Package ‘bootGOF’

October 12, 2022

Title Bootstrap Based Goodness-of-Fit Tests

Version 0.1.0

Description Bootstrap based goodness-of-fit tests. It allows to perform rigorous statistical tests to check if a chosen model family is correct based on the marked empirical process. The implemented algorithms are described in (Dikta and Scheer (2021) <doi:10.1007/978-3-030-73480-0>) and can be applied to generalized linear models without any further implementation effort. As far as certain linearity conditions are fulfilled the resampling scheme are also applicable beyond generalized linear models. This is reflected in the software architecture which allows to reuse the resampling scheme by implementing only certain interfaces for models that are not supported natively by the package.

Imports checkmate (>= 2.0.0), R6 (>= 2.4.1)

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.0

URL https://github.com/MarselScheer/bootGOF

BugReports https://github.com/MarselScheer/bootGOF/issues

Suggests testthat, covr, roxygen2, pkgdown, devtools, tinytest, mockery, knitr, rmarkdown, minpack.lm, MASS

VignetteBuilder knitr

NeedsCompilation no

Author Marsel Scheer [aut, cre], Gerhard Dikta [aut]

Maintainer Marsel Scheer <scheer@freescience.de>

Repository CRAN

Date/Publication 2021-06-24 07:30:02 UTC
R topics documented:

GOF_glm_info_extractor ........................................... 2
GOF_glm_sim_param ............................................. 3
GOF_glm_trainer .................................................. 4
GOF_lm_info_extractor .......................................... 5
GOF_lm_sim_param ................................................ 6
GOF_lm_trainer ................................................... 6
GOF_model ......................................................... 7
GOF_model_info_extractor ....................................... 9
GOF_model_resample ............................................. 10
GOF_model_simulator .......................................... 11
GOF_model_test .................................................. 11
GOF_model_trainer ............................................. 13
GOF_sim_wild_rademacher .................................... 14
Rn1_CvM .......................................................... 15
Rn1_KS ........................................................... 15
Rn1_statistic ...................................................... 16
rrademacher ........................................................ 17

Index 18

GOF_glm_info_extractor

Implements the "interface" GOF_model_info_extractor for for generalized linear models

Description

This class is specialized in extracting various information from an object of class "glm"

Super class

bootGOF::GOF_model_info_extractor -> GOF_glm_info_extractor

Methods

Public methods:

• GOF_glm_info_extractor$yhat()
• GOF_glm_info_extractor$y_minus_yhat()
• GOF_glm_info_extractor$beta_x_covariates()
• GOF_glm_info_extractor$clone()

Method yhat(): see GOF_model_info_extractor

Usage:
GOF_glm_info_extractor$yhat(model)

Arguments:
GOF_glm_sim_param

model see GOF_model_info_extractor

Method y_minus_yhat(): see GOF_model_info_extractor

Usage:
GOF_glm_info_extractor$y_minus_yhat(model)

Arguments:
model see GOF_model_info_extractor

Returns: see GOF_model_info_extractor

Method beta_x_covariates(): see GOF_model_info_extractor

Usage:
GOF_glm_info_extractor$beta_x_covariates(model)

Arguments:
model see GOF_model_info_extractor

Returns: see GOF_model_info_extractor

Method clone(): The objects of this class are cloneable with this method.

Usage:
GOF_glm_info_extractor$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

GOF_glm_sim_param Implements the "interface" GOF_model_simulator for for generalized linear models

Description
after the GLM was fitted the distribution of the of the dependent variable is fully specified and used here to generate new dependent variables that follow model

Methods

Public methods:
- GOF_glm_sim_param$resample_y()
- GOF_glm_sim_param$clone()

Method resample_y(): see GOF_model_simulator

Usage:
GOF_glm_sim_param$resample_y(model)

Arguments:
model  see GOF_model_simulator
Returns:  see GOF_model_simulator

Method clone(): The objects of this class are cloneable with this method.
Usage:
GOF_glm_sim_param$clone(deep = FALSE)
Arguments:
deep  Whether to make a deep clone.

