Exploring the association between time series features and forecasting by temporal aggregation using machine learning

Bahman Rostami-Tabar[a,*], Dejan Mircetic[b,1]

[a] *Cardiff Business School 3 Colum Drive CF10 3EU Cardif*
[b] *Institute for Artificial Intelligence research and development of Serbia Fruskogorska 1 21000 Novi Sad*

Abstract

When a forecast of the total value over a number of time periods ahead is required, forecasters are presented with two temporal aggregation (TA) approaches approaches to produce required forecasts: i) aggregated forecast (AF) or ii) aggregate data using non-overlapping temporal aggregation (AD). Often, the recommendation is to aggregate data to a frequency relevant to the decision the eventual forecasts will support and then produce the forecast. However, this might not be always the best choice and we argue that both AF and AD approaches may outperform each other in different situations. Moreover, there is a lack of evidence on what indicators may determine the superirity of each approach. We design and execute an empirical experiment framework to first explore the performance of these approaches using monthly time series of M4 competition dataset. We further turn the problem into a classification supervised learning and build a machine learning algorithm to investigate the connection between time series features and the performance of temporal aggregation. This is the first study in time series forecasting that explores the association between time series features and temporal aggreagtion performance. Our findings suggest that neither AF or AD approaches perform accurately for each individual series. AF is shown to be significantly better than AD for the monthly M4 time series, especially for longer horizons. We build several machine learning approaches using a set of extracted time series features as input to predict accurately whether AD or AF oshould be used. We find out Random Forest (RF) is the most accurate approach in correctly classifying the oucome examined both by staitical measures such as missclassification error, F-statistics, and area under the curve and a utility measure. The RF approach reveals that curvature, nonlinearity, seas_pacf, unitroot_up, mean, ARCHM.LM, Coifficient of Variation, stability, linearity and max_level_shif are among the most important features in driving the predictions of the model. Our findings indicate that the strength of trend, ARCH.LM, hurst, autocorrelation lag 1 and unitroot_pp and seas_pacf may favor AF approach, while lumpiness, entropy, nonlinearity, curvature, stremgth of seasonality may increase the chance of AD performing better. We conclude the study by sumamrising the finding and present an agenda for further research.

Keywords: Temporal Aggregation, Forecasting, Time Series Feature, Exponential Smoothing, Machine Learning, Random Forest, Classification

1. Introduction

Time series forecasting has been used for many decades to inform decisions in various sectors such as business, finance, economy, supply chain and healthcare (Petropoulos et al., 2022). With advances in technology, data can often be collected at the time of transaction or service, e.g. call arrival times in a

*Corresponding author
Email addresses: rostami-tabarb@cardiff.ac.uk (Bahman Rostami-Tabar), dejan.mircetic@ivi.ac.rs (Dejan Mircetic)
[1]Equal contribution

call centre, point of sales in retail, or incidents attended in an ambulance service. In electronic databases, temporal data are generally stored in a single level of granularity.

One common assumption in time series forecasting is that the time series granularity matches the forecast requirement, i.e. to produce daily forecasts, we use daily time series. However, the level of time series granularity does not necessarily match the level of forecast granularity, driven by the decision-making process. Indeed, in practice the level of temporal granularity in the forecast requirement is often lower than the existing time series granularity. For instance, while a forecast might be required at the annual (daily) level, the time series is available at monthly (hourly) level. We also recognise that there might be cases where forecast granularity is higher than the existing series, however this requires introducing a disaggregation mechanism and is not covered in this study.

Generally, forecast granularity level and its horizon are determined by decisions made in the light of forecasts. In this paper, we consider a situation where an original time series has a higher temporal granularity (e.g. monthly) than the required forecast (e.g. annual). We aim to generate a forecast of the total value over several time periods ahead, which is referred to as forecast horizon aggregation or forecast over the lead-time period (Mohammadipour and Boylan, 2012). Therefore, the lead-time period matches the aggregation level required to aggregate the time series.

Producing an aggregated forecast over a number of periods is required in many situations to inform decisions about capacity planning, inventory management, logistics, procurement, and others (Nikolopoulos et al., 2011; Zotteri and Kalchschmidt, 2007). For example, generating a forecast for the whole lead-time period is often required to determine the level of safety stock in inventory management. In an emergency department, while the historical hourly time series of admission is available, daily forecasts might be required for rostering, while quarterly forecasts might be useful for resource planning (Rostami-Tabar and Ziel, 2020). In a supply chain, yearly forecasts might be used for procurement and budgeting decisions, while the time series might be available at a monthly granularity (Mircetic et al., 2021).

A key question then to be answered is: should the original time series be used to generate the forecast for the required horizon and then sum them up to obtain the forecast over time periods (lead-time), i.e. Aggregate Forecast (AF), or should we first aggregate the time series to match the forecast requirement granularity and then extrapolate directly at that level, i.e. Aggregate Data (AD). This has been illustrated in Figure 1.

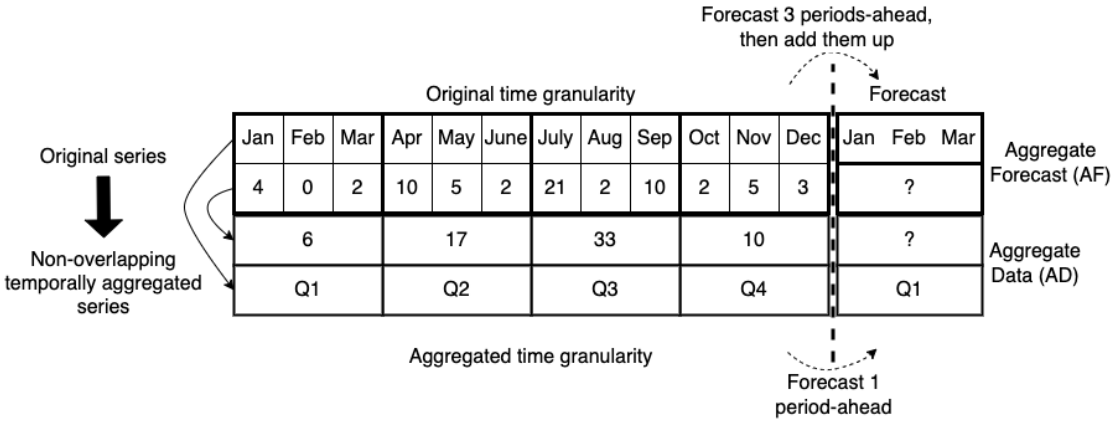

Figure 1: Aggregate forecast vs. aggregate data approaches. Assuming a montly time series is available and a forecast over one quarter (aggregation level = 3 months) is required. We first generate forecast for 3 periods ahead and then sum them up to create forecast over the quarter (Forecast by AF approach). Then, we create the temporally aggregated series by dividing the original series into the block of 3 periods. Next, we forecast for 1 period ahead (Forecast by AD approach).

For the later, we often use the non-overlapping temporal aggregation (NOTA) approach. Using this approach, the original series is divided into a consecutive non-overlapping bucket of time, starting from the most recent period backward. The size of the bucket is equal to the number of time periods required in forecasting, which is referred to as aggregation level, m. The aggregated series is then created by summing

up the values inside each bucket. The number of aggregate observations is $[N/m]$, where N is the number of periods, and the $[x]$ operator returns the integer part of x (Rostami-Tabar et al., 2019). When we aggregate data to lower frequency domain such as annual using NOTA, we loose some information and AD is losing some of the sensitivity compared to AF. AF may better capture detailed information at the higher frequency resulting in better accuracy. However, this might be also affected by forecast horizon, as using AF might not be useful when forecasting far ahead into the future.

There are some studies that investigate this question when considering forecasting at one single level of aggregation (Rostami-Tabar et al., 2013, 2014; Kourentzes et al., 2017) or forecast combinations using multiple temporal aggregation levels (Kourentzes et al., 2014) or temporal hierarchies (Athanasopoulos et al., 2017). These approaches have been applied (Nikolopoulos et al., 2011; Petropoulos and Kourentzes, 2014) in both intermittent demand (Nikolopoulos, 2021) and fast-moving demand contexts (Athanasopoulos et al., 2017).

The overall conclusion is that both aggregate forecast and aggregate data approaches may outperform each other. Their performance may depend on the presence of the autocorrelation in the original series, aggregation level, forecast horizon and the employed forecasting method (see, e.g., Boylan and Babai (2016);Rostami-Tabar et al. (2021);Rostami-Tabar et al. (2014);Nikolopoulos et al. (2011)). We should note that this study only applies to the case of a single level of aggregation, we can extend this study to examine the case of using multiple level of aggregation, temporal hierarchies or multi-output forecast in the future.

Despite recent developments in this area, there is still a lack indications on which temporal aggregation approach should be used to forecast a time series, given its features. To our knowledge, this is the first study that explores the association between time series features and model performance in the context of forecasting by temporal aggregation (TA). The need for such research has also been emphasised by @babai2022demand in a review article. This study contributes to the area of time series forecasting and intends to shed lights on how the performance of temporal aggregation approaches (i.e. both AF and AD) is associated with time series features. To that end, we use 48,000 time series from the monthly M4-competition dataset. First, we examine how the features of time series changes going from a high granularity level (e.g. monthly) to a low granularity (e.g. annual). We then build machine learning models to describe the association between the original time series features and the forecasting performance of temporal aggregation approaches. Next, we use models' outputs to discover which features are critical in predicting accurately the performance of temporal aggregation approaches, followed by an interpretation of features associated with the forecasting performance. This will help us to provide recommendations to forecasters and decision-makers on which approach to use.

The research objectives are as following:

1. We measure 42 features of the time series at the original level (e.g. monthly) and at various levels of temporal aggregation (e.g. quarterly, annual) using the monthly M4 competition.

2. We reveal how time series features change as we aggregate data from high frequency (e.g. monthly) to low frequency (e.g. annual).

3. We assess the forecast accuracy performance of AD and AF approaches for the forecasts generated by the the Exponential Smoothing State Space (ETS) model.

4. We build machine learning models using time series features as predictors to accurately predict which approach (AD or AF) performs better.

5. We examine the association between time series features and the forecasting performance of these approaches.

The rest of the paper is organised as follows: section 2 provides a brief overview of the use of temporal aggregation in time series forecasting. Section 3 describes the empirical experiment design, forecasting approaches, method and forecast accuracy metrics. Section 4 describes time series features and presents the time series features for monthly M4 competition dataset, followed by analysing how non-overlapping temporal aggregation affects time series features. We then examine the forecasting performance of AD

and AF approaches. Section 5 presents machine learning algorithms and their performance on accurately classifying the performance of AD and AF for a given time series and its features. Section 6 presents the important features and their connection with the performance of temporal aggregation approaches. Section 7 provides concluding remarks and an agenda for future research.

2. Research background

In practice, a time series is generally stored at a single level of time granularity. When the time series is available at a higher level of granularity (e.g. hourly), it is often expected to generate a forecast at a lower granularity level (e.g. daily) over several time periods. Therefore, a forecast of the total value over several time periods ahead (i.e. lead-time or aggregation level) is required (Mohammadipour and Boylan, 2012). In these situations, an obvious option, that is often recommended in practice (Goodwin, 2018), would be to first transform the higher granularity time series into the lower granularity that matches the forecast requirement, and then produce the forecast. The transformation is generally performed using non-overlapping temporal aggregation.

Another approach is to aggregate forecast rather than data. In that case, we first produce forecasts using the higher granular time series for the forecast horizon (i.e. multiple periods ahead) and then aggregate them.

The application of NOTA approach in time series forecasting has been initially studied in the econometric literature. They evaluated how NOTA may change the structure of Autoregressive Integrated Moving Average (ARIMA) processes (Wei, 1978; Rossana and Seater, 1995). This literature is in favor of aggregate data using NOTA. They show that it leads to accuracy gains under the assumption of ARIMA process and using an optimal conditional mean forecast.

Rostami-Tabar et al. (2013) and Rostami-Tabar et al. (2014) further explored analytically the effect of NOTA on forecasting performance at both aggregated forecast horizon (lead-time) and the disaggregated level using Mean Squared Error (MSE). They assume that Single Exponential Smoothing (SES) forecasting method is applied to an ARIMA(1,1) time series. They determine the conditions under which aggregate data outperforms the aggregate forecast approach. They show that the superiority of each approach depends on the process parameters that are affecting the time series features, parameters of the forecasting method, and aggregation levels. They concluded that NOTA performs better when autocorrelation is not highly positive. In contrast, they show that high positive autocorrelation as one of the key features of time series, favorites AF approach. Rostami-Tabar et al. (2019) evaluated the impact of NOTA on forecasting demand and orders in a supply chain. They assume that the demand time series follows an ARMA(1,1), the stock policy is order-up-to-level and optimal forecasting is used to produce forecasts. They showed that although the NOTA does not lead to an accuracy improvement in terms of MSE at the retailer's level, however it can lead to MSE reduction at the manufacturer level and a reduction of the bullwhip effect.

Mohammadipour and Boylan (2012) also assessed analytically the effect of temporal aggregation when the time series process is integer autoregressive moving average, INARMA(p, q), processes. They demonstrated that aggregate data leads to lower MSE compared to aggregate forecast approach when the value of the autoregressive parameter is high.

