Package ‘coefplot’

August 29, 2016

Type Package
Title Plots Coefficients from Fitted Models
Version 1.2.4
Date 2016-01-09
Author Jared P. Lander
Maintainer Jared P. Lander <packages@jaredlander.com>
Description Plots the coefficients from model objects. This very quickly shows the user the point estimates and confidence intervals for fitted models.
License BSD_3_clause + file LICENSE
LazyLoad yes
Depends ggplot2 (>= 2.0.0)
Imports plyr, reshape2, useful, stats
Enhances glmnet, maxLik
ByteCompile TRUE
Suggests testthat
RoxygenNote 5.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2016-01-10 09:03:26

R topics documented:

buildModelCI ................................................................. 2
buildModelCI.default .................................................... 3
buildPlotting.default .................................................... 5
coeffplot ................................................................. 7
coeffplot.data.frame ..................................................... 8
coeffplot.default ......................................................... 10
coeffplot.glm ............................................................... 12
coeffplot.lm ............................................................... 13
Description

Construct Confidence Interval Values

Usage

buildModelCI(model, ...)

Arguments

model

A Fitted model such as from lm, glm

... Arguments passed on onto other methods
Details

Takes a model and builds a data.frame holding the coefficient value and the confidence interval values.

Value

A data.frame listing coefficients and confidence bands.

Author(s)

Jared P. Lander

See Also

coefplot multiplot

Examples

data(diamonds)
model1 <- lm(price ~ carat + cut, data=diamonds)
coefplot:::buildModelCI(model1)
coefplot(model1)

buildModelCI.default  buildModelCI.default

Description

Construct Confidence Interval Values

Usage

## Default S3 method:
buildModelCI(model, outerCI = 2, innerCI = 1,
intercept = TRUE, numeric = FALSE, sort = c("natural", "magnitude",
"alphabetical"), predictors = NULL, strict = FALSE, coefficients = NULL,
newNames = NULL, decreasing = TRUE, name = NULL,
interceptName = "(Intercept)", ...)

Arguments

model  A Fitted model such as from lm, glm
outerCI  How wide the outer confidence interval should be, normally 2 standard deviations. If 0, then there will be no outer confidence interval.
innerCI  How wide the inner confidence interval should be, normally 1 standard deviation. If 0, then there will be no inner confidence interval.
intercept  logical; Whether the Intercept coefficient should be plotted
numeric   logical; If true and factors has exactly one value, then it is displayed in a horizontal graph with continuous confidence bounds.; not used for now.
sort      Determines the sort order of the coefficients. Possible values are c("natural", "magnitude", "alphabetical")
predictors A character vector specifying which variables to keep. Each individual variable has to be specified, so individual levels of factors must be specified. We are working on making this easier to implement, but this is the only option for now.
strict    If TRUE then predictors will only be matched to its own coefficients, not its interactions
coefficients A character vector specifying which factor variables to keep. It will keep all levels and any interactions, even if those are not listed.
newNames  Named character vector of new names for coefficients
decreasing logical; Whether the coefficients should be ascending or descending
name      A name for the model, if NULL the call will be used
interceptName  Specifies name of intercept it case it is not the default of "(Intercept)".
...       See Details for information on factors, only and shorten

Details
Takes a model and builds a data.frame holding the coefficient value and the confidence interval values.

Value
A data.frame listing coefficients and confidence bands.

Author(s)
Jared P. Lander

See Also
ccoefplot multiplot

Examples

data(diamonds, package='ggplot2')
model1 <- lm(price ~ carat + cut, data=diamonds)
coefplot:::buildModelCI(model1)
coefplot(model1)
Description

Build ggplot object for coefplot

Usage

buildPlotting.default(modelCI, title = "Coefficient Plot", xlab = "Value", ylab = "Coefficient", lwdInner = 1, lwdOuter = 0, pointSize = 3, color = "blue", cex = 0.8, textAngle = 0, numberAngle = 0, shape = 16, linetype = 1, outerCI = 2, innerCI = 1, multi = FALSE, zeroColor = "grey", zeroLWD = 1, zeroType = 2, numeric = FALSE, fillColor = "grey", alpha = 1/2, horizontal = FALSE, facet = FALSE, scales = "free", value = "Value", coefficient = "Coefficient", errorHeight = 0, dodgeHeight = 1)

