M4 Competition: FFORMS framework

Team members: Thiyanga Talagala (PhD student), George Athanasopoulos, Rob J Hyndman

1 Introduction

We use Feature-based FORecast-Model Selection (FFORMS) framework introduced in (Talagala, Hyndman & Athanasopoulos 2018) over the course of the M4 forecasting competition. The underlying approach involves computing a vector of features from the time series we will be forecasting which are then used to select the forecasting model. The model selection process is carried out using a classification algorithm – we use the time series features as inputs, and the best forecasting model as the output. A Random Forest approach is used to develop the classifier. The classification algorithm can be built in advance of the forecasting exercise (so it is an "offline" procedure). Then, when we have a new time series to forecast, we can quickly compute its features, use the pre-trained classification algorithm to identify the best forecasting model, and produce the required forecasts. Thus, the "online" part of our algorithm requires only feature computation, and the application of a single forecasting model, with no need to estimate large numbers of models within a class, or to carry out a computationally-intensive cross-validation procedure. This framework is implemented in the open source R package `seer` and is publicly available on github (https://github.com/thiyangt/seer).

2 FFORMS framework: application to M4 competition data

As shown in Figure 1, the FFORMS framework consists of two main components: i) *offline phase*, which includes the development of a classification model and ii) *online phase*, use the classification model developed in the offline phase to identify "best" forecast-model. We develop separate classifiers for yearly, monthly, quarterly, weekly, daily and hourly series. Now we will illustrate the implementation of FFORMS framework over the course of the M4 competition.

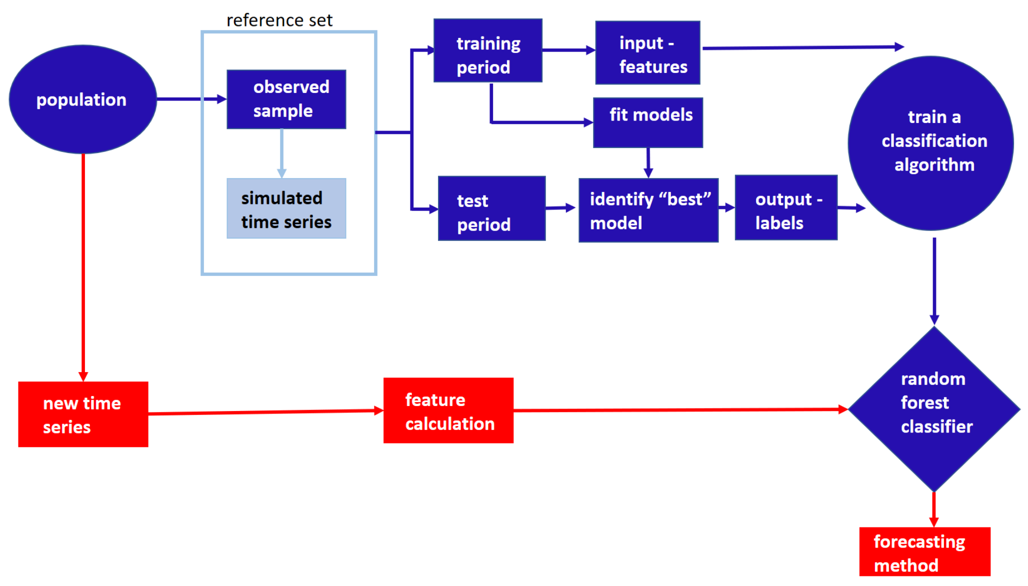

Figure 1: *FFORMS (Feature-based FORecast-Model Selection) framework. The offline phase is shown in blue and the online phase by red.*

2.1 FFORMS framework: offline phase

2.1.1 observed sample

We split the time series in the M4 competition into training set and test set. The time series in the training set are used as the set of observed time series. The time series in the test set are used to evaluate the classification models. Further, for yearly, quarterly and monthly time series in addition to the time series provided in the M4 competition we used the time series of M1 and M3 competitions. Table 1 summarizes the number of time series in the observed sample and the test set in each frequency category.