The potential forecasting benefit of TA was investigated by Willemain et al. (1994) and Nikolopoulos et al. (2011) in the context of intermittent time series. Willemain et al. (1994) empirically compared the forecast accuracy improvement of AF and AD approaches. They showed that aggregating time series can lead to more accurate forecasts. Nikolopoulos et al. (2011) showed that NOTA approach may offer considerable improvements in terms of forecast accuracy. Further studies (Babai et al., 2012; Petropoulos and Kourentzes, 2015; Kourentzes et al., 2014; Spithourakis et al., 2011) confirmed empirically the forecast and stock control improvement resulted from NOTA. These studies covered both intermittent and fast moving time series.

Athanasopoulos et al. (2011) investigated the benefits of aggregate forecast versus aggregate data in an empirical study consisting of 366 monthly series and various forecasting methods including state space models for exponential smoothing (ETS), ARIMA, and Theta. Their findings are in favor of aggregating forecasts. They found that aggregating forecast from either monthly or quarterly to yearly leads to more forecast accuracy improvements than yearly forecasts generated from the NOTA yearly series. Using time

series of order and point-of-sale in a retail supply chain, Jin et al. (2015) assessed the benefits of NOTA for forecast accuracy. They found that NOTA leads to more accurate forecasts when the autocorrelation of time series is not highly positive. In a study where a high frequency time series is used, Luna and Ballini (2011) showed that in daily forecast of cash withdrawals, NOTA results in similar or more accurate forecast than using hourly series.

Some studies have investigated the benefits of producing forecasts using multiple time series resulted from NOTA approach, corresponding to multiple levels of aggregation. Forecasts are generated at each level and then combined to obtain the required forecast. Kourentzes et al. (2014) recommended using multiple levels of aggregation and combining the separate forecasts (MAPA). Multiple studies in intermittent time series forecasting, promotional modeling and inventory management highlighted the gain of using multiple TA levels (Petropoulos and Kourentzes, 2014; Kourentzes and Petropoulos, 2016; Barrow and Kourentzes, 2016). Athanasopoulos et al. (2017) proposed Temporal Hierarchies Forecasting (THieF), which creates multiple temporally aggregated series using NOTA, generate forecasts and then reconciles them to obtain the required forecast.

A stream of research investigates the use of features in time series forecasting based on multiple time frequency spaces in visibility graphs. Liu et al. (2020) proposed a multiple time-frequency spaces fuzzy interval forecasting model using datasets from energy and finance. The original series is decomposed into different components, which are then used to reconstruct a group of time series at different temporal scales. Next, a prediction interval forecast is generated for the different reconstructed time series that are then aggregated using the induced-ordered weighted averaging aggregation operation to generate the final forecast. Hu and Xiao (2022a) suggested a novel time series forecasting model based on a new metric measuring nodes similarity in visibility graph. In the proposed model, time series is first converted into a visibility graph. Next, similarities between nodes are determined and finally forecasts are generated using the normalized similarity distribution. Hu and Xiao (2022b) investigated the features of time series to generate accuracy forecasts from the perspective of fuzzy interaction between nodes. They used a fuzzy cognitive visibility graph to convert the time series into a pair of directed weighted graphs. Then, the weighted multi-subgraph similarity is developed to calculate the similarity between nodes. They then proposed a novel forecasting method for time series forecasts based on fuzzy similarity distribution that can efficiently capture the spatio-temporal dependency in the time series data. The empirical results confirmed the benefits of leveraging fuzzy interaction for time series forecasting based on the visibility graphs.

Although all approaches including AF, AD and using multiple TA levels have demonstrated forecasting gain, arguably none of them are suitable to be used in every situation. Almost all studies in the literature reports an overall accuracy (e.g. average) rather than accuracy at the time series level). We argue that some time series features may favorite using a particular approach over others. However, there is no study in the literature that investigates the potential association between time series features and temporal aggregation performance, which is the main aim of this study.

3. Experiment framework

In this section, we describe the design of the empirical experiment used in this study. We first present the study framework, then describe AD and AF approaches and the ETS forecasting method, followed by forecasting error metrics.

Figure 2 illustrates the framework of the experimental design performed in this study. The framework consists of several steps which are described as follows:

1. The original monthly time series is transformed into temporally aggregated series for a given aggregation aggregation level, m.

2. The ETS forecasting method is applied to the original series to generate forecast for m periods.

3. Forecasts are generated using ETS model for temporally aggregated series (forecasts from AD).

4. Forecasts per each period are added up to obtain the forecast over the aggregated horizon (forecasts from AF).

5. Forecasts are generated for temporally aggregated series (forecasts from AD).

6. Forecast accuracy is calculated for each series.

7. A database consisting of time series features and the performance of the AF and AD approaches is constructed. Time series features are used as an input (predictors) and the winning approach (labeled as AF or AD based on the minimum error metric) creates the response variable.

8. Several machine learning (ML) models are built to accurately predict the superiority of AF/AD approaches using the data created in step 7. These models include: 1) Logistic regression (LR), 2) Linear discriminant analysis (LDA), 3) Quadratic discriminant analysis (QDA), 4) K-Nearest Neighbors (KNN), 5) Lasso, 6) Generalised Additive Model (GAM), 7) Boosting, 8) Support Vector Machine (SVM), 9) Random Forest (RF), 10) Google Brain TensorFlow model (TensorFlow), 11) Facebook's Deep learning Torch model based on tensors & neural networks (DL Torch), 12) extreme gradient boosting (XGBoost), 13) recurrent neural network (RNN), 14) convolution neural network (CNN), 15) feedforward neural network (FNN) and 16) Dynamic Time Warping (DTW). Researchers are referred to James et al. (2021) for a detail description of these approaches. Given the outperformance of the random forest among all models used in this study, we will describe it in more details in section 5.

The model can help us to identify the most important time series features that lead to the outperformance of RF. Moreover, it can also reveal how time series features are connected to the performance of AD and AF.

Due to the page restrictions, details regarding the initial set up, method of optimization and cost function of these algorithms are not presented in the paper, however they are available by request from the authors. The experiment has been conducted in R software, and the authors are willing to share the code for reproducibility.

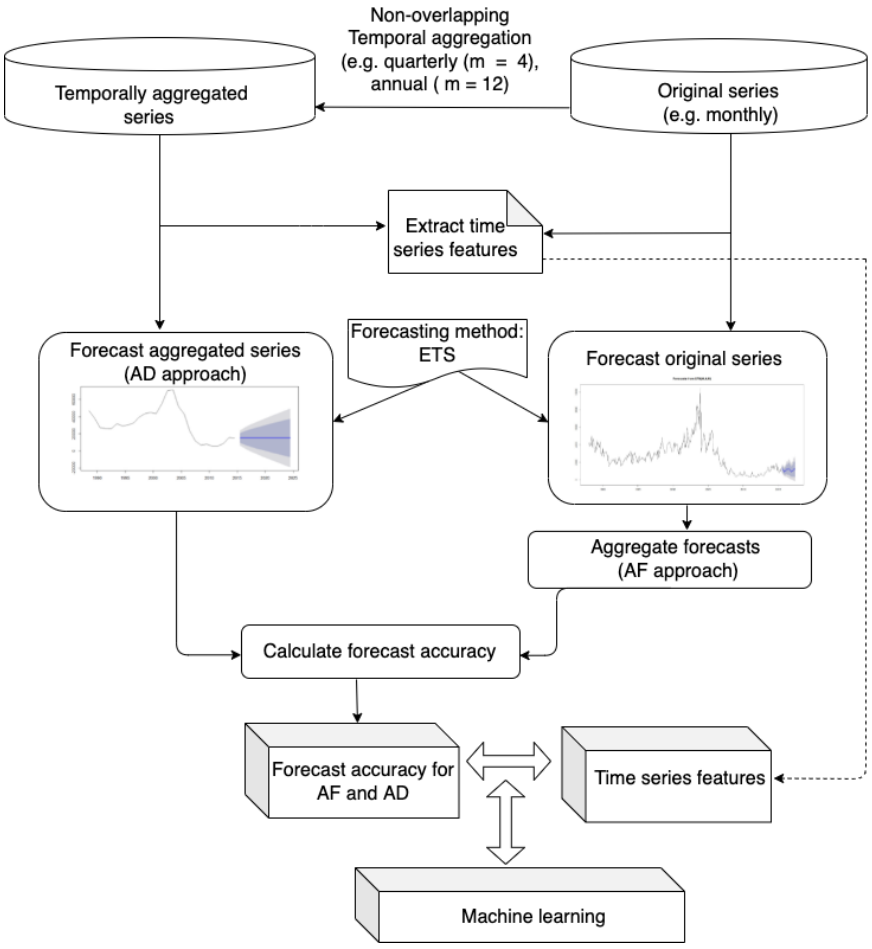

Figure 2: Design of the experiment framework

3.1. Forecasting approaches

We consider two different approaches to produce the forecast horizon aggregation (i.e. lead-time). Given that the original time granularity is monthly, we aim at generating forecasts for various aggregation levels that corresponds to 2-monthly (m = 2), quarterly (m = 3), 4-monthly (m = 4), semi-annual (m = 6) and annual (m = 12) time granularity.

The first approach is to aggregate forecasts (AF). This approach first involves generating base forecasts for m periods ahead. Base forecasts are then aggregated to create forecasts at the aggregation horizon level. The main advantage of AF is that there is no loss of information from the data because initial base forecasts are generated at the lowest disaggregated level. AF approach can capture the dynamics of the high frequency series, however they may be noisy and difficult to model. In the case of stationary time series, the AF approach may produce a more accurate forecast when the autocorrelation of the underlying demand series is highly positive (Rostami-Tabar et al., 2014). Conversely, there is an absence of researches in the case of non-stationary time series. Therefore, this research is dealing with both stationary and non-stationary time series and provides insight into the perplexity of interactions between different factors that influence on AF approach.

The second approach is to aggregate data (AD). This approach consists of applying the non-overlapping temporal aggregation approach to the original series, using the time granularity level for which the forecasting is needed. Following that the process of forecasting the temporally aggregated series for one atep ahead is performed. Benefits of AD can be seen in the reduction of the noise in the data, as well as in revelling the

more smooth patterns that exist in the series. The NOTA of the series usually sheds light on the key time series features, which are more notable and clearer as we perform the aggregation to the lower granularity levels (e.g. quarterly, annual).

3.2. Forecasting method

The exponential smoothing state space (ETS) models (Hyndman and Athanasopoulos, 2021) are used to produce out of sample forecasts, although our experiment design is model agnostic and could be expanded to any other forecasting model. ETS can capture trend, seasonality and error components in a time series through various forms such as additive, multiplicative or mixed. ETS accounts for 18 different exponential smoothing models. The automatic exponential smoothing model is used using *ets()* function in the forecast package (Hyndman and Khandakar, 2008) in R to produce forecasts for the original and aggregated time series. For each series, *ets()* identifies the most accurate model using the corrected Akaike's Information Criterion (AICc). Using an automatic forecasting method such as ETS is suitable for this study, as we can assume that the best model is selected for each series and this may help to separate the effect of the applied forecasting method from TA approaches (i.e. AF and AD).

3.3. Forecast accuracy measures

In this experiment, forecast accuracy evaluation is required at two different stages. We first use an error metric (e.g. Root Mean Squared Scaled Error) to quantify the forecast accuracy of AF and AD in generating time series forecast over lead-time for each time series and ETS method. We consider the in-sample and the out-of-sample sets in monthly M4 competition time series. We apply the ETS method to each series in the in-sample dataset and keep their forecast for the out-of-sample using time series cross validation with re-estimation. The forecast horizon for the original time series equals m, the aggregation level, and the horizon for non-overlapping aggregated series is one. We then compute the forecast errors over the test period of each time series. To evaluate the forecast accuracy and bias, several measures including Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Error (ME), Mean Absolute Percentage Error (MAPE), Mean Absolute Scaled Error (MASE) and Root Mean Squared Scaled Error (RMSSE) are used. We only present the results for RMSSE due to the space limit of the journal and also because it is the recommended error metric in M5 competition [@M5competition2022]. We share the R code and the Rmarkdown file through the GitHub repository, which provide the possibility of changing the error metric in the YAML section of the Rmarkdown to obtain results for other metrics. We should also note that similar conclusions are reached when using other accuracy metrics.

RMSSE is given by:

$$\text{RMSSE} = \sqrt{\text{mean}(q_j^2)},$$

where

$$q_j^2 = \frac{e_j^2}{\frac{1}{T-m}\sum_{t=m+1}^{T}(y_t - y_{t-m})^2},$$

where e_j is the forecast error, the difference between an observed value and its forecast. y_t is an observed value at period t, T is the length of time series, and m is the seasonal length (e.g. for monthly m=12. For non-seasonal data, $m = 1$.

Second, we need to report how accurately a classification model predicts the outcome (i.e. the most accurate approach labeled as AF and AD), when presented with a set of time series features at the original series. To that end, we report several statistical measures including misclassification error, F-statistics and area under the curve (AUC).

The misclassification error is calculated as following:

$$Misclassification_{error} = 100\% \times \left(1 - \frac{(t_p + t_n)}{N}\right),$$

where N is the total number of cases to predict, t_p is the true positive (i.e. when the model correctly predicts the outperformance of AF approach over AD) and t_n is true negative (i.e. when the model correctly predicts the underperformance of AF approach over AD).

F statistic is defined as:

$$F_{statistics} = 2 \times 100\% \times \left(\frac{(t_p/(t_p + f_p) \times t_p/(t_p + f_n))}{(t_p/(t_p + f_p) + t_p/(t_p + f_n)}\right),$$

Where f_p is false positive and f_n is false negative.

