Package 'broomExtra'

July 24, 2020

Type Package

Title Enhancements for 'broom' and 'easystats' Package Families

Version 4.0.4

Maintainer Indrajeet Patil <patilindrajeet.science@gmail.com>

Description Provides helper functions that assist in data analysis workflows involving regression analyses. The goal is to combine the functionality offered by different set of packages ('broom', 'broom.mixed', 'parameters', and 'performance') through a common syntax to return tidy dataframes containing model parameters and performance measure summaries. The 'grouped_' variants of the generics provides a convenient way to execute functions across a combination of grouping variable(s) in a dataframe.

License GPL-3 | file LICENSE

URL https://indrajeetpatil.github.io/broomExtra/, https://github.com/IndrajeetPatil/broomExtra

BugReports https://github.com/IndrajeetPatil/broomExtra/issues

Depends R (>= 3.6.0)

Imports broom, broom.mixed, dplyr, ipmisc, parameters, performance, rlang

Suggests generics, ggplot2, lavaan, lme4, MASS, mixor, orcutt, rmarkdown, spelling, testthat

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

R topics documented:

`augment` *Retrieve augmented dataframe if it exists.*

Description

Check if a `augment` method exits for a given object, either in `broom` or in `broom.mixed`. If it does, return the model summary dataframe, if not, return a `NULL`.

Usage

```
augment(x, ...)
```

Arguments

`x`	Model object or other R object with information to append to observations.
`...`	Addition arguments to `augment` method.

Value

A `tibble::tibble()` with information about data points.

Methods

No methods found in currently loaded packages.

See Also

`grouped_augment`

Examples

```
set.seed(123)
library(lme4)

# mixed-effects models (`broom.mixed` will be used)
lmm.mod <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
broomExtra::augment(lmm.mod)

# linear model (`broom` will be used)
lm.mod <- lm(Reaction ~ Days, sleepstudy)
broomExtra::augment(lm.mod)
```

---

easystats_to_tidy_names

Convert easystats *package outputs to* tidymodels *conventions.*

Description

Both broom package from tidymodels universe and parameters package from easystats universe can provide model summaries for a large number of model objects. This is a convenience function that converts naming conventions adopted in easystats to the ones adopted in the broom package.

Usage

```
easystats_to_tidy_names(x)
```

Arguments

x	A statistical model object

Examples

```
# example model object
mod <- stats::lm(formula = wt ~ am * cyl, data = mtcars)

# `tidy`-fied output
easystats_to_tidy_names(parameters::model_parameters(mod))
```

---

glance

Retrieve model summary dataframe if it exists.

Description

Check if a glance method exits for a given object, either in broom or in broom.mixed. If it does, return the model summary dataframe, if not, return a NULL. In this case, you can try the broomExtra::glance_performance function.

Usage

```
glance(x, ...)
```

Arguments

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

Methods

No methods found in currently loaded packages.

See Also

grouped_glance, glance_performance

Examples

```
set.seed(123)
library(lme4)

# mixed-effects models (`broom.mixed` will be used)
lmm.mod <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
broomExtra::glance(lmm.mod)

# linear model (`broom` will be used)
lm.mod <- lm(Reaction ~ Days, sleepstudy)
broomExtra::glance(lm.mod)
```

---

glance_performance *Model performance summary dataframes using* broom *and* easystats.

Description

Computes indices of model performance for regression models.

Usage

```
glance_performance(x, ...)
```

Arguments

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

Details

The function will attempt to get these details either using broom::glance or performance::model_performance. If both function provide model performance measure summaries, the function will try to combine them into a single dataframe.

Value

A data frame (with one row) and one column per "index".

Examples

```
set.seed(123)
mod <- lm(mpg ~ wt + cyl, data = mtcars)
broomExtra::glance_performance(mod)
```

`grouped_augment` *Augmented data from grouped analysis of any function that has* `data` *argument in its function call.*

Description

Augmented data from grouped analysis of any function that has `data` argument in its function call.

Usage

```
grouped_augment(data, grouping.vars, ..f, ..., augment.args = list())
```

Arguments

`data`	Dataframe (or tibble) from which variables are to be taken.
`grouping.vars`	Grouping variables.
`..f`	A function, or function name as a string.
`...`	<dynamic> Arguments for `.fn`.
`augment.args`	A list of arguments to be used in the relevant `S3` method.

Value

A `tibble::tibble()` with information about data points.

Methods

No methods found in currently loaded packages.

