ARTICLE IN PRESS

International Journal of Forecasting xxx (xxxx) xxx

Contents lists available at ScienceDirect

International Journal of Forecasting

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

LoMEF: A framework to produce local explanations for global model time series forecasts

Dilini Rajapaksha [a,*], Christoph Bergmeir [a], Rob J. Hyndman [b]

[a] Department of Data Science & Artificial Intelligence, Monash University, Clayton VIC 3800, Australia
[b] Department of Econometrics & Business Statistics, Monash University, Clayton VIC 3800, Australia

ARTICLE INFO

Keywords:
Local interpretability
Time series forecasting
Global models
Bootstrapping
Explainability

ABSTRACT

Global forecasting models (GFMs) that are trained across a set of multiple time series have shown superior results in many forecasting competitions and real-world applications compared with univariate forecasting approaches. One aspect of the popularity of statistical forecasting models such as ETS and ARIMA is their relative simplicity and interpretability (in terms of relevant lags, trend, seasonality, and other attributes), while GFMs typically lack interpretability, especially relating to particular time series. This reduces the trust and confidence of stakeholders when making decisions based on the forecasts without being able to understand the predictions. To mitigate this problem, we propose a novel local model-agnostic interpretability approach to explain the forecasts from GFMs. We train simpler univariate surrogate models that are considered interpretable (e.g., ETS) on the predictions of the GFM on samples within a neighbourhood that we obtain through bootstrapping, or straightforwardly as the one-step-ahead global black-box model forecasts of the time series which needs to be explained. After, we evaluate the explanations for the forecasts of the global models in both qualitative and quantitative aspects such as accuracy, fidelity, stability, and comprehensibility, and are able to show the benefits of our approach.

1. Introduction

We are living in an era of big data. Big data in the field of time series forecasting do not usually mean a single long time series. There are many applications and databases which consist of large quantities of time series from similar sources in the same domain. In this context, forecasting based on traditional univariate time series forecasting procedures leaves great potential untapped. To address this issue, many works have recently explored the capabilities of global forecasting models (GFMs, Januschowski et al., 2020) that are trained across all available time series. Most notably, the M4 forecasting competition (Makridakis, Spiliotis, & Assimakopoulos, 2018) was won by a particular type of globally trained recurrent neural network (RNN) (Smyl, 2020), the M5 competition (Makridakis, Spiliotis, & Assimakopoulos, 2022) was dominated by globally trained LightGBM models, earlier Kaggle competitions in this space were also dominated by globally trained models (Bojer & Meldgaard, 2021), and companies like Amazon (Salinas, Flunkert, Gasthaus, & Januschowski, 2020), Walmart (Bandara, Shi, Bergmeir, Hewamalage, Tran, & Seaman, 2019), Uber (Laptev, Yosinski, Li, & Smyl, 2017), and others are relying on such models.

However, the premise existing for decades in the forecasting space (most notably as a conclusion of competitions such as the M and M3 competitions (Makridakis

* Correspondence to: Faculty of Information Technology, Monash University, P.O. Box 63, Victoria 3800, Australia.
E-mail addresses:
Dilini.Rajapakshahewaranasinghage@monash.edu (D. Rajapaksha),
Christoph.Bergmeir@monash.edu (C. Bergmeir),
Rob.Hyndman@monash.edu (R.J. Hyndman).

https://doi.org/10.1016/j.ijforecast.2022.06.006
0169-2070/

Please cite this article as: D. Rajapaksha, C. Bergmeir and R.J. Hyndman, LoMEF: A framework to produce local explanations for global model time series forecasts. International Journal of Forecasting (2022), https://doi.org/10.1016/j.ijforecast.2022.06.006.

& Hibon, 2000)), that simple models are at least as accurate as more complex ones, has been rendered obsolete by such GFMs, allowing them to be considerably more complex while maintaining good generalisation and extrapolation capabilities (Montero-Manso & Hyndman, 2021). In this sense, forecasting is following other areas, such as medicine (Kaushik et al., 2020), where complex machine learning models are often more accurate than the relatively simple and interpretable predictive models established in these domains (Nanayakkara, Fogarty, Tremeer, Ross, Richards, Bergmeir, et al., 2018). However, machine learning models usually have the drawback of being black-box models that are not interpretable, leading to their slow adoption, as practitioners do not trust the models and are therewith often hesitant to rely on them in production settings. Some notable exceptions exist with machine learning models that are designed with interpretability in mind. One such model in the forecasting space is NBEATS, proposed by Oreshkin, Carpov, Chapados, and Bengio (2020), which is a deep learning model that can decompose the series into human-interpretable components (e.g., trend and seasonality). These approaches are valuable but tied to particular deep-learning methods and particular explanations. Thus, a more flexible approach in machine learning is the concept of *local interpretability* that has been developed to explain individual predictions rather than a global interpretation across all predictions.

Local interpretability is usually achieved by fitting an interpretable surrogate model locally in the neighbourhood of a particular instance for which we want to explain the output of a more complex black-box model that we will call in this context the (black-box) background model. In this way, local explanations for a specific instance can be more accurate and relevant, as opposed to global explanations that do not change across the dataset. In general, to capture the generalisation behaviour of the background model, or if no training data are available, the surrogate model is trained on new instances that are generated within the neighbourhood. Options are to perturb features or interpolate them between instances from the training set. Then, predictions from the background model on these new instances are obtained, and the surrogate model is trained to mimic those predictions. One of the first works in this space was the method of local interpretable model-agnostic explanations (LIME, Ribeiro, Singh, & Guestrin, 2016b). Here, the newly generated instances are randomly perturbed around the instance to be explained, and a linear model is fitted as a surrogate model to the instances in a selected neighbourhood of the prediction that needs to be explained. The linear model is assumed to be interpretable in terms of its coefficients. Later, Lundberg and Lee (2017) introduced Shapley additive explanations (SHAP), a local interpretability algorithm based on game theory that provides consistent additive explanations for a prediction generated by the black-box model. Both LIME and SHAP generate explanations that identify the features which positively or negatively influence the prediction of the background model of an instance. Recent literature has also focused on rule-based explainers, which provide arguably more interpretable and insightful explanations than the explanations based on feature importance. The most relevant works in this line are anchor-LIME (Ribeiro, Singh, & Guestrin, 2018), local rule-based explanations (LORE, Guidotti, Monreale, Ruggieri, Pedreschi, Turini, & Giannotti, 2018), and local rule-based model interpretability with k-optimal associations (LoRMIkA, Rajapaksha, Bergmeir, & Buntine, 2020).

In this paper, our aim is to apply the concept of local interpretability to the forecasting domain, to provide interpretability comparable to the traditional forecasting approaches for the forecasts generated by black-box GFMs such as RNNs. Here, we aim to generate these explanations by developing a procedure to fit traditional local forecasting approaches in a way that mimics the behaviour of a GFM for a particular time series. As surrogate models, we use traditional local statistical models, such as exponential smoothing, ARIMA, and others. We assume that these models are interpretable in terms of decompositions like trend and seasonality, relevant lags, and others. Therewith, we facilitate the adoption of GFMs in practice, bridging the gap in terms of interpretability between them and the traditional, well-established techniques.

In the global forecasting context, we treat (the last observations of) a single time series from the set of all series as an instance to be explained. Following the local interpretability approach, we then need to define a neighbourhood and a way to sample from the neighbourhood, to obtain data on which to train a surrogate model. Approaches to define the neighbourhood that we explore include defining a similarity measure between series, or simply to define the neighbourhood as the time series under consideration. Then, sampling from the neighbourhood can be done with a bootstrapping procedure, or if the forecasting model does not overfit or grossly underfit the data and the fit is therewith somewhat representative, sampling can even be omitted and the series used directly. We show in our work that the latter option is not only a simpler procedure but also leads to better explanations in many situations. We evaluate our local explainer methodology empirically on five different datasets with nine different local explainers to verify the fidelity, accuracy, stability, and comprehensibility of the local explainers. As such, we show that the local explainers are viable techniques to explain global forecasting models locally.

The remainder of this paper is structured as follows. Section 2 presents our approach. Section 3 discusses the experimental setup, Section 4 discusses computational performance, and Section 5 discusses the results in both qualitative and quantitative aspects. Section 6 concludes the paper and discusses future work.

2. Our approach

The proposed framework—local model explanations for forecasting (LoMEF)—is designed to make global forecasting models as interpretable as local statistical forecasting approaches, which are often inherently explainable white-box models, or offer a certain degree of

explainability through interpretable components such as trend and seasonality. As such, the main focus of our work is not to argue that these local models are explainable, but rather to lift GFMs to the same level of explainability as these simpler widely used models. Therewith, LoMEF enables forecasting practitioners to use more complex and more accurate global forecasting models, while continuing to communicate the trustworthiness of their forecasts using their most convenient statistical forecasting model. In particular, LoMEF complements more basic generic approaches of residual diagnostics or in-sample forecasting characteristics that offer more generic insights into a model's behaviour with a method that can be used to explain a particular forecast. Furthermore, the framework offers solutions for cases where the global models overfit, with bootstrapping approaches.

In this section, we first formally define the concept of local interpretability in general, and then introduce our proposed approach of local interpretability in forecasting with GFMs. Finally, we discuss the GFMs and local explainers employed in this study in detail.

2.1. Local interpretability in regression and classification

Let $X \in \mathbb{R}^{n \times m}$ be a training dataset consisting of n instances and m features, and let $\boldsymbol{z}$ be an n-dimensional vector of corresponding target values, which can be either numerical in a regression setting or categorical in a classification setting. Further, let g be a given black-box background model that produces predictions $\hat{z}$ from m-dimensional input vectors $\boldsymbol{x}$ (i.e., $g(\boldsymbol{x}) = \hat{z}$), so that an error criterion between z and $\hat{z}$ is minimised. The goal of local interpretability algorithms is to find a surrogate model e (e.g., a linear model), such that for a given instance $\boldsymbol{x}$ for which the prediction $\hat{z}$ of the background model is to be explained, $e(\boldsymbol{x}) \approx \hat{z}$.

To build e, we define a neighbourhood for $\boldsymbol{x}$ using a distance function, and now we can create a set of randomly generated instances that lie in the neighbourhood. We call this new set $\tilde{X}$. We then apply g to all instances in $\tilde{X}$ to obtain predictions $\tilde{z}$. With this, e is trained on $\tilde{X}$ to resemble $\tilde{z}$ as the target predictions. To generate $\tilde{X}$, many algorithms first filter the original training set X for instances that lie in the neighbourhood of $\boldsymbol{x}$, into a set that we call X_{filt}. Then, $\tilde{X}$ is generated from X_{filt} using a sampling procedure that perturbs single instances from X_{filt} by adding noise to the features and/or interpolates between instances of X_{filt}.

2.2. Local interpretability in forecasting

Transferring the local interpretability concept to the forecasting domain, as the background model, we want to use a GFM f that has been trained on a set of time series $Y = \{y_1, \ldots, y_p\}$, where p is the total number of time series available. Now, let us consider the predictions $\hat{y}$ of a particular time series y, obtained from the GFM as $f(y) = \hat{y}$. We want to build a local explainer e as one or more traditional forecasting models, such as ETS, ARIMA, and theta, such that $e(y) \approx \hat{y}$. Thus, it is straightforward to use the original time series y as the neighbourhood, as the local explainer e is able to be fitted on a single time series. We then need to find a sampling procedure from the neighbourhood.

Let us first consider the case where f is a purely autoregressive model that has no internal state and only uses lagged values as inputs, such as a feedforward neural network. Then, we are in a similar situation to the regression case discussed in Section 2.1. We can now first assume a procedure that does not sample any data, and all data that we use are data from the training set that is within the neighbourhood. That is, $\tilde{X}$ and X_{filt} are equal. In the forecasting context, this means that the target values we use to fit e are the predictions of the GFM along with the time series, i.e., the in-sample fit of f on y. As we want to use time series models as local explainers, we choose also the input for e to be the in-sample fit (instead of the original series). Thus, a first approach is to simply fit e on the in-sample fit of f on y. In this work, we use the one-step-ahead in-sample fit of f, even if f produces multi-step forecasts. Fig. 1 illustrates the one-step-ahead in-sample fit of the global model. Here, each red dot indicates a one-step-ahead forecast or the fit of the global model. Note that this approach is easily extendable to GFMs that are not purely autoregressive.

This approach of not sampling data has the potential drawback that the surrogate model cannot assess the generalisation capabilities of the global model, which can be a problem, especially if the global model is overfitting. In extreme cases, depending on the GFM, the fit cannot bear any meaningful information. For example, it can be identical to the original time series if the GFM produces a perfect in-sample fit. However, motivated by the findings of Montero-Manso and Hyndman (2021) that GFMs typically lead to worse fitting than per-series models but generalise better, in this paper we first explore this simple solution without sampling. In the remainder of the paper, we call this the neighbourhood-fit (NF) method.

Then, we propose a bootstrapping technique to generate samples from the neighbourhood to overcome the potential limitation of the NF approach that the local model cannot assess the generalisation behaviour of the global model if the global model is overfitting the training data. The next section discusses the neighbourhood generation methods based on bootstrapping. In what follows, we call this the neighbourhood-bootstrap (NBoot) method. Thus, in this work, we define two main approaches of explainer methods based on their neighbourhood generation procedures. An overview diagram of the proposed framework is shown in Fig. 2.

2.3. Neighbourhood generation with bootstrapping

Bootstrapping time series has the difficulty that the series can be non-stationary and have autocorrelation. Non-stationarity can be addressed by time series decomposition, as proposed in the work of Bergmeir, Hyndman, and Benítez (2016), or by using sieve bootstrapping procedures (Bühlmann, 1997), e.g., as done by Cordeiro and Neves (2009). The autocorrelation in the series can be addressed with the moving block bootstrap (MBB, Kunsch, 1989), or again with the sieve bootstrap, fitting a model

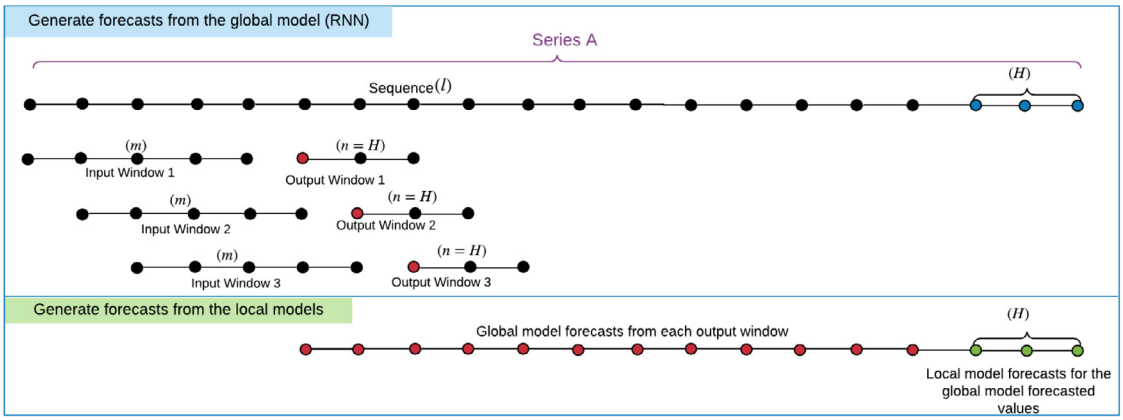


Fig. 1. A diagram that shows how to extract the fit of the global model.