The false positive rate represents the fraction of cases in which AD was incorrectly classified as a right model to use, while the AF was the correct one. Similarly, the false negative rate represents the fraction of cases in which AF was incorrectly classified as a right model to use, while AD was the correct one.

We can also illustrate the trade-off between false positives and true positives using a curve. It is called a receiver operating characteristic curve, or ROC curve [@james2021statistical]. The ROC curve is a plot of the true positive rate (t_p, sensitivity) versus the false positive rate (f_p, 1 - specificity) for a set of thresholds. We can quantify this by calculating the area under the curve, or AUC. The higher AUC, the better the model does at predictions. The maximum value of AUC is 1, which would be considered as a perfect prediction. Conversely, a model that performs no better than chance will have an AUC of 0.5 and would be considered as a poor one.

In addition to the statistical measures discussed above, we also perform the utility evaluation of different models. The utility metric approximates the costs and benefit of wrong and correct classification. For that purpose, average RMSSE error is used as a utility metric.

To assess the prediction accuracy of models, we split the data created in the step 7 of Section 3 into a training and a test set. To that end, we randomly divided the data on train and test set in a 70/30% split. The train data (33600 cases) is used for training different algorithms, while the test data (14400 cases) is used to evaluate the performance of models.

4. Time series data, features and temporal aggregation performance

In this section, we first introduce the time series features extracted for each time series. Following that, we illustrate these features for the monthly M4 competition dataset. Then, we describe how non-overlapping temporal aggregation changes these features. Finally, we discuss the forecasting performance of AD and AF applied to the monthly M4 dataset.

4.1. Time series features

Table 1 presents the description of features used in this study. For each time series (original or aggregated), we compute a set of measures from the training data. These features are also described in detail by Wang et al. (2009), Hyndman et al. (2015) and Hyndman and Athanasopoulos (2021).

Given the complexity of the relationship between the time series features and the performance of temporal aggregation approaches, we have considered all 42 features as predictor rather than using only few limited numbers of features known to users such as the strength of trend or seasonality, to develop an accurate prediction model. Using a reduced number of features either selected based on their interpretability or using dimensional reduction techniques might reduce the predictability power of the model, but this could be an interesting avenue for future research.

Table 1: Time series features considered in this study and their descriptions

Feature	Description
mean	Mean
var	Variance
cv	Coefficient of variation
trend	Strength of trend, a value close to 1 indicate highly trended series
seasonal_strength	Strength of seasonality, a value close to 1 indicate highly seaonal series
entropy	Measure of how easy the series is to forecast. Entropy close to 0 shows a series is easy to forecast
lumpiness	Lumpiness is the variance of the variances of tiled (non-overlapping) windows
flat_spots	Number of sections of the data where the series is relatively unchanging
crossing_points	Number of times a time series crosses the median
nonlinearity	Extent of nonlinearity, if values around 0 series is linear. Large values shows nonlinearity
stability	Stability is the variance of the means of tiled (non-overlapping) windows
hurst	Hurst coefficient of a time series which is a measure of long memory
spike	Prevalence of spikes in the remainder component of the STL decomposition
linearity	Linearity of the trend component of the STL decomposition
curvature	Curvature of the trend component of the STL decomposition
peak	Timing of the peaks
trough	Timing of the troughs
x_acf1	First autocorrelation coefficient
x_acf10	Sum of squares of the first ten autocorrelation coefficients
diff1_acf1	First autocorrelation coefficient from the differenced series
diff1_acf10	Sum of squares of the first ten autocorrelation coefficients from the differenced series
diff2_acf1	First autocorrelation coefficient from the twice differenced data
diff2_acf10	Sum of squares of the first ten autocorrelation coefficients from the twice differenced series
seas_acf1	Autocorrelation coefficient at the first seasonal lag
x_pacf5	Sum of squares of the first 5 partial autocorrelation coefficients
diff1x_pacf5	Sum of squares of the first 5 partial autocorrelation coefficients of differenced series
diff2x_pacf5	Sum of squares of the first 5 partial autocorrelation coefficients of second-order differenced
seas_pacf	Partial autocorrelation coefficient at the first seasonal lag
unitroot_kpss	Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
unitroot_pp	Phillips-Perron statistic for testing if a series is non-stationary
arch_acf	Sum of squares of the first 12 autocorrelations of squared series
garch_acf	Sum of squares of the first 12 autocorrelations of residuals, after fitting an GARCH(1,1)
arch_r2	R2 value of an AR model applied to the squared series
garch_r2	R2 value of an AR model applied to residuals, after fitting an GARCH(1,1)
ARCH.LM	Statistic based on the Lagrange Multiplier (LM) test for autoregressive conditional heteroscedasticity (ARCH)
e_acf1	First autocorrelation coefficient of the remainder series
e_acf10	Sum of squares of the first ten autocorrelation coefficients of the remainder series
max_level_shift	Largest mean shift between two consecutive sliding windows of the time series
time_level_shift	Timing index of the largest mean shift between two consecutive sliding windows of the time series
max_var_shift	Largest variance shift between two consecutive sliding windows of the time series
time_var_shift	Timing index of the largest variance shift between two consecutive sliding windows of the time series

4.2. Time series features of monthly M4 data

We use the monthly M4 competition time series to empirically examine the performance of AD and AF and investigate the connection between their performance and time series features. The monthly M4 data contains 48,000 time series from the Economic, Finance, Demographics and Industry areas, while also including data from Tourism, Trade, Labor and Wage, Real Estate, Transportation, Natural Resources and the Environment (Makridakis, 2018). The monthly data is the most important data for the business applications (Spiliotis et al., 2020) and therefore the largest class in M4, containing almost half of the data (48000 time series).

For each time series in the monthly M4 dataset, we extract 41 features as described in Table 1 using *tsfeatue()* function in the tsfeatures package in R (Hyndman, 2020). Additionally, we include one more feature defined as origin, which is the origin of the time series M4 competition data, i.e. Economic, Finance, Demographics and Industry.

Figure 3 and 4 illustrates the corresponding distribution of each feature. Y-axis shows the frequency in

terms of the number of time series, and X-axis indicates the range of values extracted for each feature. These figures show that the monthly M4 data covers a wide range of time series features, which makes it a suitable dataset for this research problem. We describe some significant features observed in the dataset, and readers are referred to Figure 3 and 4 for a full illustration of the features.

The spectral entropy plot indicates that there is a range of time series from easy (less noise, more systematic information) to hard (more noise, less systematic information) to forecast. Coefficient of Variation is also skewed to the right and indicates less variability for the majority of series. The trend peaks near one and is skewed to the right indicating the strong presence of the trend in this dataset. We also observe that the nonlinearity feature peaks near zero and it is skewed to the right, which indicates the lack of nonlinearity in the time series. The autocorrelation lag1 and seasonal autocorrelation lag1 is skewed to the left, and is highly positive for many time series. We also measure the seasonal autocorrelation lag1 which is highly positive. Lumpiness is extremely low for almost all time series and stability has a range from low to high values. Both seasonal partial autocorrelation and the strength of seasonality indicate a lower strength of seasonality. Both stationaity and non-stattioanirity of series are measured using unitroot_pp and unitroot_kpss statistics. They indicate a strong presence of staitonairy time series. Curvature shows a sharp distribution centered around zero, which means that most series do not have stochastic or chaotic nature. Time series of stochastic nature are associated with curves displaying positive curvature in a neighborhood of their initial points, whereas curves related to chaotic phenomena have a negative curvature. Hurst has a left skewed distribution with almost all series with a value close to one indicating the presence of the long memory in the series, which is related to the autocorrelation.

The distribution of the e_acf10 values is skewed to right and the plot shows that for most of the time series e_acf10 is close to zero, which means the leftover from trend and seasonality seems to be random.

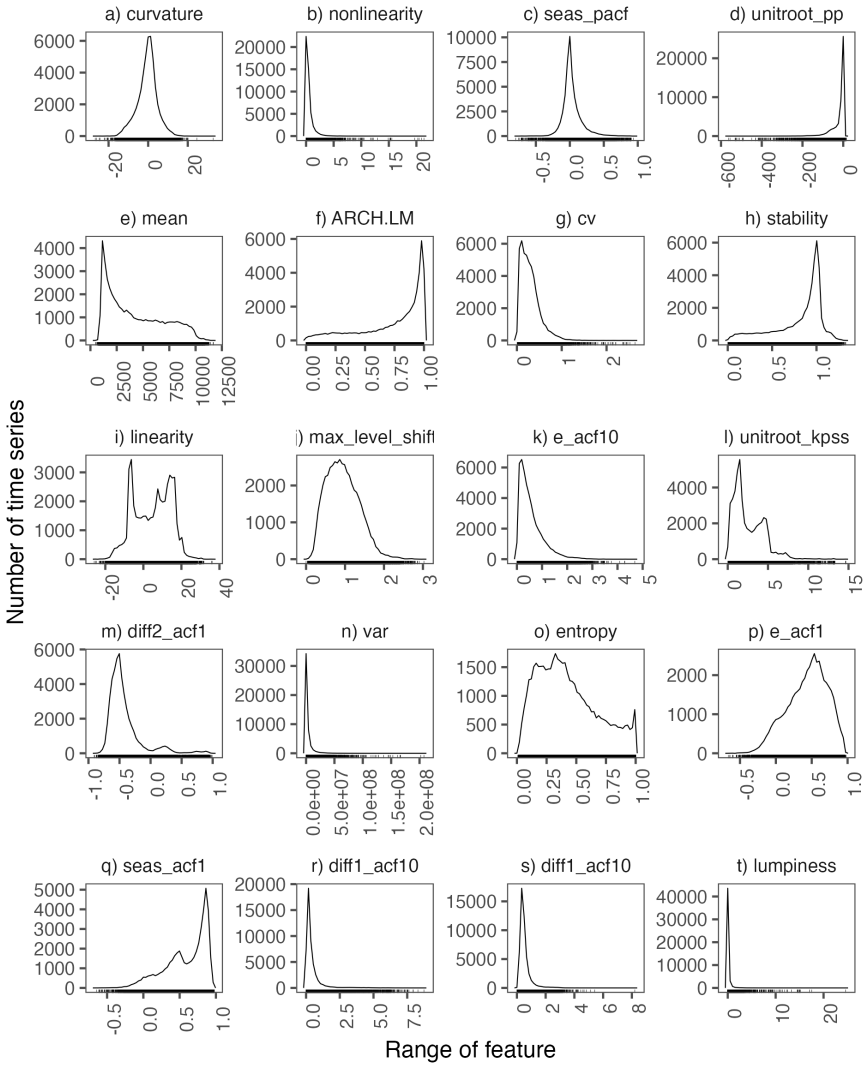

Figure 3: Features of monthly time series from M4 competition dataset

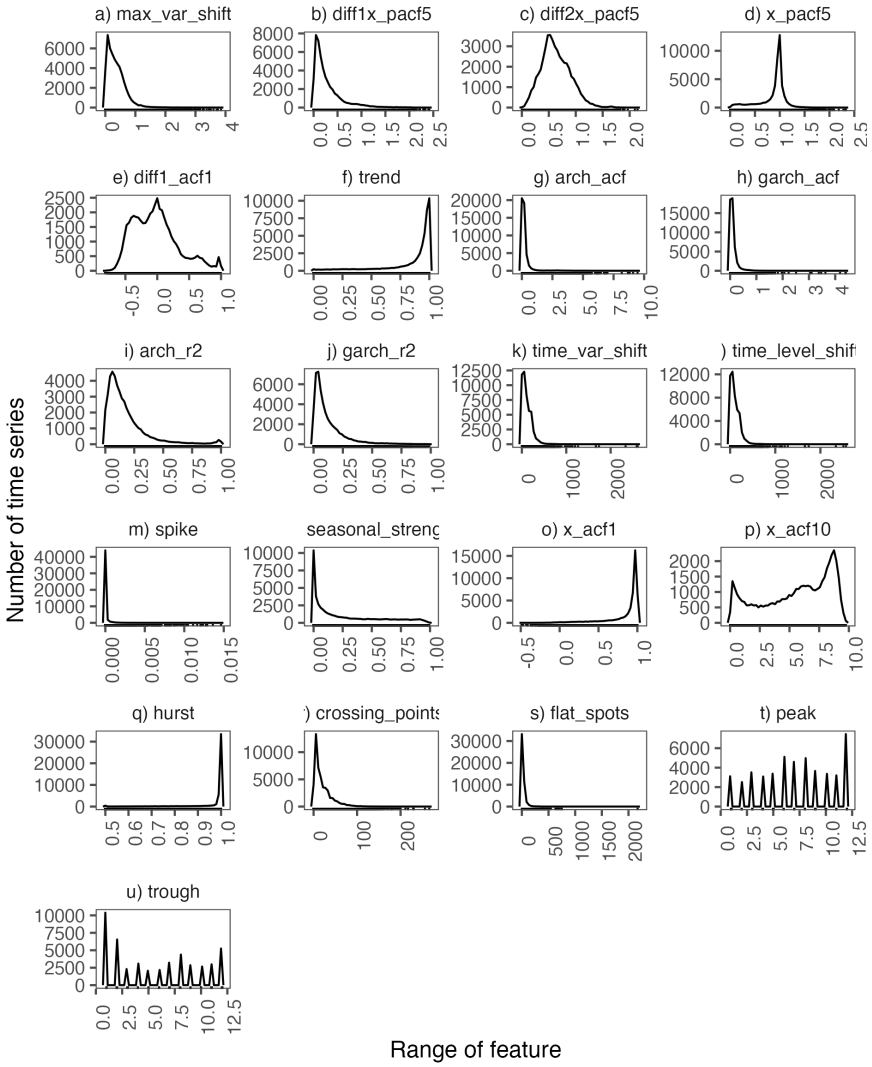

Figure 4: Features of monthly time series from M4 competition dataset (continue)

4.3. Effect of temporal aggregation on time series features

In addition to extract the features of monthly time series, we also compute features after transforming the time series to bi-monthly, quarterly, 4-monthly, semi-annual and annual granularity. We use these features to demonstrate how non-overlapping temporal aggregation may change time series features. To highlight the effect of NOTA, we first started by plotting the distribution of features at each level, however the difference between distributions was not revealing because features have log-tail distributions. Instead, we categorise features of the original series to into four categories using quantiles:

- If $0 < feature \leqslant 0.25$, Category = Very low
- If $0.25 < feature \leqslant 0.5$, Category = Very low
- If $0.5 < feature \leqslant 0.75$, Category = Very low
- If $0.75 < feature \leqslant 1$, Category = Very low

We then calculate the mean of features for each category and each time granularity. The results have been illustrated in Figure 5 and 6. They show how time series features change from monthly to annual

granularity for the time series features. The results indicate that as the level of time granularity increases from monthly to annual, some features become weaker and may disappear, while others dominate the series.

