

Package ‘caretForecast’

July 22, 2025

Title Conformal Time Series Forecasting Using State of Art Machine Learning Algorithms

Version 0.1.1

Description Conformal time series forecasting using the caret infrastructure.
It provides access to state-of-the-art machine learning models for forecasting applications. The hyperparameter of each model is selected based on time series cross-validation, and forecasting is done recursively.

License GPL (>= 3)

URL <https://github.com/Akai01/caretForecast>

BugReports <https://github.com/Akai01/caretForecast/issues>

Depends R (>= 3.6)

Imports forecast (>= 8.15), caret (>= 6.0.88), magrittr (>= 2.0.1),
methods (>= 4.1.1), dplyr (>= 1.0.9), generics (>= 0.1.3)

Suggests Cubist (>= 0.3.0), knitr (>= 1.29), testthat (>= 2.3.2)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

NeedsCompilation no

Author Resul Akay [aut, cre]

Maintainer Resul Akay <resulakay1@gmail.com>

Repository CRAN

Date/Publication 2022-10-24 07:05:25 UTC

Contents

ARml	2
conformalRegressor	4
forecast.ARml	5
get_var_imp	6
predict.conformalRegressor	7

retail	8
retail_wide	9
split_ts	9
suggested_methods	10

Index	11
--------------	-----------

ARml	<i>Autoregressive forecasting using various Machine Learning models.</i>
------	--

Description

Autoregressive forecasting using various Machine Learning models.

Usage

```
ARml(  
  y,  
  max_lag = 5,  
  xreg = NULL,  
  caret_method = "cubist",  
  metric = "RMSE",  
  pre_process = NULL,  
  cv = TRUE,  
  cv_horizon = 4,  
  initial_window = NULL,  
  fixed_window = FALSE,  
  verbose = TRUE,  
  seasonal = TRUE,  
  K = frequency(y)/2,  
  tune_grid = NULL,  
  lambda = NULL,  
  BoxCox_method = c("guerrero", "loglik"),  
  BoxCox_lower = -1,  
  BoxCox_upper = 2,  
  BoxCox_biasadj = FALSE,  
  BoxCox_fvar = NULL,  
  allow_parallel = FALSE,  
  ...  
)
```

Arguments

- | | |
|---------|--|
| y | A univariate time series object. |
| max_lag | Maximum value of lag. |
| xreg | Optional. A numerical vector or matrix of external regressors, which must have the same number of rows as y. (It should not be a data frame.). |

caret_method	A string specifying which classification or regression model to use. Possible values are found using <code>names(getModelInfo())</code> . A list of functions can also be passed for a custom model function. See http://topepo.github.io/caret/ for details.
metric	A string that specifies what summary metric will be used to select the optimal model. See <code>?caret::train</code> .
pre_process	A string vector that defines a pre-processing of the predictor data. Current possibilities are "BoxCox", "YeoJohnson", "expoTrans", "center", "scale", "range", "knnImpute", "bagImpute", "medianImpute", "pca", "ica" and "spatialSign". The default is no pre-processing. See <code>preProcess</code> and <code>trainControl</code> on the procedures and how to adjust them. Pre-processing code is only designed to work when <code>x</code> is a simple matrix or data frame.
cv	Logical, if <code>cv = TRUE</code> model selection will be done via cross-validation. If <code>cv = FALSE</code> user need to provide a specific model via <code>tune_grid</code> argument.
cv_horizon	The number of consecutive values in test set sample.
initial_window	The initial number of consecutive values in each training set sample.
fixed_window	Logical, if <code>FALSE</code> , all training samples start at 1.
verbose	A logical for printing a training log.
seasonal	Boolean. If <code>seasonal = TRUE</code> the fourier terms will be used for modeling seasonality.
K	Maximum order(s) of Fourier terms
tune_grid	A data frame with possible tuning values. The columns are named the same as the tuning parameters. Use <code>getModelInfo</code> to get a list of tuning parameters for each model or see http://topepo.github.io/caret/available-models.html . (NOTE: If given, this argument must be named.)
lambda	BoxCox transformation parameter. If <code>lambda = NULL</code> If <code>lambda = "auto"</code> , then the transformation parameter <code>lambda</code> is chosen using <code>BoxCox.lambda</code> .
BoxCox_method	<code>BoxCox.lambda</code> argument. Choose method to be used in calculating <code>lambda</code> .
BoxCox_lower	<code>BoxCox.lambda</code> argument. Lower limit for possible <code>lambda</code> values.
BoxCox_upper	<code>BoxCox.lambda</code> argument. Upper limit for possible <code>lambda</code> values.
BoxCox_biasadj	<code>InvBoxCox</code> argument. Use adjusted back-transformed mean for Box-Cox transformations. If transformed data is used to produce forecasts and fitted values, a regular back transformation will result in median forecasts. If <code>biasadj</code> is <code>TRUE</code> , an adjustment will be made to produce mean forecasts and fitted values.
BoxCox_fvar	<code>InvBoxCox</code> argument. Optional parameter required if <code>biasadj=TRUE</code> . Can either be the forecast variance, or a list containing the interval level, and the corresponding upper and lower intervals.
allow_parallel	If a parallel backend is loaded and available, should the function use it?
...	Ignored.

