International Journal of Forecasting 32 (2016) 1103–1119

Contents lists available at ScienceDirect

International Journal of Forecasting

journal homepage: www.elsevier.com/locate/ijforecast

A comparison of AdaBoost algorithms for time series forecast combination

Devon K. Barrow [a,*], Sven F. Crone [b]

[a] School of Strategy and Leadership, Coventry University, Priory Street, Coventry CV1 5FB, UK

[b] Lancaster University Management School, Department of Management Science, Lancaster LA1 4YX, UK

ARTICLE INFO

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

More recently, research on combination in machine learning has extended the popular classification algorithm AdaBoost by Freund and Schapire (1997) to regression with time series data, typically altering one or more of its com-

* Corresponding author.
E-mail address: devon.barrow@coventry.ac.uk (D.K. Barrow).

http://dx.doi.org/10.1016/j.ijforecast.2016.01.006
0169-2070/

ponents in order to create a set of seven distinct boosting algorithms for forecasting (e.g., Assaad, Bone, & Cardot, 2008; de Souza, Pozo, da Rosa, & Neto, 2010; Shrestha & Solomatine, 2006) that are 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, & Abe, 1999; Schapire, Freund, & Bartlett, 1998). As boosting combinations differ substantially from traditional forecast combination approaches, which combine predictions from different algorithms that are all estimated on the same data, they promise a novel approach to improving the accuracy of forecast combinations. However, while boosting is accepted widely as a benchmark in cross-sectional predictive classification, based on its preeminent performance in a number of empirical evaluations (Breiman, 1998; Rokach, 2009), there has been no systematic assessment of the empirical accuracy of its variants in time series regression. Although each of the seven boosting variants by different authors has proposed a different alteration of the original AdaBoost algorithm for regression, their relative accuracies have not been compared. Furthermore, none of the boosting variants have been compared to the established approaches of conventional forecast combination, individual model selection, or aggregate selection by employing simple statistical benchmarks such as the naïve, exponential smoothing and ARIMA approaches, thus making it impossible to assess their relative merits. In addition, none of the algorithms have been assessed on a representative dataset of empirical data, with most having been evaluated on only 1–3 synthetic time series, which do not allow a generalisation of their reliability to empirical industry data. Thus, in the absence of an objective assessment of the 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, making 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 that can assess multiple-step-ahead forecasts across multiple rolling origins. (II) In order to identify the relative merits of each of the seven algorithms' innovations, we decompose boosting into six archetypical meta-parameters that determine the behaviours 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 that have not been considered previously, to guide future choices and studies on meta-parameters. From this decomposition, we note that all boosting variants to date have ignored various important insights from forecasting research, including the use of simple averages when combining predictions, avoiding higher-order loss functions, and the bias of MAPE-based relative error estimation, which could potentially limit their adequacy and empirical performances. (III) As a consequence, we create a further 47 novel boosting variants with meta-parameters that are motivated by traditional forecast combination research. In evaluating them all against a set of representative benchmark algorithms, we identify a novel best combination (BC) boosting algorithm based on new meta-parameters which improves the 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 thus providing further motivation for this study. Next, Section 3 describes and decomposes the AdaBoost algorithm into its archetypical meta-parameters, in order to help us understand and evaluate the resulting boosting variants, and to derive the properties of a best combination boosting approach. Section 4 describes the experimental design and the implementation of the empirical evaluation, including the benchmark methods and the specification of the base models. This is followed by results on the statistical significance of all meta-parameter combinations in Section 5, a discussion of the forecast accuracy relative to other contenders in 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, which is the best-known and most widely applied boosting algorithm in both research and practice (Schapire, 2003).[1] Although this algorithm offers a distinct approach to the combination of predictions, and has achieved success in predictive classification, only few studies in machine learning have extended AdaBoost to regression on time series data, and there has been almost no resonance in the domains of forecasting and econometrics. The first application of boosting to regression with time series data is credited to Avnimelech and Intrator (1999), who used a modification of the original AdaBoost algorithm to forecast the laser data from the Santa Fe time series competition (Weigend & Gershenfeld, 1993). Shrestha and Solomatine (2006) later developed the AdaBoost.RT algorithm, and used it to predict the Santa Fe time series data as well. Since then, the limited research to date has focussed on what can only be described as marginal extensions of AdaBoost, where a researcher typically alters one or more of the algorithm's components in order 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) to the forecasting of tide levels; and

[1] It should be noted that other boosting approaches also exist, based on the work of Friedman (2001) and Friedman, Hastie, and Tibshirani (2000), 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 & Bühlmann, 2009), economics (Wohlrabe & Buchen, 2014) and production (Robinzonov, Tutz, & Hothorn, 2012).

an application of AdaBoost.R2 by Assaad et al. (2008), who inaccurately describe it as a new algorithm when applying a recurrent neural network learner.

With all of the aforementioned research having been published in the machine learning literature, with a focus on neural network journals, many of these extensions have ignored important insights from forecasting research within the management sciences. For example, Adaboost.R2 (Drucker, 1997), Modified AdaBoost (Goh et al., 2003), AdaBoost.RT (Shrestha & Solomatine, 2006), and the boosting algorithm of Assaad et al. (2008), all of which were developed exclusively for time series data, each use a weighted median or mean to combine the final predictions, even though the current forecasting research concurs as to the use of simple averages (see e.g. Winkler & Clemen, 1992). Similarly, from early algorithms of Adaboost.R2 to more recent developments in boosting (Assaad et al., 2008; Shrestha & Solomatine, 2006), many have suggested the use of a squared or exponential loss function in guiding the boosting algorithm on harder-to-learn examples, despite the fact that forecasting research has shown higher-order errors to bias the results (see e.g. Tashman, 2000). As a remedy, Shrestha and Solomatine (2006) suggest that AdaBoost.RT use a linear relative error that is similar in nature to the mean absolute percent error (MAPE), in spite of MAPE's documented biases in asymmetry and problems with zero and low-value observations (Hyndman & Koehler, 2006). Without input from the forecasting literature, boosting variants have ignored proven best practices from forecasting research, such as the use of the simple mean for combination, absolute error loss functions, and corresponding mean absolute errors (MAE).