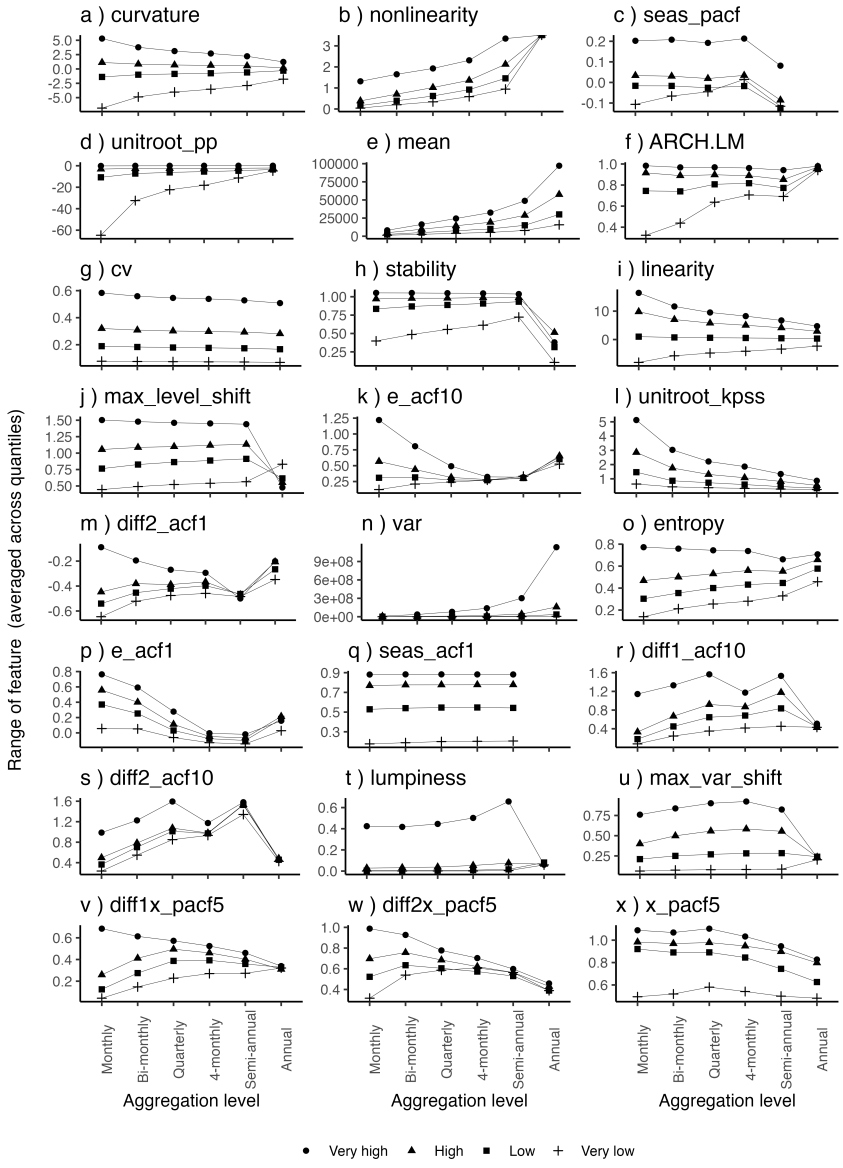

Figure 5: The effect of non-overlapping temporal aggregation on monthly time series features of M4 competition dataset

We observe a decrease in curvature as we aggregate and it approaches zero at annual level, while nonlinearity increases with aggregation indicating that series become more non-linear. Mean increases and variance decreases, however the effect is higher for very high category. We also observe that coefficient of variation decreases. Linearity is pushed toward zero if it is high or low, meaning that series at annual level losses its linearity feature. Both uniroot_pp and unitroot_kpss become close to zero with increasing aggregation level, highlighting the fact that series become more non-stationary if the monthly series is stationary, otherwise it remains almost the same. The measures related to seasonality such as seasonal strength, autocorrelation lag1 and the sum of squared of the first 10 autocorrelation decreases. It is interesting

to observe that entropy increases with aggregation if it is not very high at monthly level, meaning that series become difficult to forecast. However we see a reduction in entropy if it is very high at the monthly level. We also observe an increase for ARCH.HM especially if it is very low at the monthly level. If it is very close to one, it stays almost unchanged.

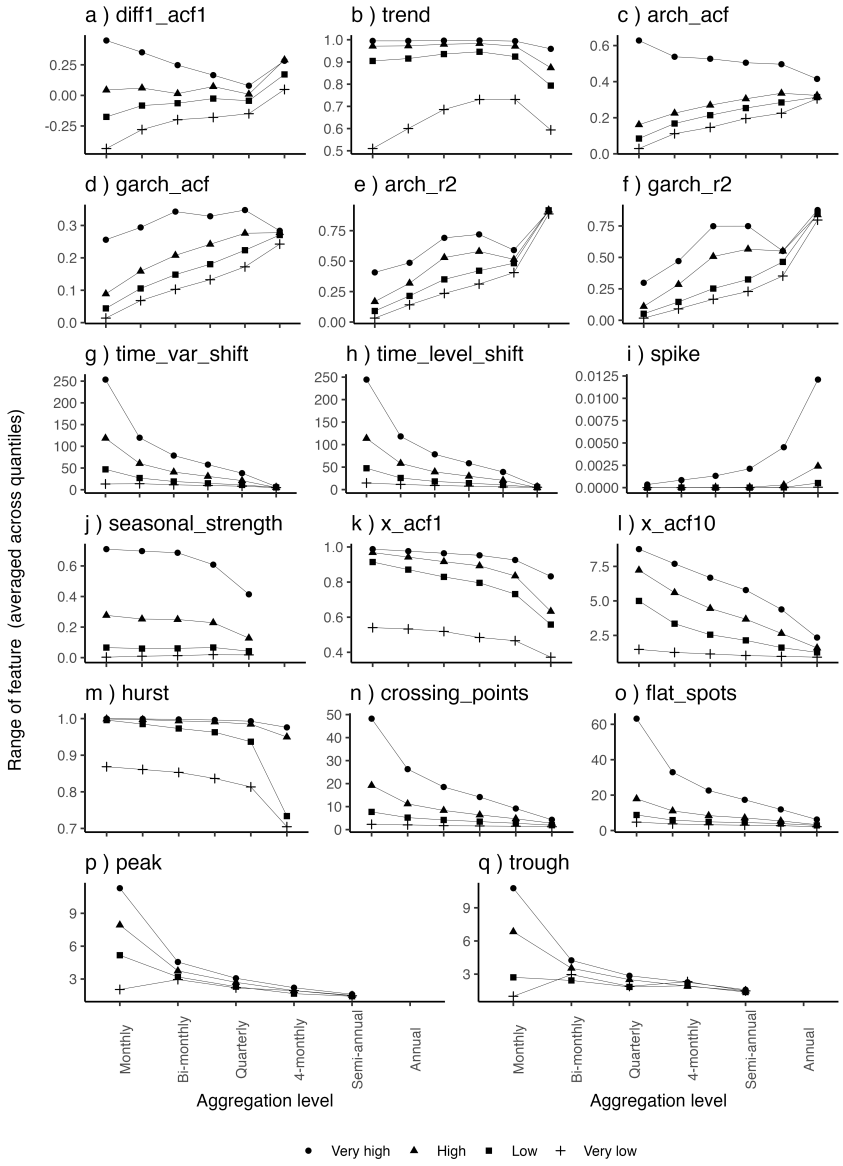

Figure 6: The effect of non-overlapping temporal aggregation on monthly time series features of M4 competition dataset (continue)

We should note that some features are not stable and show chaotic behaviors at the annual level. This might be because these features are not measurable for some time series. Therefore, they are returned as NA (not known) and are removed when calculating the mean of each category, hence the disorder of these features at the annual level.

4.4. Forecast accuracy evaluation of AD and AF approaches

In this study, we aim at generating forecasts for a forecast horizon aggregation, lead-time period, using AF and AD approaches. While the original time series granularity is monthly, we require forecasts to be produced at Bi-monthly (aggregation level = 2), Quarterly (aggregation level = 3), 4-monthly (aggregation level = 4), Semi-annual (aggregation level = 6) and annual (aggregation level = 12).

Figure 7 displays a boxplot of RMSSE measure created for all 48,000 series at various level of forecast granularities. Results demonstrate that both AF and AD approaches might outperform each other for different time series. However, AF approach is generating consistently more accurate forecasts through all aggregation levels and the difference becomes more pronounced as aggregation level increases. The difference is especially notable at the annual level. From 48000 monthly time series, the AF approach was superior in 30,147 cases (63%) compared to 17,853 cases (37%) for AD when forecasting annual time granularity. These results might be surprising, because common recommendation based on practice and in the literature (see Section reflit) is to use AD over AF. However, we should note that almost all these studies report the overall accuracy improvement rather than the forecast accuracy at the series level. Our results show that recommending AD to forecast over the forecast horizon aggregation/lead-time might not necessarily lead to more accurate forecast. Moreover, AF is shown to be a very competitive forecasting strategy. This might be due to the time series features of the monthly time series discussed in section 4.2.

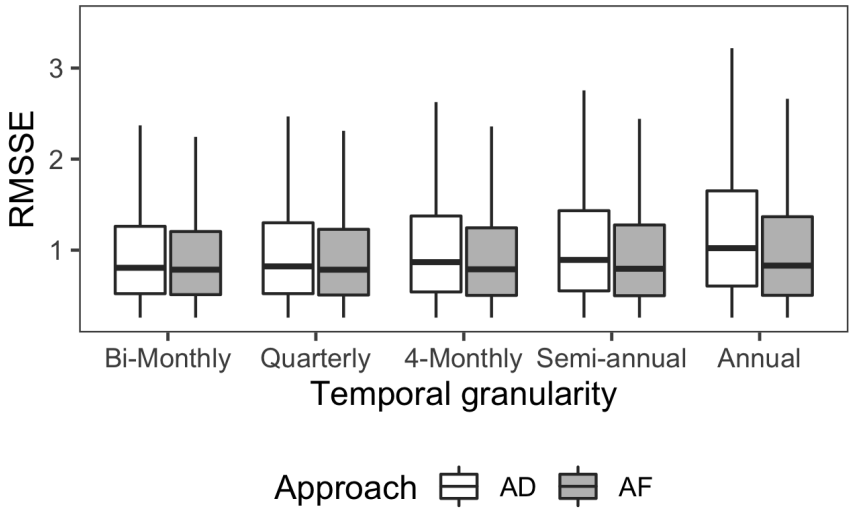

Figure 7: RMSSE errors through aggregation levels.

We have also conducted a statistical test using MCB (Multiple Comparison with the Best method) (Koning et al., 2005) to assess the statistical significance in the performance of AF and AD approaches.

From Figure 8, it is evident that there is a statistical difference between forecasts generated from AF and AD. The forecasts generated from AD approach are less accurate, while the strategy of aggregating forecasts proves to be more accurate. The results were cross-validated through multiple forecast horizons (h=12, 24 and 60) and via several other error metrics as described in 3.3. The main conclusions are almost perfectly aligned with results demonstrated in Figure 8.