---

GOF_glm_trainer  Implements the "interface" GOF_model_trainer for for generalized linear models

Description
refits an object of class "glm" to a new data set

Methods
Public methods:
• GOF_glm_trainer$refit()
• GOF_glm_trainer$clone()

Method refit(): see GOF_model_trainer
Usage:
GOF_glm_trainer$refit(model, data)
Arguments:
model  see GOF_model_trainer
data  see GOF_model_trainer
Returns:  see GOF_model_trainer

Method clone(): The objects of this class are cloneable with this method.
Usage:
GOF_glm_trainer$clone(deep = FALSE)
Arguments:
deep  Whether to make a deep clone.
Description
This class is specialized in extracting various information from an object of class "lm"

Super class

bootGOF::GOF_model_info_extractor -> GOF_lm_info_extractor

Methods

Public methods:
• GOF_lm_info_extractor$yhat()
• GOF_lm_info_extractor$y_minus_yhat()
• GOF_lm_info_extractor$beta_x_covariates()
• GOF_lm_info_extractor$clone()

Method yhat(): see GOF_model_info_extractor
Usage:
GOF_lm_info_extractor$yhat(model)
Arguments:
model see GOF_model_info_extractor
Returns: see GOF_model_info_extractor

Method y_minus_yhat(): see GOF_model_info_extractor
Usage:
GOF_lm_info_extractor$y_minus_yhat(model)
Arguments:
model see GOF_model_info_extractor
Returns: see GOF_model_info_extractor

Method beta_x_covariates(): see GOF_model_info_extractor
Usage:
GOF_lm_info_extractor$beta_x_covariates(model)
Arguments:
model see GOF_model_info_extractor
Returns: see GOF_model_info_extractor

Method clone(): The objects of this class are cloneable with this method.
Usage:
GOF_lm_info_extractor$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.
**Description**

After the classical linear model was fitted, the normal distribution of the dependent variable is fully specified and used here to generate new dependent variables that follow the model.

**Methods**

**Public methods:**

- `GOF_lm_sim_param$resample_y()`
- `GOF_lm_sim_param$clone()`

**Method `resample_y()`**: generates/resamples the dependent variables based on the parameteric nature defined by `model`.

*Usage:*

```
GOF_lm_sim_param$resample_y(model)
```

*Arguments:*

- `model` see `GOF_model_simulator`

*Returns:*) see `GOF_model_simulator`

**Method `clone()`**: The objects of this class are cloneable with this method.

*Usage:*

```
GOF_lm_sim_param$clone(deep = FALSE)
```

*Arguments:*

- `deep` Whether to make a deep clone.

**Description**

Refits an object of class "lm" to a new data set.
**Methods**

**Public methods:**
- `GOF_lm_trainer$refit()`
- `GOF_lm_trainer$clone()`

**Method** `refit()`: see `GOF_model_trainer`

*Usage:*

```r
GOF_lm_trainer$refit(model, data)
```

*Arguments:*

- `model` see `GOF_model_trainer`
- `data` see `GOF_model_trainer`

*Returns:* see `GOF_model_trainer`

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```r
GOF_lm_trainer$clone(deep = FALSE)
```

*Arguments:*

- `deep` Whether to make a deep clone.

---

**GOF_model**

*Convenience function for creating a GOF-test for statistical models*

---

**Description**

Simplifies the creation of an instance of `GOF_model_test`, the actual work horse for performing a goodness-of-fit-test.

**Usage**

```r
GOF_model(
  model,
  data,
  nmb_boot_samples,
  simulator_type,
  y_name,
  Rn1_statistic,
  gof_model_resample_class = GOF_model_resample,
  gof_model_test_class = GOF_model_test
)
```
Arguments

model of class ‘lm’ or ‘glm’. Caution with MASS::glm.nb, see vignette ‘New-Models’ for more details.
data see GOF_model_test
nmb_boot_samples see GOF_model_test
simulator_type either "parameteric" or "semi_parameteric_rademacher"
y_name see GOF_model_test
Rn1_statistic see GOF_model_test
gof_model_resample_class no need to change this parameter. Here the class used for resampling the model (GOF_model_resample) is injected. This parameter simply makes it easier to test the convenience function properly.
gof_model_test_class no need to change this parameter. Here the class used for performing the GOF test (GOF_model_test) is injected. This parameter simply makes it easier to test the convenience function properly.

Value

instance of GOF_model_test

Examples

