Package 'tornado'

January 21, 2024

Title Plots for Model Sensitivity and Variable Importance

Version 0.1.3

Description Draws tornado plots for model sensitivity to univariate changes. Implements methods for many modeling methods including linear models, generalized linear models, survival regression models, and arbitrary machine learning models in the caret package. Also draws variable importance plots.

License GPL-3

Encoding UTF-8

Suggests testthat, caret, glmnet, randomForest, knitr, rmarkdown

RoxygenNote 7.3.0

Imports survival, assertthat, ggplot2, scales, grid, gridExtra, rlang

VignetteBuilder knitr

URL https://github.com/bertcarnell/tornado

BugReports https://github.com/bertcarnell/tornado/issues

NeedsCompilation no

Author Rob Carnell [aut, cre]

Maintainer Rob Carnell <bertcarnell@gmail.com>

Repository CRAN

Date/Publication 2024-01-21 17:30:02 UTC

R topics documented:

`importance` *Generic Importance Plot*

Description

Generic Importance Plot

Usage

```
importance(model_final, ...)
```

Arguments

`model_final`	a model object
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance.glm` `importance.lm` `importance.cv.glmnet` `importance.survreg`

`importance.cv.glmnet` *Plot Variable Importance for a GLMNET model*

Description

Plot Variable Importance for a GLMNET model

Usage

```
## S3 method for class 'cv.glmnet'
importance(model_final, model_data, form, dict = NA, nperm = 500, ...)
```

Arguments

`model_final`	a model object
`model_data`	the data used to fit the model
`form`	the model formula
`dict`	a variable dictionary for plotting
`nperm`	the number of permutations used to calculate the importance
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance`

Examples

```
if (requireNamespace("glmnet", quietly = TRUE))
{
  form <- formula(mpg ~ cyl*wt*hp)
  mf <- model.frame(form, data = mtcars)
  mm <- model.matrix(mf, mf)
  gtest <- glmnet::cv.glmnet(x = mm, y = mtcars$mpg, family = "gaussian")
  imp <- importance(gtest, mtcars, form, nperm = 50)
  plot(imp)
}
```

`importance.glm` *GLM variable importance plot*

Description

GLM variable importance plot

Usage

```
## S3 method for class 'glm'
importance(model_final, model_null, dict = NA, ...)
```

Arguments

`model_final`	a model object
`model_null`	a glm object for the null model
`dict`	a dictionary to translate the model variables to plotting variables
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance`

Examples

```
gtest <- glm(mpg ~ cyl*wt*hp + gear + carb, data=mtcars, family=gaussian)
gtestreduced <- glm(mpg ~ 1, data=mtcars, family=gaussian)
imp <- importance(gtest, gtestreduced)
plot(imp)

gtest <- glm(mpg ~ cyl + wt + hp + gear + carb, data=mtcars, family=gaussian)
gtestreduced <- glm(mpg ~ 1, data=mtcars, family=gaussian)
imp <- importance(gtest, gtestreduced)
plot(imp)

gtest <- glm(vs ~ wt + disp + gear, data=mtcars, family=binomial(link="logit"))
gtestreduced <- glm(vs ~ 1, data=mtcars, family=binomial(link="logit"))
imp <- importance(gtest, gtestreduced)
plot(imp)
```

`importance.lm` *Linear Model variable importance plot*

Description

Linear Model variable importance plot

Usage

```
## S3 method for class 'lm'
importance(model_final, model_null, dict = NA, ...)
```

Arguments

`model_final`	a model object
`model_null`	a `lm` object for the null model
`dict`	a dictionary to translate the model variables to plotting variables
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance`

Examples

```
gtest <- lm(mpg ~ cyl*wt*hp + gear + carb, data=mtcars)
gtestreduced <- lm(mpg ~ 1, data=mtcars)
imp <- importance(gtest, gtestreduced)
plot(imp)

gtest <- lm(mpg ~ cyl + wt + hp + gear + carb, data=mtcars)
gtestreduced <- lm(mpg ~ 1, data=mtcars)
imp <- importance(gtest, gtestreduced)
plot(imp)
```

`importance.survreg` *Create a variable importance plot for a survreg model*

Description

Create a variable importance plot for a survreg model

Usage

```
## S3 method for class 'survreg'
importance(model_final, model_data, dict = NA, nperm = 500, ...)
```

Arguments

`model_final`	a model object
`model_data`	the data used to fit the model
`dict`	a plotting dictionary for models terms
`nperm`	the number of permutations used to calculate the importance
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance`

Examples

```
model_final <- survival::survreg(survival::Surv(futime, fustat) ~ ecog.ps*rx + age,
                      data = survival::ovarian,
                      dist = "weibull")
imp <- importance(model_final, survival::ovarian, nperm = 50)
plot(imp)
```

