Package ‘seer’

October 14, 2022

Type Package
Title Feature-Based Forecast Model Selection
Version 1.1.8
Maintainer Thiyanga Talagala <tstalagala@gmail.com>
Description A novel meta-learning framework for forecast model selection using time series features. Many applications require a large number of time series to be forecast. Providing better forecasts for these time series is important in decision and policy making. We propose a classification framework which selects forecast models based on features calculated from the time series. We call this framework FFORMS (Feature-based FORecast Model Selection). FFORMS builds a mapping that relates the features of time series to the best forecast model using a random forest. ‘seer’ package is the implementation of the FFORMS algorithm. For more details see our paper at <https://www.monash.edu/business/econometrics-and-business-statistics/research/publications/ebs/wp06-2018.pdf>.
License GPL-3
URL https://thiyangt.github.io/seer/
BugReports https://github.com/thiyangt/seer/issues
Depends R (>= 3.2.3)
Imports stats, urca, forecast (>= 8.3), dplyr, magrittr, randomForest, forecast, stringr, tibble, purrr, future, furrr, utils, tsfeatures
Encoding UTF-8
RoxygenNote 7.2.1
Suggests testthat (>= 2.1.0), covr, repmisc, knitr, rmarkdown, ggplot2, tidyr, Mcomp, GGally
NeedsCompilation no
Author Thiyanga Talagala [aut, cre] (<https://orcid.org/0000-0002-0656-9789>), Rob J Hyndman [ths, aut] (<https://orcid.org/0000-0002-2140-5352>), George Athanasopoulos [ths, aut]
Repository CRAN
Date/Publication 2022-10-01 07:10:02 UTC
R topics documented:

accuracy_arima .................................................. 3
accuracy_ets ..................................................... 3
accuracy_mstl .................................................... 4
accuracy_nn ....................................................... 4
accuracy_rw ....................................................... 5
accuracy_rwd ..................................................... 5
accuracy_snaive .................................................. 6
accuracy_stlar .................................................... 6
accuracy_tbats ..................................................... 7
accuracy_theta .................................................... 7
accuracy_wn ....................................................... 8
acf5 ............................................................. 8
acf_seasonalDiff ................................................ 9
build_rf .......................................................... 9
cal_features ..................................................... 10
cal_m4measures .................................................. 11
cal_MASE ........................................................ 12
cal_medianscaled ............................................... 12
cal_sMAPE ....................................................... 13
cal_WA ........................................................... 13
classify_labels ................................................. 14
classlabel ......................................................... 14
combination_forecast_inside .................................. 15
convert_msts ..................................................... 15
e_acf1 ............................................................ 16
fcast_accuracy ................................................... 16
fforms_combinationforecast .................................. 17
fforms_ensemble ................................................. 18
holtWinter_parameters ......................................... 19
prepare_trainingset ............................................. 19
rf_forecast ....................................................... 20
sim_arimabased ............................................... 21
sim_etsbased .................................................... 22
sim_mstlbased ................................................... 23
split_names ...................................................... 24
stlar ............................................................. 24
unitroot ......................................................... 25

Index 26
**accuracy_arima**

*Calculate accuracy measure based on ARIMA models*

**Description**

Calculate accuracy measure based on ARIMA models

**Usage**

```r
accuracy_arima(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function

**Value**

a list which contains the accuracy and name of the specific ARIMA model.

---

**accuracy_ets**

*Forecast-accuracy calculation*

**Description**

Calculate accuracy measure based on ETS models

**Usage**

```r
accuracy_ets(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function

**Value**

a list which contains the accuracy and name of the specific ETS model.
**accuracy_mstl**  
*Calculate accuracy based on MSTL*

**Description**  
Calculate accuracy based on MSTL

**Usage**  
```
accuracy_mstl(ts_info, function_name, length_out, mtd)
```

**Arguments**
- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function
- `mtd`: Method to use for forecasting the seasonally adjusted series

**Value**  
accuracy measure calculated based on multiple seasonal decomposition

---

**accuracy_nn**  
*Calculate accuracy measure calculated based on neural network forecasts*

**Description**  
Calculate accuracy measure calculated based on neural network forecasts

**Usage**  
```
accuracy_nn(ts_info, function_name, length_out)
```

**Arguments**
- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function

**Value**  
accuracy measure calculated based on neural network forecasts
**accuracy_rw**

*Calculate accuracy measure based on random walk models*

**Description**

Calculate accuracy measure based on random walk models

**Usage**

```r
accuracy_rw(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function

**Value**

returns accuracy measure calculated based on random walk model

---

**accuracy_rwd**

*Calculate accuracy measure based on random walk with drift*

**Description**

Calculate accuracy measure based on random walk with drift

**Usage**

