Package 'fable.ata'

June 19, 2023

Type Package

Title 'ATAforecasting' Modelling Interface for 'fable' Framework

Version 0.0.6

Date 2023-06-11

Maintainer Ali Sabri Taylan <alisabritaylan@gmail.com>

Description

Allows ATA (Automatic Time series analysis using the Ata method) models from the 'ATAforecasting' package to be used in a tidy workflow with the modeling interface of 'fabletools'. This extends 'ATAforecasting' to provide enhanced model specification and management, performance evaluation methods, and model combination tools. The Ata method (Yapar et al. (2019) <doi:10.15672/hujms.461032>), an alternative to exponential smoothing (described in Yapar (2016) <doi:10.15672/HJMS.201614320580>, Yapar et al. (2017) <doi:10.15672/HJMS.2017.493>), is a new univariate time series forecasting method which provides innovative solutions to issues faced during the initialization and optimization stages of existing forecasting methods. Forecasting performance of the Ata method is superior to existing methods both in terms of easy implementation and accurate forecasting. It can be applied to non-seasonal or seasonal time series which can be decomposed into four components (remainder, level, trend and seasonal).

Imports stats, rlang, tibble, dplyr, tsibble, distributional, tsbox, lubridate

Depends fabletools (>= 0.3.3), ATAforecasting (>= 0.0.60)

License GPL (>= 3)

URL https://alsabtay.github.io/fable.ata/

BugReports https://github.com/alsabtay/fable.ata/issues

Encoding UTF-8

RoxygenNote 7.2.3

ByteCompile true

NeedsCompilation no

Author Ali Sabri Taylan [aut, cre, cph]
(<https://orcid.org/0000-0001-9514-934X>),
Hanife Taylan Selamlar [aut, cph]
(<https://orcid.org/0000-0002-4091-884X>),
Guckan Yapar [aut, ths, cph] (<https://orcid.org/0000-0002-0971-6676>)

Repository CRAN

Date/Publication 2023-06-19 19:00:02 UTC

R topics documented:

`AutoATA` *ATAforecasting: Automatic Time Series Analysis and Forecasting using Ata Method with Box-Cox Power Transformations Family and Seasonal Decomposition Techniques*

Description

Returns ATA(p,q,phi)(E,T,S) applied to time series data. The Ata method based on the modified simple exponential smoothing as described in Yapar, G. (2016) <doi:10.15672/HJMS.201614320580> , Yapar G., Capar, S., Selamlar, H. T., Yavuz, I. (2017) <doi:10.15672/HJMS.2017.493> and Yapar G., Selamlar, H. T., Capar, S., Yavuz, I. (2019) <doi:10.15672/hujms.461032> is a new univariate time series forecasting method which provides innovative solutions to issues faced during the initialization and optimization stages of existing methods. Forecasting performance of the Ata method is superior to existing methods both in terms of easy implementation and accurate forecasting. It can be applied to non-seasonal or seasonal time series which can be decomposed into four components (remainder, level, trend and seasonal). This methodology performed well on the M3 and M4-competition data.

Usage

```
AutoATA(formula, ...)
```

Arguments

`formula`	Model specification (see "Specials" section).
`...`	Other arguments

Value

A model specification.

Specials

The _specials_ define the methods and parameters for the components (level, trend, seasonality, accuracy, transform, holdout) of an ATA method.

There are a couple of limitations to note about ATA method:

- It supports only additive error term. - It does not support exogenous regressors. - It does not support missing values. You can complete missing values in the data with imputed values (e.g. with [tsibble::fill_gaps()], [tidyr::fill()], or by fitting a different model type and then calling [fabletools::interpolate()]) before fitting the model.

level: The ‘level‘ special is used to specify the form of the level term.

```
level(parP = NULL, level_fixed = TRUE, initial_level = "none")
```

‘parP‘	The value of the smoothing parameter for the level. If ‘p = 0‘, the level will not change over time. Conversely,
‘level_fixed‘	If TRUE, "pStarQ" –> First, fits ATA(p,0) where p = p* is optimized for q=0. Then, fits ATA(p*,q) where q is
‘initial_level‘	If NULL, "none" is default. If "none", ATA Method calculates the pth observation in `data` for level. If "mean"

trend: The ‘trend‘ special is used to specify the form of the trend term and associated parameters.

```
trend(type = "A", parQ = NULL, initial_trend = "none", opt_trend = "none",
      parPHI = NULL, parPHI_range = c(0.8, 1.0), parPHI_increment = 0.01,
      uroot_test = "adf", uroot_alpha = 0.05, uroot_type = "level")
```

