

Package ‘fableCount’

April 5, 2024

Type Package

Title INGARCH and GLARMA Models for Count Time Series in Fable Framework

Description Provides a tidy R interface for count time series analysis. It includes implementation of the INGARCH (Integer Generalized Autoregressive Conditional Heteroskedasticity) model from the 'tscount' package and the GLARMA (Generalized Linear Autoregressive Moving Averages) model from the 'glarma' package. Additionally, it offers automated parameter selection algorithms based on the minimization of a penalized likelihood.

Version 0.1.0

Depends R (>= 4.0.0), fabletools (>= 0.3.0), tscount (>= 1.4.3), glarma (>= 1.6.0), fable (>= 0.3.4)

Imports stats, dplyr (>= 1.0.0), tsibble (>= 0.9.0), tibble, tidyr, distributional, lubridate, stringr, tsibbledata

Suggests Rcpp, utils, rlang, covr, feasts, forecast, knitr, rmarkdown, testthat

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

Author Gustavo Almeida [aut, cre] (<<https://orcid.org/0009-0005-7266-5866>>),
Marcel Vieira [aut] (<<https://orcid.org/0000-0002-0456-380X>>),
Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq [fnd],
JFSalvando Todos - Plataforma de Análises Estatísticas [fnd, cph]

Maintainer Gustavo Almeida <gustavoalmeidasilva@ice.ufjf.br>

Repository CRAN

Date/Publication 2024-04-05 09:13:03 UTC

R topics documented:

fitted.GLARMA 2

fitted.INGARCH	3
forecast.GLARMA	3
forecast.INGARCH	4
glance.GLARMA	5
glance.INGARCH	6
GLARMA	7
INGARCH	8
residuals.GLARMA	10
residuals.INGARCH	11
tidy.GLARMA	11
tidy.INGARCH	12
Index	13

fitted.GLARMA	<i>Extract fitted values from a fable model</i>
---------------	---

Description

Extracts the fitted values.

Usage

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

Arguments

object	A model for which forecasts are required.
...	Other arguments passed to methods

Value

A vector of fitted values.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_gla = GLARMA(Beer ~ pq(1,0))) |>
  dplyr::select(manual_gla) |>
  fitted()
```

fitted.INGARCH	<i>Extract fitted values from a fable model</i>
----------------	---

Description

Extracts the fitted values.

Usage

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

Arguments

object	A model for which forecasts are required.
...	Other arguments passed to methods

Value

A vector of fitted values.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1))) |>
  dplyr::select(manual_ing) |>
  fitted()
```

forecast.GLARMA	<i>Forecast a model from the fable package</i>
-----------------	--

Description

Produces forecasts from a trained model.

Usage

```
## S3 method for class 'GLARMA'
forecast(object, new_data, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	Tsibble, it has to contains the time points and exogenous regressors to produce forecasts for.
...	Other arguments passed to methods

Details

Predict future observations based on a fitted GLM-type model for time series of counts. Further informations about the forecast method can be obtained typing `?glarma::forecast`

Value

A list of forecasts.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_gla = GLARMA(Beer ~ pq(1,0))) |>
  dplyr::select(manual_gla) |>
  fabletools::forecast(h = 2)
```

forecast.INGARCH	<i>Forecast a model from the fable package</i>
------------------	--

Description

Produces forecasts from a trained model.

Usage

```
## S3 method for class 'INGARCH'
forecast(object, new_data, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	Tsibble, it has to contains the time points and exogenous regressors to produce forecasts for.
...	Other arguments passed to methods

Details

Predict future observations based on a fitted GLM-type model for time series of counts. For 1 step ahead, it returns parametric forecast, based on the 'distr' param especificed distribution, for multiples steps forecast, the distribution is not know analytically, so it uses a parametric bootstrap

Value

A list of forecasts.

Examples

```
# 1 step ahead parametric forecast
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1) + PQ(1,1))) |>
  dplyr::select(manual_ing) |>
  fabletools::forecast(h = 1)

# Multiples steap ahead parametric bootstrap forecast
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1) + PQ(1,1))) |>
  dplyr::select(manual_ing) |>
  fabletools::forecast(h = 4)
```

glance.GLARMA

*Glance a GLARMA model***Description**

Construct a single row summary of the GLARMA model.

Usage

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

Arguments

```
x          model or other R object to convert to single-row data frame
...        other arguments passed to methods
```

Format

A data frame with 1 row, with columns:

sigma2 The unbiased variance of residuals. Calculated as $\text{'sum(residuals^2) / (num_observations - num_pararameters + 1)'$

log_lik The log-likelihood

AIC Akaike information criterion

Value

A one row tibble summarising the model's fit.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_ing = GLARMA(Beer ~ pq(1,1))) |>
  dplyr::select(manual_ing) |>
  glance()
```

`glance.INGARCH`*Glance a INGARCH model*

Description

Construct a single row summary of the INGARCH model.

