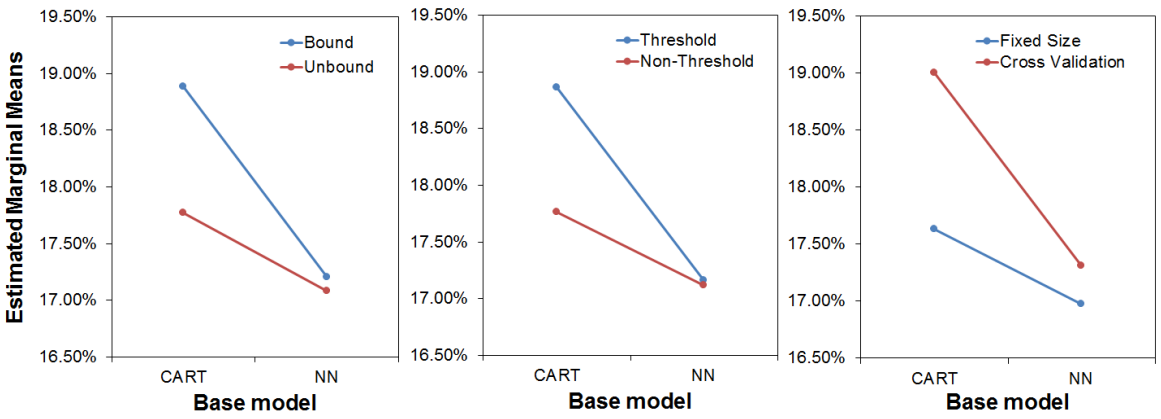


Figure 2: Estimated marginal means plot for test set SMAPE of base models across all time series.

In comparison, other factors show less pronounced average deviations of forecast errors on the test data, such as bound vs. unbound (18.03% vs 17.42%) and threshold vs. non-threshold based loss estimation (18.01% vs. 17.44%), cross validation vs. fixed size of the ensemble (18.15% vs. 17.30%), and Ensemble vs. Model estimated stopping (17.89% vs. 17.56%). These results are consistent across base models, although even less pronounced for MLPs than CART, and would indicate that these meta-parameter choices impact very little on empirical accuracy (irrespective of their statistical significance). However, the significant improvement of accuracy from using an unbound and non-threshold based loss calculation using a fixed ensemble size, together with the use of a MLP base learner, is noteworthy and novel. Given the performance of the MLP relative to the CART base learner and considering that the base learner is not normally considered a meta-parameter adding to the functioning of the Boosting algorithm, we do not proceed further with a detailed analysis of the base learner. All other meta-parameter are assessed in detail in the sections below in an attempt to facilitate a deeper understanding of the direction and magnitude of the main factors, and of potential interactions between them.

5.3 Impact of stopping criteria on forecast performance

To investigate the significance of stopping criteria and their interaction with other meta-parameters, we analyse the estimated marginal means of the forecast error for bounded and unbounded stopping across factor levels for loss function type, loss function and combination size in Figure 3. (As results are consistent across data properties, these graphs are omitted.)

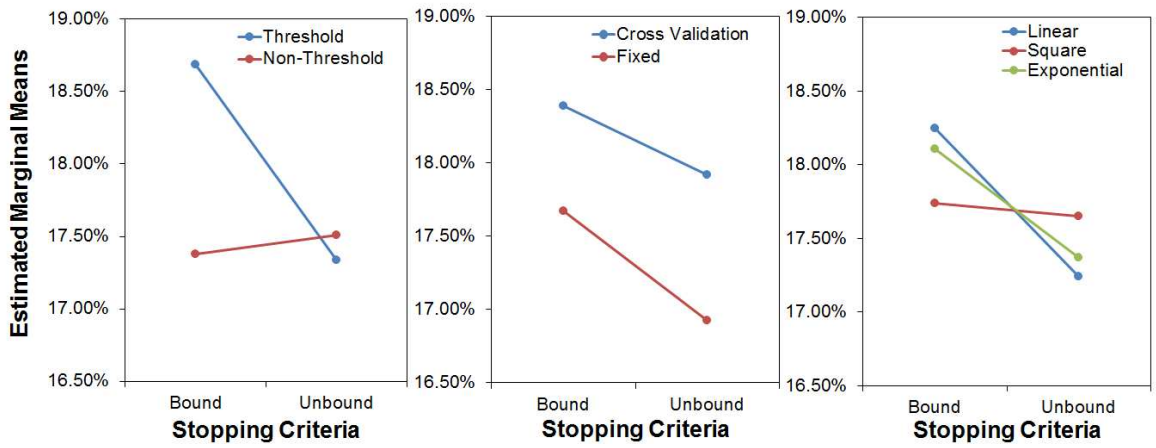


Figure 3: Estimated marginal means plots of test set error across all time series for various factors.

The analysis of marginal means shows that for bounded errors, a significant difference in mean forecast error exists between threshold and non-threshold based estimation, with non-threshold loss functions producing lower forecast errors, $mm_{non-threshold} = \{0.007; 0.013\}$. In contrast, for unbounded errors the non-threshold loss functions show a negative impact of $mm_{non-threshold} = \{-0.004; -0.002\}$. Given a bounded stopping criteria, a squared loss function significantly outperforms both the linear and exponential loss functions, estimated at $mm_{squared} = \{-0.002; -0.004\}$ and $mm_{squared} = \{-0.001; -0.005\}$ improvement over linear and exponential respectively. However, both the exponential and linear loss functions show significantly lower errors than the squared loss function under the unbounded and bounded average error criterion, rendering the previous interaction irrelevant. Given these interactions, the type and choice of loss function used significantly depends on the stopping criteria enforced. A non-threshold based, squared loss function is significantly better when average loss is bounded, yet a linear (or exponential) loss function is preferred when there is no bound on the average error, with both threshold and non-threshold based estimation providing similar low errors. This holds true across both MLP and CART base learners. Fixed size combinations are again better than combination size determined using cross-validation early stopping across both bounded and unbounded parameter values, showing no significant interaction effects.