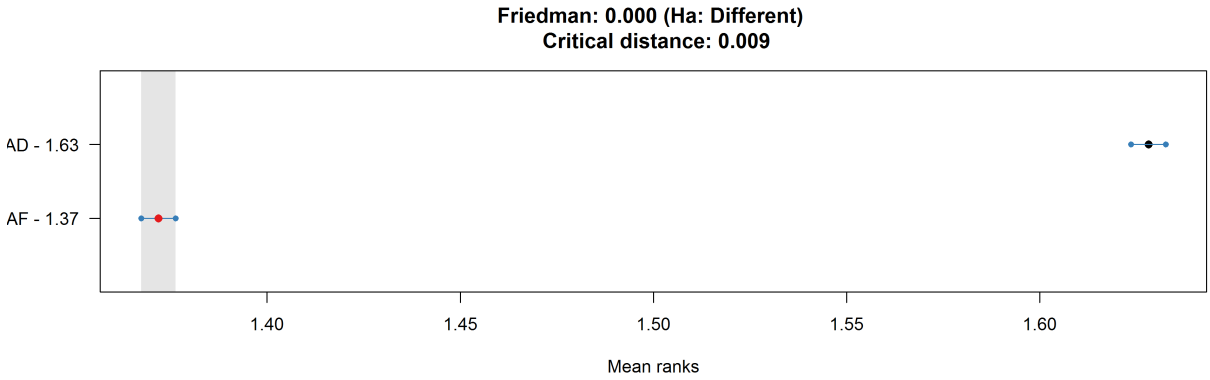

Figure 8: Performance of AF and AD models evaluated through MCB test. RMSSE values are used for computing the ranks and a 95 percentile confidence level.

In Figure 7, we showed that both AF and AD have diverging performance through different levels of aggregation. Moreover, Figure 5 and 6 demonstrate the evolution of the time series features through these levels. We believe that the presence of certain time series features might favorite AD or AF. We should note that the results presented for the rest of the paper are based on forecasts produced for the annual level (aggregation level = 12) using AF and AD approaches. This is because we see the highest difference in the performance at the annual level, and when the time series granularity is monthly, very often forecast is required at the annual level.

In the next section, we build machine learning models to explore the association between time series features and the forecasting performance of AF and AD performance.

5. Machine learining models

In this section, we use several models to shed lights on the association between time series features and the performance of these approaches. Using the time series features extracted at monthly time granularity and the RMSSE error metric generated for each approach (i.e. AF and AD), we build machine learning algorithms to reveal relevant sets of features affecting the performance of these approaches. The first step in building such a model is to construct a dataset consisting of time series features and model class labels for a given time series. Therefore, we turn this problem into a classification supervised learning, using features as predictors and the winning approach labeled as AD or AF as the response variable or the outcome for each time series. We discover, extract and present the details on which time series features are the most influential on the accuracy of AF and AD approaches. In addition to interpretability, the algorithm should be able to accurately predict which model to use with a given set of time series features in the original series. We have used the RMSSE at the annual level given the pronounced difference in accuracy at that level.

Figure 9 shows a grid of scatterplots showing the bivariate interactions between various pairs of features. Each point corresponds to one time series. The figure also highlights the winning approach (i.e. either AF or AD) when forecasting for annual time granularity. The black dots represent the situations where AD was more accurate, while yellow dots represent the opposite. Given the number of features used in this study, it is unfeasible to include the scatterplot matrix for all pairs of features. Figure 9 shows only the pairs plot for 10 features.

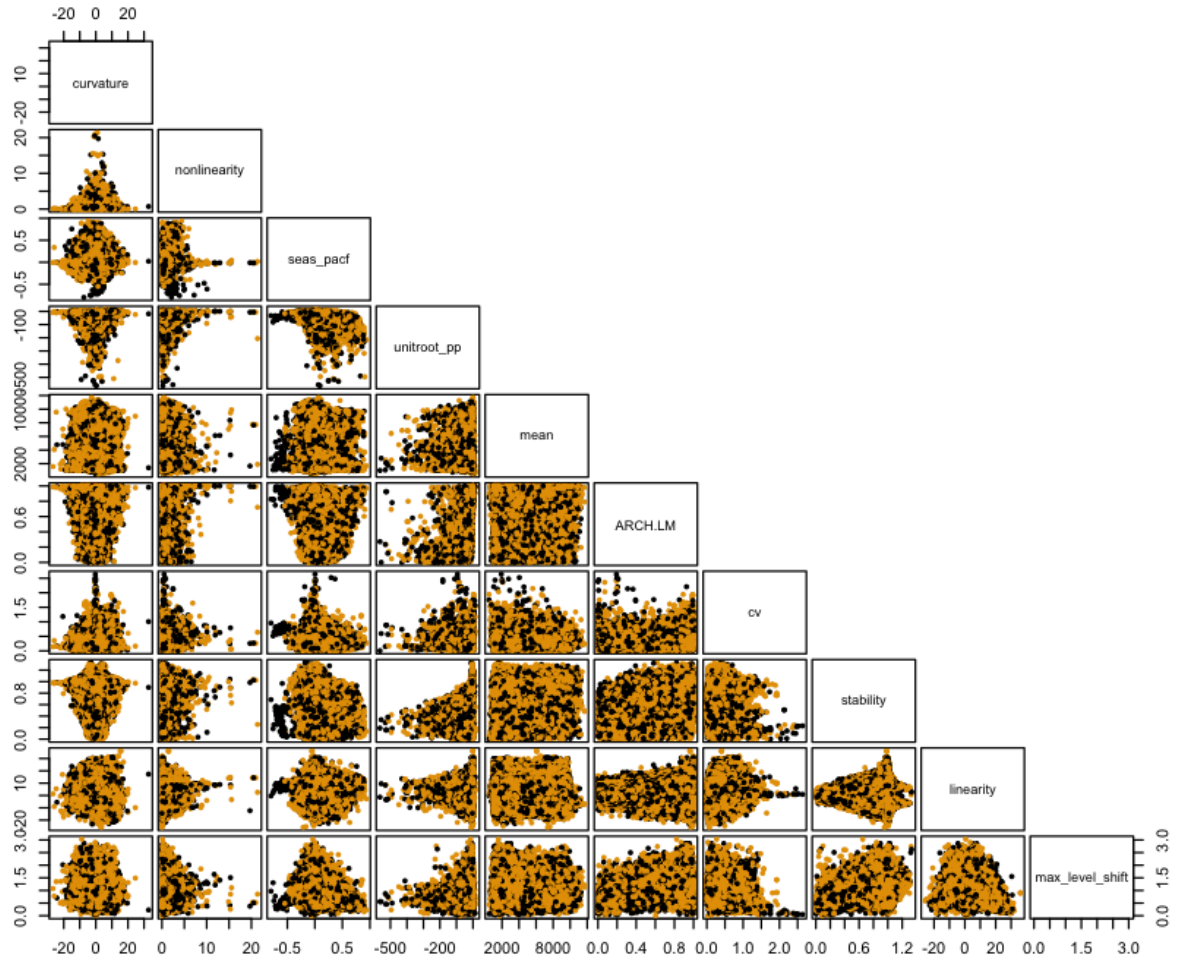

Figure 9: Time series features matrix. Each point indicates one single time series with a feature in x-axis and another at y-axis. The black and yellow colors show the outperformance of AD and AF, respectively.

Figure 9 demonstrates a perplexed relationship between features and AD/AF approaches, with a lot of noise, confounding features and lack of clear decision boundaries between AD and AF performance. It is clear that some features might have linear or non-linear relationships with each other, while others might be unrelated. For most of features presented in Figure 9, there is no strong relationship.

Considering the complicated boundaries and relationship between predictors, we develop several machine learning algorithms to discover patterns and connections between time series features and the accuracy of the AF and AD approaches.

5.1. Evaluating the prediction accuracy fo models in classification

In this section, we examine the prediction accuracy of the ML models in classifying whether AD or AF should be used to forecast the horizon aggregation of a given time series, based on its features. We report the prediction accuracy of the ML classifiers using measures discussed in section 3.3. We have designed an interactive ShinyApp that includes the prediction accuracy of ML approaches and can be accessed by readers [2].

[2]https://supplychainanalytics.shinyapps.io/Evaluation_of_ML_models/.

Table 2: Comparison of different ML models

	F-statistics	Misclassification error	AUC
LR	0.6083881	0.4383333	0.5676151
LDA	0.7655532	0.3718750	0.5130521
QDA	0.7121420	0.3929861	0.5499207
KNN	0.6882851	0.3911806	0.5814969
Lasso	0.6368112	0.4252083	0.5677939
GAM	0.6042137	0.4383333	0.5707890
RF	0.7773134	0.3367361	0.5704724
Boosting	0.7646663	0.3640972	0.5267067
SVM	0.7646663	0.3640972	0.5318491
DTW	0.7093325	0.3888889	0.5616327
FNN	0.7649972	0.3724306	0.5128323
DL Torch	0.7649434	0.3678472	0.5230656
XG Boost	0.7688268	0.3423611	0.5728495
TensorFlow	0.7669575	0.3690972	0.5166527
RNN	0.7659351	0.3710417	0.5142032
CNN	0.7662412	0.3700694	0.5158014

Table 2 demonstrates that RF model has the best performance with the lowest misclassification error, highest F statistics and one of the best AUC statistics.

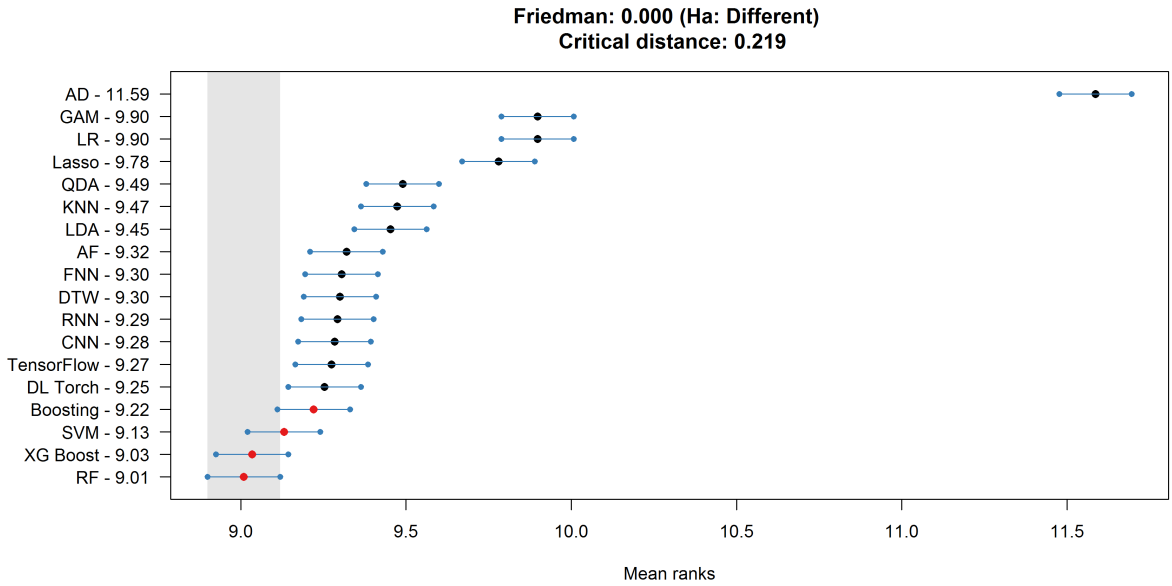

Figure 10: Performance of different models evaluated through MCB test. RMSSE errors are used for computing the ranks and a 95 percentile confidence level.

The Figure 10 shows the result of MCB test conducted to examine the difference between the performance of ML algorithms. The results show that there is a statistically significant difference between the performance of RF and all other models. According to the MCB test, AD approach generated the most inaccurate forecasts and therefore drastically underperformed compared to the others. All other ML models have much

Table 3: Cost and benefit matrix

		Actual	
		AD	AF
Predicted	AD	0.902872	1.6568039
	AF	1.520416	0.8669348

better performance, and the RF comes out as the ultimate winner in terms of mean rank classification accuracy on the test data set.

We also perform the utility evaluation of ML models. The results are presented in the Table 3. The Table 3 represents the average values of RMSSE for AD and AF approaches when the correct classification was AD and AF, respectively. When the true model to use is AD (17,853 causes), the average RMSSE error is 0.9028 for AD, and 1.5204 for AF. On the contrary, when the true model to use is AF (30,147 causes), the average RMSSE error is 1.6568 for AD and 0.8669 for AF. In this way, we design the average costs and benefits associated with classification decisions and we are able to evaluate the ML models performances via practical utility measure contribution connected to their usage, and not just via standard statistical measures. This kind of performance evaluation has much more value for practitioners assessing the practical values of each ML model. The presented utility evaluation can be further extended in practice by linking the monetary costs to the decision based on the forecast. However, domain knowledge on the area where forecasts are implemented is required.

Figure 11 demonstrates that RF has the smallest utility metric (cumulative average RMSSE) while performing the classification task on the test data set. We have also included the Ideal model, a model that always predict correctly, to highlight the deviation of ML approaches from this benchmark and to reveal the maximum possible margin for reducing the overall cumulative RMSSE. Therefore, our result shows that RF also provides the most accurate result for a given time series examined by utility metric.

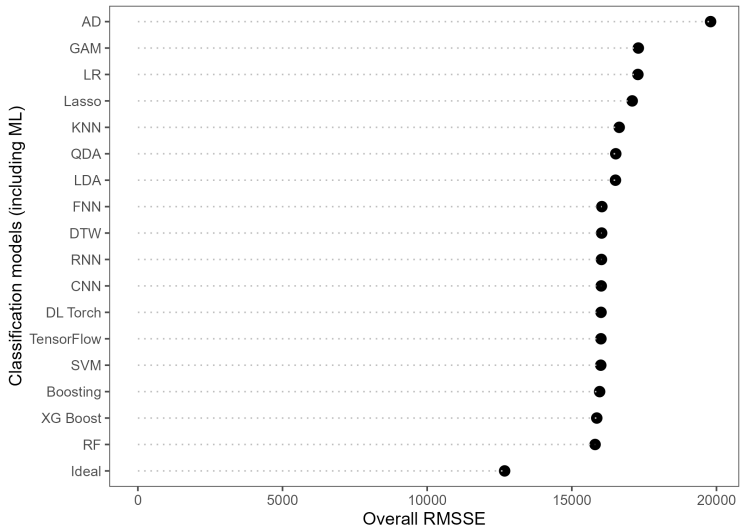

Figure 11: Utility accuracy comparison

We should note that we have conducted a comprehensive study and searched for optimal setups of each model rather than using automatic ML models. Additional information about the setup of models can be found in the Appendix or in the [GitHub repository](https://github.com/bahmanrostamitabar/time-searies-featute-temporal-aggregation).