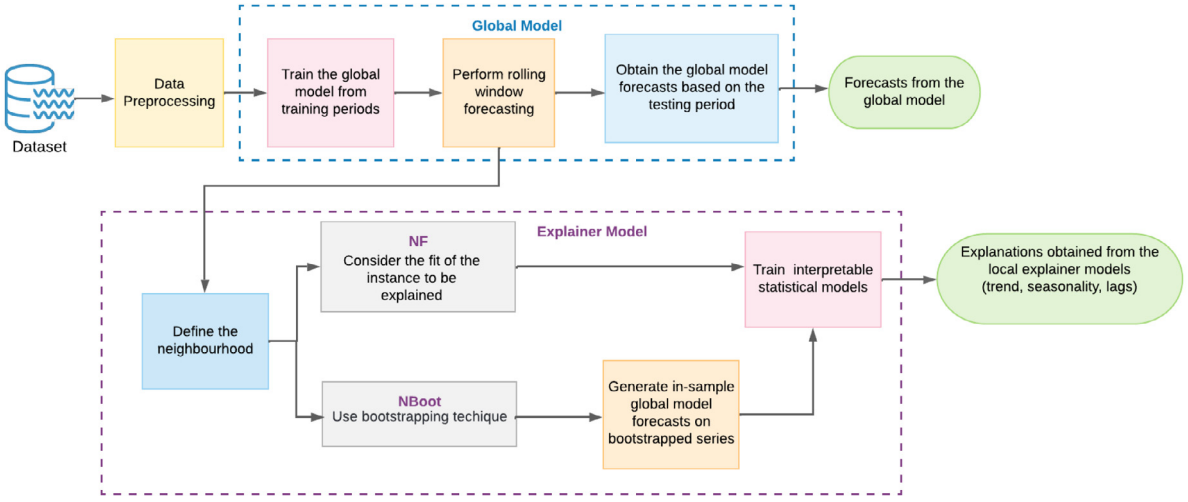


Fig. 2. An overview diagram of the framework to generate local explanations for the global model predictions.

and then bootstrapping the residuals, assuming that they are uncorrelated. Also, more sophisticated bootstrapping procedures have been introduced for time series, such as the tapered block bootstrap (Paparoditis & Politis, 2001), the extended tapered block bootstrap (Shao, 2010b), the dependent wild bootstrap (DWB, Shao, 2010a), or the linear process bootstrap (LPB, McMurry & Politis, 2010). However, in this work, we focus on common and established approaches based on MBB, decomposition, and sieve bootstrapping.

In particular, we use the procedure from Bergmeir et al. (2016) which uses the MBB technique together with STL decomposition. Here, a remainder is extracted after performing the STL decomposition on the original series. Then, the decomposed trend and seasonality are added to the bootstrapped series. In what follows, we call this approach the neighbourhood STL-MBB (NSTL) method. Another possibility we use is a sieve bootstrapping procedure where we fit a model and then bootstrap the residuals. Similar to the NSTL approach, the MBB technique is used to generate the new set of remainders. We call this approach neighbourhood sieve-MBB (NSieve) in the following. For NSTL we use the `bld.mbb.bootstrap` function from the `forecast` package in R (Hyndman et al., 2022; Khandakar & Hyndman, 2008).

After generating the neighbourhood as bootstrapped series, we generate in-sample global model forecasts on these series, and fit our local explainer models to these. Note that, when considering the local explainer methods using bootstrapping procedures, one local explainer model is built per bootstrapped time series, so that it is necessary to select local explainers where the explanations can be summarised across the bootstrapped series.

2.4. Global model forecasting procedure

Many different types of global forecasting models have been proposed recently, from the earlier works of Smyl and Kuber (2016), Wen, Torkkola, Narayanaswamy, and Madeka (2017), Salinas et al. (2020), and Bandara, Bergmeir, and Smyl (2020), to the winning method of the M4 by Smyl (2020), to Oreshkin et al. (2020) and others. As our work is agnostic towards the global model used, we aim for a relatively standard long short-term memory (LSTM) network

implementation, following the guidelines of Hewamalage, Bergmeir, and Bandara (2021). We detail the exact global model that we use in the following.

Recurrent neural networks with long short-term memory cells. RNNs are neural networks that have a feedback loop to propagate an inner state (Elman, 1990), which makes them appropriate for sequential modelling tasks such as natural language processing or speech recognition. LSTMs are a special type of RNN that solve problems of vanishing and exploding gradients in traditional RNNs by gating mechanisms, which enable them, together with the feedback loops, to capture non-linear long-term temporal dependencies in sequences. They have recently gained popularity in the forecasting space, in particular because the winning method of the M4 competition (Smyl, 2020) is based on LSTMs. Following the guidelines of Hewamalage et al. (2021), we use a stacked architecture combined with LSTMs with peephole connections (Gers, Schmidhuber, & Cummins, 2000) and perform a number of preprocessing steps.

Data preprocessing. Training and validation periods are chosen according to Hewamalage et al. (2021). We reserve a separate section from each time series as the validation period to perform hyperparameter tuning. The global LSTM model is built across a group of time series and may contain observations in different value ranges. Consequently, it may be beneficial to perform a normalisation to scale the observations. We perform a mean scaling transformation as the normalisation strategy, which divides the whole time series y_i by its corresponding mean value, as indicated in Eq. (1):

$$y_{i,normalised} = \frac{y_i}{\frac{1}{T_i}\sum_{t=1}^{T_i} y_{i,t}} \tag{1}$$

Thereafter, we take the logarithm of each series to stabilise the variance in the time series. As the time series in our experiments are all non-negative and may contain zero values, we apply the log transformation, as defined in Eq. (2):

$$y_{i,log\_scaled} = \begin{cases} \log(y_i), & \text{if } \min(Y) > 0, \\ \log(y_i + 1), & \text{if } \min(Y) = 0 \end{cases} \tag{2}$$

Here, Y is the whole set of time series, and y_{i,log_scaled} is the time series i after logarithmic scaling. As such, if any of the series used for modelling contains a zero, all series are incremented by one before applying the logarithm. Otherwise, the logarithm is applied directly.

In our approach, we use Fourier terms as additional inputs (Hyndman & Athanasopoulos, 2021), where a set of sine and cosine terms are used to model the seasonality of the time series (Harvey & Shephard, 1993). Following Bandara, Bergmeir, and Hewamalage (2021a) we train the RNN model together with the preprocessed time series and Fourier terms, where the Fourier terms are added as an external variable to capture the periodic effects of the time series.

Assume $y_{i,t}$ is an observation of time series i in a particular point of time t. Eq. (3) defines the approach of approximating the periodic terms for the observation $y_{i,t}$:

$$\sin\left(\frac{2\pi kt}{s_1}\right), \cos\left(\frac{2\pi kt}{s_1}\right), \ldots, \sin\left(\frac{2\pi kt}{s_n}\right), \cos\left(\frac{2\pi kt}{s_n}\right) \tag{3}$$

Here, n is the number of seasonalities in the time series, such that s_n represents the nth seasonality of time series y_i, and $k = 1, \ldots, K$, so that there are K sine and cosine pairs for each seasonality. The variable K controls the regularity of the periodic patterns, and we determine its value using a grid-search algorithm within the range of 1 to 25.

We use the multi-input multi-output (MIMO) strategy when producing forecasts. Here, we transform a time series y_i into multiple pairs of $(T_i - n - m)$ records, where T_i is the length of series y_i, n is the length of the input window, and m is the length of the output window. Thus, each record has $(m + n)$ observations. The MIMO strategy produces forecasts for the whole output window at once. Hence, it avoids the accumulation of error in each forecasting step of the time series when training neural networks. Following the aforementioned procedure, to train the global model, we use $(T_i - m)$ observations from time series y_i and reserve the last output window as the validation period for parameter tuning.

Global model training. The RNN models we use have several hyperparameters that need to be tuned: namely, the cell dimension, mini-batch size, maximum number of epochs, epoch size, number of hidden layers, L2-regularisation weight, standard deviation of random normal initialiser, and standard deviation of the Gaussian noise. In line with Hewamalage et al. (2021), we perform automatic hyperparameter tuning using sequential model-based algorithm configuration optimisation (SMAC, Hutter, Hoos, & Leyton-Brown, 2011). We use continuous coin betting (COCOB) as the learning algorithm to select an optimal learning rate when training the global model (Orabona & Tommasi, 2017). The model generates the final forecasts after training the model using the optimal hyperparameters. To address parameter uncertainty due to the randomly chosen initial weights and the stochastic nature of the optimisation procedure of the NN, we train on ten different Tensorflow graph seeds and average the resulting forecasts that are then used as final forecasts in the evaluation, effectively building an ensemble of 10 models. Although the hyperparameter tuning has associated uncertainty, we do not re-tune the hyperparameters for separate seeds, as it would be computationally too expensive. The initial hyperparameter ranges of the RNN model and their optimal values are shown in Tables F.17 and F.18 in Appendix F.

Postprocessing global model forecasts. The forecasts are postprocessed as follows. First, we take the exponential of the values to reverse the log transformation. Note that the mean in the log-transformed scale is the median when transformed back to the original scale of the data, and no bias adjustment is performed (Hyndman & Athanasopoulos, 2021; Kolassa, 2020), so that the forecasts considered

in this work are usually medians of the forecasting distribution. Then, we subtract 1 if the original dataset contains zero values. Afterwards, we de-normalise by multiplying the forecasts of every series by its corresponding mean. Then, if the original dataset that we consider contains integer values, we round the forecasts to the nearest integer. This is not a necessity for the forecasts, but we assume that integer forecasts could be required in these use cases. Finally, if the original dataset that we consider contains only non-negative values, we convert all negative values to zero.

2.5. Traditional local forecasting models used as local explainer models

In this section, we briefly discuss the forecasting models that we use as surrogate explainer models. A more detailed description of these methods and our way to interpret them can be found in Appendix A.

The traditional models we consider in this work are exponential smoothing (Brown & Meyer, 1961), seasonal and trend decomposition using loess (STL, Cleveland, Cleveland, McRae, & Terpenning, 1990), the multiple STL (MSTL) algorithm (Bandara, Hyndman, & Bergmeir, 2021c), linear autoregressive models (ARs, fitted locally; we also consider a global version often called pooled regression, PR), dynamic harmonic regression ARIMA (DHR_ARIMA, Hyndman & Athanasopoulos, 2021), a trigonometric Box–Cox ARMA trend seasonal (TBATS) model (De Livera, Hyndman, & Snyder, 2011), Prophet (Taylor & Letham, 2018), and the theta model (Assimakopoulos & Nikolopoulos, 2000). Here, we assume that these traditional statistical forecasting models are interpretable through components such as relevant lags, trend, seasonality, and other attributes. In particular, exponential smoothing decomposes a time series into trend, seasonality, and remainder when producing forecasts. STL decomposition is an approach to decompose time series with a single seasonality additively into seasonality, trend, and remainder. Similarly, MSTL can be used to decompose time series with multiple seasonalities. TBATS also performs a decomposition of the series into complex seasonal patterns, trend, and irregular components, and Prophet provides human-interpretable parameters such as decomposed seasonalities, trend, and holiday effects. For all these methods, the decomposed components can be used as interpretable components when explaining forecasts.

For exponential smoothing, another interpretable characteristic is the chosen model—for example, if a model with an additive or multiplicative trend or seasonality is chosen, etc. Similarly, for the AR, PR, and DHR_ARIMA models, their coefficients and other model parameters, such as the amount of differencing, the model order, or the amount of Fourier terms, can be considered as the interpretable components when explaining the forecasts. In this work, we use PR as a local explainer in the NBoot approach. Hence, it produces explanations that are still local to the time series that need to be explained. Finally, as the theta method is identical to simple exponential smoothing with drift (Hyndman & Billah, 2003), it can be considered explainable due to its simplicity.

3. Experimental setup

In this section, we describe the benchmark datasets, local benchmark models, error metrics, and the evaluation criteria of the proposed LoMEF framework used for our experiments.

3.1. Datasets

The aim of our paper is to explain predictions of global models in situations where the global models have high accuracy and are preferred over traditional univariate models. Based on previously reported results (Bandara et al., 2020; Hewamalage et al., 2021) and preliminary experiments not reported here, we selected five main benchmark time series datasets on which global forecasting models perform well. A brief description of the benchmark datasets is as follows.

- Ausgrid-Energy half-hourly dataset (AusGridHH): This dataset consists of data related to the energy consumption of 300 households in Australia (AusGrid, 2019). In our experiments, we select data related to general consumption (GC letter code in the dataset), which is one of the primary energy consumption categories. We extract a subset of half-hourly data from July to October 2012.
- Ausgrid-Energy weekly dataset (AusGridW): This dataset is an aggregated version of the Ausgrid-Energy half-hourly dataset. Before aggregation, we omitted one series which has missing values for more than eight consecutive months. Thus, this dataset contains 299 weekly series. We use the full period for which data is available, namely from July 2010 to June 2013.
- San Francisco Traffic hourly dataset (SFTrafficH): This dataset contains the hourly traffic data on San Francisco Bay Area freeways from 2015 to 2016 (Caltrans, 2020).
- NN5 weekly dataset (NN5W): This dataset is an aggregated version of the NN5-daily dataset used in the NN5 forecasting competition and consists of data on cash withdrawals from ATMs in the UK (Ben Taieb, Bontempi, Atiya, & Sorjamaa, 2012). The missing values of the original daily dataset on a particular day are replaced by the median across the same days of the week along the whole series. Then the daily time series are aggregated into weekly values.
- Kaggle Web Traffic daily dataset (WebTrafficD): This dataset was used in the Kaggle Wikipedia Web Traffic forecasting competition (Google, 2017). It consists of the daily web traffic (number of hits) of around 145,000 Wikipedia pages. Following the configurations used in the work of Hewamalage et al. (2021), in our experiments, we use the first 997 series from the dataset, for the period of July 1, 2015 to December 31, 2016. Unlike the other datasets in our study, this dataset is count data. As this may be a requirement for certain applications (though a forecast as a mean does not have to be a count, even for count

Table 1
Dataset statistics.

Dataset	No. of series	Length	Seasonal cycles	Horizon
NN5W	111	105	52	8
AusGridW	299	148	52	8
WebTrafficD	997	549	No	60
SFTrafficH	963	726	(24, 168)	24
AusGridHH	300	4704	(48, 336)	96

Table 2
Benchmark models for the datasets.