These findings are of interest, as they indicate that using bounded errors, following the

implementation of bounded errors and weak learner in classification, yields no improvements for Boosting in time series regression. Similarly, using a linear loss function, which does not overemphasise extreme errors from outliers, combined with unbounded estimation outperforms all other loss functions and stopping criteria in time series Boosting. As such, best practices from time series prediction and forecast combination equally hold for Boosting on time series data, leading to the development of distinct and novel Boosting algorithm variants.

5.4 Impact of combination and combination size on forecast performance

We analyse the estimated marginal means for fixed and cross validation combination size factor levels against other main factors, see Figure 4.

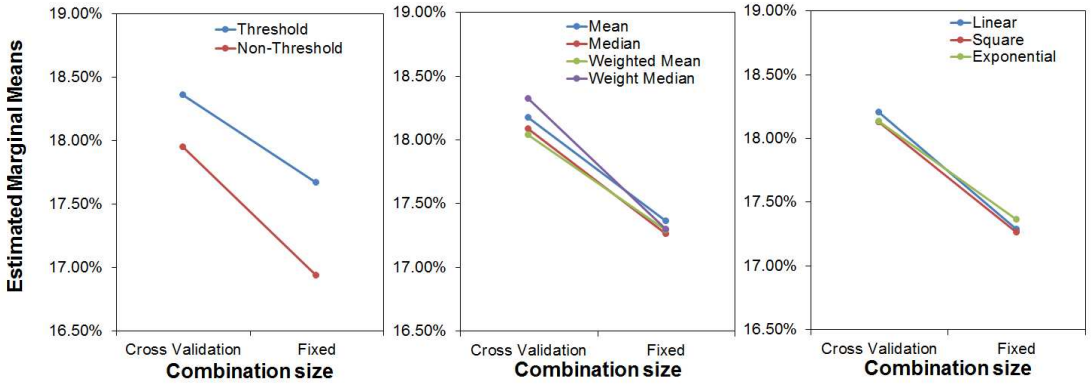


Figure 4: Estimated marginal means plots of test set error across all time series for various factors.

Results indicate consistently lower errors from fixed size estimation across all other meta-parameters without any interaction effects. Additionally the performance of non-threshold versus threshold loss type is more pronounced for fixed size combination. Together with results in Table 4, we observe that this is particularly true when stopping criteria is bounded, that is, when combination size tends to be smaller. In this case the effect of early stopping cross-validation is to make combinations even smaller through early stopping, suggesting that larger combinations are not only better, and but needed to see the full benefits of different loss function types. We also note that the weighted median and weighted mean, the combination operators of choice in Boosting for forecasting, do not significantly outperform the simple mean or simple median. This confirms the findings from forecasting research, where no significant

improvements were found from applying more complex, weighted approaches to forecast combination. As such, it also questions the estimation of the β_k parameter used for base model weighting, potentially allowing a much more substantial simplification of the Boosting algorithm paradigm for time series prediction.

The results hold across data conditions of data subsets, as Table 6 indicates that a fixed size combination provides a lower mean SMAPE than cross validation early stopping across both long and short time series, seasonal and non-seasonal time series. This is supported by Skurichina, Kuncheva, and Duin (2002) who find that Boosting performs best for large training sample sizes. One explanation for the performance of cross validation could be that cross-validated prediction errors can incur large error variances, especially for small datasets, and can therefore become unreliable (Efron 1983). We conclude that different data conditions impact the meta-parameter choice, but since a fixed size combination is computationally less demanding than cross validation early stopping and not less accurate, this is considered sufficient.

Table 5: Test set SMAPE based on combination size method across time series type.

	Cross Validation	Fixed
Long non-seasonal	18.93%	**18.02%**
Long Seasonal	13.59%	**13.38%**
Short non-seasonal	22.24%	**21.75%**
Short Seasonal	17.88%	**16.76%**

5.5 Implications of meta-parameter choice on forecast performance

To summarise the analysis of meta-parameters for Boosting, MLP base models significantly outperform CART models (on this dataset). Fixed size forecast combinations significantly outperform cross validation early stopping, and including models with an average loss $\bar{L} > 0.5$ can improve the out-of-sample performance across all time series patterns, rendering the estimation of a loss bound unnecessary for Boosting on time series. This contrasts findings in classification where the standard Boosting practice of bounded average error is enforced and shown to be successful (Quinlan 1996; Avnimelech and Intrator 1999; Dietterich 2000), indicating the uniqueness of time series regression and the potential to develop specific

Boosting meta-parameters. Surprisingly the choice of loss function is not statistically significant although marginally more accurate (across all base models), yet using a linear loss function seems most plausible due to simplicity, robustness and parsimony. For combination, the widely used weighted median combination method is significantly inferior to all other combination methods on out-of-sample forecast accuracy; instead, the mean and the weighted mean give the smallest error, in concurrence with forecast combination research. Overall, the impact of most AdaBoost meta-parameters depends on the base model. Therefore, while statistically significant improvements in forecast accuracy can be obtained from the right choice of meta-parameters, the choice of MLP or CART base learner is one that must be considered first.

As a further result of our balanced study of existing and novel meta-parameters, we identify those factor-realisations that most influence accuracy and derive a novel AdaBoost variant as a combination of these, also drawing on insights from forecast combination research. The novel Boosting algorithm consequently applies the simple, unweighted arithmetic mean for forecast combination, using an unbounded threshold based, model calculated linear loss with a fixed combination size of MLP base learners. As the novel AdaBoost variant combines the best meta-parameters of our study, we name it AdaBoost.BC (with BC for Best Combination). AdaBoost.BC promises to be a valid contender for Boosting on time series data given its unique combination of meta-parameters. However, its relative performance in comparison to other benchmark algorithms, such as Bagging, statistical forecast combination, or simple model selection also needs to be evaluated, which is documented in the next section.