```r
accuracy_rwd(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info`: list containing training and test part of a time series
- `function_name`: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out`: number of measures calculated by the function

**Value**

accuracy measure calculated based on random walk with drift model
accuracy_snaive  
*Calculate accuracy measure based on snaive method*

**Description**

Calculate accuracy measure based on snaive method

**Usage**

accuracy_snaive(ts_info, function_name, length_out)

**Arguments**

- **ts_info**: list containing training and test part of a time series
- **function_name**: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- **length_out**: number of measures calculated by the function

**Value**

accuracy measure calculated based on snaive method

accuracy_stlar  
*Calculate accuracy measure based on STL-AR method*

**Description**

Calculate accuracy measure based on STL-AR method

**Usage**

accuracy_stlar(ts_info, function_name, length_out)

**Arguments**

- **ts_info**: list containing training and test part of a time series
- **function_name**: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- **length_out**: number of measures calculated by the function

**Value**

accuracy measure calculated based on stlar method
**accuracy_tbats**

*Calculate accuracy measure based on TBATS*

**Description**

Calculate accuracy measure based on TBATS

**Usage**

```
accuracy_tbats(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info` : list containing training and test part of a time series
- `function_name` : function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out` : number of measures calculated by the function

**Value**

accuracy measure calculated based on TBATS models

---

**accuracy_theta**

*Calculate accuracy measure based on Theta method*

**Description**

Calculate accuracy measure based on Theta method

**Usage**

```
accuracy_theta(ts_info, function_name, length_out)
```

**Arguments**

- `ts_info` : list containing training and test part of a time series
- `function_name` : function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- `length_out` : number of measures calculated by the function

**Value**

returns accuracy measure calculated based on theta method
accuracy_wn

*Calculate accuracy measure based on white noise process*

**Description**

Calculate accuracy measure based on white noise process

**Usage**

accuracy_wn(ts_info, function_name, length_out)

**Arguments**

- **ts_info**: list containing training and test part of a time series
- **function_name**: function to calculate the accuracy function, the arguments of this function should be forecast, training and test set of the time series
- **length_out**: number of measures calculated by the function

**Value**

returns accuracy measure calculated based on white noise process

acf5

*Autocorrelation-based features*

**Description**

Computes various measures based on autocorrelation coefficients of the original series, first-differenced series and second-differenced series

**Usage**

acf5(y)

**Arguments**

- **y**: a univariate time series

**Value**

A vector of 3 values: sum of squared of first five autocorrelation coefficients of original series, first-differenced series, and twice-differenced series.

**Author(s)**

Thiyanga Talagala
acf_seasonalDiff

Autocorrelation coefficients based on seasonally differenced series

Description

Autocorrelation coefficients based on seasonally differenced series

Usage

acf_seasonalDiff(y, m, lagmax)

Arguments

y a univariate time series
m frequency of the time series
lagmax maximum lag at which to calculate the acf

Value

A vector of 3 values: first ACF value of seasonally-differenced series, ACF value at the first seasonal lag of seasonally-differenced series, sum of squares of first 5 autocorrelation coefficients of seasonally-differenced series.

Author(s)

Thiyanga Talagala

build_rf

build random forest classifier

Description

train a random forest model and predict forecast-models for new series

Usage

build_rf(
  training_set,
  testset = FALSE,
  rf_type = c("ru", "rcp"),
  ntree, seed,
  import = FALSE,
  mtry = 8
)
**Arguments**

- **training_set**: data frame of features and class labels
- **testset**: features of new time series, default FALSE if a testset is not available
- **rf_type**: whether ru(random forest based on unbiased sample) or rcp(random forest based on class priors)
- **ntree**: number of trees in the forest
- **seed**: a value for seed
- **import**: Should importance of predictors be assessed?, TRUE of FALSE
- **mtry**: number of features to be selected at each node

**Value**

a list containing the random forest and forecast-models for new series

**Description**

Computes relevant time series features before applying them to the model

**Usage**