Dataset	Benchmark models
NN5W	TBATS, DHR_ARIMA, Prophet, PR
AusGridW	TBATS, DHR_ARIMA, Prophet, STL_ETS
WebTrafficD	ETS, theta, PR
SFTrafficH	TBATS, Prophet, DHR_ARIMA, MSTL_ETS
AusGridHH	TBATS, Prophet, DHR_ARIMA, MSTL_ETS

data), we round the final forecasts to the closest non-negative integer. Furthermore, as the dataset does not exhibit seasonality, in this dataset we do not use Fourier terms with the RNN.
- Kaggle Web Traffic100 daily dataset (WebTraffic100D): For the experiments of the NBoot approach, randomly sampled records were selected from the WebTrafficD dataset, since a considerable amount of memory is required when dealing with the full dataset.

Table 1 shows summary statistics of the datasets used in this study. We note that all series have equal lengths within a dataset. Similar to the heuristics proposed by Bandara et al. (2020) and Hewamalage et al. (2021), we chose an input window size of 1.5 × horizon, assuming that the horizon has been chosen in relation to the autocorrelation structure and seasonality of the series.

3.2. Benchmark models

We use ETS, TBATS, DHR_ARIMA, Prophet, STL_ETS, theta, and MSTL_ETS as the benchmark models.

Table 2 illustrates the benchmark models used for each dataset. Note that the benchmark models (local models) are trained on the actual data, in contrast with the same models used as local explainers trained on the respective sampled data for explanation.

The forecasting literature has established that the performance and accuracy of forecasting models will usually heavily depend on the characteristics of the dataset (Petropoulos, Makridakis, Assimakopoulos, & Nikolopoulos, 2014). This enables practitioners to make informed decisions on method selection when facing new data. We argue that these approaches can be translated directly to the case when selecting the best explainer model among multiple explainers, by considering the input data of the local explainers as the underlying data from which to extract characteristics. In our experiments we use a simplified procedure and select the local models and the local explainer models for each dataset, considering the length of the series, number of seasonal periods, and type of seasonality. For example, seasonal series use seasonal local models and seasonal local explainers, and non-seasonal series use non-seasonal local models and non-seasonal local explainer models, accordingly. In our experimental setup, for the non-seasonal datasets like the WebTrafficD dataset, we use non-seasonal ETS and theta as the local models and the local explainers. To forecast the time series with complex seasonalities we use TBATS, Prophet, and DHR_ARIMA in the NN5W, AusGridW, SFTrafficH, and AusGridHH datasets. Moreover, we categorise the benchmark models based on the number of seasonalities in the dataset. We use MSTL_ETS for hourly datasets with multiple seasonalities, and we use STL_ETS for weekly datasets with a single seasonality.

3.3. Error metrics

To measure the performance of our models, we use three different error metrics commonly used in the forecasting space, namely, the mean absolute scaled error (MASE, Hyndman & Koehler, 2006), root mean squared error (RMSE), and mean absolute error (MAE). The MASE is a standard scale-free error measure common in forecasting, used in, e.g., the M4 (Smyl, 2020) competition. The MAE and RMSE are scaled measures that therewith avoid many of the problems that scale-free measures have, and we use them as our datasets have the same scale across different series. Eqs. (4), (5), and (6) define the MASE, RMSE, and MAE measures, respectively (Hyndman & Koehler, 2006). Though some works in the literature argue against the use of different error measures together (Kolassa, 2020), we use the measure for which the forecasts have been optimised (RMSE as we use quadratic loss) as the primary error measure, and report the other measures for sanity checking and to assess how much accuracy is lost under these other measures.

$$\text{MASE} = \frac{\frac{1}{h}\sum_{t=M+1}^{M+h}|\hat{y}_t - y_t|}{\frac{1}{M-S}\sum_{t=S+1}^{M}|y_t - y_{t-S}|} \tag{4}$$

$$\text{RMSE} = \sqrt{\frac{1}{h}\sum_{t=1}^{h}(\hat{y}_t - y_t)^2} \tag{5}$$

$$\text{MAE} = \frac{1}{h}\sum_{t=1}^{h}|\hat{y}_t - y_t| \tag{6}$$

Here, y_t and $\hat{y}_t$ denote the observation and the forecast at time t, respectively, M is the number of data points in the training series, S is the seasonality of the dataset, and h is the forecast horizon.

3.4. Evaluating the local explainers

When it comes to evaluating the local explainers, a question is what are the desirable qualities of such explainers. Following the general literature on local explainability (Carvalho, Pereira, & Cardoso, 2019; Yang & Kim, 2019), we evaluate our local explainers along the following dimensions.

- Fidelity – How well does the explainer approximate the forecast of the GFM?
- Comprehensibility – How interpretable are the explanations?
- Stability – How consistent are the explanations of the explainer across multiple runs?
- Accuracy – How well does an explainer forecast unseen data?

Of these, fidelity and comprehensibility are arguably the most important properties. Fidelity ensures that the explainer actually explains the relevant prediction and can be trusted. Comprehensibility ensures that the explanations can be understood and are therefore useful. Fidelity and accuracy are closely related in the sense that if the GFM is accurate and the explainer has fidelity, the explainer predictions are accurate. However, if the GFM is inaccurate, the explainer should follow the GFM and try to achieve high fidelity at the cost of accuracy. As such, accuracy becomes irrelevant in these situations (Molnar, 2018). Furthermore, there is a general tradeoff between accuracy and interpretability in machine learning (Ribeiro, Singh, & Guestrin, 2016a), which is overcome by local post hoc explainers, such as the ones used in our paper, by changing it effectively to a tradeoff between interpretability and local accuracy. Nonetheless, a simpler model will arguably be more comprehensible but may need to tradeoff fidelity. Similarly, the stability of a model is closely related to the variance (Molnar, 2018), and through the well-known bias–variance tradeoff (Hastie, Tibshirani, & Friedman, 2009), simpler explainers will tend to be more stable, and complex explainers and explainers that have non-deterministic components, such as a neighbourhood sampling step, will have less stability.

To perform the evaluations, we first calculate error measures among the actual observations, global model forecasts, local statistical benchmark model forecasts, and the local explainer model forecasts, as follows. Here, $Error(y_t, \hat{y}_t)$ can be one of MAE, RMSE, or MASE.

- *Error(global, explainer)*: Error between the global model forecasts and forecasts generated from the local explainer model.
- *Error(actual, global)*: Error between the actual observations and the forecasts generated from the global model.
- *Error(actual, local)*: Error between the actual observations and the forecasts generated from the local statistical model.
- *Error(global, local)*: Error between the global model forecasts and the forecasts generated from the local statistical model.
- *Error(actual, explainer)*: Error between the actual observations and the forecasts generated from the local explainer model.

Based on these primary error measures, we can then calculate secondary error measures as follows:

$$Fidelity\_Actual = Error(global, explainer) - Error(global, actual) \quad (7)$$

Fidelity_Actual, as defined in Eq. (7), verifies whether the local explainer forecasts are closer to the GFM forecasts than the GFM forecasts to the actuals. Thus, a value of *Fidelity_Actual* of less than zero shows that the explainer model resembles the behaviour of the global model on the given time series better than the global model is able to predict the actuals. As such, this measure is a rather indirect approximation of the performance of the explainer, as it depends on the accuracy of the global model and is more useful for accurate GFMs than for models that do not perform well. It can be seen as a form of sanity checking, where positive values should be used for further investigation but do not necessarily indicate a problem.

$$Fidelity\_Local = Error(global, explainer) - Error(global, local) \quad (8)$$

Fidelity_Local, as defined in Eq. (8), assesses whether the local explainer forecasts are closer (indicated by a negative value of the measure) to the GFM forecasts than the local model forecasts to the GFM forecasts. This measure is arguably the most important fidelity measure among those used in our work. It shows whether the explainer actually resembles the GFM, compared with a similar model trained directly on the data without the involvement of the GFM. The main drawback of this measure is that it does not directly compare the local model with the explainer, and as such only measures whether the local model or the explainer is relatively closer to the global model. There may be situations when both the local model and the explainer are far away from the global model (but close to each other). Especially in situations when the global model overfits the data, the explainer will be trained on data close to the original data, and therefore the explainer model will be close to the local model. This measure may often still show negative values even in this situation, as long as the explainer is relatively closer to the global model than the local model, even though both are far from the global model.

$$Fidelity\_with\_Explainer = Error(global, explainer) - Error(local, explainer) \quad (9)$$

Eq. (9) shows the formula to verify whether the local explainer is closer to the global model than the local model. If the value is less than zero, the explainer model is closer to the global model than the local model, and vice versa, and therefore a good explainer. The *Fidelity_with_Explainer* measure evaluates the fidelity of the explainer by examining whether the explainer is mimicking the global model better than the local model, regardless of the accuracies of the global and local models. As such, it is able to indicate problems with overfitting, in contrast to the *Fidelity_Local* measure. The drawback of this measure is that if the local model forecasts and the explainer model forecasts are on different sides of the global model, this measure may still show negative values, even though the explainer is further away from the global model than the local model is from the global model.

Hence, all fidelity measures defined thus far indicate important properties of the explainer, and the different fidelity measures enable us to evaluate different aspects

of the fidelity of the local explainer. Thus, we use an average over all of them to achieve a general indication of fidelity, as shown in Eq. (10):

$$Fidelity\_Average = (Fidelity\_Actual + Fidelity\_Local + Fidelity\_with\_Explainer)/3 \quad (10)$$

Next, we define measures of accuracy for the local explainers. Accuracy is arguably less important, as the main goal of a local explainer is to explain well, not to predict well.

$$Acc\_Global\_LocalModel = Error(actual, global) - Error(actual, local) \quad (11)$$

Eq. (11) shows the formula to verify whether the global model performs better than the local model. If the value is less than zero, the global model performs better than the local model, and vice versa. This measure does not directly evaluate the local explainer, but for a complete picture and to put the other evaluations into context, it is necessary to know whether the global model outperforms the local models.

$$Acc\_Explainer\_LocalModel = Error(actual, explainer) - Error(actual, local) \quad (12)$$

Acc_Explainer_LocalModel, defined in Eq. (12), assesses whether the local explainer performs better than the local model when forecasting. If the value is less than zero, the explainer model performs better than the local model, and vice versa. Thus, *Acc_Explainer_LocalModel* evaluates the accuracy of the explainer compared to the local model. We expect negative values as the results of this measure when the global model is performing better than the local model. In other words, we expect negative results for this measure when the results of the *Acc_Global_LocalModel* measure are negative.

$$Acc\_Explainer\_GlobalModel = Error(actual, explainer) - Error(actual, global) \quad (13)$$

Eq. (13) shows the formula to assess whether the local explainer performs better than the global model. If the value is less than zero, the explainer model performs better than the global model, and vice versa. In other words, *Acc_Explainer_GlobalModel* evaluates the accuracy of the explainer compared to the global model.

To understand the broader view of the model performance, we further calculate the mean and median values across time series of the per-series performance measures (*Fidelity_Actual*, *Fidelity_Local*, *Fidelity_with_Explainer*, *Acc_Global_LocalModel*, *Acc_Explainer_LocalModel*, and *Acc_Explainer_GlobalModel*). In our results both mean and median values across time series of the per-series performance measures are similar. Hence, we here report the mean values of the performance measures, and the median values are listed separately in the supplementary materials. These measures evaluate the fidelity and accuracy dimensions of the local explainers. For further clarity, the fidelity measures are summarised in Fig. 3.

To evaluate the stability of the local explainer models, we calculate the *Error*(*global*, *explainer*) over independent runs of the bootstrapping explainer method, as well as the NF method, and afterwards measure the stability of these errors with boxplots and the interquartile range (IQR) over these runs. We note that, as the NF method executes the local explainer on only a single time series, stability here effectively measures the stability of the local forecasting model.

To evaluate the comprehensibility of the explainer models, we follow a functionally grounded evaluation procedure (proxy tasks) in our experiments (Doshi-Velez & Kim, 2017). This technique can be applied when the class of local explainer models has already been validated as being interpretable, e.g., using a user study or, as in our case, assuming that these models are well established in practice for interpretations.

Furthermore, we show many examples of how the models can be used for interpretations in practice, e.g., with components such as trend, seasonality, model coefficients, and other characteristics of the models. Those are considered as a proxy for the quality of the explanations, and a qualitative evaluation with examples is given in Section 5.2.

Table 3
Execution times for the NN5 weekly dataset.

Dataset	No. of series	RNN	Prophet	DHR_ARIMA	TBATS
NN5W	111	994	40	20	420

Times are in seconds.

4. Computational performance

The computational performance of our framework heavily depends on the execution time of the local explainer models. While oftentimes the notion in forecasting is that local models are trained much faster than global models, this is mainly true for single series and less so when forecasts on a large set of series are required (for some timing results along these lines, see e.g., Godahewa, Bergmeir, Webb, Hyndman, & Montero-Manso, 2021). Generally, the idea of local explainability is that the explanations are only generated for a few specific series where explanations are needed. Hence, in our LoMEF framework, the local model is used as a local explainer only in the neighbourhood of the single intended series, where the execution time of the local explainer model is relatively low. Table 3 illustrates the average time taken for an RNN global model and Prophet, DHR_ARIMA, and TBATS local models for the NN5 weekly dataset in the experiments of Bandara, Hewamalage, Liu, Kang, and Bergmeir (2021b) and Godahewa et al. (2021). Those experiments were run on an Intel(R) Core(TM) i7-8700 processor (3.2 GHz) with 65 GB of main memory.

The proposed NF approach considers the fit of the global model as the neighbourhood of the series to be explained. Hence, the time taken to execute the Prophet, DHR_ARIMA, and TBATS local explainer models on the fit of the global model will be similar to the execution time of the local models mentioned in Table 3. For example, if we consider Prophet as the local explainer, the total time to produce the RNN forecasts for the whole dataset would

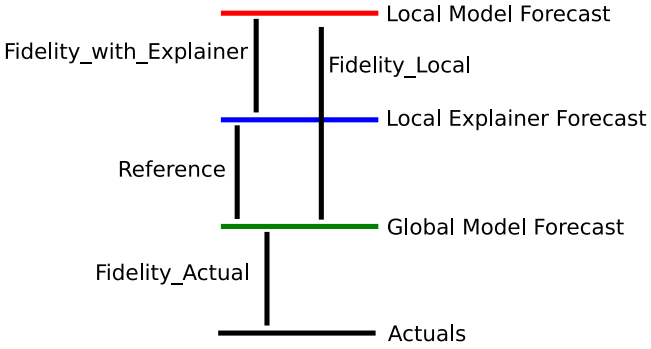


Fig. 3. Illustration of the different fidelity measures. The fidelities all calculate an error between the respective forecasts/actuals, and then subtract their error from the "Reference" error between the GFM and the local explainer. As such, if the measures are negative, this indicates the desired property that the reference error is smaller than the error currently under consideration. The *Fidelity_Average* measure is an average of all the measures, indicating average performance against the reference.

Table 4
Mean performance measures of the NF approach based on the RMSE measure for datasets on different local explainer models.