At the same time, the different boosting variants have not been compared against each other either conceptually or in an empirical evaluation of their predictive accuracy, which does not allow their individual merits in different data conditions to be assessed. Of the few studies that have attempted empirical evaluations, the experimental designs are often found to be lacking in scientific rigor (see e.g. the recommendations by Tashman, 2000): all studies forecast fewer than five time series in total, assess the accuracy only from a single time origin, use biased error metrics, and provide no assessment of the statistical significance of the findings. More importantly, the performance of boosting on time series is not compared to those of other competing algorithms. A few studies have compared the performance of boosting to those of other machine learning approaches (usually bagging; see the studies by Avnimelech & Intrator, 1999; Deng, Jin, & Zhong, 2005; or Shrestha & Solomatine, 2006), but none have compared it to those of the statistical benchmarks of forecast combinations, simple model selection, or even basic statistical forecasting algorithms, that are employed regularly in forecasting practice. Reflecting on the relevance of an empirical evaluation to a method's efficacy, this is a significant research gap, which our study seeks to remedy.

To facilitate a comprehensive understanding of AdaBoost across disciplines, as well as to allow replications of the extensions and experiments, the algorithm is described formally 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 in general below, with detailed discussions of each meta-parameter choice being provided in the following subsections.

Initially, a dataset $S = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\rangle$ is created from a time series, where the dataset consists of N pairs of observations $(\mathbf{x}_i, y_i)$, for $i = 1, \ldots, N$, where $\mathbf{x}_i$ is a vector comprising 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 a weight $w_i = 1$, for all $i = 1, \ldots, N$, thus creating a uniform distribution. Based on its weight, each observation i is then assigned a probability $p_i^k = w_i^k / \sum w_i^k$ of being included in the training set at 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) that is capable of learning such a mapping, and calculates the loss suffered when predicting each observation L_i^k, with $L_i^k \in [0, 1]$, based on the choice of loss function (see Section 3.2) that specifies 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):

$$\bar{L}_k = \sum_{i=1}^{N} L_i^k p_i^k. \tag{1}$$

Based on this average model loss, a measure of the predictive accuracy β_k of the resulting forecast model is calculated. The way in which 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 these 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 behaviours of all boosting variants (Bühlmann & Yu, 2010; Rokach, 2009): 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 the existing literature to highlight their impacts and interactions, and contrast their choices with research findings on best practice from the forecast combination literature in order to suggest improved meta-parameters.

3.2. Loss function

The loss function controls the magnitude of the relative error that is attributed to a predicted observation, and thus the weight update of that observation for the next iteration $k+1$. We begin by considering 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; otherwise, it is classed as a good forecast. The resulting loss L_i^k for observation i is then given by the threshold function:

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

with two classes of errors for poor and good forecasts respectively. In contrast, AdaBoost.R2, proposed by Drucker (1997), utilises the absolute percentage error (APE), which is 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:

$$L_i^k = \begin{cases} APE_i^k, & f_k(\mathbf{x_i}) \neq y_i \\ 0, & f_k(\mathbf{x_i}) = y_i. \end{cases} \tag{6}$$

Both the impact of loss functions on the time series forecast accuracy and interactions with other meta-parameters have largely been ignored, with the only evaluation of the linear, squared and exponential loss of AdaBoost.R2 on time series data being that by Assaad et al. (2008). When considering the loss function type, only Shrestha and Solomatine (2006) evaluated the threshold loss function in Eq. (4) against a non-threshold loss function, with no other studies investigating further. In this study, we extend the functional form of the threshold and non-threshold loss functions in Eqs. (4) and (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 by Drucker (1997):

$$APE_i^k = \frac{|f_k(\mathbf{x_i}) - y_i|^2}{D^2}, \tag{7}$$

to a threshold squared *ARE* loss function, the performance of which has not been assessed previously:

$$ARE_i^k = \frac{(f_k(\mathbf{x_i}) - y_i)^2}{y_i^2}. \tag{8}$$

From a forecasting perspective, the formulation and choice of loss functions seems to ignore established research findings for time series prediction. The relative errors suggested for AdaBoost.RT are similar in nature to MAPE, despite its documented biases in the asymmetry of over- versus under-forecasting, and issues with zero values (see for example the discussion by Hyndman & Koehler, 2006). The potential problems with observations close to zero are notable, with small absolute errors leading to large yet irrelevant percentage errors.

Similarly, the suggestion of using a squared or even higher-order error loss ignores the biases that have been found when using such error metrics in forecasting research (see e.g. Tashman, 2000), which would emphasise training on outliers with a higher squared error contribution. The formulation of the non-threshold loss function might induce further problems for non-stationary time series, where the errors from higher levels of the series are compared to those of lower levels. On the other hand, 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, there has been no systematic evaluation assessing the effects of different loss functions in boosting, i.e., linear, squared and exponential, and different loss function types, threshold versus non-threshold, on their relative merit for forecasting.

3.3. Stopping criteria

The variable β_k measures how much confidence we have in the predictions produced by model k, and is given by (for example, 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 is used by most boosting variants, e.g., AdaBoost.R2 (Drucker, 1997), Threshold-based Boosting (Avnimelech & Intrator, 1999), and Modified AdaBoost (Goh et al., 2003).

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

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

where 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 the 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., 2010). We refer to this as an unbounded stopping criterion. Shrestha and Solomatine (2006) observed that the combined prediction could still outperform the best individual prediction even when including models with $\bar{L} > 0.5$. In addition, Quinlan (1996) showed empirically that while it is possible to obtain better performance on training data with fewer boosting iterations and $\bar{L}_k \leq 0.5$

in some cases, AdaBoost generally performed better when more models were included.

As before, from a forecasting perspective, it does not seem to make sense to use a minimum average loss when interpreting the loss bound stopping criteria. Note that time series forecasting has no theoretical definition of a 'weak' learner (which achieves a binary classification accuracy of marginally greater than 50% on a balanced dataset), yet the performance requirement for base models in boosting for regression still relates to a stopping criterion that is taken from classification. In forecasting, however, a measure of the performance does not have a natural upper bound for interval-scaled observations, as the performance of the predictive algorithm will be bounded by the signal-to-noise ratio that is inherent in the data. As a consequence, we assess the choice of the 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 for $L_k > 0.5$ 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 et al. (2008) combine the final forecast f_c using a weighted median of all of the models' predictions f_k:

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

In contrast, the final model combination in AdaBoost.RT (Shrestha & Solomatine, 2006), BCC Algorithm (de Souza et al., 2010) and Modified Adaboost (Goh et al., 2003) is estimated using the weighted mean:

$$f_c(\mathbf{x}) = \sum_k \beta_k f_k(\mathbf{x}) \Big/ \sum_k \beta_k. \tag{12}$$