`importance.train` *Importance Plot for the caret::train objects*

Description

Importance Plot for the caret::train objects

Usage

```
## S3 method for class 'train'
importance(model_final, ...)
```

Arguments

`model_final`	a model object
`...`	arguments passed to other methods

Value

an object of type `importance_plot`

`type`	the type of importance plot
`data`	the importance data required for the plot

See Also

`importance`

Examples

```
if (requireNamespace("caret", quietly = TRUE) &
    requireNamespace("randomForest", quietly = TRUE))
{
 model_final <- caret::train(x = subset(mtcars, select = -mpg), y = mtcars$mpg, method = "rf")
  imp <- importance(model_final)
  plot(imp)
}
```

`plot.importance_plot` *Plot an Importance Plot object*

Description

Plot an Importance Plot object

Usage

```
## S3 method for class 'importance_plot'
plot(
  x,
  plot = TRUE,
  nvar = NA,
  col_imp_alone = "#69BE28",
  col_imp_cumulative = "#427730",
  geom_bar_control = list(fill = "#69BE28"),
  ...
)
```

Arguments

`x`	a `importance_plot` object
`plot`	boolean to determine if the plot is displayed, or just returned
`nvar`	the number of variables to plot in order of importance
`col_imp_alone`	the color used for the variance explained by each variable alone
`col_imp_cumulative`	the color used for the cumulative variance explained
`geom_bar_control`	list of arguments to control the plotting of `ggplot2::geom_bar`
`...`	future arguments

Value

the plot

Examples

```
gtest <- lm(mpg ~ cyl + wt + hp + gear + carb, data = mtcars)
gtestreduced <- lm(mpg ~ 1, data = mtcars)
imp <- importance(gtest, gtestreduced)
plot(imp)

gtest <- survival::survreg(survival::Surv(futime, fustat) ~ ecog.ps*rx + age,
                           data = survival::ovarian,
                           dist = "weibull")
imp <- importance(gtest, survival::ovarian, nperm = 50)
plot(imp)
```

`plot.tornado_plot` *Plot a Tornado Plot object*

Description

Plot a Tornado Plot object

Usage

```
## S3 method for class 'tornado_plot'
plot(
  x,
  plot = TRUE,
  nvar = NA,
  xlabel = "Model Response",
  sensitivity_colors = c("grey", "#69BE28"),
  geom_bar_control = list(width = NULL),
  geom_point_control = list(fill = "black", col = "black"),
  ...
)
```

Arguments

`x`	a `tornado_plot` object
`plot`	boolean to determine if the plot is displayed, or just returned
`nvar`	the number of variables to plot
`xlabel`	a label for the x-axis
`sensitivity_colors`	a two element character vector of the bar colors for a lower value and upper value
`geom_bar_control`	a list of `ggplot2::geom_bar` options
`geom_point_control`	a list of `ggplot2::geom_point`
`...`	future arguments

Value

the plot

Examples

```
gtest <- lm(mpg ~ cyl*wt*hp, data = mtcars)
tp <- tornado(gtest, type = "PercentChange", alpha = 0.10, xlabel = "MPG")
plot(tp)
```

print.importance_plot *print data in an* importance_plot

Description

print data in an importance_plot

Usage

```
## S3 method for class 'importance_plot'
print(x, ...)
```

Arguments

x	the object to be printed
...	further arguments passed to print.data.frame

Examples

```
gtest <- glm(vs ~ wt + disp + gear, data=mtcars, family=binomial(link="logit"))
gtestreduced <- glm(vs ~ 1, data=mtcars, family=binomial(link="logit"))
g <- importance(gtest, gtestreduced)
print(g)
```

print.tornado_plot *print data in a* tornado_plot

Description

print data in a tornado_plot

Usage

```
## S3 method for class 'tornado_plot'
print(x, ...)
```

Arguments

x	the object to be printed
...	further arguments passed to print.data.frame

Examples

```
gtest <- lm(mpg ~ cyl*wt*hp, data = mtcars)
tp <- tornado(gtest, type = "PercentChange", alpha = 0.10, xlabel = "MPG")
print(tp)
```

quantile.ordered *Quantile for Ordered Factors*

Description

Quantile for Ordered Factors

Usage

```
## S3 method for class 'ordered'
quantile(x, probs = seq(0, 1, 0.25), ...)
```

Arguments

x	an ordered factor
probs	the desired quatiles
...	arugments passed on

Value

ordered factor levels at the desired quantiles

Examples

```
quantile(ordered(rep(c("C","B","A"), each=30), levels=c("C","B","A")),
         probs <- seq(0, 1, 0.25))
```

tornado *Generic tornado plotting method*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
tornado(model, type, alpha, dict, ...)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

See Also

tornado.lm, tornado.glm, tornado.cv.glmnet, tornado.survreg, tornado.coxph, tornado.train

---

tornado.coxph *Cox Proportional Hazards Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'coxph'
tornado(model, type = "PercentChange", alpha = 0.1, dict = NA, modeldata, ...)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.
modeldata	the data used to fit the model
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

