Contents lists available at ScienceDirect

International Journal of Forecasting

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

Weighted ensemble of statistical models

Maciej Pawlikowski *, Agata Chorowska

ProLogistica Soft, Ostrowskiego 9, 53-238, Wroclaw, Poland

ARTICLE INFO

Keywords:
Combining forecasts
Time series
Automatic forecasting
Time series clustering
Forecasting competitions

ABSTRACT

We present a detailed description of our submission for the M4 forecasting competition, in which it ranked 3rd overall. Our solution utilizes several commonly used statistical models, which are weighted according to their performance on historical data. We cluster series within each type of frequency with respect to the existence of trend and seasonality. Every class of series is assigned a different set of models to combine. Combination weights are chosen separately for each series. We conduct experiments with a holdout set to manually pick pools of models that perform best for a given series type, as well as to choose the combination approaches.

1. Introduction

Combining forecasts has been shown to greatly improve the forecast quality (Clemen, 1989). Averaging predictions produced by different models usually outperforms the individual methods. Our submission for the M4 competition relies heavily on this technique. We take several commonly used statistical models and weight their outputs according to their performance on a holdout set. The main challenge was in choosing a pool of algorithms and a combination approach.

We categorize M4 data with regard to the frequency (monthly, weekly, etc.), as well as the existence of trend and seasonality. For every category, we select a distinct pool of models. A slightly different way of calculating weights is used for every frequency.

Section 2 is an overview of our combination methodology. Section 3 briefly summarizes the computations. Section 4 describes additional heuristics that helped us further improve the accuracy. Sections 5 and 6 provide the discussion and the conclusion. In the Appendix (available online) we illustrate the complete forecast calculation process for an example time series.

2. Method overview

Our forecasting method is comprised of five steps:

1. Clustering time series.
2. Choosing the model pool for each cluster.
3. Measuring performance of the models with rolling origin evaluation.
4. Determining weights for models in the pool.
5. Calculating the final forecast.

The method is parameterized by the model pools, the number of origins N and the error averaging formula f used in the rolling origin evaluation, as well as the model weighting formula g. The choice of each parameter is performed separately. For a given cluster, the model pool is chosen first, then the rolling origin evaluation parameters, and finally, the weighting formula. Each series is split into the training part and the holdout part, where the length of the holdout part is equal to the respective forecast horizon. We monitor the Overall Weighted Average (OWA) error (Makridakis, 2018) on the holdout part to choose the best parameter values. The OWA was the main accuracy measure in the M4 competition.

2.1. Clustering time series

We group all series into classes according to the frequency and the existence of seasonality and trend.

* Corresponding author.
E-mail address: maciej.pawlikowski@prologistica.pl
(M. Pawlikowski).

https://doi.org/10.1016/j.ijforecast.2019.03.019
0169-2070/© 2019 International Institute of Forecasters. Published by Elsevier B.V. All rights reserved.

To detect seasonality we use a 90% autocorrelation test similar to the one used in M4 benchmarks, while to detect trend we employ a Mann-Kendall test (Gilbert, 1987) with 95% confidence.

Trend and seasonality detection are omitted for daily series during clustering because we did not find it beneficial. Splitting daily series into multiple classes did not help us lower the holdout set error. We also do not use it for the frequencies with low representation in the dataset (hourly and weekly) in order to avoid possible overfitting of model pools when the classes get too small.

Since the yearly series are not seasonal, this means we had 13 classes in total: 2 for yearly series, 4 for both monthly and quarterly, and a single class for each of weekly, daily, and hourly data.

2.2. *Choosing the model pool*

Each class of series is assigned a distinct model pool. The list of individual models we choose from, alongside respective R packages and functions, is given below. To calculate forecasts we used R 3.5.0 with packages "forecast" (version 8.2) (Hyndman & Khandakar, 2008) and "forecTheta" (version 2.2) (Fiorucci, Louzada, & Yiqi, 2016).

- *Naïve models* - Naïve 1, Naïve 2 (Makridakis, 2018)
- *Exponential Smoothing (ETS) models* - simple ETS [forecast::ses], ETS with automatic model choice [forecast::ets], with or without damped trend (Hyndman, Koehler, Snyder, & Grose, 2002)
- *Theta models* - Theta method (Assimakopoulos & Nikolopoulos, 2000) [forecast::thetaf], Optimized Theta method (Fioruci, Pellegrini, Louzada, & Petropoulos, 2015) [forecTheta::otm]
- *ARIMA models* - ARIMA with automatic parameter choice [forecast::auto.arima]
- *Linear regression models* - linear regression of a series on various types of a trend (constant, linear, logarithmic) [stats::lm], optionally applied to deseasonalized data

Seasonality in automatic ETS and ARIMA methods is handled by their implementations in the "forecast" package. In other cases we use classical multiplicative decomposition.