Dataset	Local Explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	Prophet	**−9.154***	**−3.152***	**−1.908**	**−4.738***	**−0.739**	**−0.222**	0.517
	DHR_ARIMA	**−10.353***	**−3.311***	**−2.233***	**−5.299***	**−0.422**	**−0.450**	**−0.028**
	TBATS	**−11.119***	**−4.206***	**−1.787**	**−5.704***	**−3.041***	**−2.111***	0.931
AusGridW	Prophet	**−9.170***	**−2.950***	8.039	**−4.081***	**−4.352**	**−2.209***	2.143
	DHR_ARIMA	**−11.223***	**−3.658***	5.289	**−3.197***	**−3.088**	**−1.689**	1.399
	TBATS	**−12.602***	**−2.231**	2.104	**−4.243***	**−3.248**	**−0.741**	2.508
	STL_ETS	**−9.967***	**−5.769***	0.678	**−5.019***	**−6.435***	**−4.240***	2.194
WebTrafficD	ETS	**−20.205**	**−9.778***	**−8.909***	**−12.964***	**−4.706**	**−4.708**	**−0.002**
	Theta	**−20.322***	**−6.033***	**−5.365***	**−10.573***	**−2.248**	**−2.198**	0.050
SFTrafficH	TBATS	**−0.007***	**−0.008***	**−0.006***	**−0.006***	**−0.003***	**−0.004***	**−0.001***
	Prophet	**−0.006***	**−0.007***	**−0.004***	**−0.006***	**−0.008***	**−0.006***	0.002
	DHR_ARIMA	**−0.006***	**−0.009***	**−0.005***	**−0.006***	**−0.005***	**−0.007***	**−0.001***
	MSTL_ETS	**−0.002**	**−0.006***	**−0.003***	**−0.003***	**−0.006***	**−0.004***	0.002
AusGridHH	TBATS	**−0.136***	**−0.156***	**−0.133***	**−0.142***	**−0.088***	**−0.090***	**−0.002**
	Prophet	**−0.106***	**−0.178***	**−0.136***	**−0.140***	**−0.120***	**−0.110***	0.010
	DHR_ARIMA	**−0.119***	**−0.079***	**−0.026***	**−0.074***	**−0.029***	**−0.029***	0.000
	MSTL_ETS	0.013	**−0.135***	**−0.169***	**−0.097***	**−0.186***	**−0.085**	0.101

Asterisks indicate statistically significant results, and boldface font indicates negative values (which are desirable).

be 994 s, whereas the generation of an explanation for a single series would take approximately $(40/111) \approx 0.36$ s. In contrast, in the NBoot approach, the execution time taken to explain the forecast of a single series is the time taken to fit the local explainer model per series times the number of bootstrapped series generated when defining the neighbourhood of the series to be considered. If, for example, we use 100 bootstrapped time series, in the example above, the explanation would take 36 s for a single time series.

5. Results and discussion

In this section, we first discuss the quantitative results of our study, measuring the quality of the local explainers in terms of the fidelity, accuracy, and stability dimensions of the explainer models. Then, we provide examples of explanations of the local explainers for a qualitative assessment to examine the quality of local explainer models in terms of their comprehensibility. Finally, we provide a brief discussion on the approach of selecting local explainer models according to the datasets.

Note that the reported results for the NBoot approach are from the NSTL method. In our experiments, the NSTL explainer method outperforms the NSieve method. Since both of these methods demonstrate the potential benefits of generating neighbourhoods with bootstrapping, we report the results of the best-performing method, which is NSTL under the NBoot approach. The other results related to the NSieve approach are provided in the supplementary materials.

5.1. Quantitative results

In the following, we discuss the quantitative results of the NF and NBoot explainer methods.

5.1.1. Evaluating the accuracy and fidelity of the explainer methods

Table 4 shows the mean values of the performance measures *Fidelity_Actual*, *Fidelity_Local*, *Fidelity_with_Explainer*, *Fidelity_with_Explainer*, *Acc_Explainer_Global Model*, *Acc_Explainer_LocalModel*, and *Acc_Global_LocalModel* relevant to the NF explainer method

Table 5
Mean and median performance measures based on RMSE measures of the datasets on different local explainer models using the NBoot approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
				Mean values				
NN5W	PR	**−6.755***	0.967	**−0.266**	**−2.365**	**−2.018**	**−1.865**	0.501
	ETS	–	–	–	–	–	–	–
WebTraffic100D	PR	**−14.156***	**−1.547***	**−1.661***	**−5.788***	**−0.399**	**−0.416**	**−0.017**
	ETS	**−15.282***	**−10.474***	**−10.391***	**−12.049***	**−4.773**	**−4.824**	**−0.050**
AusGridW	PR	**−10.922***	0.852	2.353	**−2.572 ***	**−4.780***	0.974	5.754
	ETS	–	–	–	–	–	–	–
				Median values				
NN5W	PR	**−6.443***	**−0.079**	0.918	**−2.411**	**−0.946**	**−0.626**	0.115
	ETS	–	–	–	–	–	–	–
WebTraffic100D	PR	**−9.302***	**−0.753***	**−0.957***	**−3.846***	0.052	0.075	0.000
	ETS	**−10.186***	**−2.946***	**−3.027***	**−6.097***	**−0.018**	**−0.030**	0.000
AusGridW	PR	**−6.459***	0.748	**−1.051**	**−2.474***	**−2.807***	0.840	5.538
	ETS	–	–	–	–	–	–	–

Asterisks indicate statistically significant results, and boldface font indicates negative values (which are desirable).

for the RMSE measure. As the results of the explainer methods based on the bootstrapping procedure, Table 5 shows the mean and median values of the performance measures relevant to the NBoot approach for the RMSE measure. Note that for the NBoot approach, no results are reported for the NN5W and AusGridW datasets with the ETS explainer, as ETS cannot be applied to weekly datasets due to their long seasonal cycles. In the following, we discuss the results related to the RMSE. The results for the MASE and MAE measures are similar and lead to the same conclusions. They are reported in Tables B.7, B.8, C.9, and C.10 in the Appendix.

Considering the NF explainer method, all of the mean *Fidelity_Actual* values of the RMSE are negative except the mean *Fidelity_Actual* value of the MSTL local explainer which is used in the AusGridHH dataset. With regard to the NBoot approach, for all three datasets, the mean *Fidelity_Actual* measures show negative values. This implies that in most cases the forecasts produced by the global models are closer to the forecasts produced by the explainers than the actual values. Therefore, the analysis of *Fidelity_Actual* allows us to conclude that the explainers approximate the global model considerably better than the actuals.

In both explainer methods, the *Fidelity_Local* measures show negative values in all cases, except for the NN5W and AusGridW datasets when the NBoot approach is used, which indicates that the explainer model forecasts approximate the forecasts of the global model better than the local model forecasts approximate the global model under both techniques. The worse performance of NBoot for weekly datasets with the PR explainer is likely due to PR being a global model that trains across all the bootstrapped series, and as such, the explainer model is less localised.

The mean *Fidelity_with_Explainer* measures show negative values except for the NF explainer methods in the AusGridW dataset. As such, except for AusGridW, the forecasts from the explainers are considerably closer to the forecasts of the global model compared with the forecasts of the local models. The positive values for AusGridW show a certain degree of overfitting of the GFM (which we could confirm with a check of some example series). Thus, the results on the AusGridW dataset illustrate well the limitation of the NF approach, which may lead to explainers that are very close to the local models, if the in-sample fit of the GFM is close to the original data. To overcome this limitation, we perform the NBoot technique on the AusGridW dataset. With NBoot, the mean of *Fidelity_with_Explainer* is still positive, although the negative values we now obtain for the median show that NBoot is able to overcome the problem in the majority of cases.

The summarised mean fidelity, *Fidelity_Average*, is negative for all cases in the NF and NBoot explainer approaches, implying that the local explainer models generally approximate the global model forecasts well.

The mean *Acc_Global_LocalModel* measures show negative values in both approaches across all the explainer models, which implies that the actual values are closer to the global model forecasts than the local model forecasts. This indicates that the global model is better than the local model in terms of accuracy measured against the actual values (which is not surprising, as the datasets have been selected in such a way that global models perform well, as outlined in Section 3.1).

The mean *Acc_Explainer_LocalModel* performance measures show negative values for all three types of error measures with regard to both the NF and NBoot approaches. This implies that the forecasts of the explainer model are closer to the actual values than the forecasts of the local model. Thus, the performance of the explainer model is better than that of the local model when compared with the actual values, which is an interesting property and an interesting avenue to achieve more accurate but still simple local models.

Considering both the NF and NBoot explainer approaches, the mean *Acc_Explainer_GlobalModel* measures show mostly positive values, except for few cases. This implies that the forecasts of the global model are

closer to the actual values than the forecasts of the explainer model. Thus, the performance of the global model is better than that of the explainer model, which is not surprising, as the main aim of an explainer model is to explain well, not to achieve the best possible accuracy.

We furthermore perform Student's t-tests, as implemented in the function `t.test` in the `stats` package of the R programming language (R Core Team, 2021), to evaluate the statistical significance of our results. In particular, we perform one-sided tests to test whether the values are significantly smaller than zero. Assuming independence in the results between the datasets and local explainers, but dependence across error measures (RMSE, MASE, and MAE), we use a Bonferroni correction separately on each error measure to control the family-wise error across multiple tests, and to avoid spurious results. We use an initial significance level of 0.05, and as we perform 350 tests in total per error measure (RMSE, MASE, and MAE), this leads to a corrected significance level of $\alpha \approx 1.4 \times 10^{-4}$. The statistically significant results are marked with an asterisk (*) in Tables 4 and 5 of the main paper, and in Tables B.7, C.9, B.8, and C.10 in the Appendix. Furthermore, in the Appendix, Tables D.11, D.12, D.13, E.14, E.15, and E.16 show the corresponding unadjusted *p*-values. In these tables with *p*-values, values in boldface font indicate values that are statistically significant.

In summary, by analysing *Fidelity_Actual*, *Fidelity_Local*, and *Fidelity_with_Explainer*, we conclude that the local explainer model is explaining the global model for the given benchmark datasets and their particular local explainers well. In both the fidelity and accuracy performance measures of the explainers, the NF approach outperforms the NBoot approach. In the NBoot approach, we add noise during the bootstrapping process, which may make the explanations less local and bring the explanations further away from the actual values. However, NBoot is able to overcome problems that NF has in situations where the GFM is overfitting.

By analysing the results of *Acc_Explainer_LocalModel*, we have shown that it is possible to increase the accuracy of the forecasts of the local models by training them as local explainers following our methodology.

We further perform experiments to identify the best local explainer model for each dataset, considering the fidelity and accuracy dimensions of the performance metrics. Fig. 4 shows the summary plots for the NN5W, WebTrafficD, SFTrafficH, and AusgridHH datasets. In each of these radar plots, the performance measures are shown on the individual axes. Each axis shares the same tick mark where the higher values increase the error, and the lower values decrease the error. In particular, we consider the models to be better when they cover a smaller area in the radar plot, so that a minimal area covered yields the best local explainers. Using this criterion, according to Fig. 4, the best local explainer models for NN5W, WebTrafficD, SFTrafficH, and AusgridHH are the TBATS, ETS, DHR_ARIMA, and Prophet explainers, respectively. Here, we do not consider the AusgridW dataset, as it already revealed positive values in Table 4 that deviated from the expected values for the fidelity and accuracy evaluation metrics.

Table 6
Stability of the proposed explainer methods based on bootstrapping.

Dataset	Local explainer	NF	NBoot
NN5W	PR	0.000	0.229
	ETS	–	–
WebTraffic100D	PR	0.000	0.073
	ETS	0.000	0.069
AusGridW	PR	0.000	0.284
	ETS	–	–

5.1.2. Evaluating the stability of the explainer methods

For this experiment, we consider the same datasets and explainer models that were used when producing explanations using the NF and NBoot approaches. Fig. 5 shows boxplots of the median RMSE between the global model forecast and the PR local explainer on the NN5W dataset, the PR local explainer on the WebTraffic100D dataset, the ETS local explainer on the WebTraffic100D dataset, and the PR local explainer for the AusGridW dataset, over 10 independent runs. Note that for NF, the PR model is fitted on a single time series and effectively falls back to an AR model. Table 6 shows the according IQRs. From the plots and the table, we see that the NF approach is unsurprisingly the most stable approach, as it does not introduce uncertainty through a bootstrapping procedure, and only reflects the uncertainty in the fitting of the local model.

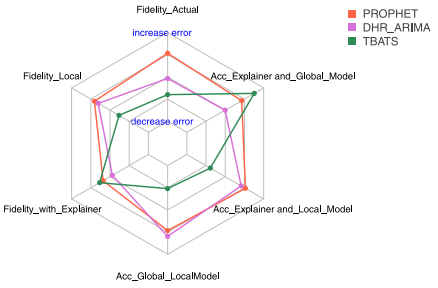
5.2. Qualitative results

This section qualitatively evaluates the comprehensibility of the explanations, with example explanations generated using the different local explainers for two explainer approaches.

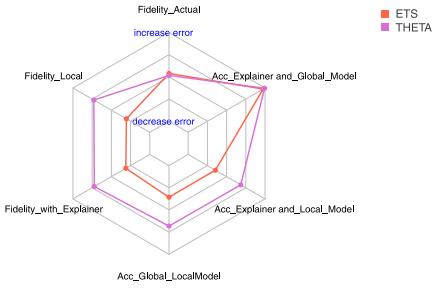
5.2.1. Example explanations for the NF method

In the following, we discuss two examples for different local explainer models that can explain the forecasts via decompositions.

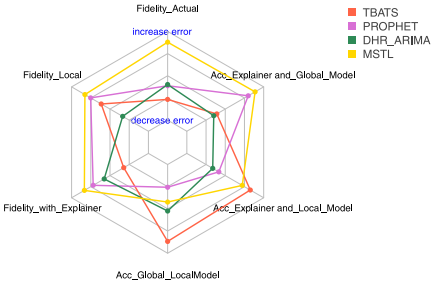
Example 1: Decomposition explanations using the MSTL local explainer. Fig. 6(a) shows a plot of forecasts based on the MSTL model for a single time series of the San Francisco Traffic hourly dataset. The figure allows us to visually verify fidelity and accuracy as follows. We can see that the global RNN model outperforms the local MSTL model, since the RNN forecasts are closer to the actuals than the local MSTL model. Moreover, we can observe that the MSTL explainer model approximates the global model better than the MSTL local model. Both of these observations show that the MSTL explainer is performing as expected. Fig. 6(b) gives an example of explanations of the global model forecasts provided by the TBATS explainer model. It provides components (ideally of the forecasts, but in our plot of the training set), namely, daily and weekly seasonalities and the trend as the decomposed components, to give insights into the forecast.



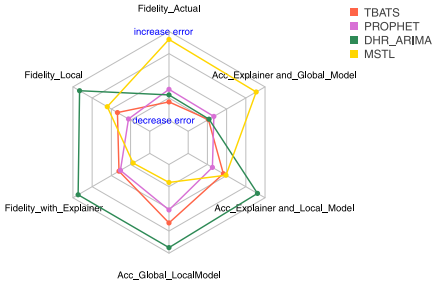
(a) Summary plot for NN5W dataset



(b) Summary plot for WebTrafficD dataset



(c) Summary plot for SFTrafficH dataset



(d) Summary plot for AusgridHH dataset

Fig. 4. Summary plots for the fidelity and accuracy error measures for the benchmark datasets.