Similarly to forecasting research, substantial research has been invested in the area of machine learning in an attempt to establish a combination method for boosting, yet no consensus has been achieved. For example, Assaad et al. (2008) found that the weighted median combination gives a better predictive performance than the weighted average. Bone, Assaad, and Crucianu (2003) and Deng et al. (2005) both concluded that the weighted median is better than the weighted mean because it is less sensitive to outliers, while Avnimelech and Intrator (1999) found no statistically significant difference between the two. Surprisingly, boosting research with time series has neglected the simpler approach of equally weighted combinations (Kourentzes, Barrow, & Crone, 2014), with only threshold-based boosting (Avnimelech & Intrator, 1999) using a simple median for combination. This omission is deemed particularly important because the simple arithmetic mean is the method that is applied most widely in forecasting, making it a sensible benchmark. We propose to investigate both the weighted and equal-weighted means 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 model k currently being constructed, not on that of the combined predictions. However, it is possible that the training error of the combined model may drop to zero before other termination conditions are reached, or before the training error of any single model may drop to zero. It would be reasonable to assume that, in such situations, further iterations will increase the complexity, but not improve the performance on the training set data. In such cases, 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 for calculating the loss at each iteration using the combined model output up to each iteration respectively. We refer to this as the ensemble loss calculation, where we calculate the loss L_i^k of the combined model output at each iteration k, rather than the single model output created at 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 considered explicitly in past boosting studies, non-threshold-based estimation such as AdaBoost.R2 uses an unbounded stopping criterion, allowing many models to be added in order to obtain arbitrary ensemble sizes. As the size of an ensemble is often considered to be of significance in both forecasting and machine learning, we consider it to be another implicit meta-parameter, and evaluate two competing methodologies for determining the combination size.

The first is cross-validation early stopping. Under this scheme, each time series is split into three disjoint datasets, namely training, validation and test sets. If the addition of more models to the combination results in an increase in the MSE on the validation set, training stops; this being a type of early stopping to prevent overfitting. The second scheme includes a pre-specified number of models (50) in the final combination, without the use of a validation set. By assessing the final accuracy on a fixed number of observations that are withheld for testing, the additional observations become available for training. This choice of 50 models was based on both common practice in the literature, and observations of the convergence of the in-sample error training on a 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, which may allow better learning and permit the model to converge with the underlying data generating process.

3.7. Base model

Although boosting remains a generic algorithm, irrespective of the underlying base model used, and therefore the base model is not one of the archetypical meta-parameters, we discuss it briefly here in order to

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: The numbers in boldface correspond to the seven existing boosting variants that have already been introduced by prior research studies; those in italics are the 42 novel boosting variants derived from combinations of existing meta-parameters; and those underlined are the 47 novel boosting variants motivated by forecasting research.
= Algorithm Id; Mod = Model loss calculation; Ens = Ensemble loss calculation.

consider it in our systematic empirical evaluation. Traditionally, the predictive algorithm underlying boosting is required to be a weak learner (see also Section 3.3). In the absence of a formal performance criterion for a weak learner in forecasting, such as $L_k < 0.5$ 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 build up the complexity of the combined model adaptively, 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 rise quickly, even as early as the second iteration from overfitting, and the out-of-sample performance will become very poor. As a consequence, decision trees constitute the most widely boosted base models in classification, e.g., see Drucker (1997) and Shrestha and Solomatine (2006), with the popular classification and regression tree (CART) algorithm by Breiman (1984) being applied frequently. Regression trees are powerful, effective and easily interpretable, and are able to select relevant features automatically. However, CARTs have not been applied widely in forecasting, and the studies on boosting for time series regression have generally applied artificial neural networks. The popular multilayer perceptron (MLP), a class of universal approximator (Hornik, 1991), can facilitate weak or strong learning through architectures of different complexity, with the numbers of layers and hidden nodes being set accordingly. In the absence of an interpretation of 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. Descriptions of the setups of both base learners and details on their parameterisations are given in the Appendix.

3.8. Meta-parameter summary

The archetypical meta-parameters that we identified above, namely 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 meta-parameter realisations from the seven boosting variants, e.g., the weighted mean and weighted median combination methods, and introduce novel meta-parameter realisations based on forecasting research, e.g., the 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 the loss function and type, stopping criteria and combination method, and are shown in italics, while 47 novel boosting variants, underlined in Table 1, are motivated by forecasting research and distinguished by the use of the ensemble loss update method.

The remaining seven correspond to the seven algorithms proposed previously, and are highlighted in boldface. These are the AdaBoost.R2 by Drucker (1997) and its variants, labelled algorithms #78, #86 and #94 and which implement linear, squared and exponential loss with a non-threshold-based bounded loss function and a weighted median combination. AdaBoost.RT, proposed by Shrestha and Solomatine (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., 2010), and #50 to the original Threshold-based Boosting (Avnimelech & Intrator, 1999).

The meta-parameter framework has three further benefits. First, in addition to the seven existing AdaBoost variants, the combination of all of the meta-parameters proposed individually as innovations facilitates the creation of 89 novel boosting algorithms which have not been considered previously in the literature. Second, a complete enumeration of all 96 existing or potentially viable boosting algorithms in a balanced design allows not only a systematic assessment of their relative accuracies, 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. Third, combining the meta-parameter choices with the highest positive marginal contributions to the predictive accuracy allows us to propose a novel algorithm called Best Combination Boosting (AdaBoost.BC, #28), which is determined from the results of the empirical evaluation experiment outlined below.[2]

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

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 et al., 2011). Created as a subset of the renowned M3-competition, multiple empirical studies have established it as a valid, reliable, and representative benchmark dataset for computational intelligence (CI) methods. The dataset consists of a representative set of 50 long (L) and 50 short (S) monthly industry time series, which in turn contain an equal balance of 50 seasonal (S) and 50 non-seasonal (NS) patterns. The shortest of these time series is 68 months long, and the longest 144 months. Finally, the last 11 time series are series that exhibit particular complexities in their time series patterns. The balanced design of the dataset allows us to analyse the accuracy of the boosting algorithms across typical industrial data conditions, testing their relative performances across long vs. short and seasonal vs. non-seasonal series, and complex data patterns. Six sample time series are shown in Fig. 1.

4.2. Forecast performance evaluation

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

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

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 rankings of competing methods' forecast errors using the nonparametric Friedman test (Friedman, 1937, 1940) and the post-hoc Nemenyi test (Nemenyi, 1963), providing further evidence on the differences across meta-parameters and forecasting benchmarks methods. Neither test imposes assumptions regarding the distributions of the errors, and both are based on the mean ranking of the forecast errors of each method over each forecast origin. The Friedman test outputs a p-value that evaluates whether or not at least one of the methods is statistically different from the rest. The Nemenyi test outputs a critical distance that is based on the mean ranking of each method and depends on the number of methods, the sample size and the significance level. This is used to assess whether the methods have significantly different mean rankings, i.e., whether the rankings differ from each other by more than the critical distance.