```r
cal_features(
  tslist,
  seasonal = FALSE,
  m = 1,
  lagmax = 2L,
  database,
  h,
  highfreq
)
```

**Arguments**

- **tslist**: a list of univariate time series
- **seasonal**: if FALSE, restricts to features suitable for non-seasonal data
- **m**: frequency of the time series or minimum frequency in the case of msts objects
- **lagmax**: maximum lag at which to calculate the acf (quarterly series-5L, monthly-13L, weekly-53L, daily-8L, hourly-25L)
- **database**: whether the time series is from mcomp or other
- **h**: forecast horizon
- **highfreq**: whether the time series is weekly, daily or hourly
Value

dataframe: each column represent a feature and each row represent a time series

Author(s)

Thiyanga Talagala

cal_m4measures  Mean of MASE and sMAPE

Description

Calculate MASE and sMAPE for an individual time series

Usage

cal_m4measures(training, test, forecast)

Arguments

training  training period of a time series

test  test period of a time series

forecast  forecast obtained from a fitted to the training period

Value

returns a single value: mean on MASE and sMAPE

Author(s)

Thiyanga Talagala

Examples

require(Mcomp)
require(magrittr)
ts <- Mcomp::M3[[1]]$x
fcast_arima <- auto.arima(ts) %>% forecast(h=6)
cal_m4measures(M3[[1]]$x, M3[[1]]$xx, fcast_arima$mean)
cal_MASE

*Mean Absolute Scaled Error (MASE)*

**Description**
Calculation of mean absolute scaled error

**Usage**

```r
cal_MASE(training, test, forecast)
```

**Arguments**

- `training`: training period of the time series
- `test`: test period of the time series
- `forecast`: forecast values of the series

**Value**
returns a single value

**Author(s)**
Thiyanga Talagala

---

cal_medianscaled

*scale MASE and sMAPE by median*

**Description**
Given a matrix of MASE and sMAPE for each forecasting method and scaled by median and take the mean of MASE-scaled by median and sMAPE-scaled by median as the forecast accuracy measure to identify the class labels

**Usage**

```r
cal_medianscaled(x)
```

**Arguments**

- `x`: output form the function `fcast_accuracy`, where the parameter `accuracyFun = cal_m4measures`

**Value**
a list with accuracy matrix, vector of arima models and vector of ets models the accuracy for each forecast-method is average of scaled-MASE and scaled-sMAPE. Median of MASE and sMAPE calculated based on forecast produced from different models for a given series.
**Description**

Calculation of symmetric mean absolute percentage error

**Usage**

\[
\text{cal\_sMAPE}(\text{training}, \text{test}, \text{forecast})
\]

**Arguments**

- **training**: training period of the time series
- **test**: test period of the time series
- **forecast**: forecast values of the series

**Value**

returns a single value

**Author(s)**

Thiyanga Talagala

---

**Description**

Weighted Average(WA) calculated based on MASE, sMAPE for an individual time series

**Usage**

\[
\text{cal\_WA}(\text{training}, \text{test}, \text{forecast})
\]

**Arguments**

- **training**: training period of a time series
- **test**: test period of a time series
- **forecast**: forecast obtained from a fitted to the training period

**Value**

returns a single value: WA based on MASE and sMAPE


Author(s)
Thiyanga Talagala

classify_labels

Classify labels according to the FFORMS framework

Description
This function further classify class labels as in FFORMS framework

Usage
classify_labels(df_final)

Arguments
df_final a dataframe: output from split_names function

Value
a vector of class labels in FFORMS framework

classlabel

identify the best forecasting method

Description
identify the best forecasting method according to the forecast accuracy measure

Usage
classlabel(accuracy_mat)

Arguments
accuracy_mat matrix of forecast accuracy measures (rows: time series, columns: forecasting method)

Value
a vector: best forecasting method for each series corresponding to the rows of accuracy_mat

Author(s)
Thiyanga Talagala
**combination_forecast_inside**

*This function is call to be inside fforms_combination*

---

**Description**

Given weights and time series in a two separate vectors calculate combination forecast.

**Usage**

```r
combination_forecast_inside(x, y, h)
```

**Arguments**

- `x`: weights and names of models (output based on fforms.ensemble)
- `y`: time series values
- `h`: forecast horizon

**Value**

List of combination forecasts corresponds to point, lower and upper.

**Author(s)**

Thiyanga Talagala

---

**convert_msts**

*Convert multiple frequency time series into msts object*

---

**Description**

Convert multiple frequency (daily, hourly, half-hourly, minutes, seconds) time series into msts object.

**Usage**