Examples

```
gtest <- survival::coxph(survival::Surv(stop, event) ~ rx + size + number,
                         survival::bladder)
torn <- tornado(gtest, modeldata = survival::bladder, type = "PercentChange",
            alpha = 0.10)
plot(torn, xlabel = "Risk")
```

---

tornado.cv.glmnet *GLMNET Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'cv.glmnet'
tornado(
  model,
  type = "PercentChange",
  alpha = 0.1,
  dict = NA,
  modeldata,
  form,
  s = "lambda.1se",
  ...
)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.
modeldata	the raw data used to fit the glmnet model
form	the model formula
s	Value(s) of the penalty parameter lambda at which predictions are required. Default is the value s="lambda.1se" stored on the CV object. Alternatively s="lambda.min" can be used. If s is numeric, it is taken as the value(s) of lambda to be used.
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

See Also

tornado

Examples

```
if (requireNamespace("glmnet", quietly = TRUE))
{
  form <- formula(mpg ~ cyl*wt*hp)
  mf <- model.frame(form, data = mtcars)
```

```
 mm <- model.matrix(form, data = mf)
 gtest <- glmnet::cv.glmnet(x = mm, y= mtcars$mpg, family = "gaussian")
torn <- tornado(gtest, modeldata = mtcars, form = formula(mpg ~ cyl*wt*hp), s = "lambda.1se",
                type = "PercentChange", alpha = 0.10)
 plot(torn, xlabel = "MPG")
}
```

`tornado.glm` *GLM Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'glm'
tornado(model, type = "PercentChange", alpha = 0.1, dict = NA, ...)
```

Arguments

`model`	a model object
`type`	`PercentChange`, `percentiles`, `ranges`, or `StdDev`
`alpha`	the level of change, the percentile level, or the number of standard deviations
`dict`	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements `old` and `new`. The `old` element must contain each variable in the model.
`...`	further arguments, not used

Value

a `tornado_plot` object

`type`	the type of tornado plot
`data`	the data required for the plot
`family`	the model family if available

See Also

tornado

Examples

```
gtest <- glm(mpg ~ cyl*wt*hp, data = mtcars, family = gaussian)
torn <- tornado(gtest, type = "PercentChange", alpha = 0.10)
plot(torn, xlabel = "MPG")
```

---

tornado.lm *Linear Model Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'lm'
tornado(model, type = "PercentChange", alpha = 0.1, dict = NA, ...)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

See Also

tornado

Examples

```
gtest <- lm(mpg ~ cyl*wt*hp, data = mtcars)
torn <- tornado(gtest, type = "PercentChange", alpha = 0.10)
plot(torn, xlabel = "MPG")
```

---

tornado.survreg *Survreg Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'survreg'
tornado(model, type = "PercentChange", alpha = 0.1, dict = NA, modeldata, ...)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.
modeldata	the data used to fit the model
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

See Also

tornado

Examples

```
gtest <- survival::survreg(survival::Surv(futime, fustat) ~ ecog.ps + rx,
                           survival::ovarian,
                           dist='weibull', scale=1)
torn <- tornado(gtest, modeldata = survival::ovarian, type = "PercentChange",
             alpha = 0.10, xlabel = "futime")
plot(torn, xlabel = "Survival Time")
```

---

tornado.train *Caret Tornado Diagram*

Description

A tornado plot is a visualization of the range of outputs expected from a variety of inputs, or alternatively, the sensitivity of the output to the range of inputs. The center of the tornado is plotted at the response expected from the mean of each input variable. For a given variable, the width of the tornado is determined by the range of the variable, a multiplicative factor of the variable, or a quantile of the variable. Variables are ordered vertically with the widest bar at the top and narrowest at the bottom. Only one variable is moved from its mean value at a time. Factors or categorical variables have also been added to these plots by plotting dots at the resulting output as each factor is varied through all of its levels. The base factor level is chosen as the input variable for the center of the tornado.

Usage

```
## S3 method for class 'train'
tornado(
  model,
  type = "PercentChange",
  alpha = 0.1,
  dict = NA,
  class_number = NA,
  ...
)
```

Arguments

model	a model object
type	PercentChange, percentiles, ranges, or StdDev
alpha	the level of change, the percentile level, or the number of standard deviations
dict	a dictionary to translate variables for the plot. The dictionary must be a list or data.frame with elements old and new. The old element must contain each variable in the model.

class_number	for classification models, which number of the class that will be plotted
...	further arguments, not used

Value

a tornado_plot object

type	the type of tornado plot
data	the data required for the plot
family	the model family if available

See Also

tornado

Examples

```
if (requireNamespace("caret", quietly = TRUE) &
    requireNamespace("randomForest", quietly = TRUE))
{
  gtest <- caret::train(x = subset(mtcars, select = -mpg), y = mtcars$mpg, method = "rf")
  torn <- tornado(gtest, type = "PercentChange", alpha = 0.10)
  plot(torn, xlabel = "MPG")
}
```

Index