Example 2: Decomposition explanations using the Prophet local explainer. Fig. 7(a) shows the forecast plot based on the Prophet model for a single time series of the Ausgrid half-hourly dataset. The plot shows that the global RNN model outperforms the local Prophet model, and that the Prophet explainer model approximates the global model better than the Prophet local model approximates the global model. Both of these observations allow us to conclude that the Prophet explainer is performing as expected. Fig. 7(b) shows the explanations for the forecasts generated by the global model using the Prophet explainer. Here, the Prophet explainer provides daily, yearly, and weekly seasonalities and the trend as components of the forecasts that can be used to attribute parts of the forecasts of the global model to these components directly.

5.2.2. Example explanations for bootstrapped neighbourhoods

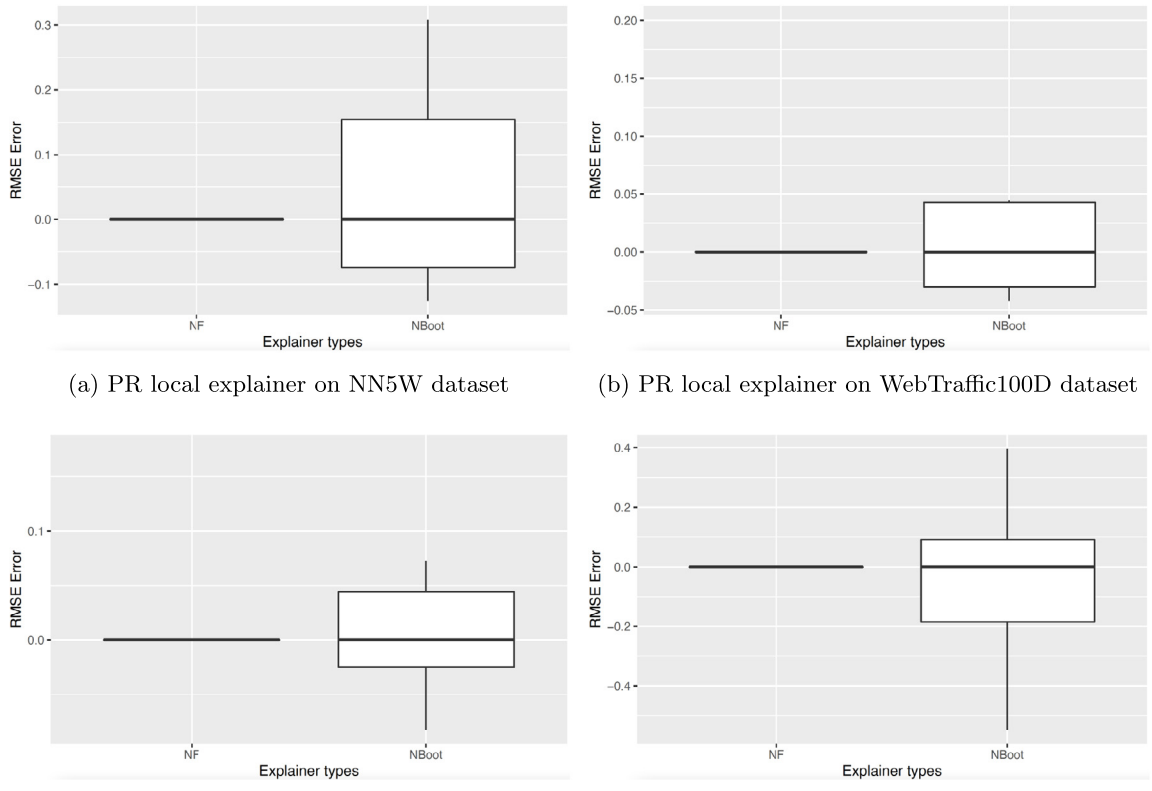
In contrast to the NF approach, in the NBoot approach, the explanations are produced by training an interpretable statistical model on the in-sample global model forecasts of the bootstrapped series. Therefore, in this case, there are N explainer models, where N is the number of bootstrapped series per original series. In our experiments, we bootstrap 100 series from the weekly series and 50 series from the daily series. With N explainer models instead of just one, it now also becomes important to be able to summarise the models. In the following, we illustrate this for the NBoot approach and for PR and ETS as the local explainers.

Example 1: Decomposition explanations using the ETS local explainer. Fig. 8 shows the forecast plot based on the ETS model for a single time series of the Kaggle Web Traffic daily dataset. Here, the forecasts of the explainer are calculated by bagging, i.e., by averaging the forecasts over the bootstrapped series. From the plot, we can see that the global RNN model outperforms the local ETS model, and that the ETS explainer model approximates the global model better than the ETS local model approximates the global model. Both of these observations allow us to conclude that the ETS explainer is performing as expected using the NBoot approach.

Fig. 9 summarises the resulting ETS models fitted on the bootstrapped series, following a procedure from Petropoulos, Hyndman, and Bergmeir (2018). As an explanation, Fig. 9 shows us that the predictions are based on no seasonality and a trend that is between no trend and a linear trend; i.e., a weak linear trend can be assumed.

Example 2: Decomposition explanations using a PR local explainer. Though PR is a global model, as outlined above, here we use it as a local explainer by fitting it across all bootstrapped series for a particular series to be explained.

Fig. 10 shows the forecast plot based on the PR model for a single time series of the NN5 weekly dataset. From



(a) PR local explainer on NN5W dataset

(b) PR local explainer on WebTraffic100D dataset

(c) ETS local explainer on WebTraffic100D dataset

(d) PR local explainer on AusGridW dataset

Fig. 5. Distribution of RMSEs across 10 independent runs of the proposed explainer methods.

the plot, we see that the global RNN model outperforms the PR model, and that the PR explainer model approximates the global model better than the (local) PR model.

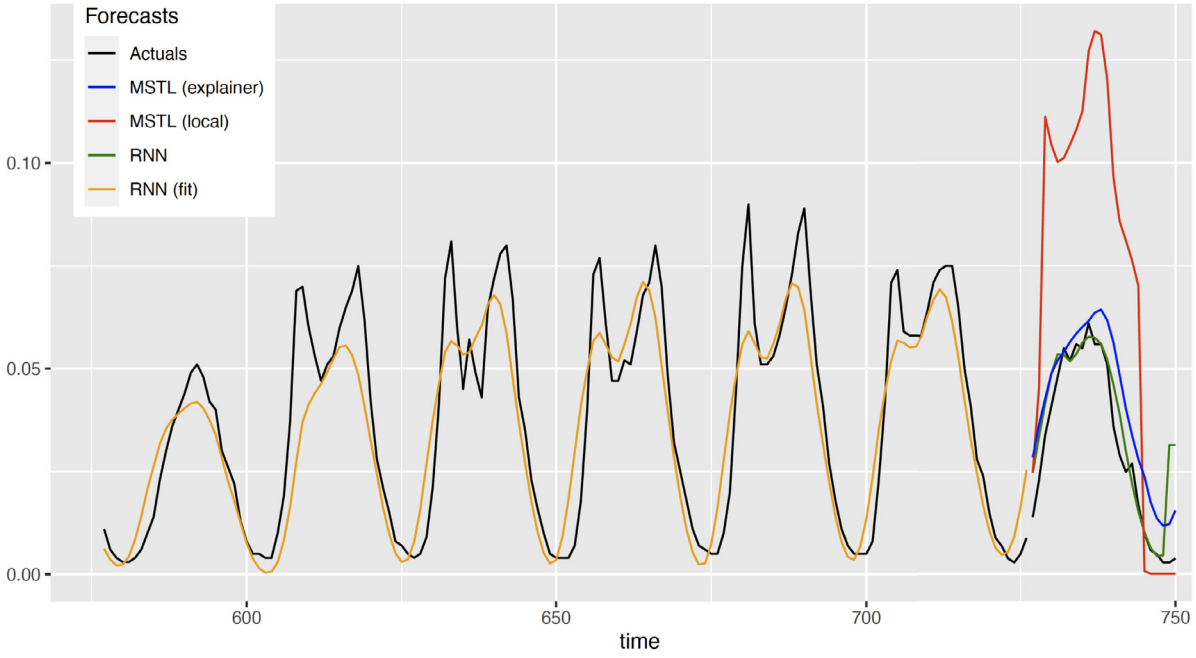
Fig. 11 shows the PR interpretation based on the coefficients of the regression function. Similar to the explanations provided by the LIME algorithm (Ribeiro et al., 2016b), here we plot the coefficients in a bar graph as the explanation of feature importance for the global model forecast, as the coefficients show how much a feature contributes towards a higher or lower prediction. From the figure, we can see that lag1 contributes towards higher values, whereas lag2 and lag8 contribute to lower values of the global model prediction.

6. Conclusions and future work

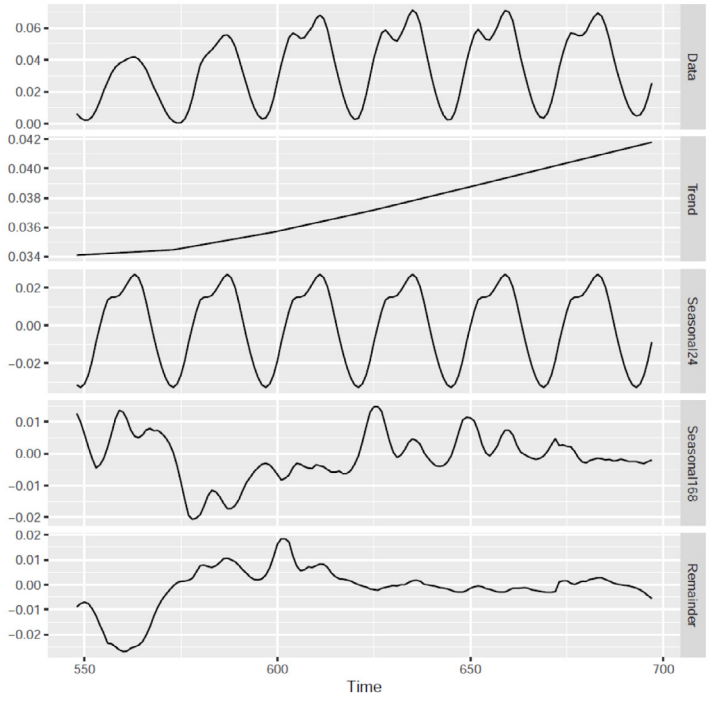
We presented local model explanations for forecasting (LoMEF), a model-agnostic framework for the local explainability of global forecasting models. It employs statistical forecasting techniques that can be deemed interpretable in a locally defined neighbourhood to explain the global model forecast of a particular time series using interpretable components such as trend, seasonality, coefficients, and other model attributes. We proposed two approaches to define the neighbourhood, one based on bootstrapping and the other based on model fitting. The bootstrapping procedure of the neighbourhood is appropriate when the global model is overfitting to the training dataset. Otherwise, we recommend the simpler procedure based on model fitting.

We evaluated our framework on five different benchmark datasets from energy, web traffic, and road traffic forecasting applications. The experiments performed in both quantitative and qualitative aspects showed that the two approaches in our LoMEF framework lead to explainers that approximate the global model forecast well and lead to comprehensible explanations. Thus, LoMEF promises to be a versatile tool in the forecaster's toolbox to explain GFMs through local models.

However, our approach also has some limitations. The NF approach is usually the approach of choice, as it runs much quicker than the NBoot approach and is as localised as possible to the series under consideration. Its main limitation is that it depends on the in-sample fit of the GFM. If the fit is not meaningful, through heavy under- or over-fitting of the GFM, the NF approach's ability to provide faithful explanations is limited. If the GFM fits the data perfectly, the local explainer and the local model are in fact identical. The *Fidelity_with_Explainer* measure is able to detect such situations, where the local model and the explainer coincide better than the explainer and the GFM.



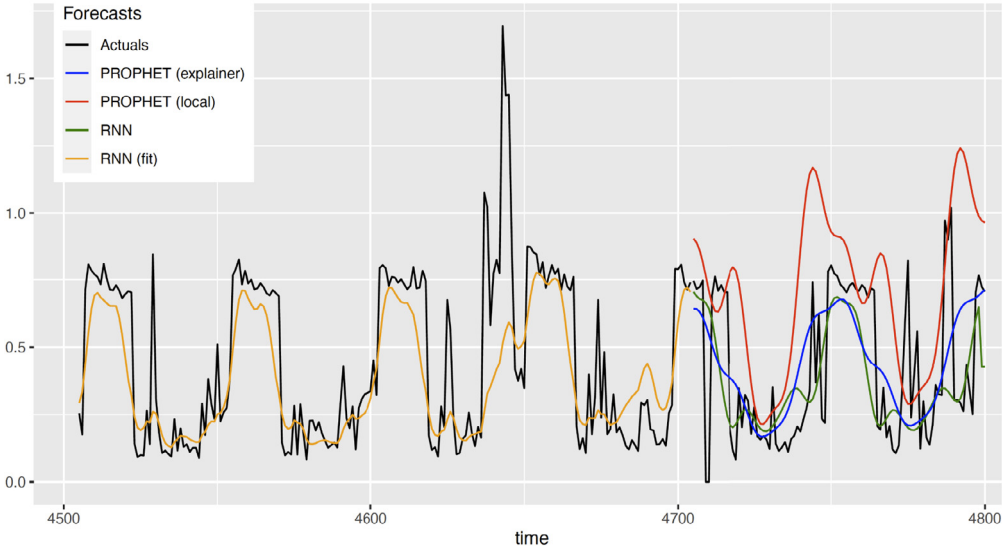
(a) MSTL local explainer forecasts



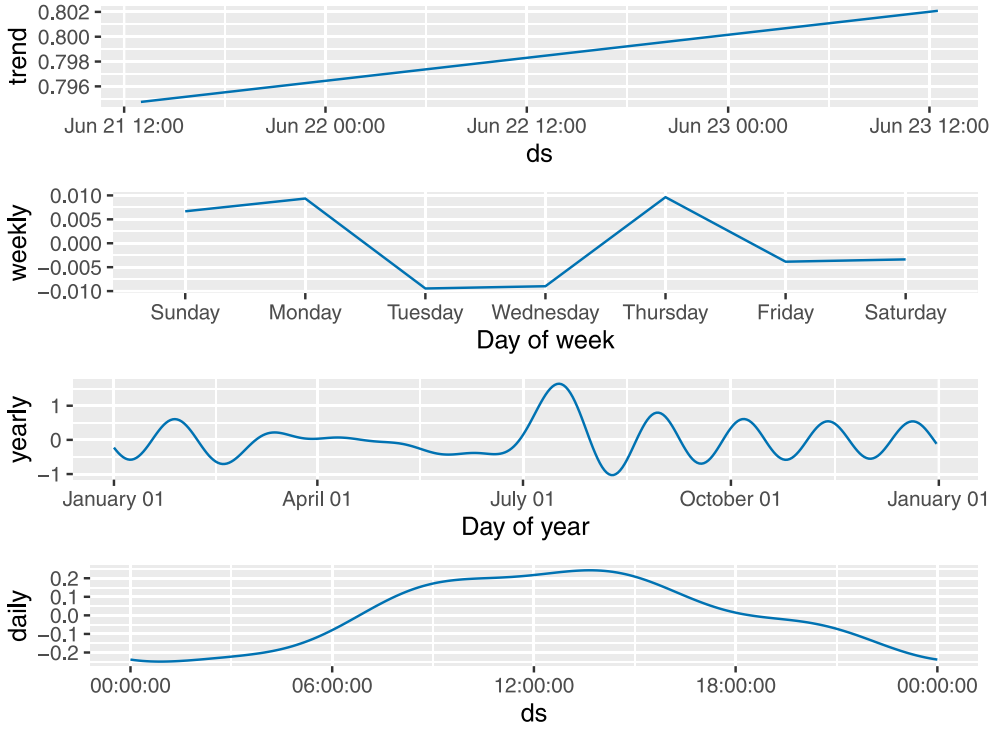
(b) Decomposition plot for MSTL interpretation

Fig. 6. MSTL local explainer forecasts and interpretation.

