Method Description

General Information

Type of Entry (*Academic, Practitioner, Researcher, Student*)	Practitioner
First Name	Maciej
Last Name	Pawlikowski
Country	Poland
Type of Affiliation (*University, Company-Organization, Individual*)	Company-Organization
Affiliation	ProLogistica Soft

Team Members

1st Member	
First Name	Agata
Last Name	Chorowska
Country	Poland
Affiliation	ProLogistica Soft
2nd Member	
First Name	Olena
Last Name	Yanchuk
Country	Poland
Affiliation	ProLogistica Soft

Information about the method utilized

Name of Method	Weighted Ensemble of Statistical Models
Type of Method (*Statistical, Machine Learning, Combination, Other*)	Combination
Short Description (up to 200 words)	Our solution utilizes several common statistical models, which are weighted according to their performance on historical data. We classify series in each time period with respect to the existence of trend and/or seasonality. Each type of series is assigned a different set of models to consider. We performed experiments with holdout set to manually pick sets of models that perform the best for each series type, as well as to choose the weighting schemes. We additionally exploit the fact that in M4 data set some series are very highly correlated to fragments of other series. This last approach is used for producing forecasts of Daily and Hourly data.

Weighted Ensemble of Statistical Models

1. Introduction

We developed our solution in statistical software R 3.5.0 on Windows 7 Professional. We make heavy use of the `forecast` 8.2 package, which provides implementations of a lot of popular models. For instructions on how to use the software, refer to the `README.txt` file attached to the source code.

The final predictions are weighted averages of forecasts produced by statistical methods. We utilized the following models:

- Naive
- Naive2 (M4 benchmark)
- Simple Exponential Smoothing [`forecast::ses`]
- Exponential Smoothing (with automatic parameters choice) [`forecast::ets`]
- Damped Exponential Smoothing (automatic) [`forecast::ets, damped = T`]
- ARIMA (automatic) [`forecast::auto.arima`]
- Simple Theta [`forecast::thetaf`, modified to never crash on very short series]
- Optimised Theta [`forecTheta::otm`, same as above]
- econometric models (linear regression on different types of trend) [custom]

All of the above can be modified with one or more of the following:

- deseasonalization – the model is applied to a deseasonalized series. The seasonality is restored at the end. [`stats::decompose, type = "mult"`]
- frequency change – the model is applied to a series after changing frequency
- series trimming – the model is applied only to some tail of a series

In case of Daily and Hourly sets, we also find highly correlated segments among the data, and use that information to produce forecasts.

2. The Algorithm

For a given series X:

1. Choose a set of models to combine. The selection is based upon X's characteristics: period (Yearly, Daily, etc.), existence of seasonality and/or trend.
2. Using chosen models, create one-step-ahead forecasts for the last N prefixes of X and measure their accuracy according to sMAPE. The number N depends on X's period and is typically slightly smaller than forecast horizon.
3. For each model, summarize errors of different prefixes. The summarization method is a weighted mean with either equal or exponential weights. This gives us scores that measure performance of chosen models on X's history.

4. From the scores, create a set of weights, that will be used in calculation of final forecasts. The method of translating scores to weights depends on X's period. Usually the weights are squares of inverted scores.
5. Calculate future forecasts for X for all models. The final prediction is defined as a weighted mean of those forecasts, using weights described in step 4.

Special cases:

A) We omit the trend and seasonality detection for Daily series (because we did not find it beneficial), as well as for the time periods with small amount of data points (Hourly and Weekly).

B) In case of Daily series, we found that the best performing single statistical method was Naive. Because of that, we employed a very simple test to check if a series should be predicted with Naive. A series passes the test if the Naive sMAPE error on the last forecast horizon is smaller than some threshold. During experiments we settled on a threshold of 0.05, which allowed roughly 90% of all Daily series to pass. After step 5, predictions of those series are replaced by Naive forecasts.