In addition to test errors, which assess the true ex ante out-of-sample accuracy of the boosting meta-parameters, we also assess the 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 observations that remain 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 performances evenly across different forecasting methods and combination approaches, we always report the forecast errors on the 18 holdout observations as the test set error, the preceding 14 observations as the validation set error, and the remainder as the training error.[3]

4.3. The multifactorial ANOVA analysis

To quantify the impact and significance of each meta-parameter on the forecasting accuracy, we conduct a multifactorial analysis of the variance (ANOVA), with extended multiple comparison tests of estimated marginal means on the out-of-sample forecast errors. The experimental setup follows a balanced factorial design, with each meta-parameter in Table 1 being modelled as a different factor treatment of equal cell sizes. The loss function, loss function type, loss calculation method, combination method, stopping criteria, base model and combination size method are modelled as fixed main effects in order to test whether the factor levels show different linear effects on the forecast error with the test dataset. In addition to the test error, we also assess the impacts of training and validation errors on the dependent variables, to control for potential in-sample overfitting. A meta-parameter is considered to have a relevant impact on the forecast performance if it is consistently significant at the 0.01 level using Pillai's trace statistic,[4] as measured across all datasets. In addition, in order for a meta-parameter to be considered to have an impact on the out-of-sample performance irrespective of the

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

[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 of the test can be found in several advanced statistical texts, such as that by Foster, Barkus, and Yavorsky (2005).

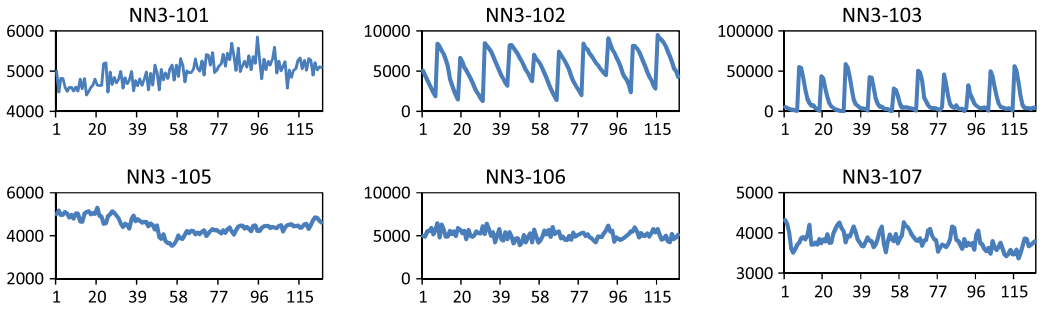

Fig. 1. Plot of 6 time series from the NN3 dataset.

in-sample data, it must also prove to be significant for the individual test set. We disregard a significant Box's test of the 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 impacts 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 to account for unequal variances in the factor cells.[5] The test allows us to evaluate the positive or negative impact on the forecast accuracy of a given factor for each factor level. We estimate marginal means across the training, validation and test datasets, and use $mm_{factorlevel} = \{\text{training; test}\}$ to represent the difference in the marginal mean forecast error between two factor levels (ignoring validation errors). A positive value (impact) indicates a decrease in the forecast error, and vice versa.

4.4. Ensemble benchmark methods

We assess the accuracy of boosting relative to those of other ensemble benchmarks and statistical forecasting methods. A popular predecessor to boosting, bagging (short for bootstrap and aggregating), is also based on the idea of forming multiple models that are estimated using different samples of the original training set, and then combining them (Breiman, 1996). Recent studies have applied bagging to a variety of forecasting applications with positive results (e.g., Hillebrand & Medeiros, 2010; Lee & Yang, 2006; Lin & Zhu, 2007; Stock & Watson, 2005). 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 also implement ensembles of MLP averaging over predictions from multiple initialisations of the same neural network (Hansen & Salamon, 1990). For this study, we utilise 50 initialisations of the connecting weights of an identical MLP architecture, and average their 50 predictions over the complete forecasting horizon from each time origin. For both MLP ensembles and bagging, the fixed size of 50 forecasts was found to provide adequate results for both the MLP and CART benchmarks. In addition, we also evaluated ensembles of MLP with the size of the ensemble determined through cross-validation, thus estimating error improvements from additional base models. Furthermore, we also compare all methods to simple model selection, where the base-learner with the lowest validation error is selected as the forecast, constituting the prediction of only a single 'best' model.

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 (the training, validation and test subsets combined) and for each subset separately, both across all base models and for MLP and CART separately across all data subsets.

The results confirm that the performances of boosting methods depend significantly on the choices of all meta-parameters (at the 0.01 level) both across all data (i.e., training, validation, and test combined) and for most data subsets individually: the 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 has no significant impact on the performance for either the validation or test datasets, and the choice of combination method has no significant impact on the test set error. A closer algorithmic examination indicates that 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 the training data versus selected insignificant effects on the validation and test data could indicate overfitting, and warrants further analysis.

[5] Tukey's post-hoc test is used to assess more detailed interactions in the MANOVA. Smaller ANOVA models are built using meta-parameters that have been found to have a significant impact on the 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, and linear and exponential, and provides mean differences between the performances of each factor level, together with a p-value to indicate whether or not the two differ significantly.

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 (two-tailed).
** Highly significant at the 0.01 level (two-tailed).

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 (two-tailed).
** Highly significant at the 0.01 level (two-tailed).

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

The loss function type, stopping criteria and combination size are highly significant across each of the five time series data conditions, but all of the other factors appear to be insignificant on one or two data conditions. More factors prove insignificant for shorter (seasonal) time series than for longer ones. 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 indicates the interaction between meta-parameters and time series patterns, pointing to the importance of analysis across different data conditions. These statistical significance results are relevant because they show that all of the meta-parameters identified in this study have a significant impact on the forecast accuracy of boosting, across data conditions and base learners, but that the impact varies. This not only justifies our research question and experimental design, but also calls into question the validity of studies which combine the meta-parameters of boosting (or other) algorithms arbitrarily without any systematic evaluation of their partial contributions.

Surprisingly, the insignificant effects of the combiner and loss function in boosting on the test error performance are in contrast to the majority of the forecast combination literature. A more detailed analysis of each factor treatment, i.e., the choice of combiners, including the mean, median, weighted mean and weighted median, might reveal possible explanations for the lack of influence due to the similarity of the selected approaches, and is conducted below.