Among several algorithms, the RF provided the most accurate results according to the several criteria. Therefore, we further discuss this approach in the next section in more details.

5.2. Building the random forest algorithm

Individually, these trees tend to overfit the data and generate future forecasts with large variance. But, at the same time, they produce a low bias. Therefore, in order to reduce the variance, the Bagging process is building many trees on slightly different train data (since the data is bootstraped-resampled) and averages forecasts of all the trees in one unique forecast. For the sake of improving the Bagging process, Breiman (2001) introduced the RF as an algorithm that mimics the Bagging process, but uses only a small random sample of the available features, for building each classification tree. The idea behind this is instead of just "shaking" the data via bootstraped-resampled process, we are introducing additional randomness in the process by "shaking" the features for the forecasting via random sampling of just part of the features. Therefore, each time a split in a tree is considered, a random sample of m predictors is chosen as a split candidates from the full set of p features. A fresh sample of m features is taken at each split, and typically $m \approx \sqrt{p}$ (Friedman et al., 2001). By forcing this process each time using the subsample of fresh features, the RF is decorrelating each tree. In this way, the process of averaging many different trees will reduce the variance more efficiently than in the case of correlated trees.

In order to generate the RF model, there are several questions and tuning parameters that need to be optimized so that the final model could generate reliable and accurate forecasts. For that purpose, Friedman et al. (2001) have proposed the following methodology presented in the Algorithm 1.

Algorithm 1 Random Forest methodology.

1: For $b = 1$ to B:

(a) Draw a bootstrap sample **Z*** of size N from the training data.

(b) Grow a random forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

i Select m variables at random from the p variables.

ii Pick the best variable/split-point among the m.

iii Split the node into two daughter nodes.

2: Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\widehat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

Classification: Let $\widehat{C}_b(x)$ be the class prediction of the bth random-forest tree.
Then $\widehat{C}_{rf}^B(x) = majority\ vote\ \{C_b(x)\}_1^B$.

The first question is how many features to randomly choose for training in each individual tree (i.e. tree depth)? The question of how many trees to use for fitting the RF model emerges, which also needs to be answered in parallel. In order to determine the optimal number of features and trees for building an RF model, the evaluation process is performed with a different number of features and trees in each split. Therefore, the resulting process represents the optimization problem of seeking the minimum of the surface generated by features in each split, trees and resulting out of the bag (OOB) forecasting error. (Please refer to Figure 12). OOB represents the misclassification error of the RF model while performing the classification tasks. Also, OOB is the approximation of the cross validated test error, since the process of fitting bootstrapped data sets consists of training the model on just part of the original data and then predicting the remaining part of the original data (i.e. out of bag data). By averaging OOB errors from many different tree models (which comprise the final random forest model), the final OOB is obtained.

Figure 12 reveals that increasing the number of trees used for generating RF model reduces the misclassification error. A similar scenario exists with features used in each split, where increasing the number of features (up to a certain point) reduce misclassification error. The presented surface is highly erratic and undulating, making it hard to visually determine global minimum. For that reason, the performance of the

four best RF models with feature splits (m = 6, 10, 21 and 42) are shown through a different number of trees used for fitting RF model (Figure 13).

The Figure 13 represents the optimal number of splits (randomly selected features) used during the process of building each classification tree, according to OOB forecasting error. As demonstrated in Figure 13 the OOB error is minimized on the depth of 10. Therefore, the optimal number of features (i.e. tree depth) for the RF model is 10.

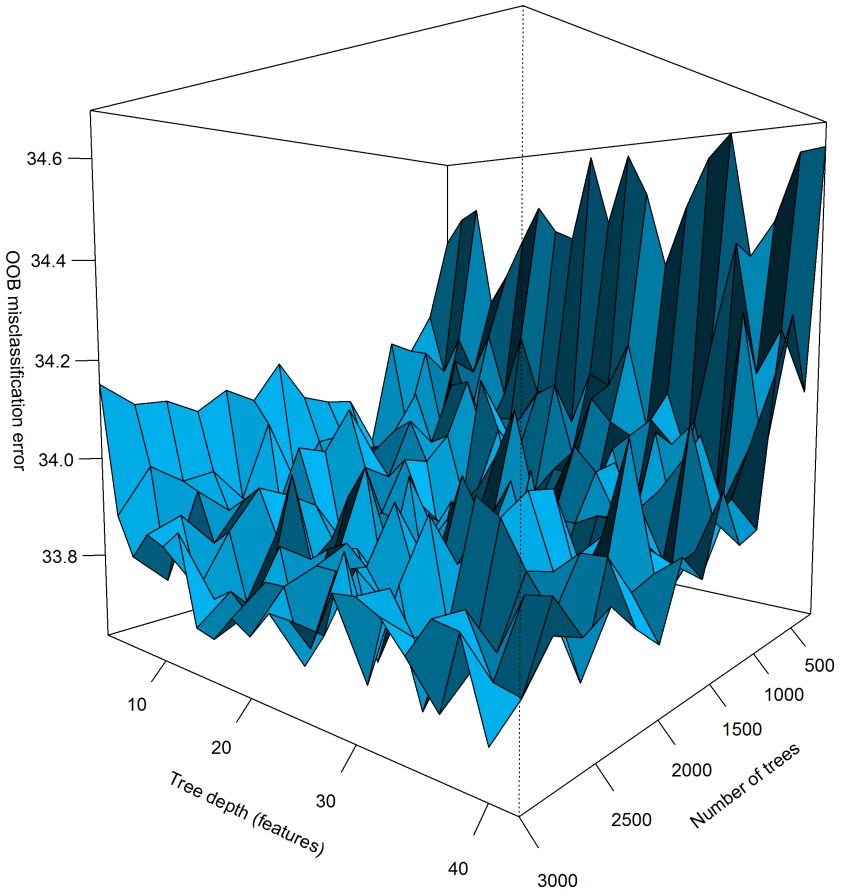

Figure 12: Surface plot of the number of features (depth), number of trees and misclassification error of random forest model.

The second tuning question is how many trees should be used for generating the RF model? One of the good merits that help in this process is the resilience of RF on overfitting (Friedman et al., 2001). Averaging process during the RF building phase remedies the negative effects of choosing a large number of trees for fitting the model, since this will not result in an increase in variance on the test set. This implies that the penalty of choosing a too large number of trees on models' overall performance is minor. The criteria for this judgment is based on the OOB misclassification error, which RF model produces on a given data set. Therefore, the optimal number of trees is chosen by determining the number of trees where the OOB misclassification error is stabilized (Figure 13). From Figure 13, it is clear that error is stabilized by using the 3000 trees.

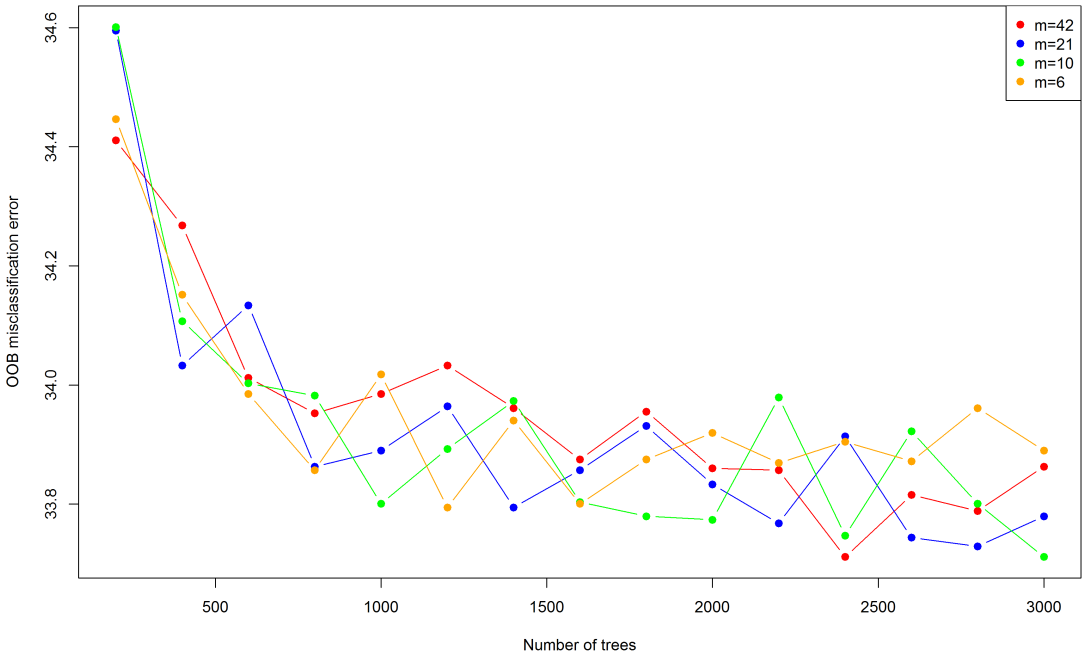

Figure 13: Determining the optimal depth and the of number the trees in the random forest model.

6. Association between time series features and AF/AD performance

Since the RF consists of a large number of trees, it is no longer possible to represent the resulting statistical learning procedure using a single tree, and it is not clear which variables are most important to the procedure (James et al., 2013). Therefore, the RF improves prediction accuracy at the expense of interpretability. Instead, it is possible to obtain the overall summary of the importance of each feature by measuring the mean decrease in the Gini index.

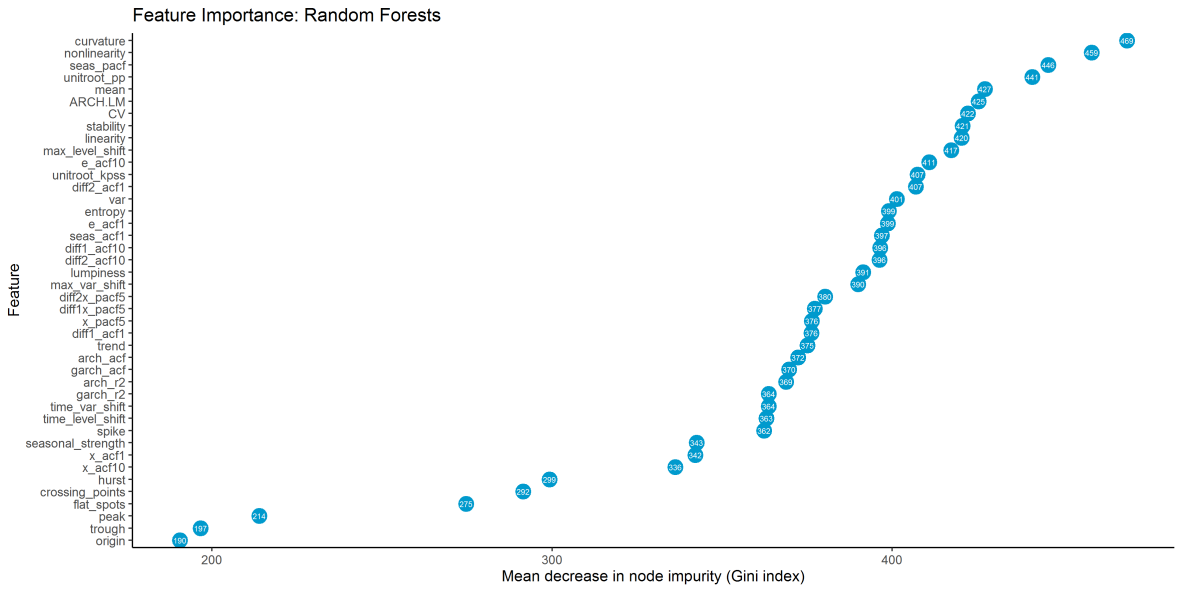

Figure 14: Predictor features importance spectrum for the M4 data. A feature importance is computed using the mean decrease in Gini index.

The Figure 14 demonstrates the overall importance of the features used in building trees during the random successive process of generating the RF model. Importance of the features is determined by the decrease in total amount of the node impurity by splits over a given predictor and averaging over all trees in RF. The decrease of the node impurity is measured by the Gini index. A large values of the Gini index indicates the important features. Clearly some features prove to be more important than others in classifying accurately the AF versus AD approach. The most important features are: *curvature*, *nonlinearity*, *seas_pacf*, *unitroot_up*, *mean* and *ARCHM.LM*; while *origin*, *through*, *peak* and *flats spots* seems to be the least important for predicting which temporal aggregation method to use (i.e. AF or AD). The quantity being modeled here is the probability of correctly choosing the AD versus AF, and the opposite. The least important features have approximately half of the importance as the most significant ones. Difference in a decrease of node impurity from the *curvature* to the *origin* feature is more than 250 units, indicating the wide range of features importance.