‘type‘	The form of the trend term: either none ("N"), additive ("A"), multiplicative ("M") or damped variants (
‘parQ‘	The value of the smoothing parameter for the slope. If ‘q = 0‘, the slope will not change over time. Con
‘parPHI‘	The value of the dampening parameter for the slope. If ‘phi = 0‘, the slope will be dampened immediat
‘parPHI_range‘	If ‘phi=NULL‘, ‘phi_range‘ provides bounds for the optimised value of ‘phi‘.
‘parPHI_increment‘	If ‘phi=NULL‘, ‘parPHI_increment‘ provides increment step for searching ‘phi‘. If NULL, ‘parPHI_in
‘initial_trend‘	If NULL, "none" is default. If "none", ATA Method calculates the qth observation in `data` for trend. If
‘trend_opt‘	Default is ‘none‘. If ‘fixed‘ is set, "pBullet" –> Fits ATA(p,1) where p = p* is optimized for q = 1. If ‘s
‘uroot_test‘	Type of unit root test before all type seasonality test. Possible values are "adf", "pp" and "kpss".
‘uroot_alpha‘	Significant level of the unit root test, possible values range from 0.01 to 0.1.
‘uroot_type‘	Specification of the deterministic component in the regression for unit root test. Possible values are "lev
‘uroot_maxd‘	Maximum number of non-seasonal differences allowed.

season: The ‘season‘ special is used to specify the form of the seasonal term and associated parameters. To specify a nonseasonal model you would include ‘season(method = "N")‘.

```
season(type = "A", test = TRUE, period = NULL, method = "decomp",
     suroot_test = "correlogram", suroot_tcrit = 1.28, suroot_uroot = TRUE, suroot_m = NULL)
```

‘type‘	The form of the seasonal term: either none ("N"), additive ("A") or multiplicative ("M").
‘test‘	Testing for stationary and seasonality. If TRUE, the method firstly uses `test="adf"`, Augmented Dickey-Fu
‘period‘	The periodic nature of the seasonality. This can be a number indicating the number of observations in each s
‘method‘	A string identifying method for seasonal decomposition. If NULL, "decomp" method is default. Possible va
‘suroot_test‘	Type of seasonal unit root test to use. Possible values are "correlogram", "seas", "hegy", "ch" and "ocsb".
‘suroot_tcrit‘	t-value for autocorrelogram.
‘suroot_alpha‘	Significant level of the seasonal unit root test, possible values range from 0.01 to 0.1.
‘suroot_uroot‘	If TRUE, unit root test for stationary before seasonal unit root test is allowed.
‘suroot_m‘	Deprecated. Length of seasonal period: frequency of data for nsdiffs.
‘suroot_maxD‘	Maximum number of seasonal differences allowed.

accuracy: The ‘accuracy‘ special is used to the optimization criterion for selecting the best ATA Method forecasting.

```
accuracy(criteria = "sMAPE", nmse = 3, ic = "AIC")
```

‘criteria‘	Accuracy measure for optimization of the best ATA Method forecasting. IF NULL, ‘sMAPE‘ is default.
‘nmse‘	If ‘accuracy.type == "AMSE"‘, ‘nmse‘ provides the number of steps for average multistep MSE ‘(2<=nmse<=30)‘.
‘ic‘	The information criterion used in selecting the model.

transform: The ‘transform‘ special is used to provide the applicability of different types of transformation techniques for the data to which the ATA method will be applied.

```
transform(method="none", order = "none", lambda = NULL, shift = 0,
          bcMethod = "guerrero", bcLower = 0, bcUpper = 5)
```

‘method‘	Transformation method –> "Box_Cox", "Sqrt", "Reciprocal", "Log", "NegLog", "Modulus", "BickelDoksum", "
‘order‘	Default is "none. If "before", Box-Cox transformation family will be applied and then seasonal decomposition t
‘lambda‘	Box-Cox power transformation family parameter. If NULL, data transformed before model is estimated.
‘shift‘	Box-Cox power transformation family shifting parameter. If NULL, data transformed before model is estimatec
‘bcMethod‘	Choose method to be used in calculating lambda. "guerrero" is default. Other method is "loglik".
‘bcLower‘	Lower limit for possible lambda values. The lower value is limited by -5. Default value is 0.
‘bcUpper‘	Upper limit for possible lambda values. The upper value is limited by 5. Default value is 5.

holdout: The ‘holdout‘ special is used to improve the optimized parameter value obtained for the ATA Method forecasting.

```
holdout(holdout = FALSE, adjustment = TRUE, set_size = NULL, onestep = FALSE)
```