5.2. Accuracy of the meta-parameter choice

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

The results show that the lowest overall test error is obtained using an unbound stopping criterion, 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 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 achieves lower training errors than the MLPs (8.92% vs. 13.41%), regardless of the choices of other factors, but vice versa on the validation (16.51% vs. 11.89%) and test (18.31% vs. 17.14%) sets. This indicates substantial in-sample overfitting by CART relative to MLPs, which is a novel insight for boosting on time series data, and is in contrast to the results obtained for bagging (Bauer & Kohavi, 1999; Breiman, 1996). In addition, we also noted that the CART errors are more sensitive to other meta-parameter choices than the MLPs',

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

Stopping	Model	Size	Loss update	Training		Validation		Test	
				Loss function type					
				Non-threshold	Threshold	Non-threshold	Threshold	Non-threshold	Threshold
CART	Bound	Cross-valid	Ens	9.00%	9.75%	16.78%	18.28%	18.95%	20.29%
			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-valid	Ens	8.89%	9.12%	16.60%	17.04%	18.58%	18.90%
			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-valid	Ens	13.14%	13.20%	14.04%	14.33%	17.15%	17.56%
			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-valid	Ens	13.76%	12.95%	14.79%	13.80%	17.69%	17.07%
			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 methods with the lowest forecast errors in each of the training, validation and test datasets are highlighted in boldface. The methods with the lowest forecast errors 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.

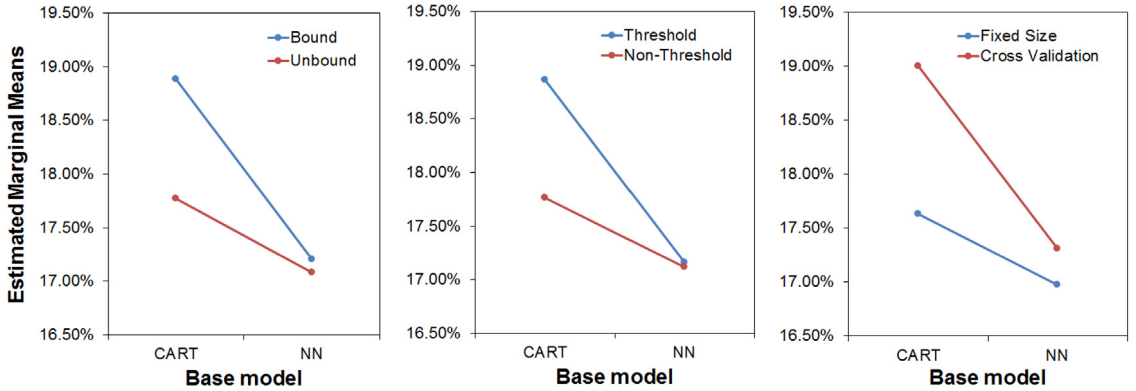

Fig. 2. Estimated marginal means plot for the test set SMAPEs of base models across all time series.

but with different factor treatments having only very small impacts. As there are no interaction effects between the base model of CART or MLPs and other meta-parameters (see Fig. 2), the results are consistent, clearly identifying CART as being inferior to MLPs as a base model for boosting across all data conditions of the time series in this dataset.

In comparison, less pronounced average deviations of forecast errors on the test data are seen for other factors, 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, though they are less pronounced for MLPs than CART, and indicate that these meta-parameter choices have very little impact on the empirical accuracy (irrespective of their statistical significance). However, the significant improvement in accuracy from using an unbound and non-threshold-based loss calculation with a fixed ensemble size, together with 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 that adds to the functioning of the boosting algorithm, we do not go into more detail in our analysis of the base learner. All of the other meta-parameters 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 interactions with other meta-parameters, we analyse the estimated marginal means of the forecast error for bounded and unbounded stopping across factor levels for the loss function type, loss function and combination size in Fig. 3. (As the results are consistent across data properties, these graphs are omitted.)

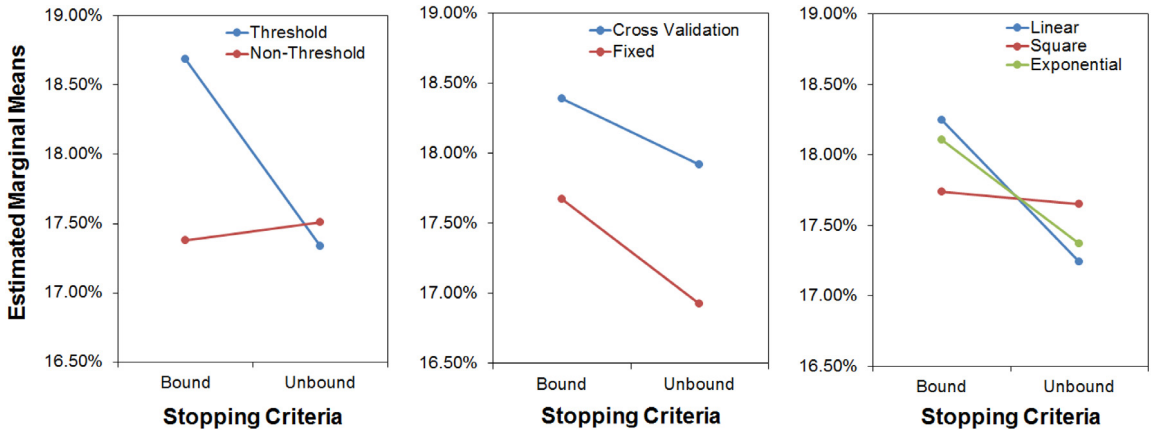

Fig. 3. Estimated marginal mean plots of the test set error across all time series for various factors.

The analysis of marginal means shows that there is a significant difference in mean forecast errors between threshold and non-threshold based estimation for bounded errors, with non-threshold loss functions producing lower forecast errors, $mm_{non\text{-}threshold} = \{0.007; 0.013\}$. In contrast, the non-threshold loss functions for unbounded errors show $mm_{non\text{-}threshold} = \{-0.004; -0.002\}$ to have a negative impact. Given a bounded stopping criterion, a squared loss function outperforms both the linear and exponential loss functions significantly, with estimated improvements of $mm_{squared} = \{-0.002; -0.004\}$ and $mm_{squared} = \{-0.001; -0.005\}$ 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 criteria, rendering the previous interaction irrelevant. Given these interactions, the type and choice of the loss function used depend significantly on the stopping criteria enforced. A non-threshold-based, squared loss function is significantly better when the average loss is bounded, but a linear (or exponential) loss function is preferred when there is no bound on the average error, with threshold-based and non-threshold-based estimation providing similar low errors. This holds true across both the MLP and CART base learners. Again, fixed size combinations are better than a 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 because they indicate that, following the implementation of bounded errors and weak learner in classification, using bounded errors 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 hold equally 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 the forecast performance

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

The results indicate that fixed size estimation consistently has lower errors across all other meta-parameters without any interaction effects. In addition, the difference in performance between non-threshold and threshold loss types is more pronounced for fixed size combination. Taken together with the results in Table 4, we observe that this is particularly true when the stopping criteria are bounded, that is, when the combination size is smaller. In this case, the effect of early stopping cross-validation is to make the combinations even smaller by stopping early, suggesting that larger combinations are not only better, but required in order 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 outperform the simple mean or simple median significantly. This confirms the findings from forecasting research, where no significant improvements have been found from applying more complex, weighted approaches to forecast combination. As such, it also calls into question the estimation of the β_k parameter that is 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, and both 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, though different data conditions do have an impact on the meta-parameter choice, a fixed size combination is sufficient, since it is computationally less demanding than cross-validation early stopping, and not less accurate (see Table 5).