The NBoot approach is able to address this problem, since it uses bootstrapping techniques to generate a neighbourhood. However, the procedure is rather slow and, depending on the bootstrapping procedure, certain assumptions about the data need to be made. Furthermore, noise added during the bootstrapping procedure may make the procedure less localised (as indicated by the *Fidelity_Local* measure), especially when using a global model across the bootstraps as an explainer. When using a local explainer per bootstrap, the NBoot approach



(a) Prophet local explainer forecasts



(b) Decomposition plot for Prophet interpretation

Fig. 7. Prophet local explainer forecasts and interpretation.

generates multiple local explanations for the forecasts generated for a single series, and it may be a challenge to aggregate and summarise explanations.

In the future, we hope to validate the robustness of the LoMEF framework by applying it to different types of global forecasting models and different types of time series datasets used in various applications (e.g., medical applications). When providing these explanations, the multivariate factors of the time series are not considered in our framework. Therefore, this will be an excellent research question on interpretability in forecasting. Moreover, similar to other ML models, this model is susceptible to biases in the data. A plethora of works in this line

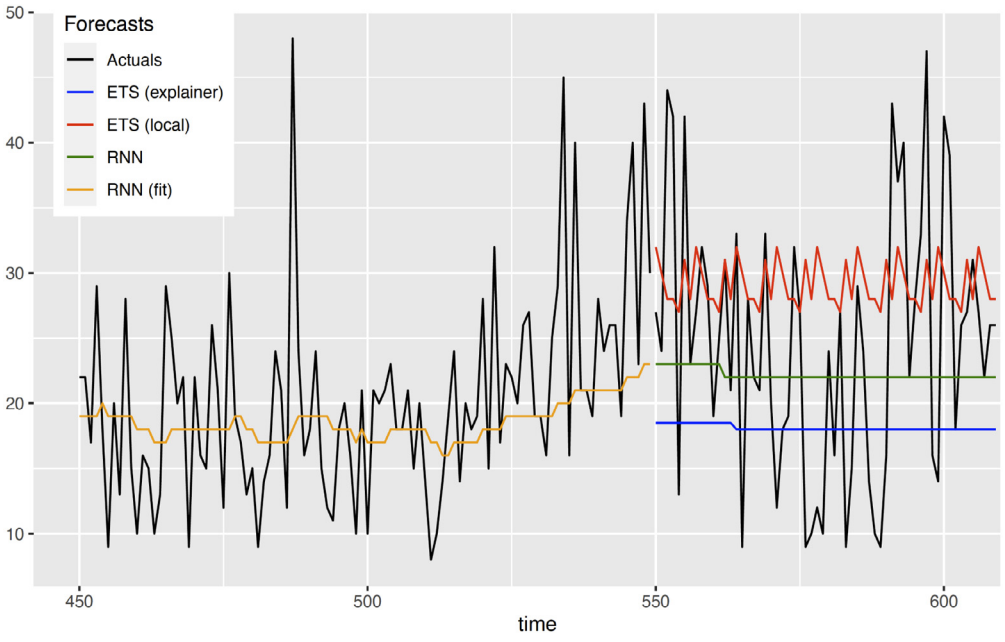


Fig. 8. ETS interpretation: Forecast plot.

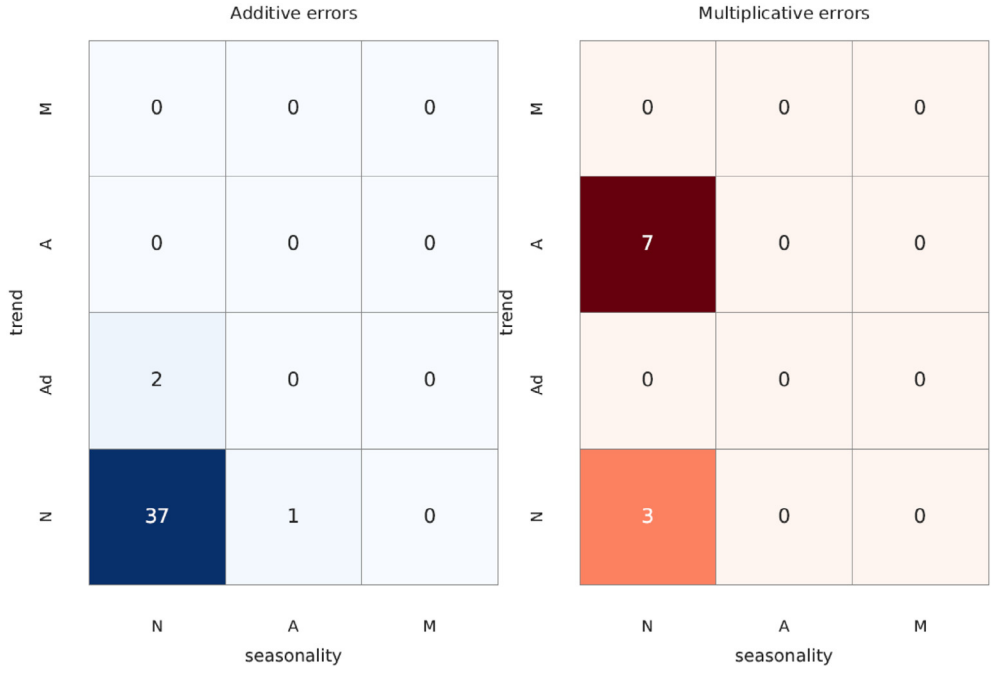


Fig. 9. ETS interpretation: Chosen model summary plot.

show that post hoc explanations can assist in detecting model biases. Uncovering model biases using post hoc explanations was not addressed in the current work but would be an excellent research question for future work. Furthermore, theoretically defining the "right" level of locality for explanations is unexplored in the time series forecasting domain, and this could also be explored in the future.

Acknowledgments

This research was supported by the Australian Research Council under grant DE190100045, a Facebook Statistics for Improving Insights and Decisions research award, Monash University Graduate Research funding, and MASSIVE – high-performance computing facility, Australia. We thank the anonymous referees and the associate

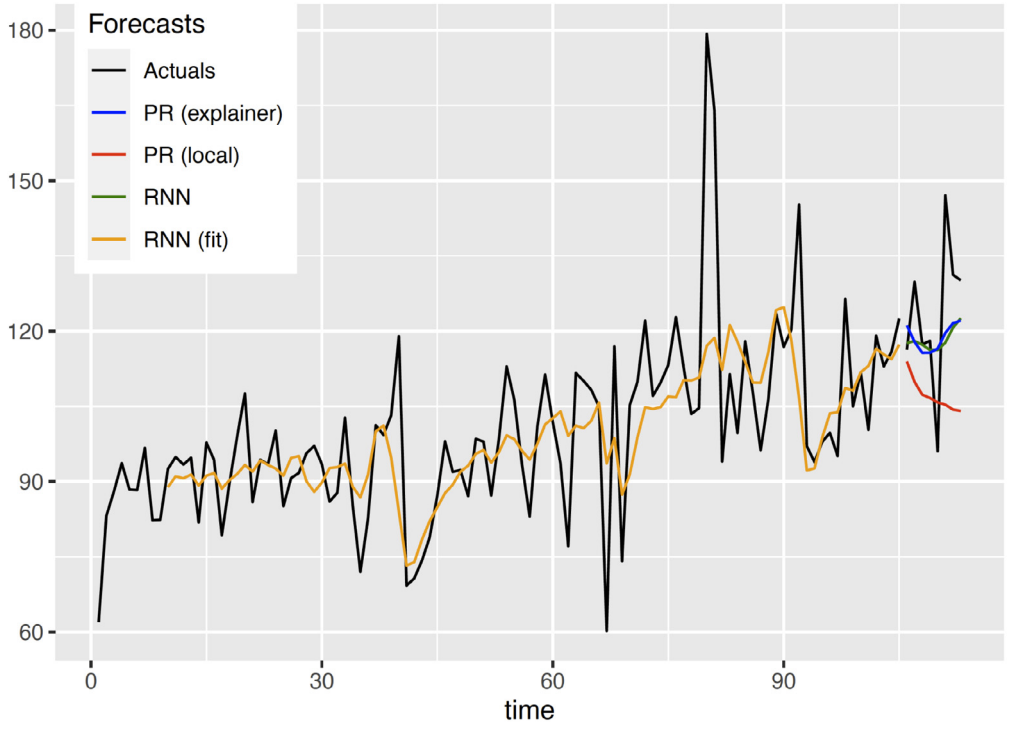


Fig. 10. PR interpretation: Forecast plot.

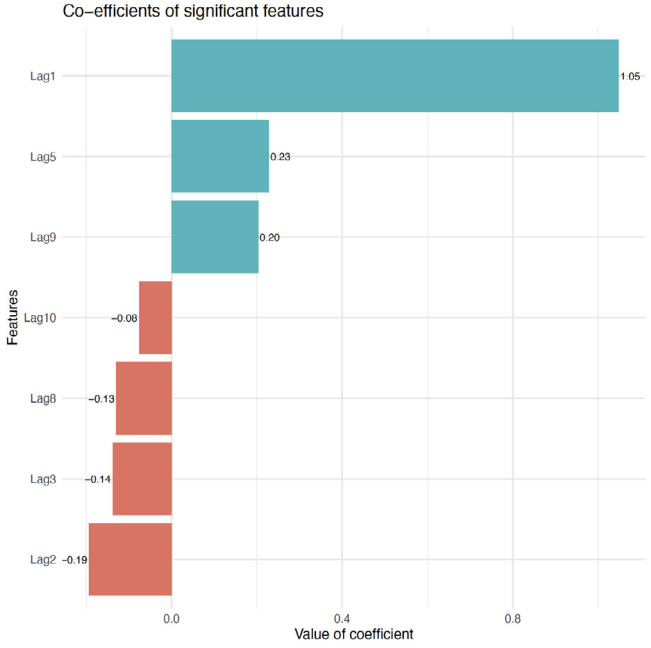


Fig. 11. PR interpretation: Coefficients of significant features.

editor for their comments that led to various improvements in the paper. We furthermore thank Brian Seaman for discussions that led to some ideas used in the paper.

Appendix A. Local explainer models

Exponential smoothing

Exponential smoothing is a very popular family of univariate forecasting models. It uses decompositions and weighted averages of past observations, where the weights are decaying exponentially over time (Brown & Meyer, 1961). In its simplest form, called simple exponential smoothing, it does not model trend or seasonality. More sophisticated models have been introduced subsequently to model trends and seasonalities (Brown, 1957; Winters, 1960) in additive and multiplicative ways.

In this work, we use the `ets` function from the `forecast` package (Hyndman et al., 2022; Khandakar & Hyndman, 2008) in the R programming language. The function fits different exponential smoothing models with different trends and seasonalities, and then selects one model based on a criterion such as the AICc. Thus, both the type of model (for example, a model with a local linear trend and a multiplicative seasonality) as well as the decomposed time series components (trend, seasonality, remainder) can be used to interpret the predictions.

STL decomposition

Seasonal and trend decomposition using loess (STL, Cleveland et al., 1990) is a statistical approach of decomposing a time series x_t additively into three components—seasonality S_t, trend T_t, and residuals R_t—as shown in Eq. (A.1):

$$x_t = S_t + T_t + R_t \tag{A.1}$$

The components can be used to interpret the forecast. STL is limited to series with a single seasonality. If a series has multiple seasonalities, multiple STL (MSTL) decomposition can be used (Bandara et al., 2021c).

Similar to STL, MSTL decomposes the time series into seasonal components, trend, and remainder. It extracts multiple seasonal components by applying the STL procedure iteratively. Thus, Eq. (A.1) can be extended to Eq. (A.2):

$$x_t = S_t^1 + S_t^2 + \cdots + S_t^n + T_t + R_t \tag{A.2}$$

Here, $S_t^1, S_t^2, \ldots, S_t^n$ denote the multiple seasonal components. Again, we consider the decomposed components of the time series as interpretable features of a forecast. We use the implementations of STL and MSTL available in the functions `stl` and `mstl` from the `stats` and `forecast` packages in R (Hyndman et al., 2022; Khandakar & Hyndman, 2008; R Core Team, 2021).

Linear autoregressive models

We also consider univariate linear autoregressive models, since they can be fitted both locally and globally (also called pooled regression, PR) but with the same interpretability through their coefficients. In the pooled regression, coefficients are the same across all time series. In our experiments we use PR as a local explainer in the NBoot approach. In this context, the term "global" means training across the bootstrapped series of the instance that needs to be explained. Hence, it produces explanations that are still local to the time series under consideration. To implement the PR models, we use the `glm` function in the `stats` package of the R programming language (R Core Team, 2021).

Dynamic harmonic regression ARIMA

Dynamic harmonic regression (DHR) is a statistical forecasting approach which is often used to forecast series with long seasonal periods, such as weekly and daily series, where ARIMA and ETS tend to produce less accurate results (Hyndman & Athanasopoulos, 2021).

In the approach of DHR_ARIMA, we use Fourier terms to model the seasonality, and ARIMA errors to model the short-term time series dynamics, while assuming that the seasonal pattern is constant over time. In terms of interpretation, the value of K Fourier terms, selected through grid search, gives an indication of the complexity of the seasonal pattern. The value of the first two coefficients, i.e., for $\cos(2\pi t/s)$ and $\sin(2\pi t/s)$, can be interpreted as the relative importance of the corresponding seasonal period. We use the `auto.arima` function from the `forecast` package in R (Hyndman et al., 2022; Khandakar & Hyndman, 2008) together with the Fourier terms generated by the function `fourier` from the same package to implement the model.

Trigonometric Box–Cox ARMA trend seasonal model: TBATS

TBATS is a state-of-the-art approach to forecast time series with complex multiple seasonalities (De Livera et al., 2011). To model complex multiple seasonalities, TBATS uses trigonometric representation terms based on Fourier series. Additionally, TBATS has the ability to model both linear and non-linear time series with single seasonalities, multiple seasonalities, high-period seasonalities, non-integer seasonalities, and dual calendar effects. As a local explainer, TBATS performs a decomposition of the series into complex seasonal patterns, trend, and irregular components. We use TBATS in its implementation in the `tbats` function from the `forecast` package in R.