Table 1: *Composition of the time series in the observed sample and the test set*

Frequency	Observed Sample			Test set
	M1	M3	M4	M4
Yearly	181	645	22000	1000
Quarterly	203	756	23000	1000
Monthly	617	1428	47000	1000
Weekly	-	-	259	100
Daily	-	-	4001	226
Hourly	-	-	350	64

2.1.2 simulated time series

As described in Talagala, Hyndman & Athanasopoulos (2018), we augment the reference set by adding multiple time series simulated based on each series in the M4 competition. We use several standard automatic forecasting algorithms to simulate multiple time series from each series. Table 2 shows the different automatic forecasting algorithms used under each frequency category. The automated ETS and ARIMA are implemented using `ets` and `auto.arima` functions available in the forecast package in R (Hyndman et al. 2018). The `stlf` function in the forecast package (Hyndman et al. 2018) is used to simulate multiple time series based on multiple seasonal decomposition approach. As shown in Table 2 we fit models to each time series in the M4 competition database from the corresponding algorithm and then simulate multiple time series from the selected models. Before simulating time series from daily and hourly series we convert the time series into multiple seasonal time series (msts) objects. For daily time series with length less 366 the frequency is set to 7 and if the time series is long enough to take more than a year (length > 366), the series is converted to a multiple seasonal time series objects with frequencies 7 and 365.25. For hourly series, if the series length is shorter than 168, frequency is set to 24, if the length of the series is greater than 168 and less than or equals to 8766 only daily and weekly seasonality are allowed setting the frequencies to 24 and 168. In this experiment the length of the simulated time series is set to be equal to: length of the training period specified in the M4 competition + length of the forecast horizon specified in the competition. For example, the series with id "Y13190" contains a training period of length 835. The length of the simulated series generated based on this series is equals to 841 (835+6).

Table 2: *Automatic forecasting algorithms used to simulate time series*

Algorithm	Y	Q	M	W	D	H
automated ETS	✓	✓	✓			
automated ARIMA	✓	✓	✓			
forecast based on multiple seasonal decomposition				✓	✓	✓

As shown in Figure 1, the observed time series and the simulated time series form the reference to build our classification algorithm. Once we create the reference set for random forest training we split each time series in the reference set into training period and test period.

2.1.3 Input: features

The FFORMS framework operates on the features of the time series. For each time series in the reference set features are calculated based on the training period of the time series.

Table 3: *Time series features*

	Feature	Description	Y	Q/M	W	D/H
1	T	length of time series	✓	✓	✓	✓
2	trend	strength of trend	✓	✓	✓	✓
3	seasonality 1	strength of seasonality corresponds to frequency 1	-	✓	✓	✓
4	seasonality 2	strength of seasonality corresponds to frequency 2	-	-	-	✓
5	linearity	linearity	✓	✓	✓	✓
6	curvature	curvature	✓	✓	✓	✓
7	spikiness	spikiness	✓	✓	✓	✓
8	e_acf1	first ACF value of remainder series	✓	✓	✓	✓
9	stability	stability	✓	✓	✓	✓
10	lumpiness	lumpiness	✓	✓	✓	✓
11	entropy	spectral entropy	✓	✓	✓	✓
12	hurst	Hurst exponent	✓	✓	✓	✓
13	nonlinearity	nonlinearity	✓	✓	✓	✓
14	alpha	ETS(A,A,N) $\hat{\alpha}$	✓	✓	✓	-
15	beta	ETS(A,A,N) $\hat{\beta}$	✓	✓	✓	-
16	hwalpha	ETS(A,A,A) $\hat{\alpha}$	-	✓	-	-
17	hwbeta	ETS(A,A,A) $\hat{\beta}$	-	✓	-	-
18	hwgamma	ETS(A,A,A) $\hat{\gamma}$	-	✓	-	-
19	ur_pp	test statistic based on Phillips-Perron test	✓	-	-	-
20	ur_kpss	test statistic based on KPSS test	✓	-	-	-
21	y_acf1	first ACF value of the original series	✓	✓	✓	✓
22	diff1y_acf1	first ACF value of the differenced series	✓	✓	✓	✓
23	diff2y_acf1	first ACF value of the twice-differenced series	✓	✓	✓	✓
24	y_acf5	sum of squares of first 5 ACF values of original series	✓	✓	✓	✓
25	diff1y_acf5	sum of squares of first 5 ACF values of differenced series	✓	✓	✓	✓
26	diff2y_acf5	sum of squares of first 5 ACF values of twice-differenced series	✓	✓	✓	✓
27	seas_acf1	autocorrelation coefficient at first seasonal lag	-	✓	✓	✓
28	sediff_acf1	first ACF value of seasonally-differenced series	-	✓	✓	✓
29	sediff_seacf1	ACF value at the first seasonal lag of seasonally-differenced series	-	✓	✓	✓
30	sediff_acf5	sum of squares of first 5 autocorrelation coefficients of seasonally-differenced series	-	✓	✓	✓
31	lmres_acf1	first ACF value of residual series of linear trend model	✓	-	-	-
32	y_pacf5	sum of squares of first 5 PACF values of original series	✓	✓	✓	✓
33	diff1y_pacf5	sum of squares of first 5 PACF values of differenced series	✓	✓	✓	✓
34	diff2y_pacf5	sum of squares of first 5 PACF values of twice-differenced series	✓	✓	✓	✓

The description of the features calculated under each frequency category is shown in Table 3. A comprehensive description of the features used in the experiment is given in Talagala, Hyndman & Athanasopoulos (2018).