Arguments

modelCI An object created by buildModelCI
title The name of the plot, if NULL then no name is given
xlab The x label
ylab The y label
lwdInner The thickness of the inner confidence interval
lwdOuter The thickness of the outer confidence interval
pointSize Size of coefficient point
color The color of the points and lines
cex The text size multiplier, currently not used
textAngle The angle for the coefficient labels, 0 is horizontal
numberAngle The angle for the value labels, 0 is horizontal
shape The shape of the points
linetype The linetype of the error bars
outerCI How wide the outer confidence interval should be, normally 2 standard deviations. If 0, then there will be no outer confidence interval.
innerCI How wide the inner confidence interval should be, normally 1 standard deviation. If 0, then there will be no inner confidence interval.
multi logical; If this is for multiplot then leave the colors as determined by the legend, if FALSE then make all colors the same
zeroColor The color of the line indicating 0
zeroLWD The thickness of the 0 line
zeroType  The type of 0 line, 0 will mean no line
numeric   logical; If true and factors has exactly one value, then it is displayed in a horizontal graph with continuous confidence bounds.
fillColor The color of the confidence bounds for a numeric factor
alpha     The transparency level of the numeric factor’s confidence bound
horizontal logical; If the plot should be displayed horizontally
facet     logical; If the coefficients should be faceted by the variables, numeric coefficients (including the intercept) will be one facet
scales    The way the axes should be treated in a faceted plot. Can be c("fixed", "free", "free_x", "free_y")
value     Name of variable for value metric
coefficient Name of variable for coefficient names
errorHeight Height of error bars
dodgeHeight Amount of vertical dodging

Details

This function builds up the ggplot layer by layer for coefplot.lm

Value

a ggplot graph object

Author(s)

Jared P. Lander www.jaredlander.com

See Also

c coefplot.default coefplot multiplot

Examples

data(diamonds)
model1 <- lm(price ~ carat + cut, data=diamonds)
theCI <- coefplot:::buildModelCI(model1)
coefplot:::buildPlotting.default(theCI)
coefplot(model1)
Description

Provides an S3 generic method for plotting coefficients from a model so it can be extended to other model types.

A graphical display of the coefficients and standard errors from a fitted model

Usage

coefplot(model, ...)

Arguments

model The fitted model with coefficients to be plotted
...

See coefplot.lm for argument details

Details

Currently, methods are available for lm, glm and rxLinMod objects.

coefplot is the S3 generic method for plotting the coefficients from a fitted model.

This can be extended with new methods for other types of models not currently available.

A future iteration of coefplot.glm will also allow for plotting the coefficients on the transformed scale.

See coefplot.lm for specific documentation and the return value.

Value

A ggplot2 object or data.frame. See details in coefplot.lm for more information

Author(s)

Jared P. Lander

See Also

coefplot.lm
Examples

data(diamonds)
head(diamonds)
model1 <- lm(price ~ carat + cut*color, data=diamonds)
model2 <- lm(price ~ carat*color, data=diamonds)
model3 <- glm(price > 10000 ~ carat*color, data=diamonds)
coefplot(model1)
coefplot(model2)
coefplot(model3)
coefplot(model1, predictors="color")
coefplot(model1, predictors="color", strict=TRUE)
coefplot(model1, coefficients=c("(Intercept)", "color.Q"))
coefplot(model1, predictors="cut", coefficients=c("(Intercept)", "color.Q"), strict=TRUE)
coefplot(model1, predictors="cut", coefficients=c("(Intercept)", "color.Q"), strict=FALSE)
coefplot(model1, predictors="cut", coefficients=c("(Intercept)", "color.Q"), strict=TRUE, newNames=c(color.Q="Color", "cut4="Fourth"))
coefplot(model1, predictors=c("(Intercept)", "carat"), newNames=c(carat="Size"))
coefplot(model1, predictors=c("(Intercept)", "carat"),
newNames=c(carat="Size", "(Intercept)"="Constant"))

description

Dotplot for coefficients

Usage

## S3 method for class 'data.frame'
coefplot(model, title = "Coefficient Plot",
         xlab = "Value", ylab = "Coefficient", lwdInner = 1, lwdOuter = 0,
         pointSize = 3, color = "blue", cex = 0.8, textAngle = 0,
         numberAngle = 0, shape = 16, linetype = 1, outerCI = 2, innerCI = 1,
         multi = FALSE, zeroColor = "grey", zeroLWD = 1, zeroType = 2,
         numeric = FALSE, fillColor = "grey", alpha = 1/2, horizontal = FALSE,
         facet = FALSE, scales = "free", value = "Value",
         coefficient = "Coefficient", errorHeight = 0, dodgeHeight = 1, ...)

Arguments

model A data.frame like that built from coefplot(..., plot=FALSE)
title The name of the plot, if NULL then no name is given
xlab The x label
ylab The y label
coefplot.data.frame