Prophet

Prophet (Taylor & Letham, 2018) is another popular univariate forecasting software based on decompositions of the time series. It fits an additive decomposition model where non-linear trends are fitted with multiple seasonalities (e.g., yearly, weekly, and daily seasonality), together with holiday effects, as defined in Eq. (A.3):

$$x_t = S_t^1 + S_t^2 + \cdots + S_t^n + T_t + R_t + H_t \tag{A.3}$$

Table B.7
Mean performance measures based on the MASE measure for datasets on different local explainer models using the NF approach.

Dataset	Local explainer	Fidelity_Actual	Fidelity_Local	Fidelity_with_Explainer	Fidelity_Average	Acc_Global_LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
				Mean values				
NN5W	Prophet	**−0.375***	**−0.146***	**−0.086**	**−0.202**	**−0.033**	**−0.005**	0.028
	DHR_ARIMA	**−0.439***	**−0.147***	**−0.086**	**−0.224**	**−0.026**	**−0.017**	0.009
	TBATS	**−0.503***	**−0.246***	**−0.130**	**−0.293**	**−0.187***	**−0.140***	0.047
AusGridW	Prophet	**−0.268***	**−0.083***	0.272	**−0.026**	**−0.126***	**−0.063***	0.063
	DHR_ARIMA	**−0.335***	**−0.128***	0.179	**−0.095**	**−0.099**	**−0.066***	0.033
	TBATS	**−0.360***	**−0.087***	0.075	**−0.124**	**−0.124***	**−0.046**	0.078
	STL_ETS	**−0.302***	**−0.183***	0.049	**−0.145**	**−0.186***	**−0.127***	0.058
WebTrafficD	ETS	**−1.101***	**−0.773***	**−0.739***	**−0.871**	**−0.431**	**−0.430**	0.001
	Theta	**−1.110***	**−0.424***	**−0.394***	**−0.643**	**−0.157***	**−0.155***	0.003
SFTrafficH	TBATS	**−0.364***	**−0.480***	**−0.413***	**−0.419**	**−0.170***	**−0.182***	**−0.013**
	Prophet	**−0.232***	**−0.354***	**−0.194***	**−0.260**	**−0.453***	**−0.298***	0.155
	DHR_ARIMA	**−0.262***	**−0.547***	**−0.274***	**−0.361**	**−0.291***	**−0.308***	**−0.018**
	MSTL_ETS	**−0.032**	**−0.366***	**−0.210***	**−0.203**	**−0.334***	**−0.183***	0.152
AusGridHH	TBATS	**−0.462***	**−0.510***	**−0.446***	**−0.473**	**−0.285***	**−0.291***	**−0.006**
	Prophet	**−0.337***	**−0.680***	**−0.578***	**−0.531**	**−0.533***	**−0.449***	0.083
	DHR_ARIMA	**−0.408***	**−0.347***	**−0.160**	**−0.305**	**−0.155***	**−0.142***	0.013
	MSTL_ETS	0.110	**−0.393***	**−0.595***	**−0.293**	**−0.645***	**−0.197**	0.448

Asterisks indicate statistically significant results, and boldface font indicates negative values (which are desirable).

Table B.8
Mean performance measures based on the MAE measure for datasets on different local explainer models using the NF approach.

Dataset	Local explainer	Fidelity_Actual	Fidelity_Local	Fidelity_with_Explainer	Fidelity_Average	Acc_Global_LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	Prophet	**−6.454***	**−2.794***	**−1.427**	**−3.558***	**−0.683**	**−0.131**	0.552
	DHR_ARIMA	**−7.673***	**−2.793***	**−1.651**	**−4.039***	**−0.586**	**−0.406**	0.180
	TBATS	**−8.419***	**−4.555***	**−2.179**	**−5.051***	**−3.332***	**−2.392***	0.940
AusGridW	Prophet	**−6.093***	**−2.643***	7.969	**−0.256**	**−4.649***	**−2.045***	2.604
	DHR_ARIMA	**−8.242***	**−3.282***	5.105	**−2.140***	**−3.142**	**−1.689**	1.453
	TBATS	**−9.319***	**−1.956**	1.915	**−3.120***	**−3.000***	**−0.568**	2.432
	STL_ETS	**−7.244***	**−4.689***	2.067	**−3.289***	**−5.748***	**−3.518***	2.229
WebTrafficD	ETS	**−10.887***	**−8.669***	**−7.772***	**−9.109***	**−5.156***	**−5.215***	**−0.059**
	Theta	**−11.045***	**−5.904***	**−5.250***	**−7.400***	**−2.798**	**−2.796**	0.001
SFTrafficH	TBATS	**−0.005***	**−0.006***	**−0.005***	**−0.005***	**−0.002***	**−0.002***	0.000
	Prophet	**−0.003***	**−0.005***	**−0.003***	**−0.004***	**−0.006***	**−0.004***	0.002
	DHR_ARIMA	**−0.004***	**−0.008***	**−0.004***	**−0.005***	**−0.004***	**−0.004***	0.000
	MSTL_ETS	0.000	**−0.005***	**−0.002***	**−0.002***	**−0.005***	**−0.002***	0.003
AusGridHH	TBATS	**−0.075***	**−0.115***	**−0.102***	**−0.097***	**−0.078***	**−0.078***	0.000
	Prophet	−0.046*	**−0.142***	**−0.121***	**−0.103***	**−0.122***	**−0.103***	0.019
	DHR_ARIMA	**−0.061***	**−0.056***	**−0.016***	**−0.044***	**−0.035***	**−0.029***	0.005
	MSTL_ETS	0.055	**−0.087**	**−0.132***	**−0.054**	**−0.165***	**−0.061**	0.104

Asterisks indicate statistically significant results, and boldface font indicates negative values (which are desirable).

Table C.9
Mean performance measures based on MASE measures of the datasets on different local explainer models using the NBoot.

Dataset	Local explainer	Fidelity_Actual	Fidelity_Local	Fidelity_with_Explainer	Fidelity_Average	Acc_Global_LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	PR	**−0.236***	0.145	0.060	−0.010	**−0.087***	**−0.052**	0.034
	ETS	–	–	–	–	–	–	–
WebTraffic100D	PR	**−0.862***	**−0.120**	**−0.167***	**−0.324***	**−0.081**	**−0.091**	**−0.010**
	ETS	**−0.929***	**−0.735***	**−0.728***	**−0.451***	**−0.479**	**−0.421**	0.058
AusGridW	PR	**−0.159***	0.037	**−0.047**	**−0.092**	**−0.093***	0.072	0.218

Here, H_t represents the holiday effects. Most importantly, in Prophet, human-interpretable parameters (i.e., decomposed seasonalities, trend, and holiday effects) can be encompassed based on domain knowledge to improve the forecast. In this paper, we use the implementation of Prophet in the `prophet` package in R (Taylor & Letham, 2018).

Theta

The theta method was initially introduced by Assimakopoulos and Nikolopoulos (2000) and attracted the interest of forecasting practitioners after winning the M3 forecasting competition (Makridakis & Hibon, 2000). It was later shown by Hyndman and Billah (2003) that the

Table C.10
Mean performance measures based on MAE measures of the datasets on different local explainer models using the NBoot approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	PR	**−3.879***	1.072	**−0.277**	−1.028	**−2.556**	**−2.033**	0.523
	ETS	–	–	–	–	–	–	
WebTraffic100D	PR	**−7.896***	**−1.489***	**−1.550***	**−3.645***	**−0.924***	**−0.807**	0.118
	ETS	**−8.826***	**−9.175***	**−9.126***	**−9.043***	**−5.470**	**−5.511**	**−0.041**
AusGridW	PR	**−7.908***	0.895	1.358	**−1.885**	**−3.965***	1.568	5.533

Table D.11
Unadjusted *p*-values for mean RMSE measures of the datasets for different local explainer models using the NF approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	Prophet	**2.190E−18**	**2.435E−08**	2.134E−04	**2.831E−16**	1.436E−01	3.476E−01	8.794E−01
	DHR_ARIMA	**2.226E−23**	**5.478E−09**	**2.519E−05**	**3.213E−19**	2.538E−01	1.976E−01	4.720E−01
	TBATS	**5.584E−23**	**6.950E−10**	1.514E−03	**7.598E−20**	**6.954E−07**	**4.983E−05**	9.962E−01
AusGridW	Prophet	**5.439E−12**	**5.639E−09**	1.000E+00	**1.786E−07**	1.991E−04	**3.034E−06**	9.791E−01
	DHR_ARIMA	**4.533E−17**	**5.756E−11**	1.000E+00	**1.909E−09**	1.506E−03	1.733E−04	9.324E−01
	TBATS	**3.698E−19**	5.226E−04	9.916E−01	**1.909E−09**	1.435E−04	1.571E−01	9.985E−01
	STLETS	**7.380E−14**	**1.283E−27**	8.121E−01	**5.041E−15**	**2.548E−09**	**2.475E−15**	9.908E−01
WebTrafficD	ETS	4.244E−16	**1.480E−10**	**1.212E−10**	**1.089E−18**	3.546E−04	3.240E−04	4.947E−01
	Theta	**3.750E−16**	**4.276E−06**	**3.301E−06**	**5.651E−13**	2.313E−02	2.419E−02	6.443E−01
SFTrafficH	TBATS	**1.068E−78**	**1.729E−121**	**1.070E−84**	**5.792E−144**	**5.388E−17**	**1.623E−28**	**1.447E−12**
	Prophet	**1.806E−51**	**7.091E−182**	**1.461E−68**	**1.307E−129**	**2.253E−189**	**8.686E−150**	1.000E+00
	DHR_ARIMA	**1.984E−51**	**3.659E−166**	**6.729E−54**	**3.420E−130**	**2.720E−60**	**4.436E−99**	**1.176E−15**
	MSTLETS	2.585E−04	**3.117E−77**	**2.843E−18**	**1.419E−34**	**2.189E−66**	**3.198E−37**	1.000E+00
AusGridHH	TBATS	**1.245E−67**	**6.560E−19**	**1.973E−15**	**5.588E−29**	**1.983E−10**	**9.561E−11**	1.565E−01
	Prophet	**1.642E−49**	**5.192E−47**	**7.933E−37**	**5.509E−60**	**2.166E−31**	**1.721E−30**	1.000E+00
	DHR_ARIMA	**5.191E−60**	**4.320E−57**	**3.081E−11**	**8.130E−64**	**1.910E−11**	**5.424E−18**	4.452E−01
	MSTLETS	7.082E−01	**1.641E−06**	**2.203E−20**	**1.036E−06**	**1.884E−27**	4.016E−04	1.000E+00

Values that are significant according to the Bonferroni-corrected significance level of 1.4×10^{-4} are shown in boldface.

Table D.12
Unadjusted *p*-values for mean MASE measures of the datasets for different local explainer models using the NF approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	Prophet	**5.601E−18**	**1.220E−06**	2.123E−03	**8.808E−14**	1.447E−01	4.318E−01	8.990E−01
	DHR_ARIMA	**5.788E−24**	**2.087E−09**	4.433E−04	**1.616E−20**	1.883E−01	2.640E−01	6.748E−01
	TBATS	**2.693E−23**	**1.843E−10**	2.421E−04	**1.229E−18**	**1.942E−08**	**6.629E−06**	9.943E−01
AusGridW	Prophet	**2.959E−11**	**5.273E−08**	1.000E+00	**8.001E−02**	**5.919E−05**	**1.181E−05**	9.834E−01
	DHR_ARIMA	**3.708E−16**	**2.295E−14**	1.000E+00	**2.019E−07**	5.523E−04	**5.674E−06**	8.919E−01
	TBATS	**3.970E−17**	**7.418E−05**	9.973E−01	**1.298E−08**	**1.671E−05**	2.668E−02	9.988E−01
	STL_ETS	**1.917E−13**	**3.732E−28**	9.848E−01	**3.172E−09**	**1.564E−09**	**5.238E−14**	9.865E−01
WebTrafficD	ETS	**4.714E−22**	**8.557E−05**	**1.595E−04**	**3.172E−09**	1.609E−02	1.629E−02	6.384E−01
	Theta	**2.762E−22**	**7.716E−27**	**1.104E−24**	**8.788E−30**	**3.261E−07**	**4.623E−07**	9.452E−01
SFTrafficH	TBATS	**9.907E−108**	**2.046E−113**	**1.157E−92**	**1.728E−140**	**1.916E−22**	**1.037E−24**	7.013E−02
	Prophet	**2.247E−44**	**5.228E−161**	**1.583E−63**	**2.030E−115**	**3.434E−195**	**5.045E−131**	1.000E+00
	DHR_ARIMA	**2.122E−63**	**1.974E−162**	**9.674E−57**	**9.566E−135**	**7.890E−54**	**4.229E−72**	3.098E−02
	MSTL_ETS	5.427E−02	**6.423E−71**	**1.327E−26**	**4.543E−34**	**1.091E−65**	**1.171E−21**	1.000E+00
AusGridHH	TBATS	**6.712E−09**	**9.007E−29**	**1.024E−23**	**1.629E−20**	**3.758E−14**	**1.045E−16**	1.871E−01
	Prophet	**3.228E−05**	**3.803E−22**	**1.488E−21**	**5.275E−15**	**9.039E−50**	**4.749E−41**	1.000E+00
	DHR_ARIMA	**8.148E−07**	**4.004E−06**	2.162E−02	**6.830E−05**	**9.206E−24**	**4.065E−19**	9.085E−01
	MSTL_ETS	8.287E−01	**1.346E−04**	**1.448E−13**	8.453E−04	**6.323E−33**	1.209E−02	1.000E+00

Values that are significant according to the Bonferroni-corrected significance level of 1.4×10^{-4} are shown in boldface.

method is in fact identical to simple exponential smoothing with drift, where the drift parameter is chosen as half the slope of a linear trend that is fitted to the data. As the method is simple and interpretable, and has been very successful in the past, we use it as a local explainer. We use the implementation of theta in the `thetaf` function from the `forecast` package in R.