Usage

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

Arguments

`x` model or other R object to convert to single-row data frame
`...` other arguments passed to methods

Format

A data frame with 1 row, with columns:

sigma2 The unbiased variance of residuals. Calculated as $\text{'sum(residuals^2) / (num_observations - num_parameters + 1)'$

log_lik The log-likelihood

AIC Akaike information criterion

BIC Bayesian information criterion

Value

A one row tibble summarising the model's fit.

Examples

```
tsibbledata::aus_production |>  
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1))) |>  
  dplyr::select(manual_ing) |>  
  glance()
```

Description

Estimate Generalized Linear Autoregressive Moving Average model with Poisson or Negative Binomial distribution. Also is provide a automatic parameter algorithm selection for the Autorregressive and Moving Average params

Usage

```
GLARMA(
  formula,
  ic = c("aic", "bic"),
  distr = c("Poi", "NegBin"),
  method = c("FS", "NR"),
  residuals = c("Pearson", "Score"),
  trace = FALSE
)
```

Arguments

<code>formula</code>	Model specification (see "Specials" section).
<code>ic</code>	Character, can be 'AIC','BIC'. The information criterion used in selecting the model.
<code>distr</code>	Character, can be 'poisson' or 'nbinom'. The probabily distribution used for the generalized model
<code>method</code>	Character, can be 'FS' (Fisher scoring) or 'NR' (Newton-Raphson). The method of iteration to be used
<code>residuals</code>	Character, can be 'Pearson' or 'Score'. The type of residuals to be used
<code>trace</code>	Logical. If the automatic parameter algorithm is runnig, print the path to the best model estimation

Value

A model specification.

Specials

pq: pq defines the non-seasonal autoregressive and moving avarages terms, it can be define by the user, or if it's omitted, the automatic parameter selection algorithm is trigered The automatic parameter selection algorithm gonna fit the best model based on the information criterion

PQ: PQ defines the seasonal autoregressive and moving avarages terms, it can be define by the user, or if it's omitted, the automatic parameter selection algorithm is trigered (only for 'arma_to_GLARMA' algorithm) The automatic parameter selection algorithm gonna fit the best model based on the information criterion

xreg: Exogenous regressors can be included in an GLARMA model without explicitly using the `xreg()` special. Common exogenous regressor specials as specified in `['common_xregs']` can also be used. These regressors are handled using `[stats::model.frame()]`, and so interactions and other functionality behaves similarly to `[stats::lm()]`.

The inclusion of a constant in the model follows the similar rules to `['stats::lm()']`, where including `'1'` will add a constant and `'0'` or `'-1'` will remove the constant. If left out, the inclusion of a constant will be determined by minimising `'ic'`.

If a `xreg` is provided, the model forecast is not available

```
xreg(..., fixed = list())
```

`'...'` Bare expressions for the exogenous regressors (such as `'log(x)'`)

`'fixed'` A named list of fixed parameters for coefficients. The names identify the coefficient, and should match the name of the

Examples

```
# Manual GLARMA specification
tsibbledata::aus_production |>
  fabletools::model(manual_gla = GLARMA(Beer ~ pq(1,0)))

# Automatic GLARMA specification
tsibbledata::aus_production |>
  fabletools::model(auto_gla = GLARMA(Beer, ic = 'aic'))
```

INGARCH

Estimate a INGARCH model

Description

Estimate Integer-valued Generalized Autoregressive Conditional Heteroscedasticity model with Poisson or Negative Binomial distribution. Also is provide a automatic parameter algorithm selection for the Autoregressive and Moving Average params

Usage

```
INGARCH(
  formula,
  ic = c("aic", "bic", "qic"),
  link = c("identity", "log"),
  distr = c("poisson", "nbinom"),
  algorithm = c("naive_search", "arma_to_ingarch"),
  trace = FALSE
)
```


Arguments

formula	Model specification (see "Specials" section).
ic	Character, can be 'aic' 'bic' or 'qic'. The information criterion used in selecting the model.
link	Character, can be 'identity' or 'log' The link function used for the generalized model
distr	Character, can be 'poisson' or 'nbinom'. The probability distribution used for the generalized model
algorithm	Character, specifies the automatic parameter selection algorithm. Can be 'naive_search' or 'arma_to_ingarch'. If 'naive_search' is selected, a search in a 4x4 matrix parameter space is performed, where the model to minimize the criterion value is selected. If 'arma_to_ingarch' is selected, uses an auto_arma as the starting point for the selection algorithm. The 'arma_to_ingarch' is the only one to perform a seasonal adjustment
trace	Logical. If the automatic parameter algorithm is running, print the path to the best model estimation

Value

A model specification.