```r
convert_msts(y, category)
```

**Arguments**

- `y`: univariate time series
- `category`: frequency data have been collected

**Value**

A ts object or msts object
e_acf1  
*Autocorrelation coefficient at lag 1 of the residuals*

Description
Computes the first order autocorrelation of the residual series of the deterministic trend model

Usage
`e_acf1(y)`

Arguments
- `y`: a univariate time series

Value
A numeric value.

Author(s)
Thiyanga Talagala

fcast_accuracy  
*calculate forecast accuracy from different forecasting methods*

Description
Calculate forecast accuracy on test set according to a specified criterion

Usage
```r
fcast_accuracy(
  tslist,
  models = c("ets", "arima", "rw", "rwd", "wn", "theta", "stlar", "nn", "snaive", 
              "mstlarima", "mstlets", "tbats"),
  database,
  accuracyFun,
  h,
  length_out,
  fcast_save
)
```
**fforms_combinationforecast**

Combination forecast based on fforms

**Arguments**

- **tslist**: a list of time series
- **models**: a vector of models to compute
- **database**: whether the time series is from mcomp or other
- **accuracyFun**: function to calculate the accuracy measure, the arguments for the accuracy function should be training, test and forecast
- **h**: forecast horizon
- **length_out**: number of measures calculated by a single function
- **fcast_save**: if the argument is TRUE, forecasts from each series are saved

**Value**

a list with accuracy matrix, vector of arima models and vector of ets models

**Author(s)**

Thiyanga Talagala

**Description**

Compute combination forecast based on the vote matrix probabilities

**Usage**

```r
fforms_combinationforecast(
  fforms.ensemble,
  tslist,
  database,
  h,
  holdout = TRUE,
  parallel = FALSE,
  multiprocess = future::multisession
)
```

**Arguments**

- **fforms.ensemble**: a list output from fforms_ensemble function
- **tslist**: list of new time series
- **database**: whether the time series is from mcom or other
fforms_ensemble

Function to identify models to compute combination forecast using FFORMS algorithm

Description

This function identify models to be use in producing combination forecast

Usage

fforms_ensemble(votematrix, threshold = 0.5)

Arguments

votematrix a matrix of votes of probabilities based of fforms random forest classifier
threshold threshold value for sum of probabilities of votes, default is 0.5

Value

a list containing the names of the forecast models

Author(s)

Thiyanga Talagala
holtWinter_parameters  Parameter estimates of Holt-Winters seasonal method

Description

Estimate the smoothing parameter for the level-alpha and the smoothing parameter for the trend-beta, and seasonality-gamma

Usage

holtWinter_parameters(y)

Arguments

y  a univariate time series

Value

A vector of 3 values: alpha, beta, gamma

Author(s)

Thiyanga Talagala

prepare_trainingset  preparation of training set

Description

Preparation of a training set for random forest training

Usage

prepare_trainingset(accuracy_set, feature_set)

Arguments

accuracy_set  output from the fcast_accuracy
feature_set  output from the cal_features

Value

dataframe consisting features and classlabels
rf_forecast  function to calculate point forecast, 95% confidence intervals, forecast-accuracy for new series

Description

Given the prediction results of random forest calculate point forecast, 95% confidence intervals, forecast-accuracy for the test set

Usage

rf_forecast(
    predictions,
    tslist,
    database,
    function_name,
    h,
    accuracy,
    holdout = TRUE
)

Arguments

predictions  prediction results obtained from random forest classifier

tslist       list of new time series

database     whethe the time series is from mcom or other

function_name specify the name of the accuracy function (for eg., cal_MASE, etc.) to calculate accuracy measure, ( if a user written function the arguments for the accuracy function should be training period, test period and forecast).

h            length of the forecast horizon

accuracy     if true a accuracity measure will be calculated

holdout      if holdout=TRUE take a holdout sample from your data to caldulate forecast accuracy measure, if FALSE all of the data will be used for forecasting. Default is TRUE

Value

a list containing, point forecast, confidence interval, accuracy measure

Author(s)

Thiyanga Talagala
**sim_arimabased**  
*Simulate time series based on ARIMA models*

**Description**

simulate multiple time series for a given series based on ARIMA models

**Usage**

```r
sim_arimabased(
  y,
  Nsim,
  Combine = TRUE,
  M = TRUE,
  Future = FALSE,
  Length = NA,
  extralength = NA
)
```

**Arguments**

- **y**: a time series or M-competition data time series (Mcomp)
- **Nsim**: number of time series to simulate
- **Combine**: if TRUE, training and test data in the M-competition data are combined and generate a time series corresponds to the full length of the series. Otherwise, it generate a time series based on the training period of the series.
- **M**: if TRUE, y is considered to be a Mcomp data object
- **Future**: if future=TRUE, the simulated observations are conditional on the historical observations. In other words, they are possible future sample paths of the time series. But if future=FALSE, the historical data are ignored, and the simulations are possible realizations of the time series model that are not connected to the original data.
- **Length**: length of the simulated time series. If future = FALSE, the Length argument should be NA.
- **extralength**: extra length need to be added for simulated time series

**Value**

A list of time series.

**Author(s)**

Thiyanga Talagala
**sim_etsbased**  
*Simulate time series based on ETS models*

**Description**

simulate multiple time series for a given series based on ETS models

**Usage**