See Also

`augment`

Examples

```
set.seed(123)
# to speed up computation, let's use only 50% of the data

# linear model
broomExtra::grouped_augment(
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = c(cut, color),
  formula = price ~ carat - 1,
  ..f = stats::lm,
  na.action = na.omit,
  augment.args = list(se_fit = TRUE)
)
```

```
# linear mixed effects model
broomExtra::grouped_augment(
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = "cut",
  ..f = lme4::lmer,
  formula = price ~ carat + (carat | color) - 1,
  control = lme4::lmerControl(optimizer = "bobyqa")
)
```

`grouped_glance` *Model summary output from grouped analysis of any function that has* `data` *argument in its function call.*

Description

Model summary output from grouped analysis of any function that has `data` argument in its function call.

Usage

```
grouped_glance(data, grouping.vars, ..f, ...)
```

Arguments

`data`	Dataframe (or tibble) from which variables are to be taken.
`grouping.vars`	Grouping variables.
`..f`	A function, or function name as a string.
`...`	<dynamic> Arguments for `.fn`.

Methods

No methods found in currently loaded packages.

See Also

`glance`

Examples

```
set.seed(123)
# to speed up computation, let's use only 50% of the data

# linear model
broomExtra::grouped_glance(
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = c(cut, color),
  formula = price ~ carat - 1,
  ..f = stats::lm,
  na.action = na.omit
)

# linear mixed effects model
broomExtra::grouped_glance(
```

```
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = "cut",
  ..f = lme4::lmer,
  formula = price ~ carat + (carat | color) - 1,
  control = lme4::lmerControl(optimizer = "bobyqa")
)
```

`grouped_tidy` *Tidy output from grouped analysis of any function that has* `data` *argument in its function call.*

Description

Tidy output from grouped analysis of any function that has `data` argument in its function call.

Usage

```
grouped_tidy(data, grouping.vars, ..f, ..., tidy.args = list())
```

Arguments

`data`	Dataframe (or tibble) from which variables are to be taken.
`grouping.vars`	Grouping variables.
`..f`	A function, or function name as a string.
`...`	<dynamic> Arguments for `.fn`.
`tidy.args`	A list of arguments to be used in the relevant `S3` method.

Value

A `tibble::tibble()` with information about model components.

Methods

No methods found in currently loaded packages.

See Also

`tidy`

Examples

```
set.seed(123)
# to speed up computation, let's use only 50% of the data

# linear model
broomExtra::grouped_tidy(
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = c(cut, color),
  formula = price ~ carat - 1,
  ..f = stats::lm,
  na.action = na.omit,
  tidy.args = list(quick = TRUE)
```

```
)

# linear mixed effects model
broomExtra::grouped_tidy(
  data = dplyr::sample_frac(tbl = ggplot2::diamonds, size = 0.5),
  grouping.vars = "cut",
  ..f = lme4::lmer,
  formula = price ~ carat + (carat | color) - 1,
  control = lme4::lmerControl(optimizer = "bobyqa"),
  tidy.args = list(conf.int = TRUE, conf.level = 0.99)
)
```

`tidy` *Retrieve tidy dataframe if it exists.*

Description

Checks if a `tidy` method exits for a given object, either in `broom` or in `broom.mixed`. If it does, it turn an object into a tidy tibble, if not, return a `NULL`. In this case, you can try the `broomExtra::tidy_parameters` function.

Usage

```
tidy(x, ...)
```

Arguments

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

Value

A `tibble::tibble()` with information about model components.

Methods

No methods found in currently loaded packages.

See Also

`grouped_tidy`, `tidy_parameters`

Examples

```
set.seed(123)
library(lme4)

# mixed-effects models (`broom.mixed` will be used)
lmm.mod <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
broomExtra::tidy(x = lmm.mod, effects = "fixed", exponentiate = TRUE)

# linear model (`broom` will be used)
lm.mod <- lm(Reaction ~ Days, sleepstudy)
```

```
broomExtra::tidy(x = lm.mod, conf.int = TRUE)

# unsupported object (the function will return `NULL` in such cases)
broomExtra::tidy(list(1, c("x", "y")))
```

`tidy_parameters` *Tidy dataframes of model parameters using* `broom` *and* `easystats`.

Description

Computes parameters for regression models.

Usage

```
tidy_parameters(x, conf.int = TRUE, ...)
```

Arguments

`x`	An object to be converted into a tidy `tibble::tibble()`.
`conf.int`	Indicating whether or not to include a confidence interval in the tidied output.
`...`	Additional arguments that will be passed to `parameters::model_parameters` and `broom::tidy`.

Details

The function will attempt to get these details first using `parameters::model_parameters` and then using `broom::tidy`.

Value

A data frame of indices related to the model's parameters.

Examples

```
set.seed(123)
mod <- lm(mpg ~ wt + cyl, data = mtcars)
broomExtra::tidy_parameters(mod)
```

Index