C) We were able to exploit the structure of the M4 data set to significantly boost our estimated performance on Daily and Hourly data. For a given series X in those subsets, one can often find a series Y that looks extremely similar, but is longer. We assumed this is not by accident, and decided to use existing fragment of Y as a forecast of X.
To find Y, we calculate the correlation between the last window of X and every window of every series with the same period as X. During brief experiments, we chose the window length to be twice the forecast horizon. The window with the highest correlation coefficient is chosen as the most similar, and its continuation (after proper scaling and shifting) is used as a forecast of X. We do this after all previously described steps, and only if the correlation coefficient of the best fitting window exceeds a threshold. We chose 0.99 for Daily and 0.995 for Hourly data, which resulted in 33% of Daily series and 40% of Hourly series being predicted this way.
We tried similar approach for Weekly and Yearly data, but we didn't observe improvement there. Since this was a last minute addition to the procedure, we didn't have a chance to look into Quarterly and Monthly series.

Because there were no negative values present in M4 data, we assumed that the forecasts should not contain them either. The last step was to ‘fix’ the negative values in forecasts (which was a very rare occurrence), by explicitly setting them to 0.

We briefly considered Mean Squared Error for calculating weights. However, the results did not seem to improve, so we stuck to sMAPE.

3. Parameter Tuning

To apply the algorithm we need to choose the following:

- which set of models to consider for which type of series
- the value of N - how many prefixes are used in scores calculation
- summarization function f - how to summarize the errors of historical one-step-ahead forecasts into a single score for a model
- weighting function g - how to translate scores into final weights

The function f is a weighted average, with weights being either equal or exponential (the more recent error values count more). The function g takes form of one of the following:

- $\mathrm{g_{sqr}(scores)} = 1 / \mathrm{scores}^2$
- $\mathrm{g_{inv}(scores)} = 1 / \mathrm{scores}$
- $\mathrm{g_{exp}(scores)} = \exp(1 / \mathrm{scores})$

(A small epsilon is added to the denominator to avoid division by zero).

These choices were made separately for each of the six types of time period present in M4 data. To show the process, let's take Monthly data as an example. We prepared test data by cutting the last 18 observations from every series, where 18 is the Monthly forecast horizon. The cut values were used to measure the quality of the set of parameters fitted on the remaining part.

For each of parameters N, f, g, one value was chosen empirically for all Monthly series. In order to decide which models to choose, we split Monthly data set into four groups:

- with trend and seasonality
- with trend, but no seasonality
- with no trend, but with seasonality
- with neither trend, nor seasonality

For detecting seasonality, we used 90% autocorrelation test similar to the one used in M4 benchmarks. For trend detection, we employed a Mann-Kendall test with 95% confidence (`mk.test` and `smk.test` from R package `trend`).

Each group has been tested separately. We performed the tests manually, by experimenting with different subsets of models. In the end, different set of methods was assigned to each group. We provide a step-by-step example of an experiment with Yearly data, for details refer to `README.txt`.

4. Example

Let X be a yearly time series of length 30 with horizon 6. Let's assume we chose $N = 3$, $f = mean$, $g = g_{sqr}$, and a set of methods $m_1, \dots, m_5$.

1. Calculate $s_{i,j}$, where $i \in \{1,2,3,4,5\}$, $j \in \{1,2,3\}$, and $s_{i,j}$ is the sMAPE error of one-step-ahead forecast produced by m_i on $X[1: (30 - j)]$

2. Calculate weights for models: $w_i = \left((s_{i,1} + s_{i,2} + s_{i,3}) / 3\right)^{-2}$
3. Calculate $F_1, \dots, F_5$ – forecasts of X for 6 steps for $m_1, \dots, m_5$
4. The final forecast is a weighted average of $F_1, \dots, F_5$ using weights $w_1, \dots, w_5$

5. About Us

ProLogistica Soft Sp. z. o. o.
ul. Ostrowskiego 9
53-238 Wrocław, Poland
kontakt@prologistica.pl
www.prologistica.pl

ProLogistica Soft