

Package ‘regport’

July 23, 2025

Title Regression Model Processing Port

Version 0.3.0

Description Provides R6 classes, methods and utilities to construct, analyze, summarize, and visualize regression models.

License MIT + file LICENSE

URL <https://github.com/ShixiangWang/regport>,
<https://shixiangwang.github.io/regport/>

BugReports <https://github.com/ShixiangWang/regport/issues>

Imports broom.helpers, data.table, dplyr, forestploter, glue, parameters, R6, rlang (>= 0.4.11), stats, survival

Suggests parallel, roxygen2, see, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Author Shixiang Wang [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-9855-7357>>)

Maintainer Shixiang Wang <w_shixiang@163.com>

Repository CRAN

Date/Publication 2022-05-10 05:30:06 UTC

Contents

REGModel	2
REGModelList	5
Index	8

REGModel

*R6 class representing a regression model***Description**

Contains fields storing data and methods to build, process and visualize a regression model. Currently, this class is designed for CoxPH and GLM regression models.

Public fields

`data` a `data.table` storing modeling data.

`recipe` an R formula storing model formula.

`terms` all terms (covariables, i.e. columns) used for building model.

`args` other arguments used for building model.

`model` a constructed model.

`type` model type (class).

`result` model result, a object of `parameters_model`. Can be converted into `data.frame` with `as.data.frame()` or `data.table::as.data.table()`.

`forest_data` more detailed data used for plotting forest.

Methods**Public methods:**

- `REGModel$new()`
- `REGModel$get_forest_data()`
- `REGModel$plot_forest()`
- `REGModel$plot()`
- `REGModel$print()`
- `REGModel$clone()`

Method `new()`: Build a `REGModel` object.

Usage:

```
REGModel$new(
  data,
  recipe,
  ...,
  f = c("coxph", "binomial", "gaussian", "Gamma", "inverse.gaussian", "poisson",
        "quasi", "quasibinomial", "quasipoisson"),
  exp = NULL,
  ci = 0.95
)
```

Arguments:

`data` a `data.table` storing modeling data.

recipe an R formula or a list with two elements 'x' and 'y', where 'x' is for covariables and 'y' is for label. See example for detail operation.

... other parameters passing to corresponding regression model function.

f a length-1 string specifying modeling function or family of `glm()`, default is 'coxph'. Other options are members of GLM family, see `stats::family()`. 'binomial' is logistic, and 'gaussian' is linear.

exp logical, indicating whether or not to exponentiate the the coefficients.

ci confidence Interval (CI) level. Default to 0.95 (95%). e.g. `survival::coxph()`.

Returns: a REGModel R6 object.

Method `get_forest_data()`: get tidy data for plotting forest.

Usage:

```
REGModel$get_forest_data(separate_factor = FALSE, global_p = FALSE)
```

Arguments:

separate_factor separate factor/class as a blank row.

global_p if TRUE, return global p value.

Method `plot_forest()`: plot forest.

Usage:

```
REGModel$plot_forest(ref_line = NULL, xlim = NULL, ...)
```

Arguments:

ref_line reference line, default is 1 for HR.

xlim limits of x axis.

... other plot options passing to `forestploter::forest()`. Also check <https://github.com/adayim/forestploter> to see more complex adjustment of the result plot.

Method `plot()`: print the REGModel\$result with default plot methods from **see** package.

Usage:

```
REGModel$plot(...)
```

Arguments:

... other parameters passing to `plot()` in `see::plot.see_parameters_model` function.

Method `print()`: print the REGModel object

Usage:

```
REGModel$print(...)
```

Arguments:

... unused.

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```

library(survival)
test1 <- data.frame(
  time = c(4, 3, 1, 1, 2, 2, 3),
  status = c(1, 1, 1, 0, 1, 1, 0),
  x = c(0, 2, 1, 1, 1, 0, 0),
  sex = c(0, 0, 0, 0, 1, 1, 1)
)
test1$sex <- factor(test1$sex)

# -----
# Build a model
# -----

# way 1:
mm <- REGModel$new(
  test1,
  Surv(time, status) ~ x + strata(sex)
)
mm
as.data.frame(mm$result)
if (require("see")) mm$plot()
mm$print() # Same as print(mm)

# way 2:
mm2 <- REGModel$new(
  test1,
  recipe = list(
    x = c("x", "strata(sex)"),
    y = c("time", "status")
  )
)
mm2