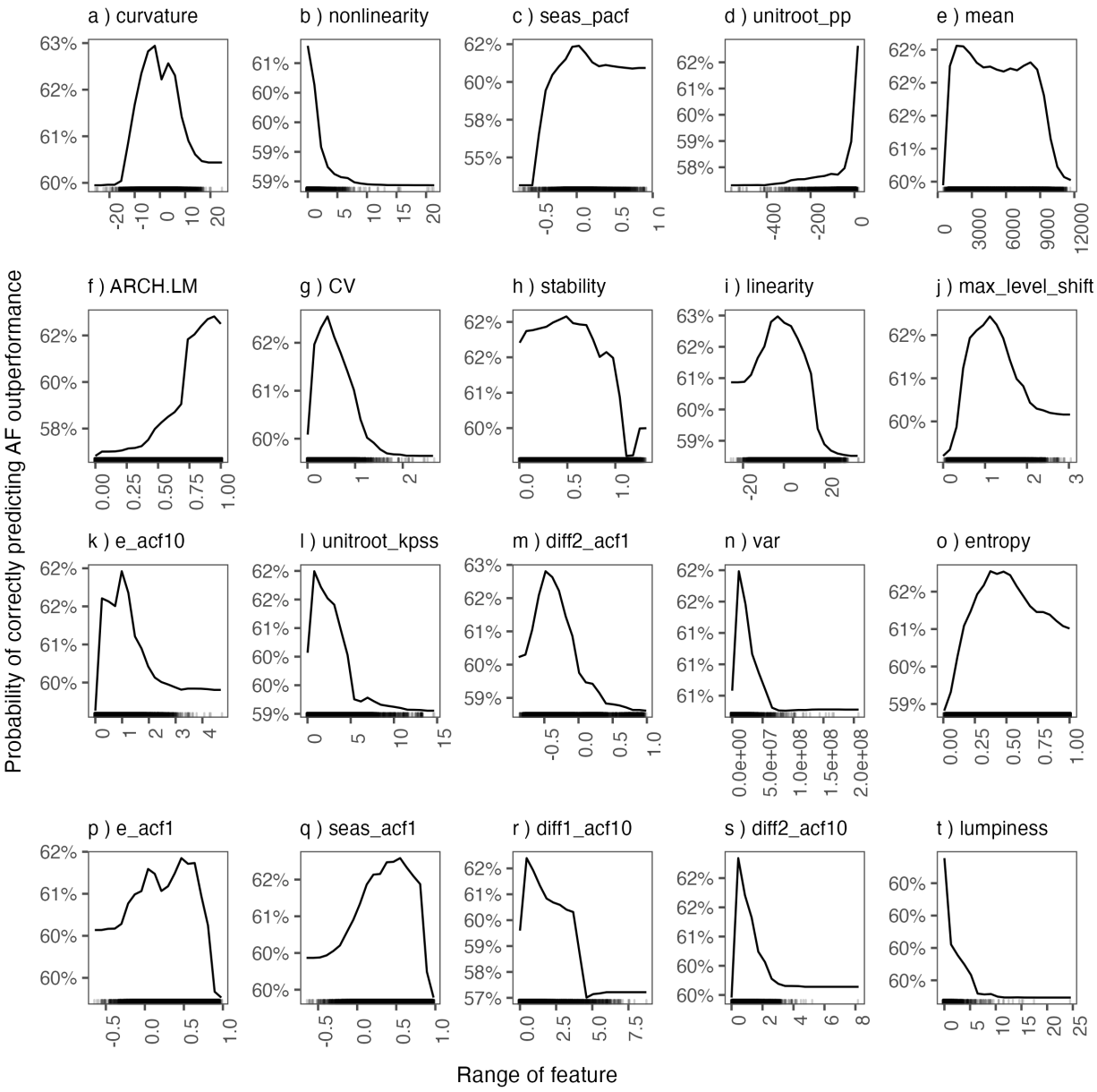

Figure 15: Partial dependence plot - the range of features vs. the probability of classifying AF versus AD

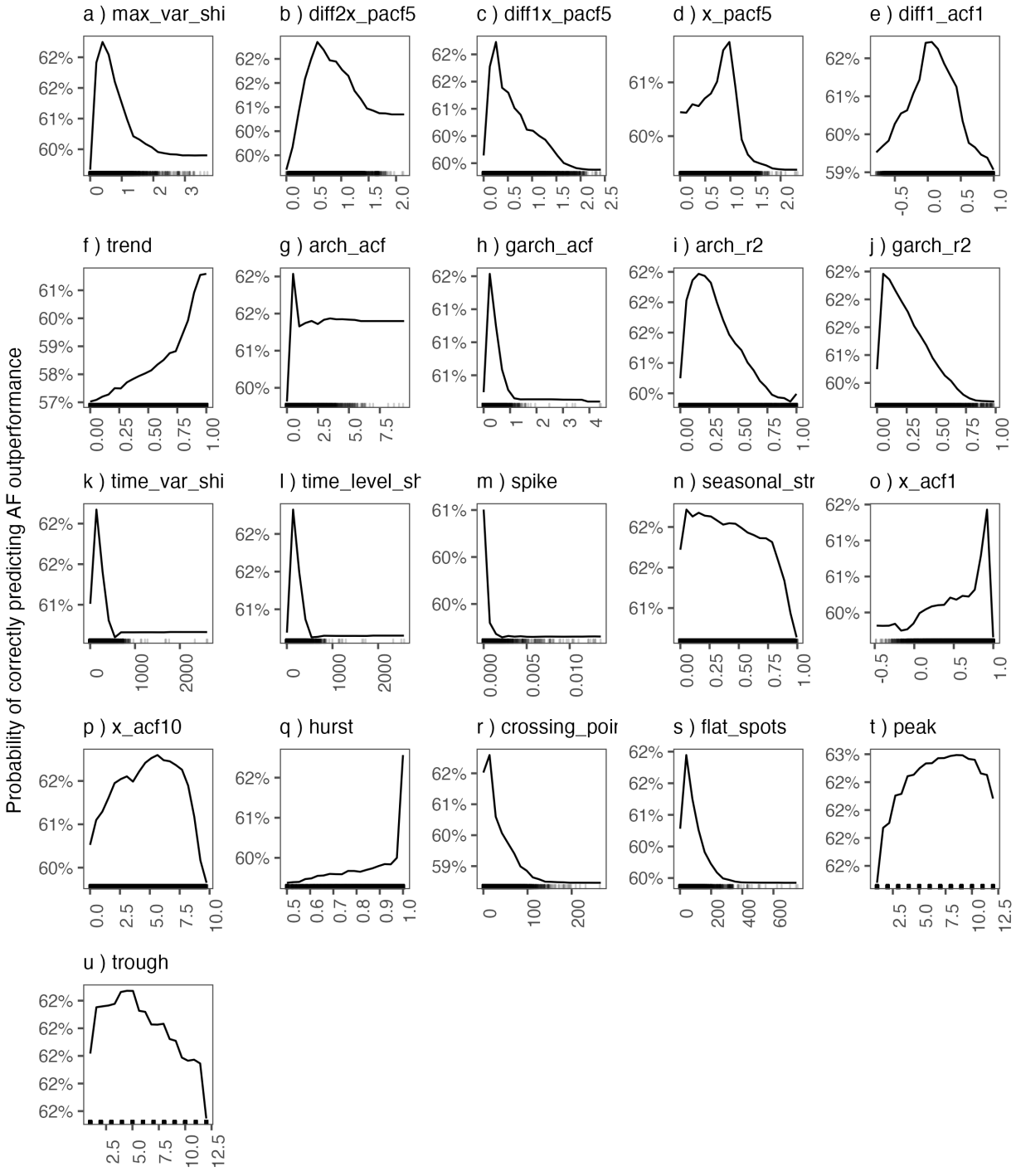

Figure 16: Partial dependence plot - the range of features vs. the probability of classifying AF versus AD (continue)

Figure 15 and 16 represent the partial dependence plot for all features. The x-axis represents the range of values extracted for each feature and the y-axis is the probability of classifying AF versus AD. Therefore,

the dependent value is the probability of correctly predicting the aggregating forecasts i.e. AF approach.

The plots illustrate the marginal effect of the selected features on the response (outcome) after integrating out all the other variables and their average effect on the response (James et al., 2013).

The rug marks at the bottom of each plot show the distribution of feature across the variable range. Note that here the data density is lower near the edges. This may cause unstable behavior of the curves in those areas. The vertical scales of the plots show the probability of AF approach providing more accurate forecast for each feature as values of that feature changes. From Figure 15 and 16, it is clear that the features have a mixed effect on classifying AF and AD approaches. While the effect might be clear for feature such as trend or nonlinearity, it is less clear for other such as mean or coefficient of variation.

We observe that increasing trend, ARCH.LM, hurst, autocorrelation lag 1, unitroot_pp and seas_pacf may increases the chance of AF performing better, therefore AF become preferable. However, increasing lumpiness, entropy, nonlinearity, curvature, strength of seasonality may increase the chance of AD performing better, so the strong presence of these features may favorite AD over AF. It is important to note that we are less interested in the exact probability in these plots. Instead, we are interested in discovering how changing time series feature may increase/decrease the chance of AF or AD.

7. Conclusions

In time series forecasting, a forecast time granularity required to inform a decision, might be different from the time series granularity itself. For instance, if a time series is stored at a higher frequency (e.g. monthly), forecasts might be required at a lower frequencies (e.g. quarterly, annual). This is very common in modern organisations as data can be collected in the finest time granularities. Therefore, there are situations where a forecast of the total value over a time period ahead (e.g. horizon aggregation/lead-time) is needed. To generate such a forecast for a given time series, we may consider two options: i) first generate forecasts, followed by adding them up to obtain the forecast horizon aggregation (AF), or ii) first aggregate the time series using non-overlapping temporal aggregation and then generate the forecast (AD). In this paper, we design and execute an empirical experiment framework using the monthly M4 competition data to i) explore the forecasting performance of these approaches; and ii) investigate the association between time series features and forecasting performance of temporal aggregation approaches (e.g. AF or AD).

There is a common assumption that series at the higher level of temporal aggregation (lower frequency) are smoother with less noise and more cleaner patterns, which often implies that forecasts created at the higher temporal aggregation levels are more accurate. This practice may exist in supply chains, where practitioners are usually advised to aggregate the series to the frequency levels aligned with their decision making horizons and then to create forecast for the period of interest. In this paper, we questioned this practice and explore the comparative performance of AF and AD approaches. We conducted a comparative research of the forecasting performance using 48,000 monthly series from M4 data, at different levels of temporal aggregation corresponding to generating forecasts at bi-monthly, quarterly, 4–monthly, semi-annual and annual levels. Results demonstrate that there is a significant number of series for which aggregate forecast (63%) outperforms aggregate data (37%).

Given the fact that there is a lack of rules and indications on which approach should be used for a given time series, we further investigate the association between time series features and AF/AD forecasting performance. The need for such a research investigation has also been highlighted in the literature (Babai et al., 2022). Therefore, we seek to shed lights on the effect of temporal aggregation on time series features and further investigate how they might affect the performance of AD and AF approaches. To that end, we first construct a database consisting of features of each time series as predictor and model class labeled as AF/AD as response/outcome. Then, we build several ML models to accurately predict the correct class and consequently recommend using the most accurate approach for a given time series and its features.

Our results show that Random Forest model provides the best performance in accurately classifying approaches measured through statistical and utility metrics. Moreover, RF model is used to reveal the most important time series features in producing the accurate prediction. Additionally, we extract the partial dependence plot to describe the contribution of time series features through a probability. This helps to

show how the value of a feature may favor AF over AD approach. The main findings of this study can be summarised as follows:

- First of all, when a forecast of the total value over several time periods ahead is required (i.e. aggregation horizon or lead-time), we show that AF is a significantly better methodology to use overall. The findings indicate that neither of the approaches are always the most accurate, when the accuracy is reported for the individual time series. The findings clearly show that the most accurate forecast is not necessarily generated by non-overlapping temporal aggregation. This may call into question the justification of the common practice in such a situation.

- Second, non-overlapping temporal aggregation changes the features of time series. The magnitude of the change varies for different features. In particular, we observe that with increase in the aggregation level, the strength of seasonality, the autocorrelation, coefficient of variation, linearity, curvature and KPSS unitroot statistic decrease. However, nonlinearity, mean, variance, ARCH.LM, trend , unitroot pp statistics increase. Entropy is the only measure that both increases and decreases based on its initial value.

- Third, Random Forest model is the most accurate classifier ML algorithm in predicting which approach provides more accurate forecast given a set of time series features as input.

- Fourth, RF model revels that the top ten important features for predicting whether AF or AD should be used for a given monthly time series in M4 competition include *curvature*, *nonlinearity*, *seas_pacf*, *unitroot_up*, *mean*, *ARCHM.LM*, *Coifficient of Variation*, *stability*, *linearity* and *max_level_shift*.

- Fifth, dependence plots provide some indications on how time series features may favorite AD over AF, vice-versa. We observe that increasing seas_pacf, trend, ARCH.LM, hurst, autocorrelation lag 1 and unitroot_pp increases the chance of AF performing better. While, increasing Coefficient of Variation, entropy, nonlinearity, curvature increases the chance of AD performing better, so the strong presence of these features may favorite AD over AF.