```r
set.seed(1)
N <- 100
X1 <- rnorm(N)
X2 <- rnorm(N)
d <- data.frame(
y = rpois(n = N, lambda = exp(4 + X1 * 2 + X2 * 6)),
x1 = X1,
x2 = X2)
fit <- glm(y ~ x1, data = d, family = poisson())
mt <- GOF_model(
  model = fit,
  data = d,
  nmb_boot_samples = 100,
  simulator_type = "parametric",
  y_name = "y",
  Rn1_statistic = Rn1_KS$new())
mt$get_pvalue()
fit <- glm(y ~ x1 + x2, data = d, family = poisson())
mt <- GOF_model(
  model = fit,
  data = d,
  nmb_boot_samples = 100,
  simulator_type = "parametric",
  y_name = "y",
  Rn1_statistic = Rn1_KS$new())
mt$get_pvalue()
```
**Description**

R6 does not offer interfaces. Hence all methods are considered as abstract.

**Methods**

**Public methods:**
- `GOF_model_info_extractor$yhat()`
- `GOF_model_info_extractor$y_minus_yhat()`
- `GOF_model_info_extractor$beta_x_covariates()`
- `GOF_model_info_extractor$clone()`

**Method `yhat()`:** Abstract function that estimates/predicts the dependent variable in model.

*Usage:*

```r
GOF_model_info_extractor$yhat(model)
```

*Arguments:*

- `model` fitted model

*Returns:* estimate/prediction of the dependent variable fitted by model

**Method `y_minus_yhat()`:** abstract function that calculates the residuals on the scale of the dependent variable.

*Usage:*

```r
GOF_model_info_extractor$y_minus_yhat(model)
```

*Arguments:*

- `model` fitted model

*Returns:* residuals on the scale of the dependent variable

**Method `beta_x_covariates()`:** abstract function that calculates the inner product of estimated parameters and the independent variables.

*Usage:*

```r
GOF_model_info_extractor$beta_x_covariates(model)
```

*Arguments:*

- `model` fitted model

*Returns:* inner product of the estimated parameters and the independent variables.

**Method `clone()`:** The objects of this class are cloneable with this method.

*Usage:*

