seer: R package for feature-based forecast model selection

Thiyanga S Talagala
Rob J Hyndman
George Athanasopoulos

Monash University, Australia

UseR, 2018

Large collections of time series

- Forecasting demand for thousands of products across multiple warehouses.

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.
- Examples for time series features

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.
- Examples for time series features
  - strength of trend

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.
- Examples for time series features
  - strength of trend
  - strength of seasonality

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.
- Examples for time series features
  - strength of trend
  - strength of seasonality
  - lag-1 autocorrelation

Time series features

Objective

Develop a framework that automates the selection of the most appropriate forecasting method for a given time series by using an array of features computed from the time series.

- **Basic idea:**
  Transform a given time series $y = \{y_1, y_2, \cdots, y_n\}$ to a feature vector $F = (f_1(y), f_2(y), \cdots, f_p(y))'$.
- Examples for time series features
  - strength of trend
  - strength of seasonality
  - lag-1 autocorrelation
  - spectral entropy

Time series features

- length
- strength of seasonality
- strength of trend
- linearity
- curvature
- spikiness
- stability
- lumpiness
- first ACF value of remainder series
- parameter estimates of Holt's linear trend method
- spectral entropy
- Hurst exponent
- nonlinearity
- parameter estimates of Holt-Winters' additive method
- unit root test statistics
- first ACF value of residual series of linear trend model
- ACF and PACF based features - calculated on both the raw and differenced series

Methodology: FFORMS

FFORMS: Feature-based **FOR**ecast **M**odel **S**election

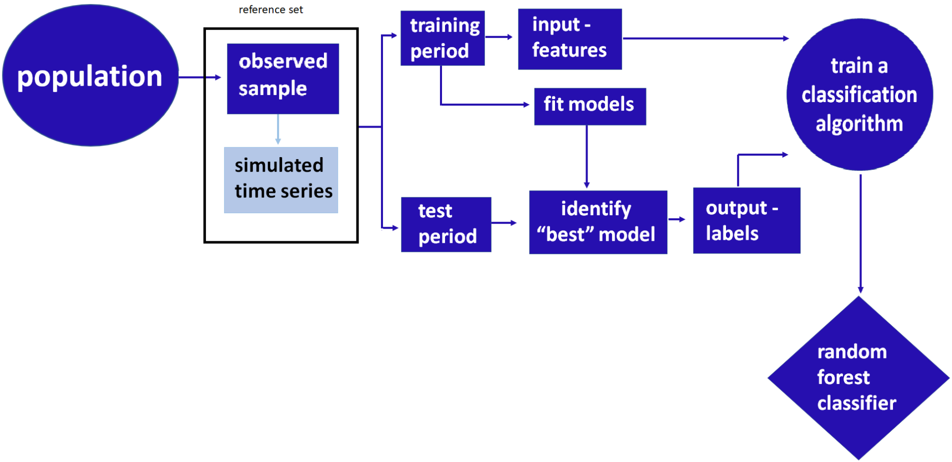
Offline

- A classification algorithm (the meta-learner) is trained.

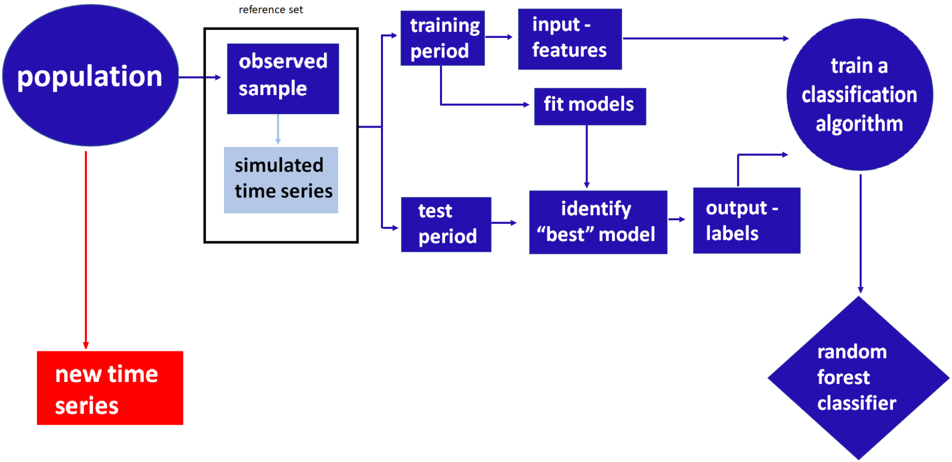
Online

- Calculate the features of a time series and use the pre-trained classifier to identify the best forecasting method.

