Package: autoTS (via r-universe)

September 11, 2024

Type Package

Title Automatic Model Selection and Prediction for Univariate Time Series

Version 0.9.11

Author Vivien Roussez

Maintainer Vivien Roussez <vivien.roussez@gmail.com>

Description Offers a set of functions to easily make predictions for univariate time series. 'autoTS' is a wrapper of existing functions of the 'forecast' and 'prophet' packages, harmonising their outputs in tidy dataframes and using default values for each. The core function getBestModel() allows the user to effortlessly benchmark seven algorithms along with a bagged estimator to identify which one performs the best for a given time series.

License GPL-3

Encoding UTF-8

LazyData true

Imports rlang, prophet, dplyr, magrittr, lubridate, tidyr, forecast, ggplot2, RcppRoll,shiny, shinycssloaders, plotly

BugReports https://github.com/vivienroussez/autots/issues

URL https://github.com/vivienroussez/autoTS

Suggests knitr,rmarkdown,stringr

VignetteBuilder knitr

RoxygenNote 7.1.0

Repository https://vivienroussez.r-universe.dev

RemoteUrl https://github.com/vivienroussez/autots

RemoteRef HEAD

RemoteSha b457c022154753b1b8eeb531c2d8db46ae06cd1c

Contents

`complete.ts` *Creates additional dates and values when NA where removed and the TS is not complete*

Description

Creates additional dates and values when NA where removed and the TS is not complete

Usage

```
complete.ts(dates, values, freq, complete = 0)
```

Arguments

`dates`	A vector of dates that can be parsed by lubridate
`values`	A vector of same size as `dates`
`freq`	A chacracter string that indicates the frequency of the time series ("week", "month", "quarter", "day").
`complete`	A numerical value (or NA) to fill the missing data points

Value

A dataframe with 2 columns : date and val, with additional rows

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"month")
values <- rnorm(length(dates))
complete.ts(dates,values,"month",complete = 0)
```

`getBestModel` *Determine best algorithm*

Description

Implement selected algorithms, train them without the last n observed data points (or n_test number of points), and compares the results to reality to determine the best algorithm

Usage

```
getBestModel(
  dates,
  values,
  freq,
  complete = 0,
  n_test = NA,
  graph = TRUE,
 algos = list("my.prophet", "my.ets", "my.sarima", "my.tbats", "my.bats", "my.stlm",
    "my.shortterm"),
  bagged = "auto",
  metric.error = my.rmse
)
```

Arguments

`dates`	A vector of dates that can be parsed by lubridate
`values`	A vector of same size as `dates`
`freq`	A chacracter string that indicates the frequency of the time series ("week", "month", "quarter", "day").
`complete`	A numerical value (or NA) to fill the missing data points
`n_test`	number of data points to keep aside for the test (default : one year)
`graph`	A boolean, if TRUE, comparison of algorithms is plotted
`algos`	A list containing the algorithms (strings, with prefix "my.") to be tested
`bagged`	A string. "auto" will use all available algoriths, skipping algos parameter. Else, specified algos of the ‘algo‘ parameter will be used
`metric.error`	a function to compute the error the each models. available functions : my.rmse and my.mae

Value

A list contraining a character string with the name of the best method, a gg object with the comparison between algorithms and a dataframe with predictions of all tried algorithms, a dtaframe containing the errors of each algorithms, the preparedTS object and the list of algorithms tested

Examples

```
library(autoTS)
dates <- seq(lubridate::as_date("2005-01-01"),lubridate::as_date("2010-12-31"),"quarter")
values <- 10+ 1:length(dates)/10 + rnorm(length(dates),mean = 0,sd = 10)

which.model <- getBestModel(dates,values,freq = "quarter",n_test = 4)

### Custom set of algorithm (including for bagged estimator)
which.model <- getBestModel(dates,values,freq = "quarter",n_test = 4,
                            algos = list("my.prophet","my.ets"),bagged = "custom")
### Use MAE instead of RMSE

which.model <- getBestModel(dates,values,freq = "quarter",n_test = 3,
                            algos = list("my.prophet","my.ets"),
                            bagged = "custom",metric.error = my.mae)
```

---

getFrequency *Determines the decimal frequency of a time series from a character string*

Description

Determines the decimal frequency of a time series from a character string

Usage

```
getFrequency(freq.alpha)
```

Arguments

freq.alpha	A character string that indicates the frequency of the time series ("week", "month", "quarter", "day").

Value

The decimal version of the frequency (useful for the forecast package functions).

Examples

```
getFrequency("week")
```

my.bats *Fit BATS algorithm and make the prediction*

Description

Fit BATS algorithm and make the prediction

Usage

```
my.bats(prepedTS, n_pred)
```

Arguments

prepedTS	A list created by the prepare.ts() function
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.bats(my.ts,n_pred=4)
```

my.ets *Fit ETS algorithm and make the prediction*

Description

Fit ETS algorithm and make the prediction

Usage

```
my.ets(prepedTS, n_pred)
```

Arguments

prepedTS	A list created by the prepare.ts() function
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
 library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.ets(my.ts,n_pred=4)
```

---

my.mae *Custom (internal) function for MAE*

Description

Custom (internal) function for MAE

Usage

```
my.mae(true, predicted)
```

Arguments

true	num vector of actual values
predicted	num vector of predicted values

Value

Num value with MAE

---

my.predictions *Make predictions with selected algorithms*

Description

Fit selected algorithms, make the predictions and combine the results along with observed data in one final dataframe.

Usage