‘holdout‘	Default is FALSE. If TRUE, ATA Method uses the holdout forecasting for accuracy measure to select the best
‘adjustment‘	Default is TRUE. If TRUE, ‘parP‘ will be adjusted by length of training, validation sets and main data set whe
‘set_size‘	If ‘holdout‘ is TRUE, this parameter divides ‘data‘ into two parts: training set (in-sample) and validation set (h
‘onestep‘	Default is FALSE. if TRUE, the dynamic forecast strategy uses a one-step model multiple times ‘h‘ (forecast h

Examples

```
library(fable.ata)
as_tsibble(USAccDeaths) %>% model(ata = AutoATA(value ~ trend("A") + season("A")))
```

---

components.ATA — *Extract estimated states from an ATA model.*

Description

Extract estimated states from an ATA model.

Usage

```
## S3 method for class 'ATA'
components(object, ...)
```

Arguments

`object`	An estimated model.
`...`	Unused.

Value

A [fabletools::dable()] containing estimated states.

---

fitted.ATA — *Extract fitted values*

Description

Extracts the fitted values from an estimated ATA model.

Usage

```
## S3 method for class 'ATA'
fitted(object, ...)
```

Arguments

`object`	An estimated model.
`...`	Unused.

Value

A vector of fitted values.

`forecast.ATA` *Forecast a model from the fable ATA model*

Description

Forecast a model from the fable ATA model

Usage

```
## S3 method for class 'ATA'
forecast(
  object,
  new_data,
  h = NULL,
  ci_level = 95,
  negative_forecast = TRUE,
  onestep = FALSE,
  ...
)
```

Arguments

`object`	The time series model used to produce the forecasts
`new_data`	A ‘tsibble‘ containing future information used to forecast.
`h`	The forecast horison (can be used instead of ‘new_data‘ for regular time series with no exogenous regressors).
`ci_level`	Confidence Interval levels for forecasting. Default value is 95.
`negative_forecast`	Negative values are allowed for forecasting. Default value is TRUE. If FALSE, all negative values for forecasting are set to 0.
`onestep`	Default is FALSE. if TRUE, the dynamic forecast strategy uses a one-step model multiple times ‘h‘ forecast horizon) where the prediction for the prior time step is used as an input for making a prediction on the following time step.
`...`	Other arguments

Value

A vector of fitted residuals.

Examples

```
library(fable.ata)
as_tsibble(USAccDeaths) %>%
  model(ata = AutoATA(value ~ trend("A") + season("M"))) %>% forecast(h=24)
```

format.ATA *Format of ATA model*

Description

Format of ATA model

Usage

```
## S3 method for class 'ATA'
format(x, ...)
```

Arguments

`x`	An estimated model.
`...`	Unused.

Value

The forecasting model's name.

glance.ATA *Glance an ATA model*

Description

Glance an ATA model

Usage

```
## S3 method for class 'ATA'
glance(x, ...)
```

Arguments

`x`	An estimated model.
`...`	Unused.

Value

A one row tibble summarising the model's fit.

Examples

```
library(fable.ata)
as_tsibble(USAccDeaths) %>%
  model(ata = AutoATA(value ~ trend("A") + season("M"))) %>% glance()
```

model_sum.ATA *Summary of ATA model*

Description

Summary of ATA model

Usage

```
## S3 method for class 'ATA'
model_sum(x, ...)
```

Arguments

`x`	An estimated model.
`...`	Unused.

Value

The model's summary specs.

report.ATA *Specialized Screen Print Function of ATA model*

Description

Specialized Screen Print Function of ATA model

Usage

```
## S3 method for class 'ATA'
report(object, ...)
```

Arguments

`object`	An estimated model.
`...`	Unused.

Value

a summary for the results of the ATAforecasting

Examples

```
 library(fable.ata)
as_tsibble(USAccDeaths) %>% model(ata = AutoATA(value ~ trend("A") + season("M"))) %>% report()
```

`residuals.ATA` *Extract model residuals*

Description

Extracts the residuals from an estimated ATA model.

Usage

```
## S3 method for class 'ATA'
residuals(object, ...)
```

Arguments

`object`	An estimated model.
`...`	Unused.

Value

A vector of residuals.

`tidy.ATA` *Tidy a ATA model*

Description

Tidy a ATA model

Usage

```
## S3 method for class 'ATA'
tidy(x, ...)
```

Arguments

`x`	An estimated model.
`...`	Unused.

Value

The model's coefficients in a 'tibble'.

Examples

```
library(fable.ata)
as_tsibble(USAccDeaths) %>%
  model(ata = AutoATA(value ~ trend("A") + season("M"))) %>% tidy()
```

Index