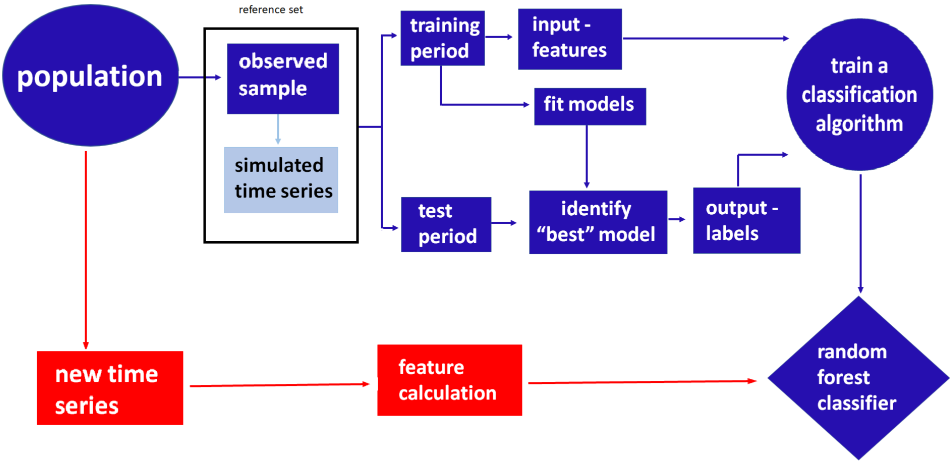
FFORMS: “offline” part of the algorithm



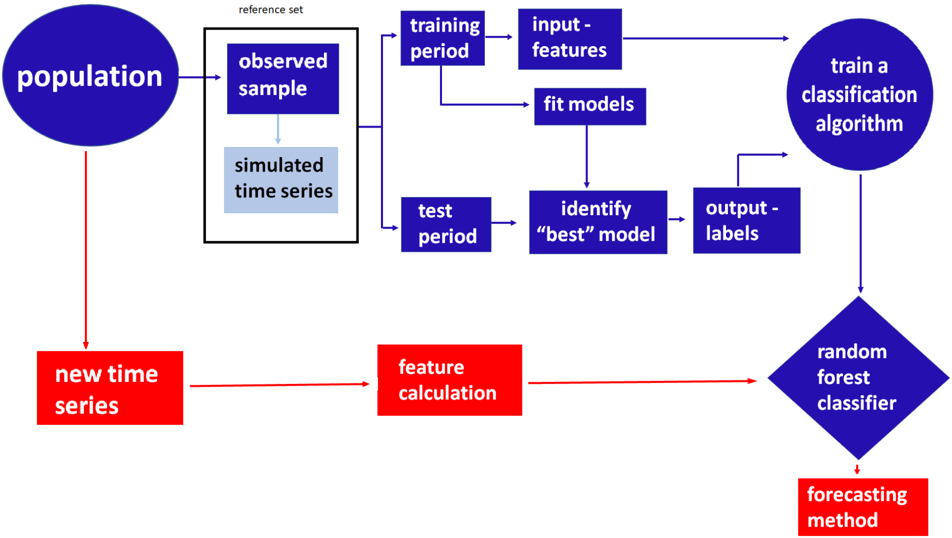
FFORMS: “online” part of the algorithm



FFORMS: “online” part of the algorithm



FFORMS: "online" part of the algorithm



R package: seer

Installation

```
devtools::install_github("thiyangt/seer")
library(seer)
```

R package: seer

Installation

```
devtools::install_github("thiyangt/seer")
library(seer)
```

Example datasets

observed time series - M1 yearly series (181)

```
library(Mcomp)
yearlym1 <- subset(M1, "yearly")
```