6. Empirical results on Forecast Accuracy

6.1 Relative performance of forecast combination algorithms

Following a systematic assessment of the impact of different meta-parameters, we assess the relative empirical accuracy of the most popular AdaBoost variants, AdaBoost.R2 and AdaBoost.RT of which both originally employed CART models, with the novel AdaBoost.BC algorithm using a MLP base model. To facilitate an assessment across base learners, both MLP

and CART base models are applied across all AdaBoost variants. Their performance is compared against established forecast combination benchmarks, including Bagging of MLPs, Ensembles of MLPs (which average predictions over multiple initialisations of network weights) created using a fixed size (Ensemble $_{\text{Fixed}}$) or cross validation early stopping (Ensemble $_{\text{Cross Valid}}$) to determine the number of MLPs to be combined, and model selection from the same pool of MLPs. This experimental design facilitates a stepwise assessment of the benefits of forecast combination, random weighted data sampling (i.e. Bagging), and error weighted sampling (i.e. Boosting). Due to the nature of the algorithms, the experiments were replicated 10 times to account for randomised starting weights of the MLPs and potentially different outcomes in relative rankings, providing the average SMAPE over 10 replication runs across 111 industry time series, fixed 12-step-horizons and rolling origins on a holdout test set of 18 observations as in Section 4.2.

Table 6 shows the forecasting performance using SMAPE, Mean Ranks on SMAPE, and Group Ranks of SMAPE based on statistical significant differences using the non-parametric Friedman and Nemenyi tests, with results sorted by descending mean rank on test SMAPE.

Table 6. Average SMAPE and nonparametric comparisons using ranks of SMAPE on training, validation and test set across all time series, averaged across 10 runs for all time series.

	SMAPE			Mean Rank on SMAPE			Group Rank on SMAPE[1]		
Method	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Ensemble $_{\text{Cross Valid}}$	13.20%	8.48%	***15.86%***	83.91	40.40	***47.39***	8	2	***1***
Ensemble $_{\text{Fixed}}$	13.22%	***8.41%***	15.88%	85.23	40.10	***47.81***	8	2	***1***
Bagging $_{\text{MLP}}$	13.72%	8.74%	16.03%	96.27	44.70	51.21	10	3	2
Bagging $_{\text{CART}}$	9.98%	14.36%	16.02%	42.22	79.12	52.45	4	8	2
AdaBoost.BC $_{\text{MLP}}$	13.35%	8.70%	15.94%	90.21	48.33	54.35	9	4	3
AdaBoost.R2 $_{\text{MLP}}$	12.30%	8.52%	16.05%	62.10	***37.86***	55.69	6	***1***	3
AdaBoost.RT $_{\text{CART}}$	8.83%	16.10%	16.56%	65.29	62.09	58.90	7	6	4
AdaBoost.BC $_{\text{CART}}$	8.83%	14.46%	16.58%	22.86	77.77	60.06	2	7,8	4
Adaboost.RT $_{\text{MLP}}$	12.50%	10.64%	16.62%	26.34	84.01	61.23	3	9	4
AdaBoost.R2 $_{\text{CART}}$	***7.18%***	14.77%	16.66%	***6.76***	77.09	67.00	***1***	7	5
Select $_{\text{MLP}}$	14.84%	9.69%	18.47%	95.22	51.29	74.49	10	5	6
Select $_{\text{CART}}$	10.23%	17.77%	20.17%	49.59	83.23	95.41	5	9	7

Note: Forecast errors in boldface highlight the best performing method in each dataset.
[1]Methods that are not significantly different (at $\alpha = 0.05$) belong to the same ranked group.

Results show that all approaches that combine predictions, including Ensembles, Bagging, and all Boosting variants, significantly outperform the selection of an individual, 'single best' base model. Model selection using MLP (Select $_{\text{MLP}}$) and CART (Select $_{\text{CART}}$) turn out to have the largest

percentage errors of 18.47% and 20.17% respectively, and are ranked 6th and 7th respectively after all other methods. This is strong evidence that forecast combination outperforms selection, and is in line with previous findings on forecast combination both in the statistical and econometric literature, where combination is found to routinely outperform attempts to select a single best model.

Second, the novel algorithm of AdaBoost.BC $_{MLP}$ proposed in this paper significantly outperforms all other existing Boosting variants (in addition to the 89 additional Boosting v developed in this paper but not assessed here due to their limited marginal benefit), including the popular algorithms of AdaBoost.R2 $_{CART}$ and AdaBoost.RT $_{CART}$. Assessing both MLP and CART base learners, our study suggest that implementing AdaBoost.R2 and AdaBoost.RT using MLPs would significantly increase accuracy of these established benchmark methods, but still underperform compared to AdaBoost.BC. As AdaBoost.BC outperforms AdaBoost.R2 and AdaBoost.RT across both base models, it provides evidence of the gains from our appropriate choice of meta-parameters, and at the same time the reliability of the findings from the multifactorial ANOVA. To indicate the generalizability of our meta-parameter study, AdaBoost.R2 with CART shows the lowest forecast error on the training set but a poor validation and test set error, confirming earlier findings on in-sample overfitting for decision trees. Similarly, overall, all forecast combination methods using a neural network MLP base learner perform significantly better than methods using CART, with the exception of Bagging $_{CART}$ which is ranked equal second out-of-sample along with Bagging $_{MLP}$.

