On combining forecasting methods using time series features

Abstract

It is well known that ensemble approaches produce improvements over single methods in statistical learning. Nevertheless, when calculating predictions over a large dataset, computation time for the whole ensemble can be prohibitive, so individual model selection becomes the preferred approach. We present a method for combining forecasting models by posing it as a classification problem using features extracted from the time series. Unlike regular classification problems, we minimize the average forecast error of the selected method rather than the classification error. Not only does this address the aim of accurate forecasting, it also provides measures of relative method accuracy across the time series, and relative difficulty across time series. In contrast, a classic classification approach would give the same importance to all series and methods. The presented classifier is compared with state-of-the-art approaches to forecasting and time series classification. The results show an improvement of error over alternative approaches. These experiments allow us to show the relevance of both the feature set and the proposed optimization approach to several collections of time series. The scalability of the approach allows it to be applied to forecasting a large collection of time series. It can also be efficiently trained to tailor specific domains or datasets.

Keywords: FFORMA (Feature-based FORecast-model Averaging), FFORMS (Feature-based FORecast-model Selection), Time series features, Forecast combination, XGBoost, M4 Competition

1 Introduction

There are essentially two general approaches to forecast a time series: i) use of single model and ii) combination forecast or forecast model averaging. There is a growing consensus that the combination forecast is a way of improving forecast accuracy. Empirical work often provides evidence that the combination of forecasts resulting from several candidate forecasting models are often superior to their individual forecasts. Combining forecasts across different models is considered as a way of reducing the risk of selecting an inappropriate model. However, the

main challenge in forecast combination is the selection of appropriate set of weights.

Granger (1969), was the first to forward the idea of "the combination of forecasts". Since then many approaches have been proposed to derive weights for forecast combination (Timmerman, 2006). Simple averages(ref_clemen1989), regression-based approaches, ... are name to few.

Recently, a few researchers have explored the use of time series features in selecting the most appropriate forecasting method. However, use of time series features to derive weights for forecast combination has been rarely addressed in the literature. In this paper we propose a general framework to obtain weights for forecast combination based on features computed from the time series. We call this framework FFORMA(Feature-based FORecast Model Averaging). The proposed FFORMA framework has been used over the course of M4 competition and placed in second in forecast accuracy in both point forecasts and prediction intervals.

The rest of the paper is organized as follows. Section 3 presents the methodology underlying the FFORMA framework followed by Section 4 that describes the application of the framework to M4 competition. Finally Section 5 presents the conclusions and future work.

2 Methodology

This section describes the rational behind the proposed framework. In the proposed framework we use meta-learning approach to derive weights for forecast combination. Figure 1 presents the Pseudo code of the proposed framework. This approach has been influenced by the work of Talagala, Hyndman & Athanasopoulos (2018).

FFORMA framework works in two phases: offline phase and online phase. In the offline phase a meta-learner is trained using a diverse collection of time series. We call the collection of time series use to train the classifier as the reference set. Each time series in the reference set is divided in to training period and test period. Features are computed based on the training period of each time series. A set of features extracted from the time series are used as inputs to the meta-learner. Further, in order to train a FFORMA meta-learning model, we need a pool of forecasting methods. The forecasting methods in the pool is applied to training period of each time series and forecast error measures are calculated over the test period. In a typical classification problem, extreme gradient boosting algorithm is trained to minimize a loss function with respect to classification error. In this experimental setting, we are not interested in the classification error but we are interested in forecast error. Hence, a customized loss function is used to include the information of forecast error rather than the classification error. For technical reasons, we use extreme

gradient boosting algorithm to train the meta-learner. Extreme gradient boosting algorithm implementation allows easy changes to the objective function. It only requires the output the gradient and hessian of the objective while other methods require re-implementation big part of the code.

Algorithm 1 The FFORMA framework - Forecast combination based on meta-learning.

Offline phase - train the classifier

Given:

$O = \{x_1, x_2, \ldots, x_n\}$: the collection of n observed time series.

L : forecast error measure calculated based on different forecasting algorithms such as ARIMA, ETS, SNAIVE, etc.

F : the set of functions to calculate time series features.

B : number of trees in the XGboost algorithm.

