Package ‘caretForecast’

May 3, 2022

Title Time Series Forecasting Using Caret Infrastructure
Version 0.0.3
Description Recursive time series forecast using Caret infrastructure.
    The models are 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 (>= 3.2.0)
Imports forecast (>= 8.15), caret (>= 6.0.88), magrittr (>= 2.0.1),
    methods (>= 4.1.1)
Suggests Cubist (>= 0.3.0), knitr (>= 1.29), testthat (>= 2.3.2)
Encoding UTF-8
LazyData true
RoxygenNote 7.1.2
NeedsCompilation no
Author Resul Akay [aut, cre]
Maintainer Resul Akay <resulakay1@gmail.com>
Repository CRAN
Date/Publication 2022-05-02 22:22:03 UTC

R topics documented:

ARml ................................................................. 2
forecast ......................................................... 4
forecast.ARMl .................................................... 6
get_var_imp ...................................................... 7
retail ............................................................. 8
retail_wide ...................................................... 9
split_ts ........................................................ 9
suggested_methods ......................................... 10
ARml

Autoregressive forecasting using various Machine Learning models.

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 = length(y) - max_lag - cv_horizon * 2,
  fixed_window = FALSE,
  verbose = TRUE,
  seasonal = TRUE,
  K = frequency(y)/2,
  tune_grid = NULL,
  lambda = "auto",
  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 names(getModelInfo()). A list of functions can also be
passed for a custom model function. See http://topepo.github.io/caret/
for details.
ARml

metric A string that specifies what summary metric will be used to select the optimal model. See ?caret::train.

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 preProcess and trainControl on the procedures and how to adjust them. Pre-processing code is only designed to work when x is a simple matrix or data frame.

cv Logical, if cv = TRUE model selection will be done via cross-validation. If cv = FALSE user need to provide a specific model via tune_grid 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 FALSE, all training samples start at 1.

verbose A logical for printing a training log.

seasonal Boolean. If seasonal = TRUE 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 getModelInfo 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 lambda = NULL If lambda = "auto", then the transformation parameter lambda is chosen using BoxCox.lambda.

BoxCox_method BoxCox.lambda argument. Choose method to be used in calculating lambda.

BoxCox_lower BoxCox.lambda argument. Lower limit for possible lambda values.

BoxCox_upper BoxCox.lambda argument. Upper limit for possible lambda values.

BoxCox_biasadj InvBoxCox 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 biasadj is TRUE, an adjustment will be made to produce mean forecasts and fitted values.

BoxCox_fvar InvBoxCox argument. Optional parameter required if biasadj=TRUE. 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?

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**

```r
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)
```

---

**forecast**  
*Forecasting an ARml object*

**Description**

Forecasting an ARml object

**Usage**

```r
forecast(
  object,
  h = frequency(object$y),
  xreg = NULL,
  level = c(80, 95),
  PI = FALSE,
  num_bs = 1000,
  ...
)
```
Arguments

object A list class of ARml
h forecast horizon
xreg Optionally, a numerical vector or matrix of future external regressors
level Confidence level for prediction intervals.
PI If TRUE, prediction intervals are produced, otherwise only point forecasts are calculated. If PI is FALSE, then level, fan, bootstrap and npaths are all ignored.
num.bs Number of bootstrapped versions to generate.
... Other arguments pased to forecast::forecast()

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
• 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), PI = TRUE) -> fc
autoplot(fc)+ autolayer(test)
accuracy(fc, test)
### Forecasting an ARml object

**Description**

Forecasting an ARml object

**Usage**

```r
# S3 method for class 'ARml'
forecast(
  object,
  h = frequency(object$y),
  xreg = NULL,
  level = c(80, 95),
  PI = FALSE,
  num_bs = 1000,
  ...
)
```

**Arguments**

- `object` A list class of ARml
- `h` forecast horizon
- `xreg` Optionally, a numerical vector or matrix of future external regressors
- `level` Confidence level for prediction intervals.
- `PI` If TRUE, prediction intervals are produced, otherwise only point forecasts are calculated. If PI is FALSE, then level, fan, bootstrap and npaths are all ignored.
- `num_bs` Number of bootstrapped versions to generate.
- `...` Other arguments passed to `forecast::forecast()`

**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
- `newxreg` A matrix containing regressors
get_var_imp

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), PI = TRUE) -> fc

autoplot(fc)+ autolayer(test)

accuracy(fc, test)

________________________________________________________________________

get_var_imp Variable importance for forecasting model.

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

```r
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), PI = TRUE) -> fc

autoplot(fc)+ autolayer(test)

accuracy(fc, test)

gget_var_imp(fc, plot = TRUE)
```

---

**retail**

*Grouped sales data from an Australian Retailer*

---

**Description**

A dataset containing 42 products' sales

**Usage**

```r
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

Sales data from an Australian Retailer in time series format

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

Split a time series into training and testing sets

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  Suggested methods for ARml

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
forecast, 4
forecast.ARMl, 6
get_var_imp, 7
InvBoxCox, 3
retail, 8, 9
retail_wide, 9
split_ts, 9
suggested_methods, 10
varImp, 7