However, despite AdaBoost.BC demonstrating the most accurate Boosting performance for time series prediction achieved to date, its performance – as that of all Boosting variants – falls short of the forecast accuracy of Bagging and the even simpler Ensemble methods. MLP Ensembles, both using fixed or cross validation size, show the lowest forecast error (15.86% and 15.88% respectively) and rank equal 1st in performance, followed by Bagging $_{MLP}$ and Bagging $_{CART}$ (with 16.03% and 16.02% test error) ranked equal 2nd and only then followed by

Boosting variants in three groups of accuracy, led by 3rd ranked AdaBoost.BC. Reflecting on the relative complexity and approach of the combination algorithms, the simplest approaches of MLP Ensembles utilise all data, equally weighted, and draw only on the diversity inherent in the neural network initialisations to combine forecast of MLPs. They consistently outperform algorithms which randomly sample (i.e. weight) observations to create additional diversity, and both outperform algorithms which create even further diversity of the base models by directing weights unto hard to predict observations. It appears that additional diversity does not add to added accuracy, raising the issue of determining an adequate level of diversity for different dataset conditions. It should however be noted, that despite the rank-based evidence that AdaBoost.BC (ranked 3rd) cannot routinely outperform Bagging with MLPs or CART (ranked equal 2nd), the average SMAPE over 10 runs of the AdaBoost.BC of 15.94% is indeed lower than that of Bagging (with 16.03% and 16.02% respectively) making it the 3rd most accurate algorithm on SMAPE error. While this indicates somewhat skewed error distributions, and the adequacy of rank-based measures, it points to some promise that selected AdaBoost variants can indeed outperform alternative forecast combination approaches, given the dataset.

Given the multifactorial evaluation in which we assess the seven existing and 89 novel Boosting candidates, it suggests that the properties of the Boosting algorithm per se might lead to inferior forecasting performance. Furthermore, both Bagging and Ensembles appear to rank high regardless of the choice of base model or Ensemble configuration, making them more robust in application. We must conclude that – given the meta-parameters currently in use, and for the (albeit considered representative) dataset assessed - none of the Boosting algorithms have the potential to outperform Bagging and Ensembles, both simpler approaches of less complexity, increased transparency and greater efficiency. In this way, the results confirm findings from other studies in forecasting, where more complex algorithms are not able to outperform simpler ones (Crone, Nikolopoulos and Hibon, 2011), providing some evidence of the external validity of our findings. Any additional findings, such as our ability to further

enhance existing algorithms drawing from insights into the meta-parameter framework, e.g. extending AdaBoost.RT by using a non-threshold based loss estimation and MLPs, or AdaBoost.R2 by using an unbound stopping rule with MLP, appear of lesser importance. However, our findings do contribute to future algorithm development in forecast combination in that they eliminate the need for many additional studies with (marginal, often arbitrary) enhancements of a Boosting algorithm leading to only irrelevant improvements in empirical accuracy.

6.2 Relative performance to NN3 competition contenders

In addition to more reliable average performance across 10 replications, we assess the practical case in which one of the 10 contenders would have to be chosen for an actual empirical forecast. Therefore, we compare a single selected AdaBoost.BC contender, selected in-sample using the lowest validation error from the 10 runs, to all other algorithms submitted to the NN3 competition (Crone, Nikolopoulos and Hibon, 2011). As NN3 competition guidelines require, we forecast 18 steps ahead from a fixed origin and a holdout test dataset of the last 18 observations. The results by SMAPE and relative ranking are provided in Table 7 with the assessed Boosting algorithms highlighted in bold.

Table 7. Average errors and ranks of errors across all time series of the NN3 competition.

User ID#	Method or Contestant Name	SMAPE	Rank SMAPE all methods	Rank SMAPE only NN/CI
B09	Wildi	14.84	1	–
B07	Theta	14.89	2	–
-	***Ensemble*** $_{Fixed}$	**15.07**	**3**	**1**
C27	Illies	15.18	4	2
B03	ForecastPro	15.44	5	–
-	**Bagging** $_{MLP}$	**15.68**	**6**	**3**
-	**AdaBoost.BC** $_{MLP}$	**15.76**	**7**	**4**
B16	DES	15.90	8	–
B17	Comb S-H-D	15.93	9	–
B05	Autobox	15.95	10	–
-	**AdaBoost.R2** $_{MLP}$	**15.98**	**11**	**5**
-	**Ensemble** $_{Cross\ Validation}$	**16.05**	**12**	**6**
C03	Flores	16.31	13	–
B14	SES	16.42	14	–
B15	HES	16.49	15	–
C46	Chen	16.55	16	–
C13	D'yakonov	16.57	17	–
-	**AdaBoost.RT** $_{CART}$	**16.77**	**18**	**7**
-	**Select** $_{MLP}$	**16.80**	**19**	**8**
B00	AutomatANN	16.81	20	9

-	**Bagging $_{CART}$**	**17.10**	**21**	**10**
-	**Adaboost.RT $_{MLP}$**	**17.58**	**22**	**11**
-	**AdaBoost.BC $_{CART}$**	**17.94**	**23**	**12**
-	**AdaBoost.R2 $_{CART}$**	**18.19**	**24**	**13**

Note: Forecast errors in boldface highlight the performance of methods evaluated in this study.
Forecast errors in boldface and underlined highlight the best performing computational Intelligence method.

The novel algorithm of AdaBoost.BC would rank 7th overall across all methods (i.e. NN/CI and statistical algorithms), and 4th on NN/CI algorithms, making it one of the best performing algorithms developed in machine learning for time series prediction. AdaBoost.BC is only marginally outperformed by Bagging with MLPs and is better than Comb S-H-D, a combination of Simple (SES), Holt (HES) and Damped (DES) (deseasonalised) Exponential Smoothing forecast. It also outperforms the individual statistical benchmarks of HES and SES. The simpler fixed size ensemble method of MLP model averaging ranks 1st amongst all computational intelligence algorithms, outperforming the Illies' Ensembles of Echo State Neural Nets with pooled training, also a neural network ensemble technique applying equal weighting and training on all data. In comparison model averaging of MLPs with cross validation ranks 12th of all algorithms and 6th amongst computational intelligence methods. In interpreting the relative accuracy, it should be noted that Table 7 shows only the top statistical contenders of the M3 competition (Makidakis and Hibon, 2000) and only the top computational intelligence (CI) contenders of the NN3. Were we to provide the complete list of all 48 algorithms from NN3 and all 24 methods from the M3 competition, all forecast combination approaches based on Boosting would rank in the top decile of empirical results. Overall, the findings again confirm the value of forecast combination for machine learning in comparison to established statistical forecasting methods, and even selected statistical benchmarks of forecast combination, which should help motivate future research in the area.