Table D.13
Unadjusted *p*-values for mean MAE measures of the datasets for different local explainer models using the NF approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	Prophet	**5.404E−14**	**4.141E−07**	3.266E−03	**3.540E−12**	1.445E−01	4.007E−01	9.192E−01
	DHR_ARIMA	**1.002E−19**	**2.037E−07**	6.889E−04	**1.238E−15**	1.608E−01	2.058E−01	6.853E−01
	TBATS	**1.510E−19**	**1.011E−10**	2.219E−04	**6.320E−18**	**1.570E−07**	**8.564E−06**	9.944E−01
AusGridW	Prophet	**2.293E−07**	**1.539E−07**	1.000E+00	3.316E−01	**4.115E−05**	**2.117E−05**	9.956E−01
	DHR_ARIMA	**8.960E−13**	**5.240E−09**	1.000E+00	**1.061E−04**	5.632E−04	1.881E−04	9.553E−01
	TBATS	**1.877E−14**	1.230E−03	9.911E−01	**1.052E−06**	**1.445E−04**	2.180E−01	9.991E−01
	STL_ETS	**4.926E−10**	**3.390E−21**	9.974E−01	**1.128E−08**	**5.320E−09**	**2.228E−12**	9.954E−01
WebTrafficD	ETS	**1.784E−15**	**2.249E−10**	**1.665E−10**	**1.300E−16**	**2.581E−05**	**3.233E−05**	2.908E−01
	Theta	**2.122E−15**	**8.827E−06**	**6.573E−06**	**4.405E−10**	5.022E−03	6.546E−03	5.068E−01
SFTrafficH	TBATS	**3.017E−91**	**5.435E−117**	**5.991E−97**	**3.493E−148**	**1.257E−14**	**1.635E−12**	8.229E−01
	Prophet	**2.441E−36**	**3.805E−137**	**8.329E−57**	**3.826E−101**	**5.589E−200**	**7.903E−107**	1.000E+00
	DHR_ARIMA	**1.497E−54**	**4.944E−137**	**4.781E−49**	**3.132E−115**	**7.527E−52**	**1.049E−59**	7.206E−01
	MSTL_ETS	5.143E−01	**8.950E−59**	**4.502E−17**	**6.076E−23**	**2.143E−56**	**4.498E−11**	1.000E+00
AusGridHH	TBATS	**3.109E−61**	**1.864E−15**	**4.808E−13**	**5.241E−22**	**5.149E−10**	**2.780E−10**	5.214E−01
	Prophet	**4.907E−28**	**2.156E−38**	**3.959E−32**	**3.412E−45**	**4.740E−35**	**2.390E−30**	1.000E+00
	DHR_ARIMA	**6.519E−48**	**7.832E−58**	**5.449E−07**	**1.044E−52**	**1.955E−22**	**2.485E−27**	9.996E−01
	MSTL_ETS	9.961E−01	5.608E−04	**5.263E−14**	1.540E−03	**1.292E−21**	5.488E−03	1.000E+00

Values that are significant according to the Bonferroni-corrected significance level of 1.4×10^{-4} are shown in boldface.

Table E.14
Unadjusted *p*-values for mean RMSE measures of the datasets for different local explainer models using the NBoot approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	PR	**9.128E−12**	7.651E−01	3.846E−01	9.310E−03	5.501E−04	3.022E−03	8.623E−01
WebTraffic100D	ETS	**9.445E−16**	**6.354E−05**	**6.883E−05**	**1.358E−08**	1.187E−02	1.101E−02	2.357E−01
	PR	**6.398E−15**	**1.518E−05**	**1.578E−05**	**6.569E−17**	5.655E−02	4.098E−02	4.593E−01
AusGridW	PR	**2.044E−14**	9.214E−01	9.984E−01	**2.623E−05**	**7.320E−08**	9.113E−01	1.000E+00

Values that are significant according to the Bonferroni-corrected significance level of 1.4×10^{-4} are shown in boldface.

Table E.15
Unadjusted *p*-values for mean MASE measures of the datasets for different local explainer models using the NBoot approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	PR	**6.164E−06**	9.922E−01	9.531E−01	4.041E−01	**1.409E−04**	2.856E−02	9.374E−01
WebTraffic100D	ETS	**8.372E−12**	**2.166E−05**	**1.901E−05**	**1.243E−09**	1.059E−03	3.423E−03	9.987E−01
	PR	**2.250E−11**	9.054E−04	**3.868E−06**	**7.610E−15**	7.678E−04	1.913E−03	2.175E−01
AusGridW	PR	**3.833E−09**	9.785E−01	9.796E−01	1.833E−02	**1.986E−06**	9.999E−01	1.000E+00

Table E.16
Unadjusted *p*-values for mean MAE measures of the datasets for different local explainer models using the NBoot approach.

Dataset	Local explainer	Fidelity _Actual	Fidelity _Local	Fidelity_with _Explainer	Fidelity _Average	Acc_Global _LocalModel	Acc_Explainer and Local Model	Acc_Explainer and Global Model
NN5W	PR	**2.454E−05**	7.897E−01	3.816E−01	1.192E−01	2.299E−04	1.254E−03	8.786E−01
WebTraffic100D	ETS	**4.056E−16**	**7.766E−05**	**7.929E−05**	**3.156E−07**	2.698E−03	2.625E−03	2.652E−01
	PR	**3.172E−15**	**6.637E−06**	**1.132E−05**	**3.637E−17**	**7.251E−05**	3.468E−04	7.666E−01
AusGridW	PR	**1.230E−10**	9.403E−01	9.582E−01	7.343E−04	**7.901E−07**	9.855E−01	1.000E+00

Values that are significant according to the Bonferroni-corrected significance level of 1.4×10^{-4} are shown in boldface.

Table F.17
Initial hyperparameter ranges of the global forecasting model.

Dataset	Cell dimension	Gaussian noise stdev	L2 regularisation	Max epoch size	Max no. epochs	Mini batch size	No. hidden layers	Random initialiser stdev
NN5W	20–50	0.0001–0.0008	0.0001–0.0008	2–10	2–25	5–30	1–2	0.0001–0.0008
AusGridW	20–50	0.0001–0.0008	0.0001–0.0008	2–10	2–25	2–50	1–2	0.0001–0.0008
WebTrafficD	20–50	0.0001–0.0008	0.0001–0.0008	2–10	2–25	200–700	1–2	0.0001–0.0008
SFTrafficH	20–50	0.0001–0.0008	0.0001–0.0008	3–6	3–10	200–300	1–2	0.0001–0.0008
AusGridHH	20–50	0.0001–0.0008	0.0001–0.0008	2–4	5–20	10–50	1–2	0.0001–0.0008

Table F.18
Optimal hyperparameter values of the global forecasting model.

Dataset	Cell dimension	Gaussian noise stdev	L2 regularisation	Max epoch size	Max no. epochs	Mini batch size	No. hidden layers	Random initialiser stdev
NN5W	35	0.00045	0.00045	6	14	18	2	0.00045
AusGridW	35	0.00045	0.00045	6	14	35	2	0.00045
WebTrafficD	22	0.00045	0.00045	6	14	450	2	0.00045
SFTrafficH	35	0.00045	0.00045	4	6	250	2	0.00045
AusGridHH	35	0.00045	0.00045	3	12	30	2	0.00045

Appendix B. Fidelity and accuracy measures for the NF method

See Tables B.7 and B.8.

Appendix C. Fidelity and accuracy measures for the NBoot method

See Tables C.9 and C.10.

Appendix D. Statistical significance of the results of the NF explainer

See Tables D.11–D.13.

Appendix E. Statistical significance of the results of NBoot approach

See Tables E.14–E.16.

Appendix F. Hyperparameter values of the global model

See Tables F.17 and F.18.

Appendix G. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ijforecast.2022.06.006.

References

Assimakopoulos, V., & Nikolopoulos, K. (2000). The theta model: A decomposition approach to forecasting. *International Journal of Forecasting*, *16*(4), 521–530.

AusGrid (2019). Solar home electricity data. Accessed: 2020-9-24, URL https://www.ausgrid.com.au/Industry/Our-Research/Data-to-share/Solar-home-electricity-data.

Bandara, K., Bergmeir, C., & Hewamalage, H. (2021a). LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(4), 1586–1599.

Bandara, K., Bergmeir, C., & Smyl, S. (2020). Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. *Expert Systems with Applications*, *140*, Article 112896.

Bandara, K., Hewamalage, H., Liu, Y.-H., Kang, Y., & Bergmeir, C. (2021b). Improving the accuracy of global forecasting models using time series data augmentation. *Pattern Recognition*, *120*, Article 108148.

Bandara, K., Hyndman, R. J., & Bergmeir, C. (2021c). MSTL: A seasonal-trend decomposition algorithm for time series with multiple seasonal patterns. *International Journal of Operational Research*, *in press*.

Bandara, K., Shi, P., Bergmeir, C., Hewamalage, H., Tran, Q., & Seaman, B. (2019). Sales demand forecast in e-commerce using a long short-term memory neural network methodology. In *International conference on neural information processing* (pp. 462–474). Springer.

Ben Taieb, S., Bontempi, G., Atiya, A. F., & Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications*, *39*(8), 7067–7083.

Bergmeir, C., Hyndman, R. J., & Benítez, J. M. (2016). Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation. *International Journal of Forecasting*, *32*(2), 303–312.

Bojer, C. S., & Meldgaard, J. P. (2021). Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, *37*(2), 587–603.

Brown, R. G. (1957). Exponential smoothing for predicting demand. *Operations Research*, *5*(1), 145.

Brown, R. G., & Meyer, R. F. (1961). The fundamental theorem of exponential smoothing. *Operations Research*, *9*(5), 673–685.

Bühlmann, P. (1997). Sieve bootstrap for time series. *Bernoulli*, *3*(2), 123–148.

Caltrans (2020). Caltrans PeMS. Accessed: 2020-9-24, URL http://pems.dot.ca.gov/.

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*(8), 832.

Cleveland, R. B., Cleveland, W. S., McRae, J., & Terpenning, I. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, *6*, 3–73.

Cordeiro, C., & Neves, M. (2009). Forecasting time series with BOOT.EXPOS procedure. *REVSTAT-Statistical Journal*, *7*(2), 135–149.

De Livera, A. M., Hyndman, R. J., & Snyder, R. D. (2011). Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, *106*(496), 1513–1527.

Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. URL https://arxiv.org/abs/1702.08608.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179–211.

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. *Neural Computation*, *12*(10), 2451–2471.

Godahewa, R., Bergmeir, C., Webb, G. I., Hyndman, R. J., & Montero-Manso, P. (2021). Monash time series forecasting archive. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Google (2017). Web traffic time series forecasting. Accessed: 2020-9-24, URL https://www.kaggle.com/c/web-traffic-time-series-forecasting.

Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., & Giannotti, F. (2018). Local rule-based explanations of black box decision systems. URL http://arxiv.org/abs/1805.10820.

Harvey, A. C., & Shephard, N. (1993). Structural time series models. In G. S. Maddala, C. R. Rao, & H. D. Vinod (Eds.), *vol. 11: Econometrics*, *Handbook of statistics* (pp. 261–302). Amsterdam: North Holland.

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

Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, *37*(1), 388–427.

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization* (pp. 507–523). Springer.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: principles and practice* (3rd ed.). OTexts, Melbourne, Australia: Accessed 2021-8-24 URL https://OTexts.com/fpp3.

Hyndman, R. J., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., et al. (2022). Forecast: Forecasting functions for time series and linear models. R package version 8.16, URL http://pkg.robjhyndman.com/forecast.

Hyndman, R. J., & Billah, B. (2003). Unmasking the Theta method. *International Journal of Forecasting*, *19*(2), 287–290.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688.

Januschowski, T., Gasthaus, J., Wang, Y., Salinas, D., Flunkert, V., Bohlke-Schneider, M., et al. (2020). Criteria for classifying forecasting methods. *International Journal of Forecasting*, *36*(1), 167–177.

Kaushik, S., Choudhury, A., Sheron, P. K., Dasgupta, N., Natarajan, S., Pickett, L. A., et al. (2020). Ai in healthcare: time-series forecasting using statistical, neural, and ensemble architectures. *Frontiers in Big Data*, *3*, 4.

Khandakar, Y., & Hyndman, R. J. (2008). Automatic time series forecasting: the forecast Package for R. *Journal of Statistical Software*, *27*(03).

Kolassa, S. (2020). Why the "best" point forecast depends on the error or accuracy measure. *International Journal of Forecasting*, *36*(1), 208–211.

Kunsch, H. R. (1989). The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, *17*(3), 1217–1241.

Laptev, N., Yosinski, J., Li, L. E., & Smyl, S. (2017). Time-series extreme event forecasting with neural networks at Uber. In *International conference on machine learning, workshop, vol. 34* (pp. 1–5).

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems 30* (pp. 4765–4774).

Makridakis, S., & Hibon, M. (2000). The M3-Competition: Results, conclusions and implications. *International Journal of Forecasting*, *16*(4), 451–476.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS One*, *13*(3), Article e0194889.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). The M5 accuracy competition: Results, findings and conclusions. *International Journal of Forecasting*, in press.

McMurry, T. L., & Politis, D. N. (2010). Banded and tapered estimates for autocovariance matrices and the linear process bootstrap. *Journal of Time Series Analysis*, *31*(6), 471–482.

Molnar, C. (2018). Interpretable machine learning. https://christophm.github.io/interpretable-ml-book/contribute.html, Accessed: 2018-5-16.

Montero-Manso, P., & Hyndman, R. J. (2021). Principles and algorithms for forecasting groups of time series: Locality and globality. *International Journal of Forecasting*, *37*(4), 1632–1653.

Nanayakkara, S., Fogarty, S., Tremeer, M., Ross, K., Richards, B., Bergmeir, C., et al. (2018). Characterising risk of in-hospital mortality following cardiac arrest using machine learning: A retrospective international registry study. *PLoS Medicine*, *15*(11), Article e1002709.

Orabona, F., & Tommasi, T. (2017). Training deep networks without learning rates through coin betting. In *Advances in neural information processing systems* (pp. 2160–2170).

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International conference on learning representations (ICLR)*.

Paparoditis, E., & Politis, D. N. (2001). Tapered block bootstrap. *Biometrika*, *88*(4), 1105–1119.

Petropoulos, F., Hyndman, R. J., & Bergmeir, C. (2018). Exploring the sources of uncertainty: Why does bagging for time series forecasting work? *European Journal of Operational Research*, *268*(2), 545–554.

Petropoulos, F., Makridakis, S., Assimakopoulos, V., & Nikolopoulos, K. (2014). 'Horses for courses' in demand forecasting. *European Journal of Operational Research*, *237*(1), 152–163.

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, URL https://www.R-project.org/.

Rajapaksha, D., Bergmeir, C., & Buntine, W. (2020). LoRMIkA: Local rule-based model interpretability with k-optimal associations. *Information Sciences*, *540*, 221–241.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Model-agnostic interpretability of machine learning. In *International Conference on Machine Learning, workshop*.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016b). Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135–1144). ACM.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. *Vol. 32*, In *Proceedings of the AAAI conference on artificial intelligence*. URL https://ojs.aaai.org/index.php/AAAI/article/view/11491.

Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, *36*(3), 1181–1191.

Shao, X. (2010a). The dependent wild bootstrap. *Journal of the American Statistical Association*, *105*(489), 218–235.

Shao, X. (2010b). Extended tapered block bootstrap. *Statistica Sinica*, *20*, 807–821.

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, *36*(1), 75–85.

Smyl, S., & Kuber, K. (2016). Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In *36th international symposium on forecasting*.

Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician*, *72*(1), 37–45.

Wen, R., Torkkola, K., Narayanaswamy, B., & Madeka, D. (2017). A multi-horizon quantile recurrent forecaster. In *Neural information processing systems, workshop*.

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science*, *6*(3), 324–342.

Yang, M., & Kim, B. (2019). BIM: Towards quantitative evaluation of interpretability methods with ground truth. URL https://arxiv.org/abs/1907.09701.