Value

A list class of forecast containing the following elements

- `x` : The input time series
- `method` : The name of the forecasting method as a character string
- `mean` : Point forecasts as a time series
- `lower` : Lower limits for prediction intervals
- `upper` : Upper limits for prediction intervals
- `level` : The confidence values associated with the prediction intervals
- `model` : A list containing information about the fitted model
- `newx` : A matrix containing regressors

Author(s)

Resul Akay

Examples

```
library(caretForecast)

train_data <- window(AirPassengers, end = c(1959, 12))

test <- window(AirPassengers, start = c(1960, 1))

ARml(train_data, caret_method = "lm", max_lag = 12) -> fit

forecast(fit, h = length(test)) -> fc

autoplot(fc) + autolayer(test)

accuracy(fc, test)
```

<code>conformalRegressor</code>	<i>Fit a conformal regressor.</i>
---------------------------------	-----------------------------------

Description

Fit a conformal regressor.

Usage

```
conformalRegressor(residuals, sigmas = NULL)
```

Arguments

residuals	Model residuals.
sigmas	A vector of difficulty estimates

Value

A conformalRegressor object

Author(s)

Resul Akay

References

Boström, H., 2022. crepes: a Python Package for Generating Conformal Regressors and Predictive Systems. In Conformal and Probabilistic Prediction and Applications. PMLR, 179. https://copa-conference.com/papers/COPA2022_paper_11.pdf

forecast.ARml	<i>Forecasting using ARml model</i>
---------------	-------------------------------------

Description

Forecasting using ARml model

Usage

```
## S3 method for class 'ARml'
forecast(object, h = frequency(object$y), xreg = NULL, level = c(80, 95), ...)
```

Arguments

object	An object of class "ARml", the result of a call to ARml.
h	forecast horizon
xreg	Optionally, a numerical vector or matrix of future external regressors
level	Confidence level for prediction intervals.
...	Ignored

Value

A list class of forecast containing the following elemets

- x : The input time series
- method : The name of the forecasting method as a character string
- mean : Point forecasts as a time series

- lower : Lower limits for prediction intervals
- upper : Upper limits for prediction intervals
- level : The confidence values associated with the prediction intervals
- model : A list containing information about the fitted model
- newxreg : A matrix containing regressors

Author(s)

Resul Akay

Examples

```
library(caretForecast)

train_data <- window(AirPassengers, end = c(1959, 12))

test <- window(AirPassengers, start = c(1960, 1))

ARml(train_data, caret_method = "lm", max_lag = 12) -> fit

forecast(fit, h = length(test), level = c(80,95)) -> fc

autoplot(fc)+ autolayer(test)

accuracy(fc, test)
```

get_var_imp	<i>Variable importance for forecasting model.</i>
-------------	---

Description

Variable importance for forecasting model.

Usage

```
get_var_imp(object, plot = TRUE)
```

Arguments

object	A list class of ARml or forecast object derived from ARml
plot	Boolean, if TRUE, variable importance will be plotted.

Value

A list class of "varImp.train". See [varImp](#) or a "trellis" plot.

Author(s)

Resul Akay

Examples

```

train <- window(AirPassengers, end = c(1959, 12))

test <- window(AirPassengers, start = c(1960, 1))

ARml(train, caret_method = "lm", max_lag = 12, trend_method = "none",
pre_process = "center") -> fit

forecast(fit, h = length(test), level = c(80,95)) -> fc

autoplot(fc)+ autolayer(test)

accuracy(fc, test)

get_var_imp(fc, plot = TRUE)

```

predict.conformalRegressor

Predict a conformalRegressor

Description

Predict a conformalRegressor

Usage

```

## S3 method for class 'conformalRegressor'
predict(
  object,
  y_hat = NULL,
  sigmas = NULL,
  confidence = 0.95,
  y_min = -Inf,
  y_max = Inf,
  ...
)

```

Arguments

object	A conformalRegressor object
y_hat	Predicted values
sigmas	Difficulty estimates

confidence	Confidence level
y_min	The minimum value to include in prediction intervals
y_max	The maximum value to include in prediction intervals
...	Ignored

Value

Prediction intervals

Author(s)

Resul Akay

retail	<i>Grouped sales data from an Australian Retailer</i>
--------	---

Description

A dataset containing 42 products' sales

Usage

retail

Format

A data class of "tbl_df", "tbl", "data.frame" with 13986 rows and 3 columns:

date date

item products

value sales

Source

<https://robjhyndman.com/data/ausretail.csv>

retail_wide	<i>Sales data from an Australian Retailer in time series format</i>
-------------	---

Description

A dataset containing 42 products' sales

Usage

```
retail_wide
```

Format

An object of class mts (inherits from ts, matrix) with 333 rows and 43 columns.
This data set is the wide format of [retail](#) data.

Source

<https://robjhyndman.com/data/ausretail.csv>

split_ts	<i>Split a time series into training and testing sets</i>
----------	---

Description

Split a time series into training and testing sets

Usage

```
split_ts(y, test_size = 10)
```

Arguments

y	A univariate time series
test_size	The number of observations to keep in the test set

Value

A list with train and test elements

Author(s)

Resul Akay

Examples

```
dlist <- split_ts(retail_wide[,1], test_size = 12)
```

suggested_methods	<i>Suggested methods for ARml</i>
-------------------	-----------------------------------

Description

Suggested methods for ARml

Usage

```
suggested_methods()
```

Value

A character vector of Suggested methods

Author(s)

Resul Akay

Examples

```
suggested_methods()
```

Index

* **datasets**

retail, [8](#)

retail_wide, [9](#)

ARml, [2](#)

BoxCox.lambda, [3](#)

conformalRegressor, [4](#)

forecast.ARml, [5](#)

get_var_imp, [6](#)

InvBoxCox, [3](#)

predict.conformalRegressor, [7](#)

retail, [8](#), [9](#)

retail_wide, [9](#)

split_ts, [9](#)

suggested_methods, [10](#)

varImp, [6](#)