7. Conclusions

This paper systematically evaluates the empirical forecast performance of Boosting algorithms in forecasting real-world industry time series. The empirical evaluation is conducted using a valid and reliable experimental design using 111 time series of the NN3 competition, ex

post evaluation across multiple rolling origins, and comparing the accuracy of all Boosting variants to established Benchmarks of Bagging, Ensemble Model Averaging, Model Selection and statistical forecasting methods. We provide empirical evidence that forecast combination outperforms the use of aggregate model selection regardless the choice of base model. Results of a multifactorial analysis of variance after decomposing Boosting into archetypical meta-parameters indicate that the choice of loss function type, loss update, stopping criteria, base model and combination size all have a (statistically) significant impact on the forecast accuracy of Boosting algorithms. Among the most important of these meta-parameters are the choice of loss function, and the method of determining the number of models included in the forecast combination. As we would expect from forecasting research, many meta-parameters proposed in this study outperform those from Boosting in Classification, including the use of a simple, equally weighted mean as combination method, an unbound loss estimation without the use of thresholds, and a linear loss function. The choice of the base model between MLP neural networks and CART decision trees also proves highly significant, and one of the most influencing determinants.

The experimental results show that the appropriate selection of meta-parameters leads to improved forecast accuracy, and allows us to derive a best combination of meta-parameter choice Boosting variant coined AdaBoost.BC, which outperforms all other Boosting variants developed for time series prediction to date. Our findings also indicate that standard meta-parameter choices may lead to suboptimal performance, so that existing algorithms of AdaBoost.RT and AdaBoost.R2 can be marginally improved by adjusting meta-parameters. However, these yield little impact, as despite the gains in accuracy achievable for Boosting from careful selection of meta-parameters, ex post model averaging using Ensembles of neural networks as well as Bagging outperform the best performing Boosting variants conflicting with previous findings (e.g., Avnimelech and Intrator 1999; Shrestha and Solomatine 2006). The findings therefore suggests the need to modify the reweighting and resampling schemes used in

the Boosting algorithms if they are to prove successful in outperforming standard combination approaches.

Our findings are limited in a number of ways, beyond that holding only for the dataset conditions we analysed. Although the dataset properties and the balanced design of the NN3 dataset, including long and short, seasonal and non-seasonal time series of monthly observations, covers a large segment of dataset properties used in industry to date, they cannot be deemed representative e.g. for daily or intraday data. Here additional studies with rigorous empirical evaluations are needed. Also, our experiments do not cover variants of gradient based Boosting approaches, which fall outside the AdaBoost properties and do not lend themselves easily to an analysis in the structure of the meta-parameter framework developed here. However, the study remains representative for the majority of Boosting algorithms used in forecasting, which are largely derived from AdaBoost. Finally, our experiment covers only meta-parameter features found in Boosting literature and those motivated from Forecasting research. Undoubtedly, many more could be considered to enhance research.

Future work should consider various extensions of Boosting along the lines of enhanced meta-parameter options. A loss function based on SMAPE would allow a direct loss estimation in the naturally bound interval of [0, 2], with the highest SMAPE error never exceeding 200%. This would also establish some congruency between the objective function to develop the algorithm with the metric it is ultimately assessed on. More importantly, theoretical advances on estimating the necessary condition for a weak learner in Boosting for regression in general, and time series prediction in particular, could help guide what base models to combine, and how to specify them. A potential starting point for defining a base learner is that it has performance similar to that of the Naïve (random walk) method. This is akin to the Mean Absolute Scaled Error (MASE) performance measure of Hyndman and Koehler (2006) which is scaled to the Naïve as a measure of relative improvement. Overall, our study has shown the benefit of combining insights from forecasting research on combination with the development of machine

learning algorithms, making this a promising approach to guide future research in forecast combination.

References

Aksu, C., & Gunter, S. I. (1992). An empirical analysis of the accuracy of SA, OLS, ERLS and NRLS combination forecasts. *International Journal of Forecasting*, 8(1), 27-43.

Assaad, M., Bone, R., & Cardot, H. (2008). A new Boosting algorithm for improved time-series forecasting with recurrent neural networks. *Information Fusion*, 9(1), 41-55.

Audrino, F. 2006. The impact of general non-parametric volatility functions in multivariate GARCH models. *Computational Statistics & Data Analysis* 50 (11):3032-3052.

Audrino, F., and P. Bühlmann. 2009. Splines for financial volatility. *Journal of the Royal Statistical Society Series B-Statistical Methodology* 71:655-670.

Avnimelech, R., & Intrator, N. (1999). Boosting regression estimators. *Neural Computation*, 11(2), 499-520.

Bates, J. M., & Granger, C. W. J. (1969). The Combination of Forecasts. *OR Quarterly*, 20(4):451-468.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, Boosting, and variants. *Machine Learning*, 36(1-2), 105-139.

Bodyanskiy, Yevgeniy, and Sergiy Popov. 2006. Neural network approach to forecasting of quasiperiodic financial time series. *European Journal of Operational Research* 175 (3):1357-1366.

Bone, R., Assaad, M., & Crucianu, M. (2003). Boosting recurrent neural networks for time series prediction. In D. W. Pearson, N. C. Steele & R. F. Albrecht (Eds.), *Artificial Neural Nets and Genetic Algorithms*. Vienna.

Bordley, R. F. (1982). The Combination of Forecasts: A Bayesian Approach. *Journal of the Operational Research Society*, 33(2), 171-174.

Breiman, L. (1984). *Classification and regression trees*. Michigan: Wadsworth Int. Group.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123-140.

Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, 26(3), 801-824.

Bühlmann, P., & Bin Yu. (2010). Boosting. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1), 69-74.