```r
GOF_model_info_extractor$clone(deep = FALSE)
```

*Arguments:*

- `deep` Whether to make a deep clone.
R6 Class representing the resampling scheme for Goodness-of-fit-tests for (linear) models

Description

Class is able to resample model fit, i.e. generate a new data set and refit the model to the new data.

Methods

Public methods:

• GOF_model_resample$new()
• GOF_model_resample$resample()
• GOF_model_resample$clone()

Method new():

Usage:
GOF_model_resample$new(gof_model_simulator, gof_model_trainer)

Arguments:
gof_model_simulator an instance that implements GOF_model_simulator
gof_model_trainer an instance that implements GOF_model_trainer

Returns: No explicit return

Method resample(): resamples the dependent variable in data and refits model to that new data set

Usage:
GOF_model_resample$resample(model, data, y_name)

Arguments:
model fitted model based on data
data used to fit model
y_name string specifying the name of the dependent variable

Returns: a resampled version of model

Method clone(): The objects of this class are cloneable with this method.

Usage:
GOF_model_resample$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
**GOF_model_simulator**  
*R6 Class representing a generator/resample of the dependent variable*

**Description**
R6 does not offer interfaces. Hence all methods are considered as abstract.

**Methods**

**Public methods:**
- `GOF_model_simulator$resample_y()`
- `GOF_model_simulator$clone()`

**Method** `resample_y()`: Abstract function that resamples/generates the dependent variable  
*Usage:*  
`GOF_model_simulator$resample_y(model)`  
*Arguments:*  
model fitted model  
*Returns:* generates the dependent variable according to the model

**Method** `clone()`: The objects of this class are cloneable with this method.  
*Usage:*  
`GOF_model_simulator$clone(deep = FALSE)`  
*Arguments:*  
deep Whether to make a deep clone.

**GOF_model_test**  
*R6 Class representing the Goodness-of-Fit test for (linear) models.*

**Description**
This class can test the null hypothesis that data follows a particular linear model, i.e. classical linear models, generalized linear models or models of the type $m(\beta^T X) + \epsilon$.

**Methods**

**Public methods:**
- `GOF_model_test$new()`  
- `GOF_model_test$get_Rn1_org()`  
- `GOF_model_test$get_Rn1_boot()`  
- `GOF_model_test$get_pvalue()`  
- `GOF_model_test$clone()`
Method new():

Usage:
GOF_model_test$new(
  model,
  data,
  nmb_boot_samples,
  y_name,
  Rn1_statistic,
  gof_model_info_extractor,
  gof_model_resample
)

Arguments:
model a fitted model
data used to fit model
nmb_boot_samples integer specifying the number of bootstrap samples to perform
y_name string specifying the name of the dependent variable in data
Rn1_statistic statistic used to map the marked empirical process to the real line. Needs to be an instance of the class that implements Rn1_statistic
gof_model_info_extractor an instance that implements GOF_model_info_extractor in order to apply it to model
gof_model_resample an instance that implements GOF_model_resample in order to apply it to model

Returns: An instance of the Class

Method get_Rn1_org(): calculates the marked empirical process for model

Usage:
GOF_model_test$get_Rn1_org()

Returns: vector ordered by the inner product of the estimated parameter and the independent variables

Method get_Rn1_boot(): calculates the marked empirical process for the resampled versions of model

Usage:
GOF_model_test$get_Rn1_boot()

Returns: list of length nmb_boot_samples where every element is a vector ordered by the inner product of the estimated parameter and the dependent variables

Method get_pvalue(): p-value for Goodness-of-Fit-test for model

Usage:
GOF_model_test$get_pvalue()

Returns: p-value for the null hypothesis that the dependent variable was generated according to model

Method clone(): The objects of this class are cloneable with this method.
Usage:
GOF_model_test$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

R6 Class representing a trainer for fitting models

Description

R6 does not offer interfaces. Hence all methods are considered as abstract.

Methods

Public methods:

- GOF_model_trainer$refit()
- GOF_model_trainer$clone()

Method refit(): Abstract function refits the model to a new data set

Usage:
GOF_model_trainer$refit(model, data)

Arguments:
model fitted model
data used for refitting the model

Returns: model refitted on data

Method clone(): The objects of this class are cloneable with this method.

Usage:
GOF_model_trainer$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
GOF_sim_wild_rademacher

Implements the "interface" GOF_model_simulator in a semi-parametric fashion