These models form a default model pool for all clusters. The final pool for each cluster is a subset of this default pool. There is a slight exception to this rule for hourly, daily and weekly data, where multiple variants of each model may be present in the default pool. For hourly data, we consider two seasonality periods (24 and 168) by including two variants of each model. Each variant assumes a different seasonality period. For weekly and daily data, we add variants of models that operate on trimmed series, which allows us to focus on the latest observations. In the case of weekly data, we trim the series to the last few years. For daily series, we leave the last several weeks. The final pool may contain multiple variants of the same model and all of them may receive non-zero weights. Additionally, for hourly, daily, and weekly data we consider only trimmed variants of the ARIMA and ETS models to reduce the computational complexity. More details on this step can be found in the source code of our submission[1] and in Appendix.

Model pool selection procedure

To determine the model pool for a class of time series, we follow the procedure below:

1. Start with a default pool, set the rolling origin evaluation parameters and the weighting formula to default values (described in later sections).
2. For each series in the class:
   (a) Split the series into a training part and a holdout part.
   (b) For each model, forecast the series with that model. Calculate the holdout set error.
   (c) Determine the combination weight for each model (Sections 2.3 and 2.4).
3. Sort the models by their mean holdout set error across all series in the class.
4. Combine all models. Compute the mean holdout set error of the combination.
5. Try removing models from the pool starting with the one with the highest holdout set error:
   (a) Remove the model. Compute the mean holdout set error of the combination.
   (b) If it increases, put the model back into the pool. Then, move to the next model.
   (c) Stop when all models have been considered.

2.3. *Rolling origin evaluation*

For each model in the pool and each series in the class, perform a rolling origin evaluation (Tashman, 2000) with a constant window size 1. The number of origins N depends only on the series' frequency, thus, for example, all monthly series share the same value. For each series, we average the symmetric Mean Absolute Percentage Error (sMAPE) across origins using a weighting function f. This produces a vector of performance scores of models that is later converted into the combination weights (described in Section 2.4). We choose f from among a simple arithmetic mean and a weighted mean with exponential weights, in which the later origins count more. The default value of f is a simple mean. Similar to N, f is chosen per frequency.

We find the best N and f after the model pool choice. For N, we check all values up to the respective forecast horizon (the default value of N) and monitor the holdout set errors of the combination. Then we check all possible values of f, holding N fixed. It is worth noting that this means the rolling origin evaluation is performed twice for a given series: once on the training part, during parameter tuning, and once on the whole series, to calculate weights for the final prediction.

[1] https://github.com/M4Competition/M4-methods/tree/master/237%20-%20prologistica

Table 1
Chosen values of N alongside respective forecast horizons h for all frequencies.

Frequency	Yearly	Quarterly	Monthly	Weekly	Daily	Hourly
N/h	3/6	8/8	10/18	13/13	8/14	24/48

Table 1 shows the chosen values of N. In most cases, we were able to make it significantly lower than forecast horizon without observing any meaningful difference in error magnitude on the holdout set. This is desirable because the computational cost of our method scales linearly with N.

For the f function, the default mean aggregation worked well for most frequencies. The two exceptions were daily and yearly series, where we observed a slight improvement using exponential weights.

We experimented with three error measures in the rolling origin evaluation: the sMAPE, the Mean Squared Error (MSE) and the OWA. We observed no noticeable difference in OWA error on the holdout set between these metrics. Since the choice of a metric for this step didn't seem to affect the accuracy of the combination, we decided to use the sMAPE as it was the most convenient.

2.4. Determining combination weights

The sMAPE errors are converted into weights using a formula g taking one of the following forms. A small epsilon is added to the denominator to avoid division by zero.

$$g_{inv}(S) = 1 \;/\; (S + \epsilon)$$
$$g_{sqr}(S) = {g_{inv}}^2(S)$$
$$g_{exp}(S) = \exp(g_{inv}(S))$$

S is a vector of performance scores for a given series calculated in the rolling origin evaluation step. All operations above are element-wise (i.e. applied separately to each element of S). The formula g is chosen per frequency, using exhaustive search while monitoring the mean holdout set error of the combination. The default value of g is g_{sqr}. g_{inv} has been chosen for hourly series and g_{exp} for weekly series. In other cases the default g_{sqr} performed the best, possibly because g was fixed to a default value during the model pool fitting phase.