Bunn, D. W. (1975). Bayesian approach to linear combination of forecasts. *OR Quarterly*, 26(2), 325-329.

Bunn, D. W. (1985). Statistical efficiency in the linear combination of forecasts. *International Journal of Forecasting*, 1(2), 151-163.

Bunn, Derek W. 1988. Combining forecasts. *European Journal of Operational Research* 33 (3):223-229.

Bunn, D. W. (1989). Forecasting with more than one model. *Journal of Forecasting*, 8(3), 161-166.

Canestrelli, E., Canestrelli, P., Corazza, M., Filippone, M., Giove, S., & Masulli, F. (2007). Local learning of tide level time series using a fuzzy approach. *Proceedings of IEEE International Joint Conference on Neural Networks* (pp. 813-1818).

Chan, C. K., B. G. Kingsman, and H. Wong. 1999. The value of combining forecasts in inventory management - a case study in banking. *European Journal of Operational Research* 117 (2):199-210.

Clements, M. P., & Hendry, D. F. (2007). An Overview of Economic Forecasting. In M. P. Clements & D. F. Hendry (Eds.), *A Companion to Economic Forecasting*.

Crone, S. F., Hibon, M., & Nikolopoulos, K. (2011). Advances in forecasting with neural networks? Empirical evidence from the NN3 competition. *International Journal of Forecasting*, 27(3), 635-660.

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International Journal of Forecasting*, 22(3), 443-473.

de Menezes, L. M., Bunn, D. W. & Taylor, J. W. (2000). Review of guidelines for the use of combined forecasts. *European Journal of Operational Research*, 120(1), 190-204.

de Souza, L. V., A. Pozo, J. M. C. da Rosa, and A. C. Neto. 2010. Applying correlation to enhance boosting technique using genetic programming as base learner. *Applied Intelligence* 33 (3):291-301.

Deng, Y. F., Jin, X., & Zhong, Y. X. (2005). Ensemble SVR for prediction of time series, *Proceedings of International Conference on Machine Learning*, 6, 3528-3534.

Diebold, F. X., and P. Pauly (1990). The use of prior information in forecast combination. *International Journal of Forecasting* 6 (4):503-508.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, Boosting, and randomization. *Machine Learning*, 40(2),139-157.

Drucker, H. (1997). Improving Regressors using Boosting Techniques. In *The Fourteenth International Conference on Machine Learning* (pp. 107-115). Morgan Kaufmann Inc.

Efron, B. (1979). Bootstrap Methods - Another Look at the Jackknife. *Annals of Statistics*, 7(1), 1-26.

Elliott, G., Granger, C.W.J., & Timmermann, A. (Ed.). (2006). *Handbook of Economic Forecasting*. Holland.

Foster, J. J., Barkus, E., & Yavorsky, C. (2005). *Understanding and using advanced statistics: a practical guide for students*. Sage.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to

Boosting. *Journal of Computer and System Sciences*, 55(1), 119-139.
Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, *14*(5), 771-780.
Friedman, J. H. (2001). Greedy function approximation: A gradient Boosting machine. *Annals of Statistics*, 29(5), 1189-1232.
Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of Boosting. *Annals of Statistics*, 28(2), 337-407.
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.
Friedman, M. (1940). A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, 11(1), 86-92.
Goh, W. Y., Lim, C. P., & Peh, K. K. (2003). Predicting drug dissolution profiles with an ensemble of boosted neural networks. *IEEE Transactions on Neural Networks*, 14(2), 459-463.
Granger, C. W. J., & Ramanathan, R. (1984). Improved methods of combining forecasts. *Journal of Forecasting*, 3(2), 197-204.
Gunter, S. I. (1992). Nonnegativity restricted least squares combinations. *International Journal of Forecasting*, 8(1), 45-59.
Hagan, M.T., Demuth, H.B., & Beale, M.H. (1996). *Neural Network Design*. PWS Publishing.
Hansen, L. K., & Salamon. P. (1990). Neural Network Ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10), 993-1001.
Hillebrand, E., & Medeiros, M. C. (2010). The Benefits of Bagging for Forecast Models of Realized Volatility. *Econometric Reviews*, 29(5-6), 571-593.
Ho, T. K. (2002). A data complexity analysis of comparative advantages of decision forest constructors. *Pattern Analysis and Applications*, 5(2), 102-112.
Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2), 251-257.
Hyndman, Rob J., and Anne B. Koehler. 2006. Another look at measures of forecast accuracy. *International Journal of Forecasting* 22 (4):679-688.
Jose, V. R. R., & Winkler, R. L. (2008). Simple robust averages of forecasts: Some empirical results. *International Journal of Forecasting*, 24(1), 163-169.
Kourentzes, N., Barrow, D. K., & Crone, S. F. (2014). Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, *41*(9), 4235-4244.
Kunsch, H. R. (1989). The Jackknife and the Bootstrap for General Stationary Observations. *Annals of Statistics*, 17(3), 1217-1241.
Langella, G., Basile, A., Bonfante, A., & Terribile, F. (2010). High-resolution space-time rainfall analysis using integrated MLP inference systems. *Journal of Hydrology*, 387(3-4), 328-342.
Lee, T. H., & Yang, Y. (2006). Bagging binary and quantile predictors for time series. *Journal of Econometrics*, 135(1-2), 465-497.
Leung, Mark T., Hazem Daouk, and An-Sing Chen. 2001. Using investment portfolio return to combine forecasts: A multiobjective approach. *European Journal of Operational Research* 134 (1):84-102.
Lin, J., & Zhu, B. Z. (2007). A novel neural network ensemble system for economic forecasting. *Advanced Intelligent Computing Theories*, 2, 1227-1233.
Macdonald, R., & Marsh, I. W. (1994). Combining exchange-rates forecasts – what is the optimal consensus measure. *Journal of Forecasting*, 13(3), 313-332.
Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., & Winkler, R. (1982). The accuracy of extrapolation (time series) methods – results of a forecasting competition. *Journal of Forecasting*, 1(2), 111-153.
Makridakis, S., C. Chatfield, M. Hibon, M. Lawrence, T. Mills, K. Ord, and L. F. Simmons. 1993. The M2-Competition - A Real-time judgmentally based Forecasting Study. *International Journal of Forecasting* 9 (1):5-22.
Makridakis, S., and M. Hibon. 2000. The M3-Competition: results, conclusions and implications. International Journal of Forecasting 16 (4):451-476.
Min, C. K., & Zellner, A. (1993). Bayesian and Non-Bayesian Methods for Combining Models and Forecasts with Applications to Forecasting International Growth-Rates. *Journal of Econometrics*, 56(1-2), 89-118.
Nemenyi, P. (1963). *Distribution-free multiple comparisons*. Princeton University.
Newbold, P., & Granger, C. W. J. (1974). Experience with forecasting univariate time series and combination of forecasts. *Journal of the Royal Statistical Society. Series A*, 137(2), 131-165.
Quinlan, J. R. (1996). *Bagging, Boosting, and C4.5*. 13th National Conference on Artificial Intelligence.
Riedel, S., & Gabrys, B. (2009). Pooling for Combination of Multilevel Forecasts. *IEEE Transactions on Knowledge and Data Engineering*, 21(12), 1753-1766.
Robinzonov, N., G. Tutz, and T. Hothorn. 2012. Boosting techniques for nonlinear time series models. *Asta-Advances in Statistical Analysis* 96 (1):99-122