Installation

```
devtools::install_github("thiyangt/seer")
library(seer)
```

Example datasets

observed time series - M1 yearly series (181)

```
library(Mcomp)
yearlym1 <- subset(M1, "yearly")
```

new time series - M3 yearly series (645)

```
yearlym3 <- subset(M3, "yearly")
```

Input: features

```r
cal_features(yearlym1[1:3], database="M3",
h=6, highfreq=FALSE)
```

```
# A tibble: 3 x 25
  entropy lumpiness stability hurst trend
    <dbl>     <dbl>     <dbl> <dbl> <dbl>
1   0.683    0.0400     0.977 0.985 0.985
2   0.711    0.0790     0.894 0.988 0.989
3   0.716    0.0160     0.858 0.987 0.989
# ... with 20 more variables: spikiness <dbl>,
#   linearity <dbl>, curvature <dbl>,
#   e_acf1 <dbl>, y_acf1 <dbl>,
#   diff1y_acf1 <dbl>, diff2y_acf1 <dbl>,
#   y_pacf5 <dbl>, diff1y_pacf5 <dbl>,
#   diff2y_pacf5 <dbl>, nonlinearity <dbl>,
#   lmres_acf1 <dbl>, ur_pp <dbl>, ur_kpss <dbl>,
#   N <int>, y_acf5 <dbl>, diff1y_acf5 <dbl>,
#   diff2y_acf5 <dbl>, alpha <dbl>, beta <dbl>
```

Output: labels

```
fcast_accuracy(yearlym1[1:3],
  models=c("arima","ets","rw","rwd","theta","nn"),
  database="M3", cal_MASE, h=6, length_out=1)
```

```
$accuracy
         arima       ets       rw       rwd
YAF2 10.527612 10.319029 13.52428 10.527612
YAF3  5.713867  7.704409  7.78949  5.225965
YAF4  8.633590  8.091416 11.55633  8.440105
         theta       nn
YAF2 12.088375 11.79341
YAF3  6.225463  6.70077
YAF4  9.952742 10.78474

$ARIMA
                      YAF2
"ARIMA(0,1,0) with drift"
                      YAF3
"ARIMA(0,1,1) with drift"
                      YAF4
"ARIMA(0,1,2) with drift"

$ETS
```

Reference set

```
accuracy_m1 <- fcast_accuracy(tslist=yearlym1,
models= c("arima","ets","rw","rwd", "theta", "nn"),
database ="M1", cal_MASE)

features_m1 <- cal_features(yearlym1, database="M1", highfreq = FALSE)

reference_set <- prepare_trainingset(accuracy_set = accuracy_m1,
feature_set = features_m1)
head(reference_set$trainingset, 1)
```

```
# A tibble: 1 x 26
  entropy lumpiness stability hurst trend
    <dbl>     <dbl>     <dbl> <dbl> <dbl>
1   0.683    0.0400     0.977 0.985 0.985
# ... with 21 more variables: spikiness <dbl>,
#   linearity <dbl>, curvature <dbl>,
#   e_acf1 <dbl>, y_acf1 <dbl>,
#   diff1y_acf1 <dbl>, diff2y_acf1 <dbl>,
#   y_pacf5 <dbl>, diff1y_pacf5 <dbl>,
#   diff2y_pacf5 <dbl>, nonlinearity <dbl>,
#   lmres_acf1 <dbl>, ur_pp <dbl>, ur_kpss <dbl>,
#   N <int>, y_acf5 <dbl>, diff1y_acf5 <dbl>,
#   diff2y_acf5 <dbl>, alpha <dbl>, beta <dbl>,
#   classlabels <chr>
```

FFORMS classifier

```r
ym3_features <- cal_features(yearlym3,
                  database="M3", highfreq = FALSE)

fforms <- build_rf(training_set = ref_set$trainingset,
          testset=ym3_features,  rf_type="rcp",
            ntree=100, seed=7, import=FALSE)

fforms$predictions %>% head(10)
```

```
##         1         2         3         4         5
## ETS-trend       rwd       rwd       rwd       rwd
##         6         7         8         9        10
##       rwd       rwd       rwd       rwd       rwd
##        11        12        13        14        15
##       rwd ETS-trend       rwd       rwd        nn
##        16        17        18        19        20
##       rwd       rwd       rwd       rwd     ARIMA
## 10 Levels: ARIMA ARMA/AR/MA ... wn
```