Output:

FFORMA meta-learner

Prepare the meta-data

For $j = 1$ to N:

1: Split x_j into a training period and test period.
2: Calculate features F based on the training period.
3: Fit L models to the training period.
4: Calculate forecasts for the test period from each model.
5: Calculate forecast error measure over the test period for all models in L.
6: Meta-data: input features (step 4), output labels (step 5).

Train a meta-learner using extreme gradient boosting algorithm

7: Train a extreme gradient boosting classifier based on the meta-data.
8: meta-learner : extreme gradient boosting classifier.

Online phase - forecast a new time series

Given:

FFORMA classifier from step 8 .

Output:

class weights for new time series x_{new}.

9: For x_{new} calculate features F.
10: For a given series, let W be the vector of weights assign for each forecasting method in L.

The extreme gradient boosting algorithm produce weights for each forecasting method. These weights can also be interpreted as the probability of each method being best. These weights can be used either to select the "best" forecasting method for each series or to combine the forecasts using weighted linear combination. Note that the accuracy of the FFORMA meta-learner depends on three main factors: i) forecasting methods used in the pool, ii) a set of time series features we considered and iii) a collection of time series used to train the classifier. Section 3 provides a more detailed description of application of the FFORMA framework over the course of M4 competition. The proposed framework is closely related to the previous work

by ref_prudencio which they use machine learning techniques to define weights for the linear combination of forecasts.

3 FFORMA framework: Application to M4 Competition

3.1 Data preprocessing

The M4 competition database consists of 100,000 real-world time series of yearly, quarterly, monthly, weekly, daily and hourly data. The frequency for yearly, quarterly, monthly and weekly data are considered as 1, 4, 12 and 52 respectively. [Q: Daily and Hourly series, what are the frequencies considered?]. We used xxxx time series out of 100000 to train a meta-learner and rest is used to evaluate the proposed framework. In addition to the time series provided in the M4 competition database we used the time series of M1 and M3 competitions (Makridakis & Hibon 2000) to the reference set. Each time series in the reference set is split into training period and test period. The length of test period of each time series was set as according to the rules of M4 competition, 6 for yearly data, 8 for quarterly, 18 for monthly, 13 for weekly, 14 for daily and 48 for hourly.

3.2 Time series features

Table xxx provides a detailed description of features used in this experiment.

[include table]

These features have been previously used by Talagala, Hyndman & Athanasopoulos (2018) and Hyndman, Wang & Laptev (2015). A detailed description of these features are provided in Talagala, Hyndman & Athanasopoulos (2018). All the features are implemented in `tsfeatures` R package by xxx.

Question: calculation of features for time series with multiple seasonality, and short time series?

3.3 Forecasting methods

We considered nine forecasting algorithms implemented in the forecast package in R. They are(the specific R calls for fitting the models are given): i) automated ARIMA algorithm (`auto.arima`), ii) automated ETS algorithm (`ets`), iii) feed-forward neural network with a single hidden layer is fitted to the lags. The number of lags is automatically selected (`nnetar`) iv) random walk with drift (`rwf` with drift=TRUE), v) TBATS model (`tbats`), vi) Theta method

forecast (`thetaf`), vii) naive forecasts (`naive`) viii) STLM-AR Seasonal and Trend decomposition using Loess with AR modeling of the seasonally adjusted series(`stlm` with modelfunction `ar`), ix) seasonal naive forecasts (`snaive`). In the case of an error when fitting the series (e.g. a series is constant), the SNAIVE forecast method is used instead.

Question: i) Calculation of ets models to daily, hourly and weekly series

3.4 Loss function and definition of "optimal" weights

3.5 Generate point forecasts

3.6 Prediction Intervals

4 Results (Do we need this?)

Fotios email: We would like to ask that this paper focuses on the clear description of the method (and possibly include flowcharts, pseudocode, etc) without any empirical evidence (that would be part of the main M4-competition paper).

5 Discussion and Conclusions

Fotios email: We would like, though, to see a short section that discusses the reasons behind the good performance of your method.

References

Hyndman, RJ, E Wang & N Laptev (2015). Large-scale unusual time series detection. In: *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*. IEEE, pp.1616–1619.

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

Talagala, TS, RJ Hyndman & G Athanasopoulos (2018). Meta-learning how to forecast time series. *Technical Report 6/18, Monash University.*