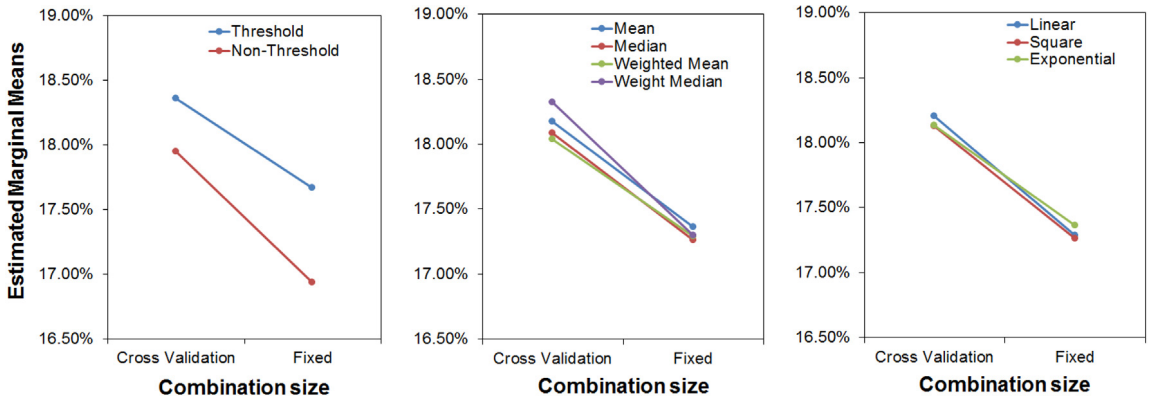

Fig. 4. Estimated marginal mean plots of test set errors across all time series for various factors.

Table 5
Test set SMAPE vales based on the combination size method across time series types.

	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 the forecast performance

To summarise our analysis of meta-parameters for boosting, MLP base models outperform CART models significantly (on this dataset). Fixed size forecast combinations outperform cross-validation early stopping significantly, and including models with an average loss of $\bar{L} > 0.5$ 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 is in contrast to the findings in classification, where the standard boosting practice of a bounded average error is enforced and shown to be successful (Avnimelech & Intrator, 1999; Dietterich, 2000; Quinlan, 1996), indicating the uniqueness of time series regression and the potential for developing specific boosting meta-parameters. Surprisingly, the choice of a loss function is not statistically significant, although it is marginally more accurate (across all base models); however, the use of a linear loss function seems most plausible, due to its simplicity, robustness and parsimony. For combination, the widely-used weighted median combination method is significantly inferior to all other combination methods for out-of-sample forecast accuracy; instead, the smallest errors come from the mean and the weighted mean, in concurrence with forecast combination research. Overall, the impacts of most of the AdaBoost meta-parameters depend on the base model. Thus, while statistically significant improvements in forecast accuracy can be obtained from the right choice of meta-parameters, the choice of the MLP or CART base learner must be considered first.

As a further result of our balanced study of existing and novel meta-parameters, we identify the factor realisations that have the greatest influence on the accuracy and derive a novel AdaBoost variant as a combination of these, also drawing on insights from forecast combination research. As a consequence, the novel boosting algorithm 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 performance relative to other benchmark algorithms, such as bagging, statistical forecast combination, or simple model selection, also needs to be evaluated, and this is documented in the next section.

6. Empirical results on forecast accuracy

6.1. Relative performances of forecast combination algorithms

Following a systematic assessment of the impacts of different meta-parameters, we assess the relative empirical accuracies of the two most popular AdaBoost variants, AdaBoost.R2 and AdaBoost.RT, both of which originally employed CART models, and the novel AdaBoost.BC algorithm using a MLP base model. To facilitate an assessment across base learners, both the MLP and CART base models are applied across all AdaBoost variants. Their performances are compared with those of established forecast combination benchmarks, including bagging of MLPs, ensembles of MLPs (which average predictions over multiple initialisations of network weights) with the number of MLPs to be combined being determined using either a fixed size ($\text{Ensemble}_{\text{Fixed}}$) or cross-validation early stopping ($\text{Ensemble}_{\text{Cross-valid}}$), 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; thus, we provide 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 performances using SMAPE, mean ranks on SMAPE, and group ranks on SMAPE, based on statistically significant differences using the non-parametric Friedman and Nemenyi tests, with the results being sorted by descending mean rank on the test SMAPE.

The results show that all approaches that combine predictions, including ensembles, bagging, and all boosting variants, outperform the selection of an individual, 'single best' base model significantly. Model selection using MLP ($Select_{MLP}$) and CART ($Select_{CART}$) turn out to have the largest percentage errors, at 18.47% and 20.17% respectively, and are ranked 6th and 7th accordingly after all other methods. This is strong evidence that forecast combination outperforms selection, and is in line with previous findings on forecast combination in the statistical and econometric literature, where combination is found to routinely outperform attempts to select a single best model.