The proposed framework can be generalised to be used with other time series data. We believe that the conclusion should remain true for lower frequency time series such as monthly and quarterly. However, a different conclusion might be reached when using higher frequency time series data such sub-daily, daily, or weekly. This will bring further complications such as the presence of multiple seasonal cycles and long seasonalities. Therefore, time series features and the way they may change with increasing the aggregation level may differ. Also, the choice of forecasting method becomes important as a method like ETS cannot handle those complications. This would be an important avenue for further research.

Given the findings of this study and the potential value of temporal aggregation in time series forecasting, the current study could be extended in a number of ways. Research into any of the following areas would prove to be useful:

- In this study, we use all features to build the ML model, an alternative approach would be to use dimension reduction approaches for all features representing the same type of information such as seasonality, autocorrelation, noise, etc and then build the model;

- The proposed framework can be used to examine the association of time series features and temporal aggregation with sub-daily and daily time series data as they present further complications.

- Given that the exact relationship between time series features and the optimal temporal aggregation approach is not known, one direction for future research could focus on using meta-learning and other advanced techniques such as transformers neural networks and attentional mechanism to further shed lights on this problem.

- In this paper, we define the AD and AF approaches to forecast a cumulative number of periods ahead, therefore it is considered as a single output approach. It might be interesting to investigate the

forecasting by temporal aggregation as a multi-output scenario [@de2022data], where a sequence of two or more future data points are of interest.

- Finally, using time-series classification techniques [@buza2018time] to classify time series data based on various factors including time series futures and investigate their association with forecast accuracy of temporal aggregation approaches might be also an interesting avenue for further research.

Reproducibility

R code and RMarkdown file to produce all results in this paper are available at `https://github.com/bahmanrostamitabar/time-searies-featute-temporal-aggregation`

Appendix

Table 4: The main setup information for ML models

Model	R package	Running time (min)	Feature enginer-ing	Activation formula
LR	stats	~3	NO	glm(marks~.,family = binomial, data=train.data)
LDA	MASS	~3	NO	lda(marks~., data=train.data)
QDA	MASS	~3	NO	qda(marks~., data=train.data)
KNN	class	~20	YES	knn(Xlag[train,], Xlag[-train,], ClassTable[train,"marks"], k=3)
LASSO	glmnet	~10	NO	cv.glmnet(x, y, family="binomial", type.measure="auc")
GAM	gam	~12	NO	gam(marks~. + s(trend, df=3) + s(seas_acf1, df = 3) + s(linearity, df = 3) + s(entropy, df = 3) + s(x_acf1, df = 3) + s(seasonal_strength, df = 3) + s(seas_acf1, df = 3), family = binomial, data = train.data)
RF	randomForest	~2200	NO	randomForest(marks~., data=train.data1, mtry=10, ntree=3000, importance = TRUE)
Boosting	gbm	~3000	YES	gbm(marks~., data = train.data01, distribution = "bernoulli", n.trees = 4500, shrinkage = 0.01, interaction.depth = 10)
SVM	e1071	~6000	YES	tune(e1071::svm, as.factor(marks)~., data = train.data, scale = FALSE, kernel = "radial", ranges = list(cost=c(0.001 , 0.01, 0.1, 1, 5, 10, 100), gamma=c(0.5, 1, 2, 3, 4)))
FNN	nnet	~10	YES	nnet(marks~., data=train.data, size=10, maxit=500, decay=0.001, rang = 0.1)
DTW	dtw	~5	YES	knn(train = trainData[, -5], test = testData[, -5], cl = trainLabels, k = k, prob = TRUE, use.all = TRUE, distance = dtw.dist)
DL Torch	torch	~180	YES	nn_module("Net", initialize = function() { self$fc1 <- nn_linear(length(features1), 40), self$fc2 <- nn_linear(40, 30), self$fc3 <- nn_linear(30, 30), self$fc4 <- nn_linear(30, 15), self$fc5 <- nn_linear(15, 15), self$fc6 <- nn_linear(15, 10), self$fc7 <- nn_linear(10, 10), self$fc8 <- nn_linear(10, 5), self$fc9 <- nn_linear(5, 5), self$fc10 <- nn_linear(5, 1) }, forward = function(x) { x %>% self$fc1() %>% nnf_relu() %>% self$fc2() %>% nnf_relu() %>% self$fc3() %>% nnf_relu() %>% self$fc4() %>% nnf_relu() %>% self$fc5() %>% nnf_relu() %>% self$fc6() %>% nnf_relu() %>% self$fc7() %>% nnf_relu() %>% self$fc8() %>% nnf_relu() %>% self$fc9() %>% nnf_relu() %>% self$fc10() })
XG Boost	xgboost	~300	YES	xgboost(data = xgboost_train, max.depth=23, nrounds=500, objective = "binary:logistic")
TensorFlow	tensorflow, keras	~30	YES	keras_model_sequential() %>% layer_dense(units = 80, activation = "relu", input_shape = c(42)) %>% layer_dropout(rate = 0.6) %>% layer_dense(units = 40, activation = "relu") %>% layer_dropout(rate = 0.3) %>% layer_dense(units = 5, activation = "relu") %>% layer_dense(units = 1, activation = "sigmoid")
RNN	keras	~30	YES	keras_model_sequential() %>% layer_lstm(units = 64, input_shape = c(42, 1)) %>% layer_dropout(rate = 0.2) %>% layer_dense(units = 1, activation = "sigmoid")
CNN	keras	~35	YES	keras_model_sequential() %>% layer_conv_1d(filters = 32, kernel_size = 3, activation = "relu", input_shape = c(42, 1)) %>% layer_max_pooling_1d(pool_size = 2) %>% layer_dropout(rate = 0.2) %>% layer_flatten() %>% layer_dense(units = 1, activation = "sigmoid")

References

George Athanasopoulos, Rob J Hyndman, Haiyan Song, and Doris C Wu. The tourism forecasting competition. *International Journal of Forecasting*, 27(3):822–844, 2011.

George Athanasopoulos, Rob J Hyndman, Nikolaos Kourentzes, and Fotios Petropoulos. Forecasting with temporal hierarchies. *European Journal of Operational Research*, 262(1):60–74, 2017.

M Zied Babai, Mohammad M Ali, and Konstantinos Nikolopoulos. Impact of temporal aggregation on stock control performance of intermittent demand estimators: Empirical analysis. *Omega*, 40(6):713–721, 2012.

M Zied Babai, John E Boylan, and Bahman Rostami-Tabar. Demand forecasting in supply chains: a review of aggregation and hierarchical approaches. *International Journal of Production Research*, 60(1):324–348, 2022.

Devon K Barrow and Nikolaos Kourentzes. Distributions of forecasting errors of forecast combinations: implications for inventory management. *International Journal of Production Economics*, 177:24–33, 2016.

John E Boylan and M Zied Babai. On the performance of overlapping and non-overlapping temporal demand aggregation approaches. *International Journal of Production Economics*, 181:136–144, 2016.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

Paul Goodwin. *Profit from your forecasting software: A best practice guide for sales forecasters*. John Wiley & Sons, 2018.

Yuntong Hu and Fuyuan Xiao. An efficient forecasting method for time series based on visibility graph and multi-subgraph similarity. *Chaos, Solitons & Fractals*, 160:112243, 2022a.

Yuntong Hu and Fuyuan Xiao. Time series forecasting based on fuzzy cognitive visibility graph and weighted multi-subgraph similarity. *IEEE Transactions on Fuzzy Systems*, 2022b.

Rob Hyndman. *tsfeatures: Time Series Feature Extraction*, 2020. URL `https://pkg.robjhyndman.com/tsfeatures/`.

Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2021. URL `https://otexts.com/fpp3`.

Rob J Hyndman and Yeasmin Khandakar. Automatic time series forecasting: the forecast package for r. *Journal of statistical software*, 27:1–22, 2008.

Rob J Hyndman, Earo Wang, and Nikolay Laptev. Large-scale unusual time series detection. In *2015 IEEE international conference on data mining workshop (ICDMW)*, pages 1616–1619. IEEE, 2015.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. Statistical learning. In *An introduction to statistical learning*, pages 15–57. Springer, 2021.

Yao Jin, Brent D Williams, Travis Tokar, Matthew A Waller, et al. Forecasting with temporally aggregated demand signals in a retail supply chain. *Journal of Business Logistics*, 36(2):199–211, 2015.

Alex J Koning, Philip Hans Franses, Michele Hibon, and Herman O Stekler. The m3 competition: Statistical tests of the results. *International Journal of Forecasting*, 21(3):397–409, 2005.

Nikolaos Kourentzes and Fotios Petropoulos. Forecasting with multivariate temporal aggregation: The case of promotional modelling. *International Journal of Production Economics*, 181:145–153, 2016.

Nikolaos Kourentzes, Fotios Petropoulos, and Juan R Trapero. Improving forecasting by estimating time series structural components across multiple frequencies. *International Journal of Forecasting*, 30(2):291–302, 2014.

Nikolaos Kourentzes, Bahman Rostami-Tabar, and Devon K Barrow. Demand forecasting by temporal aggregation: using optimal or multiple aggregation levels? *Journal of Business Research*, 78:1–9, 2017.

Gang Liu, Fuyuan Xiao, Chin-Teng Lin, and Zehong Cao. A fuzzy interval time-series energy and financial forecasting model using network-based multiple time-frequency spaces and the induced-ordered weighted averaging aggregation operation. *IEEE Transactions on Fuzzy Systems*, 28(11):2677–2690, 2020.

Ivette Luna and Rosangela Ballini. Top-down strategies based on adaptive fuzzy rule-based systems for daily time series forecasting. *International Journal of Forecasting*, 27(3):708–724, 2011.

Spyros Makridakis. M4 competitor's guide: Prizes and rules. Technical report, University of Nicosia, 2018.

Dejan Mircetic, Bahman Rostami-Tabar, Svetlana Nikolicic, and Marinko Maslaric. Forecasting hierarchical time series in supply chains: an empirical investigation. *International Journal of Production Research*, pages 1–20, 2021.

Maryam Mohammadipour and John E Boylan. Forecast horizon aggregation in integer autoregressive moving average (inarma) models. *Omega*, 40(6):703–712, 2012.

Konstantinos Nikolopoulos. We need to talk about intermittent demand forecasting. *European Journal of Operational Research*, 291(2):549–559, 2021.

Konstantinos Nikolopoulos, Aris A Syntetos, John E Boylan, Fotios Petropoulos, and Vassilis Assimakopoulos. An aggregate–disaggregate intermittent demand approach (adida) to forecasting: an empirical proposition and analysis. *Journal of the Operational Research Society*, 62(3):544–554, 2011.

Fotios Petropoulos and Nikolaos Kourentzes. Forecast combinations for intermittent demand. *Journal of the Operational Research Society*, 66(6):914–924, 2014.

Fotios Petropoulos and Nikolaos Kourentzes. Forecast combinations for intermittent demand. *Journal of the Operational Research Society*, 66(6):914–924, 2015.

Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J Bessa, Jakub Bijak, John E Boylan, et al. Forecasting: theory and practice. *International Journal of Forecasting*, 2022.

Robert J Rossana and John J Seater. Temporal aggregation and economic time series. *Journal of Business & Economic Statistics*, 13(4):441–451, 1995.

Bahman Rostami-Tabar and Florian Ziel. Anticipating special events in emergency department forecasting. *International Journal of Forecasting*, 2020.

Bahman Rostami-Tabar, M Zied Babai, Aris Syntetos, and Yves Ducq. Demand forecasting by temporal aggregation. *Naval Research Logistics (NRL)*, 60(6):479–498, 2013.

Bahman Rostami-Tabar, Mohamed Zied Babai, Aris Syntetos, and Yves Ducq. A note on the forecast performance of temporal aggregation. *Naval Research Logistics (NRL)*, 61(7):489–500, 2014.

Bahman Rostami-Tabar, M Zied Babai, Mohammad Ali, and John E Boylan. The impact of temporal aggregation on supply chains with arma (1, 1) demand processes. *European Journal of Operational Research*, 273(3):920–932, 2019.

Bahman Rostami-Tabar, Mohamed Zied Babai, and Aris Syntetos. To aggregate or not to aggregate: Forecasting of finite autocorrelated demand. *arXiv preprint arXiv:2103.16310*, 2021.

Evangelos Spiliotis, Andreas Kouloumos, Vassilios Assimakopoulos, and Spyros Makridakis. Are forecasting competitions data representative of the reality? *International Journal of Forecasting*, 36(1):37–53, 2020.

Georgios P Spithourakis, Fotios Petropoulos, M Zied Babai, Konstantinos Nikolopoulos, and Vassilios Assimakopoulos. Improving the performance of popular supply chain forecasting techniques. *Supply Chain Forum: an international journal*, 12(4):16–25, 2011.

Xiaozhe Wang, Kate Smith-Miles, and Rob Hyndman. Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. *Neurocomputing*, 72(10-12):2581–2594, 2009.

William WS Wei. Some consequences of temporal aggregation in seasonal time series models. In *Seasonal analysis of economic time series*, pages 433–448. NBER, 1978.

Thomas R Willemain, Charles N Smart, Joseph H Shockor, and Philip A DeSautels. Forecasting intermittent demand in manufacturing: a comparative evaluation of croston's method. *International Journal of forecasting*, 10(4):529–538, 1994.

Giulio Zotteri and Matteo Kalchschmidt. A model for selecting the appropriate level of aggregation in forecasting processes. *International Journal of Production Economics*, 108(1-2):74–83, 2007.