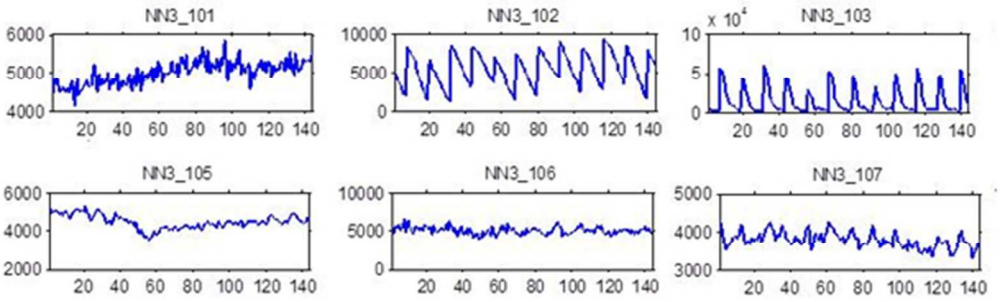
Rokach, L. (2009). Taxonomy for characterizing ensemble methods in classification tasks: A review. *Computational Statistics & Data Analysis*, 53(12), 4046-4072.
Schapire, R. E. (1990). The Strength of Weak Learnability. *Machine Learning*, 5(2), 197-227.
Schapire, R. E. (2003). The Boosting approach to machine learning. *In proceedings of MSRI Workshop on Nonlinear Estimation and Classification* (pp. 49-171).
Schapire, R. E., Freund, Y., & Bartlett, P. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5), 1651-1686.
Schapire, R. E., & Singer, Y. (1999). Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*, 37(3), 297-336.
Shrestha, D. L., & Solomatine, D. P. (2006). Experiments with AdaBoost.RT, an improved Boosting scheme for regression. *Neural Computation*, 18(7), 1678-1710.
Skurichina, M., Kuncheva, L. I., & Duin, R. P. W. (2002). Bagging and Boosting for the nearest mean classifier: Effects of sample size on diversity and accuracy. In F. Roli and J. Kittler (Eds.), *Multiple Classifier Systems*, Berlin: Springer-Verlag Berlin.
Stock, James H., & Watson, M. W. (2004). Combination forecasts of output growth in a seven-country data set. *Journal of Forecasting*, 23(6), 405-430.
Taieb, S. B., & Hyndman, R. J. (2014). A gradient boosting approach to the Kaggle load forecasting competition. *International journal of forecasting*, *30*(2), 382-394
Tashman, J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. International *Journal of Forecasting*, 16(4), 437-450.
Watson, J., Stock, H., & Mark W. (2005). *An empirical comparison of methods for forecasting using many predictors*. Princeton University.
Weigend, A. S., & Gershenfeld, N. A. (1993). Results of the Time Series Prediction Competition at the Santa Fe Institute. *IEEE International Conference on Neural Networks*, 3, 1786-1793.
Winkler, R. L., & Clemen, R. T. (1992). Sensitivity of weights in combining forecasts. *Operations Research*, 40(3), 609-614.
Wohlrabe, K., & Buchen, T. (2014). Assessing the Macroeconomic Forecasting Performance of Boosting: Evidence for the United States, the Euro Area and Germany. *Journal of Forecasting*, *33*(4), 231-242.
Wright, J. H. (2008). Bayesian Model Averaging and exchange rate forecasts. *Journal of Econometrics*, 146 (2):329-341.
Zhang, Feng. 2007. An application of vector GARCH model in semiconductor demand planning. *European Journal of Operational Research* 181 (1):288-297.
Zhang, G. Q., Patuwo, B. E., & Hu. M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14 (1):35-62.