Second, the novel $AdaBoost.BC_{MLP}$ algorithm proposed in this paper significantly outperforms all other existing boosting variants (in addition to the 89 additional boosting variants that are developed in this paper but not assessed here, due to their limited marginal benefits), including the popular algorithms of $AdaBoost.R2_{CART}$ and $AdaBoost.RT_{CART}$. Assessing both the MLP and CART base learners, our study suggests that using MLPs to implement AdaBoost.R2 and AdaBoost.RT would increase the accuracy of these established benchmark methods significantly, but that they would still perform worse than AdaBoost.BC. As AdaBoost.BC outperforms AdaBoost.R2 and AdaBoost.RT across both base models, it provides evidence of both the gains from our appropriate choice of meta-parameters, and 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 poor validation and test set errors, confirming earlier findings on in-sample overfitting for decision trees. Similarly, all forecast combination methods using a neural network MLP base learner perform significantly better overall than equivalent 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 – like those of all boosting variants – falls short of those of bagging and the even simpler ensemble methods. MLP ensembles, whether using fixed or cross-validation sizes, show the lowest forecast errors (15.86% and 15.88% respectively) and rank equal first in performance, followed by $Bagging_{MLP}$ and $Bagging_{CART}$ (with 16.03% and 16.02% test errors) ranked equal second, and only then followed by boosting variants in three groups of accuracy, led by third-ranked AdaBoost.BC. Reflecting on the relative complexity and approaches of the combination algorithms, the simplest approaches of MLP ensembles utilise all data, equally weighted, and draw only on the diversity that is inherent in the neural network initialisations for combining forecasts of MLPs. This consistently outperform both algorithms which sample (i.e., weight) observations randomly to create additional diversity, and those that create an even further diversity of the base models by directing the weight to hard-to-predict observations. It appears that an increase in diversity does not lead to an added accuracy, raising the issue of determining adequate levels of diversity for different dataset conditions. However, it should be noted that, despite the rank-based evidence that AdaBoost.BC (ranked 3rd) cannot routinely outperform bagging with either MLPs or CART (ranked equal 2nd), the average SMAPE over 10 runs of the AdaBoost.BC (15.94%) is indeed lower than that of bagging (with 16.03% and 16.02% respectively), making it the 3rd most accurate algorithm based on SMAPE error. While this indicates that the error distributions are somewhat skewed and the rank-based measures are adequate, it does suggest that selected AdaBoost variants may indeed be able to outperform alternative forecast combination approaches, given the dataset.

Given the multifactorial evaluation in which we assess the seven existing and 89 novel boosting candidates, our results suggest that the properties of the boosting algorithm per se might lead to an inferior forecasting performance. Furthermore, both bagging and ensembles appear to rank highly, regardless of the choice of base model or ensemble configuration, making them more robust for 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 or ensembles, both of which are simpler approaches with less complexity, increased transparency and greater efficiency. Thus, our results confirm the findings of other forecasting studies, that more complex algorithms may be unable to outperform simpler ones (Crone et al., 2011), providing some evidence of the external validity of our findings. Any additional findings, such as our ability to enhance existing algorithms further based on 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 the future of 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 only to irrelevant improvements in empirical accuracy.

6.2. Performances relative to those of NN3 competition contenders

In addition to considering the most reliable average performances across 10 replications, we also 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 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 et al., 2011). As the NN3 competition

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

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

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

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

User ID#	Method or contestant name	SMAPE	SMAPE rank, all methods	SMAPE rank, only NN/CI
B09	Wildi	14.84	1	–
B07	Theta	14.89	2	–
–	***Ensemble*$_{\textbf{Fixed}}$**	**15.07**	**3**	**1**
C27	Illies	15.18	4	2
B03	ForecastPro	15.44	5	–
–	**Bagging$_{\textbf{MLP}}$**	**15.68**	**6**	**3**
–	**AdaBoost.BC$_{\textbf{MLP}}$**	**15.76**	**7**	**4**
B16	DES	15.90	8	–
B17	Comb S-H-D	15.93	9	–
B05	Autobox	15.95	10	–
–	**AdaBoost.R2$_{\textbf{MLP}}$**	**15.98**	**11**	**5**
–	**Ensemble$_{\textbf{Cross-valid}}$**	**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$_{\textbf{CART}}$**	**16.77**	**18**	**7**
–	**Select$_{\textbf{MLP}}$**	**16.80**	**19**	**8**
B00	AutomatANN	16.81	20	9
–	**Bagging$_{\textbf{CART}}$**	**17.10**	**21**	**10**
–	**Adaboost.RT$_{\textbf{MLP}}$**	**17.58**	**22**	**11**
–	**AdaBoost.BC$_{\textbf{CART}}$**	**17.94**	**23**	**12**
–	**AdaBoost.R2$_{\textbf{CART}}$**	**18.19**	**24**	**13**

Note: Forecast errors in boldface indicate the performances of the methods evaluated in this study. Forecast errors in boldface and underlined indicate the best performing computational intelligence method.

guidelines required, we forecast 18 steps ahead from a fixed origin, with the last 18 observations as a holdout test dataset. The SMAPE and relative ranking results are provided in Table 7, with the boosting algorithms assessed highlighted in bold.

The novel AdaBoost.BC algorithm ranks seventh overall across all methods (i.e., NN/CI and statistical algorithms), and fourth on NN/CI algorithms, making it one of the best performing of the algorithms developed in machine learning for time series prediction. AdaBoost.BC is outperformed only marginally by bagging with MLPs, and is better than Comb S-H-D, a combination of the simple (SES), Holt (HES) and damped (DES) (deseasonalised) exponential smoothing forecasts. It also outperforms the individual statistical benchmarks of HES and SES. The simpler fixed-size ensemble method of MLP model averaging ranks first amongst all computational intelligence algorithms, outperforming the Illies' ensembles of echo state neural nets with pooled training, as well as 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 sixth amongst computational intelligence methods. When interpreting the relative accuracy, it should be noted that Table 7 shows only the top statistical contenders of the M3 competition (Makridakis & Hibon, 2000) and 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 of the 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 relative to established statistical forecasting methods, and even

selected statistical benchmarks for forecast combination, which should help to motivate future research in the area.

7. Conclusions

This paper systematically evaluates the empirical forecast performances of boosting algorithms for forecasting real-world industry time series. The empirical evaluation follows a valid and reliable experimental design, using the 111 time series of the NN3 competition with ex post evaluation across multiple rolling origins, and comparing the accuracies of all boosting variants to the 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 of the choice of base model. The results of a multifactorial analysis of the variance after decomposing boosting into archetypical meta-parameters indicate that the choice of the loss function type, loss update, stopping criteria, base model and combination size all have (statistically) significant impacts on the forecast accuracy of boosting algorithms. The most important of these meta-parameters are the choices of a loss function and the method of determining the number of models to be included in the forecast combination. As we would expect from forecasting research, many of the meta-parameters proposed in this study outperform those from boosting in classification, including the use of a simple, equally weighted mean as the combination method, an unbounded 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 influential determinants.