Specials

pq: pq defines the non-seasonal autoregressive and moving averages terms, it can be defined by the user, or if it's omitted, the automatic parameter selection algorithm is triggered. The automatic parameter selection algorithm will fit the best model based on the information criterion.

PQ: PQ defines the seasonal autoregressive and moving averages terms, it can be defined by the user, or if it's omitted, the automatic parameter selection algorithm is triggered (only for 'arma_to_ingarch' algorithm). The automatic parameter selection algorithm will fit the best model based on the information criterion.

xreg: Exogenous regressors can be included in a INGARCH model without explicitly using the 'xreg()' special. Common exogenous regressor specials as specified in ['common_xregs'] can also be used. These regressors are handled using [stats::model.frame()], and so interactions and other functionality behaves similarly to [stats::lm()].

The inclusion of a constant in the model follows the similar rules to ['stats::lm()'], where including '1' will add a constant and '0' or '-1' will remove the constant. If left out, the inclusion of a constant will be determined by minimizing 'ic'.

If a xreg is provided, the model forecast is not available

```
xreg(..., fixed = list())
```

'...' Bare expressions for the exogenous regressors (such as 'log(x)')

'fixed' A named list of fixed parameters for coefficients. The names identify the coefficient, and should match the name of the

Examples

```
# Manual INGARCH specification
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1)))

# Automatic INGARCH specification
tsibbledata::aus_production |>
  fabletools::model(auto_ing_naive =
    INGARCH(Beer,
      ic = 'aic',
      trace = TRUE,
      algorithm = 'naive_search'),
    auto_ing_arm_ing =
    INGARCH(Beer,
      ic = 'aic',
      trace = TRUE,
      algorithm = 'arma_to_ingarch'))
```

residuals.GLARMA

Extract residuals from a fable model

Description

Extracts the residuals.

Usage

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

Arguments

object	A model for which forecasts are required.
...	Other arguments passed to methods

Value

A vector of fitted residuals.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_gla = GLARMA(Beer ~ pq(1,0))) |>
  dplyr::select(manual_gla) |>
  residuals()
```

residuals.INGARCH	<i>Extract residuals from a fable model</i>
-------------------	---

Description

Extracts the residuals.

Usage

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

Arguments

object	A model for which forecasts are required.
...	Other arguments passed to methods

Value

A vector of fitted residuals.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1) + PQ(1,1))) |>
  dplyr::select(manual_ing) |>
  residuals()
```

tidy.GLARMA	<i>Tidy a fable model</i>
-------------	---------------------------

Description

Returns the coefficients from the model in a 'tibble' format.

Usage

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

Arguments

x	An object to be converted into a tidy <code>tibble::tibble()</code> .
...	Additional arguments to tidying method.

Value

The model's coefficients in a 'tibble'.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_gla = GLARMA(Beer ~ pq(1,0))) |>
  dplyr::select(manual_gla) |>
  fabletools::tidy()
```

tidy.INGARCH

Tidy a fable model

Description

Returns the coefficients from the model in a 'tibble' format.

Usage

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

Arguments

x An object to be converted into a tidy `tibble::tibble()`.
... Additional arguments to tidying method.

Value

The model's coefficients in a 'tibble'.

Examples

```
tsibbledata::aus_production |>
  fabletools::model(manual_ing = INGARCH(Beer ~ pq(1,1))) |>
  dplyr::select(manual_ing) |>
  fabletools::tidy()
```

Index

fitted.GLARMA, [2](#)
fitted.INGARCH, [3](#)
forecast.GLARMA, [3](#)
forecast.INGARCH, [4](#)

glance.GLARMA, [5](#)
glance.INGARCH, [6](#)
GLARMA, [7](#)

INGARCH, [8](#)

residuals.GLARMA, [10](#)
residuals.INGARCH, [11](#)

tibble::tibble(), [11](#), [12](#)
tidy.GLARMA, [11](#)
tidy.INGARCH, [12](#)