# Add other parameters, e.g., weights
# For more, see ?coxph
mm3 <- REGModel$new(
  test1,
  recipe = list(
    x = c("x", "strata(sex)"),
    y = c("time", "status")
  ),
  weights = c(1, 1, 1, 2, 2, 2, 3)
)
mm3$args

# -----
# Another type of model
# -----
library(stats)
counts <- c(18, 17, 15, 20, 10, 20, 25, 13, 12)
outcome <- gl(3, 1, 9)

```

```

treatment <- gl(3, 3)
data <- data.frame(treatment, outcome, counts)

mm4 <- REGModel$new(
  data,
  counts ~ outcome + treatment,
  f = "poisson"
)
mm4
mm4$plot_forest()
mm4$get_forest_data()
mm4$plot_forest()

```

REGModelList

*R6 class representing a list of regression model***Description**

Contains fields storing data and methods to build, process and visualize a list of regression model. Currently, this class is designed for CoxPH and GLM regression models.

Public fields

`data` a `data.table` storing modeling data.
`x` focal variables (terms).
`y` predicted variables or expression.
`covars` covariables.
`mlist` a list of `REGModel`.
`args` other arguments used for building model.
`type` model type (class).
`result` model result, a object of `parameters_model`. Can be converted into `data.frame` with `as.data.frame()` or `data.table::as.data.table()`.
`forest_data` more detailed data used for plotting forest.

Methods**Public methods:**

- `REGModelList$new()`
- `REGModelList$build()`
- `REGModelList$plot_forest()`
- `REGModelList$print()`
- `REGModelList$clone()`

Method `new()`: Create a `REGModelList` object.

Usage:

```
REGModelList$new(data, y, x, covars = NULL)
```

Arguments:

data a data.table storing modeling data.

y predicted variables or expression.

x focal variables (terms).

covars covariables.

Returns: a REGModelList R6 object.

Method build(): Build REGModelList object.

Usage:

```
REGModelList$build(
  f = c("coxph", "binomial", "gaussian", "Gamma", "inverse.gaussian", "poisson",
        "quasi", "quasibinomial", "quasipoisson"),
  exp = NULL,
  ci = 0.95,
  parallel = FALSE,
  ...
)
```

Arguments:

f a length-1 string specifying modeling function or family of `glm()`, default is 'coxph'. Other options are members of GLM family, see `stats::family()`. 'binomial' is logistic, and 'gaussian' is linear.

exp logical, indicating whether or not to exponentiate the the coefficients.

ci confidence Interval (CI) level. Default to 0.95 (95%). e.g. `survival::coxph()`.

parallel if TRUE, use N-1 cores to run the task.

... other parameters passing to corresponding regression model function.

Returns: a REGModel R6 object.

Method plot_forest(): plot forest.

Usage:

```
REGModelList$plot_forest(
  ref_line = NULL,
  xlim = NULL,
  vars = NULL,
  p = NULL,
  ...
)
```

Arguments:

ref_line reference line, default is 1 for HR.

xlim limits of x axis.

vars selected variables to show.

p selected variables with level' pvalue lower than p.

... other plot options passing to `forestploter::forest()`. Also check <https://github.com/adayim/forestploter> to see more complex adjustment of the result plot.

Method print(): print the REGModelList object

Usage:

```
REGModelList$print(...)
```

Arguments:

... unused.

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```
ml <- REGModelList$new(  
  data = mtcars,  
  y = "mpg",  
  x = c("factor(cyl)", colnames(mtcars)[3:5]),  
  covars = c(colnames(mtcars)[8:9], "factor(gear)")  
)  
ml  
ml$print()  
ml$plot_forest()  
  
ml$build(f = "gaussian")  
## Not run:  
ml$build(f = "gaussian", parallel = TRUE)  
  
## End(Not run)  
ml$print()  
ml$result  
ml$forest_data  
ml$plot_forest()
```

Index

`as.data.frame()`, [2](#), [5](#)

`data.table::as.data.table()`, [2](#), [5](#)

`forestploter::forest()`, [3](#), [6](#)

`glm()`, [3](#), [6](#)

`REGModel`, [2](#)

`REGModelList`, [5](#)

`stats::family()`, [3](#), [6](#)

`survival::coxph()`, [3](#), [6](#)