The experimental results show that the appropriate selection of meta-parameters leads to an improved forecast accuracy, and allows us to derive a boosting variant with the best combination of meta-parameter choices, called AdaBoost.BC, which outperforms all other boosting variants that have been developed for time series prediction to date. Our findings also indicate that standard meta-parameter choices may lead to suboptimal performances, meaning that the existing AdaBoost.RT and AdaBoost.R2 algorithms can be improved marginally by adjusting the meta-parameters. However, these have little impact, as both ex post model averaging using ensembles of neural networks and bagging outperform the best performing boosting variants, in spite of the accuracy gains that can be achieved for boosting by a careful selection of the meta-parameters (in contrast to previous findings, e.g., Avnimelech & Intrator, 1999; Shrestha & Solomatine, 2006). The findings therefore suggest that there is a 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 holding only for the dataset conditions that 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, cover a large segment of the dataset properties used in the industry to date, they cannot be deemed representative for daily or intraday data, for example. 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 within the structure of the meta-parameter framework developed here. However, the study remains representative for the majority of the boosting algorithms used in forecasting, which are derived largely from AdaBoost. Finally, our experiment covers only meta-parameter features found in the boosting literature or motivated by forecasting research. Undoubtedly, many more could be considered, so as 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 bounded interval $[0, 2]$, where the highest SMAPE error could never exceed 200%. This would also establish some congruency with the objective function, developing the algorithm using the metric it will ultimately be assessed on. More importantly, theoretical advances in estimating the necessary conditions for a weak learner in boosting for regression in general, and time series prediction in particular, could suggest what base models to combine, and how to specify them. A potential starting point for defining a base learner is that it has a 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 the relative improvement. Overall, our study has shown the benefits of combining insights from forecasting research on combination with the development of machine learning algorithms, making this a promising approach for guiding future research on forecast combination.

Appendix. Base learner parameterisation

Here, we describe the setup of each base learner and provide details on the parameterisation. We specify the MLP architecture by constructing a MLP with a single hidden layer with only two hidden nodes, using a hyperbolic tangent activation function and a linear output function (Zhang, Patuwo, & Hu, 1998). This limits the network architecture to very few degrees of freedom, in accordance with the weak learner requirement of boosting (Bühlmann & 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 for modelling monthly stochastic seasonality and potentially stochastic trends, although the NN3 dataset does not contain any time series with significant trends. All networks use a single output node with the identity activation function, and each MLP is trained directly on each time series, being scaled linearly to the interval $[-0.5, 0.5]$ to facilitate training, but without prior differencing or data transformation, so as to estimate the level, seasonality, and potential trend in the network weights and the bias terms directly. For parameterisation, the data are 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, & Beale, 1996), and the MSE is minimised up to a maximum of 1000 epochs. The algorithm requires a scalar μ_{LM} and its increase and decrease steps to be set, and we use $\mu_{LM} = 10^{-3}$, 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 the error on the validation set increases or remains the same for more than 50 epochs. In addition, network training also stops if μ_{LM} exceeds $\mu_{\max} = 10^{10}$. The network weights that give 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 when training. Identical initial weights are used across all meta-parameter choice combinations when training the MLP for boosting, to allow 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 CART algorithm is used. In contrast to traditional methods such as ordinary least squares (OLS) regression and discriminant analysis, CART does not depend on assumptions of normality or user-specified model statements. The resulting predictors are typically simple functions of the corresponding input variables, and are often easy to both use and interpret. A detailed review of the effective use of regression trees in the context of boosting is given by Schapire and Singer (1999). When the tree is constructed, the predicted value for input $\mathbf{x_i}$, corresponding to the terminal node, will be assigned to that node. The tree is built such that, at each stage, the split selected is the one that leads to the greatest reduction in the sum of the squared error between the actual values of the training set examples that correspond to a particular node, and their sample mean, which is the predicted value. The tolerance on the sum of squared errors is set to 10^{-6}, such that node splitting 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)).

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 and Data Analysis*, *50*(11), 3032–3052.

Audrino, F., & Bühlmann, P. (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.

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., & Yu, B. (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. (1988). Combining forecasts. *European Journal of Operational Research*, *33*(3), 223–229.

Canestrelli, E., Canestrelli, P., Corazza, M., Filippone, M., Giove, S., & Masulli, F. (2007). Local learning of tide level time series using a fuzzy approach. In *Proceedings of IEEE international joint conference on neural networks* (pp. 1813–1818).

Chan, C. K., Kingsman, B. G., & Wong, H. (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 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., Pozo, A., da Rosa, J. M. C., & Neto, A. C. (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. In *Proceedings of international conference on machine learning, Vol. 6* (pp. 3528–3534).

Diebold, F. X., & Pauly, P. (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. (1983). Estimating the error rate of a prediction rule—improvement on cross-validation. *Journal of the American Statistical Association*, *78*(382), 316–331.

Elliott, G., Granger, C. W. J., & Timmermann, A. (Eds.) (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 of the Japanese Society for Artificial Intelligence*, *14*(5), 771–780.

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.

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.

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.
Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*(2), 251–257.
Hyndman, R. J., & Koehler, A. B. (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.
Lee, T. H., & Yang, Y. (2006). Bagging binary and quantile predictors for time series. *Journal of Econometrics*, *135*(1–2), 465–497.
Leung, M. T., Daouk, H., & Chen, A.-S. (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. In *Advanced intelligent computing theories, Vol. 2* (pp. 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., & Hibon, M. (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. In *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., Tutz, G., & Hothorn, T. (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 and Data Analysis*, *53*(12), 4046–4072.
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, & J. Kittler (Eds.), *Multiple classifier systems*. Berlin: Springer-Verlag Berlin.
Stock, J. H., & Watson, M. W. (2004). Combination forecasts of output growth in a seven-country data set. *Journal of Forecasting*, *23*(6), 405–430.
Stock, J. H., & Watson, M. W. (2005). *An empirical comparison of methods for forecasting using many predictors*. Princeton University.
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.
Weigend, A. S., & Gershenfeld, N. A. (1993). Results of the time series prediction competition at the Santa Fe Institute. In *IEEE international conference on neural networks, Vol. 3* (pp. 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.
Zhang, F. (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.

Devon K. Barrow is a Lecturer in the Department of Strategy and Applied Management at Coventry Business School. He received his B.Sc. in Mathematics and Computer Science from University of the West Indies, an M.Sc. in Computer Science from Canterbury University, and a Ph.D. from Lancaster University Management School. His research focuses on time series prediction with neural networks and statistical methods, with a particular emphasis on forecast combination and model selection.

Sven F. Crone is an Assistant Professor of Management Science at Lancaster University Management School and the deputy director of the Lancaster Research Centre for Forecasting. He received his Diplom-Kaufmann (B.BA. and M.BA. equivalent) and Ph.D. from Hamburg University, Germany, with research fellowships in South Africa and the USA. His research focuses on forecasting, time series prediction and data mining in business applications, frequently employing methods from Computational Intelligence such as neural networks and support vector machines. His research has been published in the *European Journal of Operational Research, Journal of Operational Research Society* and *International Journal of Forecasting*. Sven is the competition chair of the IEEE CIS Data Mining Technical Committee and has organised the 2007 Neural Network Forecasting Competition (NN3) co-sponsored by the IIF, NSF and SAS, the 2008 NN5 and the current 2009 IEEE Grand Challenge on Time Series Prediction with Computational Intelligence.