The graphs displaying the mean model weights for each frequency can be found in Appendix.

2.5. Calculating final forecast

Calculate forecasts of the series for models in the pool. The final prediction is defined as a weighted mean of those forecasts, using weights described in Section 2.4. Moreover, since the M4 dataset contains no negative values, any negative forecasts are replaced with zeros.

3. Computations summary

Here we provide a brief summary of the computations described in the previous section. Let X be a yearly time series of length 30 with horizon 6. For the sake of simplicity, let's assume we chose N = 3, f = arithmetic mean, $g = g_{sqr}$, and a set of models $m_1, \ldots, m_5$.

1. Perform the rolling origin evaluation:
   (a) Calculate $e_{i,j}$, where $i \in \{1, 2, 3, 4, 5\}$, $j \in \{1, 2, 3\}$, and $e_{i,j}$ is the sMAPE error of the one-step-ahead forecast produced by m_i fitted to the first $30 - j$ terms of X
   (b) Average the evaluation results for each model: $s_i = (e_{i,1} + e_{i,2} + e_{i,3}) \;/\; 3$
2. Compute the model weights: $w_i = s_i^{-2}$
3. Calculate $F_1, \ldots, F_5$ – 6-steps-ahead forecasts created with $m_1, \ldots, m_5$ fitted to X
4. The final forecast is a weighted average of $F_1, \ldots, F_5$ using weights $w_1, \ldots, w_5$

4. Special cases

In addition to the above algorithm, we employed two heuristic procedures that improved the accuracy on the holdout set. These two additional steps occur after computing combination forecasts described in Sections 2 and 3.

4.1. Daily series

In the case of daily series, we were unable to find a model pool that would significantly outperform a single Naïve 1 model. Thus, we use forecast combinations for only a part of daily data. Since the Naïve 1 predictor is best suited for time series originating in a random walk, we have heuristically identified such cases. For each daily series, we compare the sMAPE error on the holdout set of the Naïve 1 forecast to a threshold t_{rnd}. If the error is smaller, we label the series as *random* and forecast it exclusively with Naïve 1.

We chose the value of t_{rnd} after choosing the model pool, N, f, and g. We fixed those parameters and tested 10 evenly spaced t_{rnd} values from the interval $[0.01, 0.1]$ (the threshold had to be small). For each value, we determined which series are considered *random*, replaced the combination forecast for these series with the Naïve 1 prediction, and computed the mean holdout set error for all daily series. During experiments we settled on a threshold of 0.05, which resulted in roughly 90% of daily series being considered *random*.

4.2. Forecast by analogy

We were able to use forecasting by analogy to significantly boost the holdout set accuracy for daily and hourly data. We use the correlation coefficient to determine which series should be predicted this way. The correlation is computed between the last values of a given series and every window of every series in the dataset. Properly scaled and shifted continuation of the most correlated window replaces the combination forecast for the series, provided the correlation is strong enough. The detailed process for a series $X = (x_1, \dots, x_n)$ is described below. The procedure is parameterized by the window length M and the threshold t_{cor}. The forecast horizon for X is denoted by h. $\overline{X}$ denotes the mean of the series and $\sigma(X)$ is the standard deviation.

1. Let $X' = (x_{n-M+1}, \dots, x_n)$ be the last M observations of X.
2. For every series $Y = (y_1, \dots, y_m)$ with the same frequency as X:
   (a) For i from 1 to $m - h - M + 1$, compute the correlation coefficient between the window $(y_i, \dots, y_{i+M-1})$ and X'.
3. Let $Z = (z_1, \dots, z_k)$ denote the series containing the window most correlated with X'. Let $Z' = (z_j, \dots, z_{j+M-1})$ be that window. If the correlation exceeds a threshold t_{cor}:
   (a) Let $Z'' = (z_{j+M}, \dots, z_{j+M+h-1})$ be the continuation of Z' with length h.
   (b) Replace the forecast for X with $(Z'' - \overline{Z'})\dfrac{\sigma(X')}{\sigma(Z')} + \overline{X'}$.

We arbitrarily picked $M = 2h$. The value of t_{cor} is chosen as the last parameter of our method. For each frequency, we manually tested several values from [0.95, 0.999]. For each value, we computed the holdout set error averaged across all series with the given frequency. We achieved the largest accuracy boost for 0.99 on daily data and 0.995 on hourly data. These values resulted in 33% of daily series and 40% of hourly series being predicted by analogy. For yearly and weekly series the results did not improve when using this method. We did not have enough time to try forecasting by analogy on monthly and quarterly data before the competition ended, but we did so after the test data has been released. For quarterly series, we did not observe any improvement. However, for monthly series this method decreased the mean sMAPE error on the test set from 12.747 to 12.624. With t_{cor} set to 0.995, 10% of monthly series were predicted by analogy.