Appendix A. Base learner parameterisation

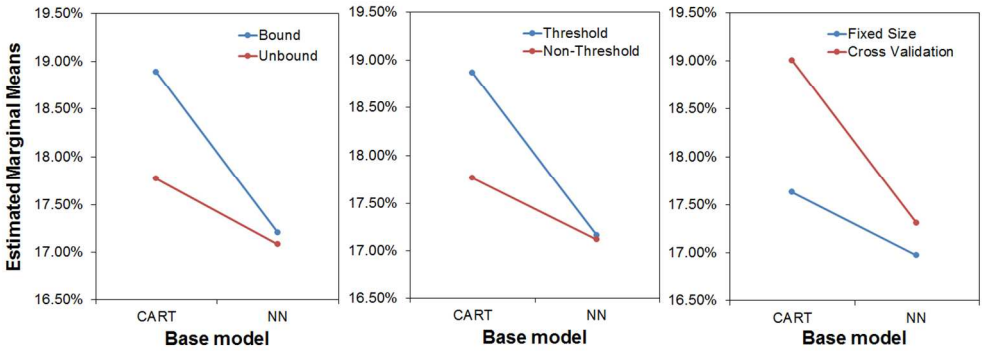
Here we provide a description of the setup of both base learners and details on the parameterisation. To specify the MLP architecture, we construct a MLP with a single hidden layer with only two hidden nodes, using a hyperbolic tangent activation function and linear output function (Zhang, Patuwo, and Hu 1998). This limits the network architecture to very few degrees of freedom in accordance with the weak learner requirement of Boosting (Bühlmann and Yu 2010). For the input layer, we employ a MLP in a univariate setup, employing continuous autoregressive lags x_t up to x_{t-13}. The input vector is sufficient to model monthly stochastic seasonality, and potentially stochastic trends, although the NN3 dataset does not contain time series with significant trends. A single output node with the identity activation function is used for all networks. Each MLP is trained directly on each time series, linearly scaled into the interval of [-0.5, 0.5] to facilitate training, but without prior differencing or data transformation, to estimate level, seasonality, and potential trend directly in the network weights and the bias terms. For parameterisation, data is presented to the MLP as an overlapping set of input vectors formed from a sliding window over the time series observations. The training algorithm used is the Levenberg-Marquardt algorithm (Hagan, Demuth, and Beale 1996), minimising the MSE up to a maximum of 1000 epochs. The algorithm requires setting a scalar μ_{LM} and its increase and decrease steps, using $\mu_{LM} = 10^{-3}$, with an increase factor of $\mu_{inc} = 10$ and a decrease factor of $\mu_{dec} = 10^{-1}$. When a validation set is used, network training stops if error on the validation set increases or remains the same for more than 50 epochs. Additionally network training stops if μ_{LM} exceeds $\mu_{max} = 10^{10}$. The network weights giving the lowest validation error during training are used in order to reduce overfitting to the training data. Each MLP is initialised multiple times with randomised starting weights to account for local minima in training. During training of the MLP for Boosting, the identical

initial weights are used across all meta-parameter choice combinations to allow for any differences in performance to be attributed solely to the choice of meta-parameters, and not to different starting weights.

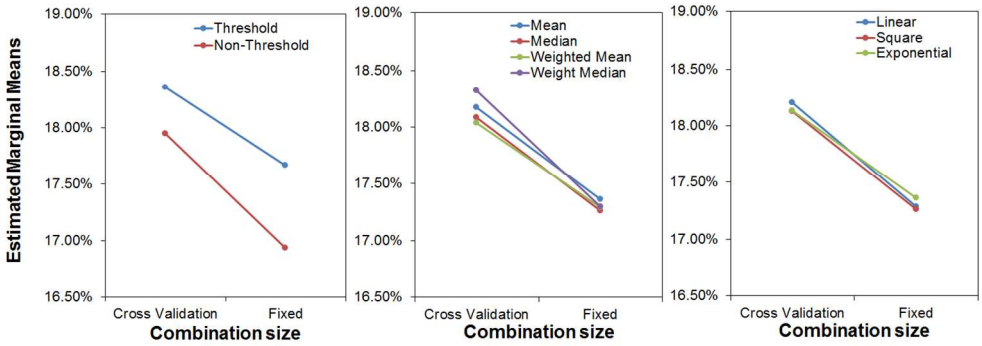
For decision trees, the popular algorithm of CART is being used. In contrast to traditional methods such as ordinary least squares (OLS) regression and discriminant analysis, CART does not depend on assumptions of normality and user-specified model statements. The resulting predictors are typically simple functions of the corresponding input variables and are often easy to use and to interpret. A detailed review of the effective use of regression trees in the context of Boosting is given in Schapire and Singer (1999). When the tree is constructed, the terminal node will have assigned to it the predicted value for input $\mathbf{x}_i$, corresponding to the node. The tree is built such that at each stage, the split selected is the one leading to the greatest reduction in the sum of squared error between the actual values of the training set examples corresponding to a particular node, and their sample mean which is the predicted value. The tolerance on the sum of squared errors is set to 1e-6 such that splitting nodes stops when the error drops below this value. We consider the standard pruning procedure with reduced-error pruning implemented in the MATLAB statistics toolbox. For both base learners, we tune Boosting's suboptimal threshold value ϕ for each time series on the validation set, estimated as the mean across multiple initialisations (see Eq. (4))



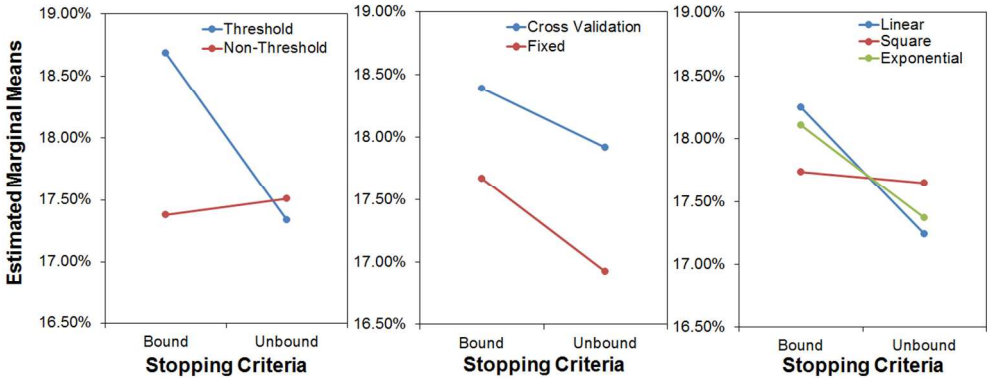
161x53mm (96 x 96 DPI)



154x55mm (240 x 240 DPI)



156x55mm (240 x 240 DPI)



358x137mm (96 x 96 DPI)