2.1.4 Output: class-labels

In addition to the class labels used by Talagala, Hyndman & Athanasopoulos (2018) we include some more class labels when applying the FFORMS framework to the M4 competition time series. The description of class labels considered under each frequency is shown in Table 4. We fit the corresponding models outlined in Table 4 to each series in the reference set. The models are estimated using the training period for each series, and forecasts are produced for the test periods.

Table 4: *Class labels*

class label	Description	Y	Q/M	W	D/H
WN	white noise process	✓	✓	✓	✓
AR/MA/ARMA	AR, MA, ARMA processes	✓	✓	✓	-
ARIMA	ARIMA process	✓	✓	✓	-
SARIMA	seasonal ARIMA	✓	✓	✓	-
RWD	random walk with drift	✓	✓	✓	✓
RW	random walk	✓	✓	✓	✓
Theta	standard theta method	✓	✓	✓	✓
STL-AR		-	✓	✓	✓
ETS-notrendnoseasonal	ETS without trend and seasonal components	✓	✓	✓	-
ETStrendonly	ETS with trend component and without seasonal component	✓	✓	✓	-
ETSdampedtrend	ETS with damped trend component and without seasonal component	✓	✓	-	-
ETStrendseasonal	ETS with trend and seasonal components	-	✓	-	-
ETSdampedtrendseasonal	ETS with damped trend and seasonal components	-	✓	-	-
ETSseasonalonly	ETS with seasonal components and without trend component	-	✓	-	-
snaive	seasonal naive method	✓	✓	✓	✓
tbats	TBATS forecasting	-	✓	✓	✓
nn	neural network time series forecasts	✓	✓	✓	✓
mstlets		-	-	✓	✓
mstlarima		-	-	-	✓

The `auto.arima` and `ets` functions in the forecast package are used to identify the suitable (S)ARIMA and ETS models. In order to identify the "best" forecast-model for each time series in the reference set we combine the mean Absolute Scaled Error (MASE) and the symmetric Mean Absolute Percentage Error (MAPE) calculated over the test set. More specifically, for each series both forecast error measures MASE and sMAPE are calculated for each of the forecast models. Each of these is respectively standardized by the median MASE and median sMAPE calculated across the methods. The model with the lowest average value of the scaled MASE and scaled sMAPE is selected as the output class-label. Most of the labels given in Table 4 are self-explanatory labels. In STL-AR, mstlets, and mstlarima, first STL decomposition method applied to the time series and then seasonal naive method is used to forecast the seasonal component. Finally, AR, ETS and ARIMA models are used to forecast seasonally adjusted data respectively.

2.1.5 Train a random forest classifier

A random forest with class priors is used to develop the classifier. We build separate random forest classifiers for yearly, quarterly, monthly, weekly, daily and hourly time series. The wrapper function called `build_rf` in the `seer` package enables the training of a random forest and returns class labels("best" forecast-model) for each time series.

2.2 FFORMS framework: online phase

2.2.1 Generate point forecasts and 95% prediction intervals for the M4 competition data

First, the corresponding features are calculated based on the full length of the training period provided by the M4 competition. Second, point forecasts and 95% prediction intervals are calculated based on the predicted class labels, in this case forecast-models. Finally, all negative values are set to zero.

3 The seer package in R

Table 5 summarizes the main functions implemented in the seer package related to the FFORMS framework. To install the package:

```r
install.packages("devtools")
devtools::install_github("thiyangt/seer")
library(seer)
```

Table 5: *Main functions in the seer package*

Function in seer	Description
simulate_arimabased	simulate time series based on ARIMA models
simulate_etsbased	simulate time series based on ETS models
simulate_mstlbased	simulate time series based on multiple seasonal decomposition approach
convert_msts	convert daily hourly time series into msts objects
cal_features	calculate features
cal_m4measures	calculate MASE and sMAPE for a given forecast-model
cal_medianscaled	scale MASE and sMAPE by median and return the average
fcast_accuracy	calculate accuracy measures for a list of time series from the specified models
prepare_trainingset	construct the training dataframe to build a random forest
build_rf	build a random forest and produce class labels
rf_forecast	calculate point forecasts and prediction intervals

References

Hyndman, R, G Athanasopoulos, C Bergmeir, G Caceres, L Chhay, M O'Hara-Wild, F Petropoulos, S Razbash, E Wang & F Yasmeen (2018). *forecast: Forecasting functions for time series and linear models*. R package version 8.3. http://pkg.robjhyndman.com/forecast.

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