Generate point foecasts and 95% prediction intervals

```
rf_forecast(fforms$predictions[1:2],
tslist=yearlym3[1:2], database="M3",
function_name="cal_MASE", h=6, accuracy=TRUE)
```

```
## $mean
##          [,1]     [,2]     [,3]     [,4]     [,5]
## [1,] 5486.429 6035.865 6585.301 7134.737 7684.173
## [2,] 4402.227 4574.454 4746.681 4918.908 5091.135
##          [,6]
## [1,] 8233.609
## [2,] 5263.362
##
## $lower
##          [,1]     [,2]     [,3]     [,4]     [,5]
## [1,] 4984.162 4893.098 4629.135 4199.745 3606.858
## [2,] 2890.401 2366.671 1959.916 1608.186 1288.666
##           [,6]
## [1,] 2848.8735
## [2,]  990.2221
##
## $upper
##          [,1]     [,2]     [,3]      [,4]
```

Augmenting the observed sample with simutated time series

```r
lapply(yearlym1[1], sim_arimabased, Nsim=2)
```

```
## $YAF2
## $YAF2[[1]]
## Time Series:
## Start = 1972
## End = 1993
## Frequency = 1
##  [1]   3600.00  70754.98  86574.90 114430.28
##  [5] 170293.02 242857.02 275962.55 334544.74
##  [9] 363978.42 384279.34 400087.38 391343.35
## [13] 448735.28 500545.96 572841.43 585433.39
## [17] 604188.86 632861.91 684580.54 727014.00
## [21] 791626.19 851251.41
##
## $YAF2[[2]]
## Time Series:
## Start = 1972
## End = 1993
## Frequency = 1
##  [1]   3600.000  -6522.126  78042.020 121578.099
##  [5] 116672.023 164651.033 162514.942 188372.664
##  [9] 191341.916 186677.219 162508.736 184740.761
## [13] 188927.351 194891.212 181393.425 238103.863
## [17] 316160.328 341523.717 411479.674 417012.639
## [21] 442201.913 420179.467
```

Augmenting the observed sample with simutated time series

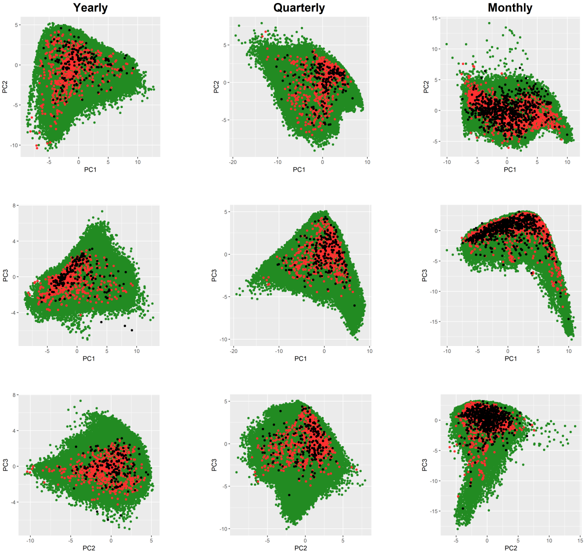
```
lapply(yearlym1[1], sim_arimabased, Nsim=2)
```

```
## $YAF2
## $YAF2[[1]]
## Time Series:
## Start = 1972
## End = 1993
## Frequency = 1
##  [1]   3600.00  70754.98  86574.90 114430.28
##  [5] 170293.02 242857.02 275962.55 334544.74
##  [9] 363978.42 384279.34 400087.38 391343.35
## [13] 448735.28 500545.96 572841.43 585433.39
## [17] 604188.86 632861.91 684580.54 727014.00
## [21] 791626.19 851251.41
##
## $YAF2[[2]]
## Time Series:
## Start = 1972
## End = 1993
## Frequency = 1
##  [1]   3600.000  -6522.126  78042.020 121578.099
##  [5] 116672.023 164651.033 162514.942 188372.664
##  [9] 191341.916 186677.219 162508.736 184740.761
## [13] 188927.351 194891.212 181393.425 238103.863
## [17] 316160.328 341523.717 411479.674 417012.639
## [21] 442201.913 420179.467
```