5. Discussion

Combining statistical models was a very popular approach in the M4 competition (Makridakis, Spiliotis, & Assimakopoulos, 2018). We believe that the key to success of our method was the careful choice of model pools and weights.

The choice of model pool was crucial in our experiments. Averaging all models never turned out optimal. We also observed that simply combining several top performing models did not result in the best choice either. For example, including the Naïve 1 model in the pool in many cases improved the accuracy, even though as a single model it often performed the worst.

After the test data has been released, we investigated the impact of the averaging function f and the weighting formula g on the three largest datasets: yearly, quarterly, and monthly. Different choices of f changed the test set error in a meaningful way only for monthly series. On the other hand, altering the formula g had a major impact on the accuracy in all cases.

In the case of series predicted by analogy, we did not use model combinations. It is possible that combining forecasting by analogy with other models would yield a better result.

We decided to use the window size 1 during the rolling origin evaluation step. This choice has been made due to its simplicity. We also tested the evaluation using one window with size N instead of N windows with size 1, but this method was significantly less accurate. There may still, however, exist a more optimal way to obtain performance scores.

The tuning of per-cluster parameters has been performed manually. We fit each of the parameters independently, in order to make it feasible given the time constraints. While manual inspection can provide intuitions about the impact of particular variables on the final performance, a proper grid search should ultimately result in a more optimal set of values. It would also enable lower-level choices, like using different values of N for different classes within the same frequency, which becomes physically impossible without automation.

Finally, the difference in the holdout set accuracy between the full list of models and the chosen pool was the largest for frequencies with a small amount of series (hourly and weekly). This might mean that a more fine-grained clustering of data could help improve the forecast quality. Considering more time series characteristics would result in smaller and more homogeneous clusters, which should make it easier to fit specialized model pools.

6. Conclusion

In this paper, we described the forecasting methodology used in the M4 competition. The core of our approach is the combination of statistical models, with rolling origin evaluation determining the weights. The most important and the hardest task was choosing which models to combine. We found the choice of the model pool to have a major impact on the accuracy, sometimes in an unexpected way. Including even weak models in the combination might improve the overall performance. At the same time, we found the fitting of the pool to be necessary, as the default setting was never optimal. We believe this should be the main takeaway from this research.

Acknowledgments

The authors would like to thank the developers of the "forecast" package for R, which has been used extensively in this research.

Funding

This work was supported by the European Regional Development Fund [grant number RPDS.01.02.01-02].

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ijforecast.2019.03.019.

References

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

Clemen, R. T. (1989). Combining forecasts: a review and annotated bibliography. *International Journal of Forecasting*, *5*(1), 559–583.

Fiorucci, J. A., Louzada, F., & Yiqi, B. (2016). Forectheta: forecasting time series by theta models. r package version 22 https://CRAN.R-project.org/package=forecTheta.

Fioruci, J. A., Pellegrini, T. R., Louzada, F., & Petropoulos, F. (2015). The optimised theta method. arXiv:1503.03529.

Gilbert, R. O. (1987). *Statistical methods for environmental pollution monitoring* (pp. 208–213). Van Nostrand Reinhold Company.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, *26*, 1–22, http://www.jstatsoft.org/article/view/v027i03.

Hyndman, R. J., Koehler, A. B., Snyder, R. D., & Grose, S. (2002). A state space framework for automatic forecasting using exponential smoothing methods. *International Journal of Forecasting*, *18*(3), 439–454.

Makridakis, S. (2018). M4 competitor's guide. https://www.m4unic.ac.cy/wp-content/uploads/2018/03/M4-Competitors-Guide.pdf.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The m4 competition: results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802–808.

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, *16*(4), 437–450.

Maciej Pawlikowski graduated from the University of Wrocław in 2018, with MSc in computer science. His forecasting practice began in April 2018, when he started working as a data analyst / programmer at ProLogistica Soft.

Agata Chorowska is Research and Development Team Leader at ProLogistica Soft responsible for development of demand forecasting and inventory optimization algorithms. Formerly she worked as a quantitative analyst at Credit Suisse. She holds a Ph.D. in mathematics (2014) and MSc in mathematics (2010) and in computer science (2010).