```
my.predictions(
  bestmod = NULL,
  prepedTS = NULL,
 algos = list("my.prophet", "my.ets", "my.sarima", "my.tbats", "my.bats", "my.stlm",
    "my.shortterm"),
  n_pred = NA
)
```

Arguments

bestmod	A list produced by the getBestModel() function (optional if prepredTS is provided)
prepedTS	A list created by the prepare.ts() function (optional if bestmod provided)
algos	A list containing the algorithms to be implemented. If bestmod is supplied, this value is ignored, and taken from the best model object Using this option will overwrite the provided list of algorithms to implement them all
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe containing : date, actual observed values, one column per used algorithm, and a column indicating the type of measure (mean prediction, upper or lower bound of CI)

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(lubridate::as_date("2000-01-01"),lubridate::as_date("2010-12-31"),"quarter")
values <- 10+ 1:length(dates)/10 + rnorm(length(dates),mean = 0,sd = 10)
### Stand alone usage
prepare.ts(dates,values,"quarter") %>%
  my.predictions(prepedTS = .,algos = list("my.prophet","my.ets"))
### Standard input with bestmodel

getBestModel(dates,values,freq = "quarter",n_test = 6) %>%
  my.predictions()
```

---

my.prophet *Fit prophet algorithm and make the prediction*

Description

Fit prophet algorithm and make the prediction

Usage

```
my.prophet(prepedTS, n_pred)
```

Arguments

prepedTS	A list created by the prepare.ts() function
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe for "next year" with 4 columns : date, average prediction, upper and lower 95

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.prophet(my.ts,n_pred=4)
```

---

my.rmse *Custom (internal) function for RMSE*

Description

Custom (internal) function for RMSE

Usage

```
my.rmse(true, predicted)
```

Arguments

true	num vector of actual values
predicted	num vector of predicted values

Value

Num value with RMSE

`my.sarima` *Fit SARIMA algorithm and make the prediction*

Description

Fit SARIMA algorithm and make the prediction

Usage

```
my.sarima(prepedTS, n_pred)
```

Arguments

`prepedTS`	A list created by the `prepare.ts()` function
`n_pred`	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
 library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.sarima(my.ts,n_pred=4)
```

`my.shortterm` *Fit short term algorithm and make the prediction*

Description

Fit short term algorithm and make the prediction

Usage

```
my.shortterm(prepedTS, n_pred, smooth_window = 2)
```

Arguments

prepedTS	A list created by the prepare.ts() function
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series). Note that this algorithm cannot predict further than one year
smooth_window	Int specifying the number of periods to consider for computing the evolution rate that will be applied for the forecast

Details

this algorithm uses data of the last year and makes the prediction taking into account the seasonality and the evolution of the previous periods' evolution

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.shortterm(my.ts,n_pred=4)
```

---

my.stlm *Fit STLM algorithm and make the prediction*

Description

Fit STLM algorithm and make the prediction

Usage

```
my.stlm(prepedTS, n_pred)
```

Arguments

prepedTS	A list created by the prepare.ts() function
n_pred	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.stlm(my.ts,n_pred=4)
```

`my.tbats` *Fit TBATS algorithm and make the prediction*

Description

Fit TBATS algorithm and make the prediction

Usage

```
my.tbats(prepedTS, n_pred)
```

Arguments

`prepedTS`	A list created by the `prepare.ts()` function
`n_pred`	Int number of periods to forecast forward (eg n_pred = 12 will lead to one year of prediction for monthly time series)

Value

A dataframe with 4 columns : date, average prediction, upper and lower 95

Examples

```
library(lubridate)
library(dplyr)
dates <- seq(as_date("2000-01-01"),as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"quarter",complete = 0)
my.tbats(my.ts,n_pred=4)
```

`prepare.ts` *Format 2 vectors in a proper object usable by all algorithms*

Description

Format 2 vectors in a proper object usable by all algorithms

Usage

```
prepare.ts(dates, values, freq, complete = 0)
```

Arguments

`dates`	A vector of dates that can be parsed by lubridate
`values`	A vector of same size as `dates`
`freq`	A chacracter string that indicates the frequency of the time series ("week", "month", "quarter", "day").
`complete`	A numerical value (or NA) to fill the missing data points

Details

Creates a list with the time series in a dataframe and a ts object, and the frequency stored in decimal and litteral values. The result is meant to be put in the prophet or forecast functions

Value

A list containing : a dataframe, a ts vector for the time series, and 2 scalars for its frequency

Examples

```
library(lubridate)
library(dplyr)
library(ggplot2)
dates <- seq(lubridate::as_date("2000-01-01"),lubridate::as_date("2010-12-31"),"quarter")
values <- rnorm(length(dates))
my.ts <- prepare.ts(dates,values,"month",complete = 0)
plot(my.ts$obj.ts)
ggplot(my.ts$obj.df,aes(dates,val)) + geom_line()
```

`runUserInterface` *Demo graphical user interface*

Description

A shiny application that allows the user to load a properly formated CSV file, benchmark the algorithms, make a prediction and download the results. Requires additional packages shiny, shinycssloaders, tidyr and plotly to be installed

Usage

```
runUserInterface()
```

Examples

```
autoTS::runUserInterface()
```

Index