Description

This is a model agnostic resampling class, where Rademacher random variables are used to add or substract the residuals from the fitted values.

Methods

Public methods:

• GOF_sim_wild_rademacher$new()
• GOF_sim_wild_rademacher$resample_y()
• GOF_sim_wild_rademacher$clone()

Method new():

Usage:
GOF_sim_wild_rademacher$new(gof_model_info_extractor)

Arguments:
gof_model_info_extractor the info extractor that is used to derive the residuals and fitted values for resampling.

Method resample_y(): a wild bootstrap using Rademacher random variables to resample the dependent variable

Usage:
GOF_sim_wild_rademacher$resample_y(model)

Arguments:
model see GOF_model_simulator

Returns: see GOF_model_simulator

Method clone(): The objects of this class are cloneable with this method.

Usage:
GOF_sim_wild_rademacher$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
Rn1_CvM  
\textit{Cramer-von-Mises-statistic for marked empirical process}

\begin{description}
\item \textbf{Description}
\begin{itemize}
\item Implements the "interface" Rn1_statistic
\end{itemize}
\item \textbf{Super class}
\begin{itemize}
\item \texttt{bootGOF::Rn1_statistic} $\rightarrow$ Rn1_CvM
\end{itemize}
\item \textbf{Methods}
\begin{itemize}
\item \textbf{Public methods:}
\begin{itemize}
\item Rn1_CvM$\cdot$calc_statistic()
\item Rn1_CvM$\cdot$clone()
\end{itemize}
\item Method \texttt{calc_statistic()}: calculates the Cramer-von-Mises statistic
\begin{itemize}
\item \textit{Usage:}
Rn1_CvM$\cdot$calc_statistic(Rn1)
\item \textit{Arguments:}
Rn1  see Rn1_statistic
\item \textit{Returns:} see Rn1_statistic
\end{itemize}
\item Method \texttt{clone()}: The objects of this class are cloneable with this method.
\begin{itemize}
\item \textit{Usage:}
Rn1_CvM$\cdot$clone(deep = FALSE)
\item \textit{Arguments:}
depth  Whether to make a deep clone.
\end{itemize}
\end{itemize}
\end{description}

Rn1_KS  
\textit{Kolmogorov-Smirnov-statistic for marked empirical process}

\begin{description}
\item \textbf{Description}
\begin{itemize}
\item Implements the "interface" Rn1_statistic
\end{itemize}
\item \textbf{Super class}
\begin{itemize}
\item \texttt{bootGOF::Rn1_statistic} $\rightarrow$ Rn1_KS
\end{itemize}
\end{description}
Methods

Public methods:
- `Rn1_KS$calc_statistic()
- `Rn1_KS$clone()

Method `calc_statistic()`: calculates the Kolmogorov-Smirnov-statistic

Usage:
`Rn1_KS$calc_statistic(Rn1)

Arguments:
- `Rn1` see `Rn1_statistic`

Returns: see `Rn1_statistic`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`Rn1_KS$clone(deep = FALSE)

Arguments:
- `deep` Whether to make a deep clone.

Rn1_statistic

R6 Class representing statistics for marked empirical processes

Description

R6 does not offer interfaces. Hence all methods are considered as abstract.

Methods

Public methods:
- `Rn1_statistic$calc_statistic()
- `Rn1_statistic$clone()

Method `calc_statistic()`: Abstract function that calculates the statistic for a given marked empirical process

Usage:
`Rn1_statistic$calc_statistic(Rn1)

Arguments:
- `Rn1` marked empirical process as a double vector

Returns: statistic based on `Rn1`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`Rn1_statistic$clone(deep = FALSE)

Arguments:
- `deep` Whether to make a deep clone.
**rrademacher**

*Generates Rademacher distributed random variables*

**Description**

Generates Rademacher distributed random variables

**Usage**

`rrademacher(n)`

**Arguments**

- **n**: number of random variables to be generated

**Value**

vector of values following the Rademacher distribution
Index

<table>
<thead>
<tr>
<th>Function/Class</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bootGOF::GOF_model_info_extractor</td>
<td>2, 5</td>
</tr>
<tr>
<td>bootGOF::Rn1_statistic</td>
<td>15</td>
</tr>
<tr>
<td>GOF_glm_info_extractor</td>
<td>2</td>
</tr>
<tr>
<td>GOF_glm_sim_param</td>
<td>3</td>
</tr>
<tr>
<td>GOF_glm_trainer</td>
<td>4</td>
</tr>
<tr>
<td>GOF_lm_info_extractor</td>
<td>5</td>
</tr>
<tr>
<td>GOF_lm_sim_param</td>
<td>6</td>
</tr>
<tr>
<td>GOF_lm_trainer</td>
<td>6</td>
</tr>
<tr>
<td>GOF_model</td>
<td>7</td>
</tr>
<tr>
<td>GOF_model_info_extractor</td>
<td>2, 3, 5, 9, 12</td>
</tr>
<tr>
<td>GOF_model_resample</td>
<td>8, 10, 12</td>
</tr>
<tr>
<td>GOF_model_simulator</td>
<td>3, 4, 6, 10, 11, 14</td>
</tr>
<tr>
<td>GOF_model_test</td>
<td>7, 8, 11</td>
</tr>
<tr>
<td>GOF_model_trainer</td>
<td>4, 7, 10, 13</td>
</tr>
<tr>
<td>GOF_sim_wild_rademacher</td>
<td>14</td>
</tr>
<tr>
<td>Rn1_CvM</td>
<td>15</td>
</tr>
<tr>
<td>Rn1_KS</td>
<td>15</td>
</tr>
<tr>
<td>Rn1_statistic</td>
<td>12, 15, 16, 16</td>
</tr>
<tr>
<td>rrادم彻</td>
<td>17</td>
</tr>
</tbody>
</table>