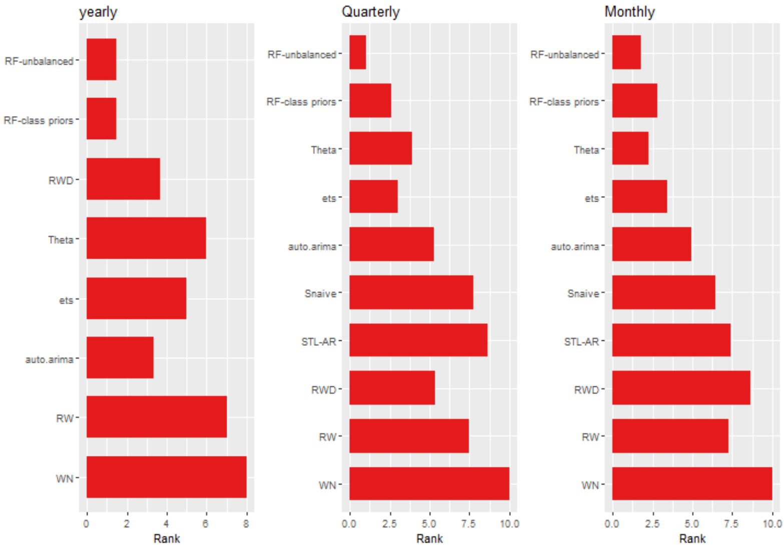
```
lapply(yearlym1[1], sim_etsbased, Nsim=2)
lapply(yearlym1[1], sim_mstlbased, Nsim=2)
```

Application: Distribution of time series in the PCA space

observed - M1 | simulated | new - M3



Results



yearly
RF-unbalanced
RF-class priors
RWD
Theta
ets
auto.arima
RW
WN
0
2
4
6
8
Rank
Quarterly
RF-unbalanced
RF-class priors
Theta
ets
auto.arima
Snaive
STL-AR
RWD
RW
WN
0.0
2.5
5.0
7.5
10.0
Rank
Monthly
RF-unbalanced
RF-class priors
Theta
ets
auto.arima
Snaive
STL-AR
RWD
RW
WN
0.0
2.5
5.0
7.5
10.0
Rank

Discussion

- FFORMS: framework for forecast model selection using meta-learning based on time series features.

Discussion

- FFORMS: framework for forecast model selection using meta-learning based on time series features.
- FFORMS algorithm uses the knowledge of the past performance of candidate forecast models on a collection of time series in order to identify the best forecasting method for a new series.

Discussion

- FFORMS: framework for forecast model selection using meta-learning based on time series features.
- FFORMS algorithm uses the knowledge of the past performance of candidate forecast models on a collection of time series in order to identify the best forecasting method for a new series.
- For real-time forecasting, our framework involves only the calculation of features, the selection of a forecast method based on the FFORMS random forest classifier, and the calculation of the forecasts from the chosen model.

Discussion

- FFORMS: framework for forecast model selection using meta-learning based on time series features.
- FFORMS algorithm uses the knowledge of the past performance of candidate forecast models on a collection of time series in order to identify the best forecasting method for a new series.
- For real-time forecasting, our framework involves only the calculation of features, the selection of a forecast method based on the FFORMS random forest classifier, and the calculation of the forecasts from the chosen model.
- We have also introduced a simple set of time series features that are useful in identifying the "best" forecast method for a given time series.

R package: seer

available at: https://github.com/thiyangt/seer

R package: seer

available at: https://github.com/thiyangt/seer

paper: https://robjhyndman.com/publications/fforms/

Email: thiyanga.talagala@monash.edu

twitter: thiyangt