```r
sim_etsbased(
  y,  
  Nsim, 
  Combine = TRUE, 
  M = TRUE, 
  Future = FALSE, 
  Length = NA, 
  extralength = NA
)
```

**Arguments**

- `y` | a time series or M-competition data time series (Mcomp)
- `Nsim` | number of time series to simulate
- `Combine` | if TRUE, training and test data in the M-competition data are combined and generate a time series corresponds to the full length of the series. Otherwise, it generate a time series based on the training period of the series.
- `M` | if TRUE, `y` is considered to be a Mcomp data object
- `Future` | if `future=TRUE`, the simulated observations are conditional on the historical observations. In other words, they are possible future sample paths of the time series. But if `future=FALSE`, the historical data are ignored, and the simulations are possible realizations of the time series model that are not connected to the original data.
- `Length` | length of the simulated time series. If `future=FALSE`, the `Length` argument should be NA.
- `extralength` | extra length need to be added for simulated time series

**Value**

A list of time series.

**Author(s)**

Thiyanga Talagala
Simulate time series based on multiple seasonal decomposition

**Description**

simulate multiple time series based on a given series using multiple seasonal decomposition

**Usage**

```r
sim_mstlbased(
  y,
  Nsim,
  Combine = TRUE,
  M = TRUE,
  Future = FALSE,
  Length = NA,
  extralength = NA,
  mtd = "ets"
)
```

**Arguments**

- `y`: a time series or M-competition data time series (Mcomp object)
- `Nsim`: number of time series to simulate
- `Combine`: if TRUE, training and test data in the M-competition data are combined and generate a time series corresponds to the full length of the series. Otherwise, it generates a time series based on the training period of the series.
- `M`: if TRUE, `y` is considered to be a Mcomp data object
- `Future`: if future=TRUE, the simulated observations are conditional on the historical observations. In other words, they are possible future sample paths of the time series. But if future=FALSE, the historical data are ignored, and the simulations are possible realizations of the time series model that are not connected to the original data.
- `Length`: length of the simulated time series. If future = FALSE, the Length argument should be NA.
- `extralength`: extra length need to be added for simulated time series
- `mtd`: method to use for forecasting seasonally adjusted time series

**Value**

A list of time series.

**Author(s)**

Thiyanga Talagala
split_names

**Description**

Split the names of ARIMA and ETS models to model name, different number of parameters in each case.

**Usage**

```r
split_names(models)
```

**Arguments**

- `models` vector of model names

**Value**

A dataframe where columns give the description of model components.

---

**stlar**

*STL-AR method*

**Description**

STL decomposition method applied to the time series, then an AR model is used to forecast seasonally adjusted data, while the seasonal naive method is used to forecast the seasonal component.

**Usage**

```r
stlar(y, h = 10, s.window = 11, robust = FALSE)
```

**Arguments**

- `y` a univariate time series
- `h` forecast horizon
- `s.window` Either the character string “periodic” or the span (in lags) of the loess window for seasonal extraction
- `robust` logical indicating if robust fitting be used in the loess procedure

**Value**

Return object of class `forecast`.

**Author(s)**

Thiyanga Talagala
unitroot

Unit root test statistics

**Description**
Computes the test statistics based on unit root tests Phillips–Perron test and KPSS test

**Usage**
```
unitroot(y)
```

**Arguments**
- `y` a univariate time series

**Value**
A vector of 3 values: test statistic based on PP-test and KPSS-test

**Author(s)**
- Thiyanga Talagala
Index

accuracy_arima, 3
accuracy_ets, 3
accuracy_mstl, 4
accuracy_nn, 4
accuracy_rw, 5
accuracy_rwd, 5
accuracy_snaive, 6
accuracy_stlar, 6
accuracy_tbats, 7
accuracy_theta, 7
accuracy_wn, 8
acf5, 8
acf_seasonalDiff, 9

build_rf, 9

cal_features, 10
cal_m4measures, 11
cal_MASE, 12
cal_medianscaled, 12
cal_sMAPE, 13
cal_WA, 13
classify_labels, 14
classlabel, 14
combination_forecast_inside, 15
convert_msts, 15
e_acf1, 16

fcast_accuracy, 16
fforms_combinationforecast, 17
fforms_ensemble, 18

holtWinter_parameters, 19

multicore, 18
multisession, 18

prepare_trainingset, 19

rf_forecast, 20

sim_arimabased, 21
sim_etsbased, 22
sim_mstlbased, 23
split_names, 24
stlar, 24

unitroot, 25