lwdInner The thickness of the inner confidence interval
lwdOuter The thickness of the outer confidence interval
pointSize Size of coefficient point
color The color of the points and lines
cex The text size multiplier, currently not used
textAngle The angle for the coefficient labels, 0 is horizontal
numberAngle The angle for the value labels, 0 is horizontal
shape The shape of the points
linetype The linetype of the error bars
outerCI How wide the outer confidence interval should be, normally 2 standard deviations. If 0, then there will be no outer confidence interval.
innerCI How wide the inner confidence interval should be, normally 1 standard deviation. If 0, then there will be no inner confidence interval.
multi logical; If this is for multiplot then leave the colors as determined by the legend, if FALSE then make all colors the same
zeroColor The color of the line indicating 0
zeroLWD The thickness of the 0 line
zeroType The type of 0 line, 0 will mean no line
numeric logical; If true and factors has exactly one value, then it is displayed in a horizontal graph with continuous confidence bounds.
fillColor The color of the confidence bounds for a numeric factor
alpha The transparency level of the numeric factor’s confidence bound
horizontal logical; If the plot should be displayed horizontally
facet logical; If the coefficients should be faceted by the variables, numeric coefficients (including the intercept) will be one facet
scales The way the axes should be treated in a faceted plot. Can be c("fixed", "free", "free_x", "free_y")
value Name of variable for value metric
coefficient Name of variable for coefficient names
errorHeight Height of error bars
dodgeHeight Amount of vertical dodging
... Further Arguments

Details

A graphical display of the coefficients and standard errors from a fitted model, this function uses a data.frame as the input.

Value

a ggplot graph object
Author(s)

Jared P. Lander

Examples

```r
data(diamonds)
head(diamonds)
model1 <- lm(price ~ carat + cut*color, data=diamonds)
model2 <- lm(price ~ carat*color, data=diamonds)
df1 <- coefplot(model1, plot=FALSE)
df2 <- coefplot(model2, plot=FALSE)
coefplot(df1)
coefplot(df2)
```

Description

Dotplot for coefficients

Usage

```r
## Default S3 method:
coefplot(model, title = "Coefficient Plot",
        xlab = "Value", ylab = "Coefficient", innerCI = 1, outerCI = 2,
        lwdInner = 1, lwdOuter = 0, pointSize = 3, color = "blue",
        shape = 16, cex = 0.8, textAngle = 0, numberAngle = 0,
        zeroColor = "grey", zeroLWD = 1, zeroType = 2, facet = FALSE,
        scales = "free", sort = c("natural", "magnitude", "alphabetical"),
        decreasing = FALSE, numeric = FALSE, fillColor = "grey", alpha = 1/2,
        horizontal = FALSE, factors = NULL, only = NULL, shorten = TRUE,
        intercept = TRUE, interceptName = "(Intercept)", coefficients = NULL,
        predictors = NULL, strict = FALSE, newNames = NULL, plot = TRUE, ...)
```

Arguments

- `model` The model to plot.
- `title` The name of the plot, if NULL then no name is given
- `xlab` The x label
- `ylab` The y label
- `innerCI` How wide the inner confidence interval should be, normally 1 standard deviation. If 0, then there will be no inner confidence interval.
- `outerCI` How wide the outer confidence interval should be, normally 2 standard deviations. If 0, then there will be no outer confidence interval.
`coefplot.default`

- `lwdInner`: The thickness of the inner confidence interval
- `lwdOuter`: The thickness of the outer confidence interval
- `pointSize`: Size of coefficient point
- `color`: The color of the points and lines
- `shape`: The shape of the points
- `cex`: The text size multiplier, currently not used
- `textAngle`: The angle for the coefficient labels, 0 is horizontal
- `numberAngle`: The angle for the value labels, 0 is horizontal
- `zeroColor`: The color of the line indicating 0
- `zeroLWD`: The thickness of the 0 line
- `zeroType`: The type of 0 line, 0 will mean no line
- `facet`: logical; If the coefficients should be faceted by the variables, numeric coefficients (including the intercept) will be one facet. Currently not available.
- `scales`: The way the axes should be treated in a faceted plot. Can be c("fixed", "free", "free_x", "free_y"). Currently not available.
- `sort`: Determines the sort order of the coefficients. Possible values are c("natural", "normal", "magnitude", "size", "alphabetical")
- `decreasing`: logical; Whether the coefficients should be ascending or descending
- `numeric`: logical; If true and factors has exactly one value, then it is displayed in a horizontal graph with continuous confidence bounds. Currently not available.
- `fillColor`: The color of the confidence bounds for a numeric factor. Currently not available.
- `alpha`: The transparency level of the numeric factor’s confidence bound. Currently not available.
- `horizontal`: logical; If the plot should be displayed horizontally. Currently not available.
- `factors`: Vector of factor variables that will be the only ones shown
- `only`: logical; If factors has a value this determines how interactions are treated. True means just that variable will be shown and not its interactions. False means interactions will be included.
- `shorten`: logical or character; If FALSE then coefficients for factor levels will include their variable name. If TRUE coefficients for factor levels will be stripped of their variable names. If a character vector of variables only coefficients for factor levels associated with those variables will the variable names stripped. Currently not available.
- `intercept`: logical; Whether the Intercept coefficient should be plotted
- `interceptName`: Specifies name of intercept it case it is not the default of "(Intercept)".
- `coefficients`: A character vector specifying which factor coefficients to keep. It will keep all levels and any interactions, even if those are not listed.
- `predictors`: A character vector specifying which coefficients to keep. Each individual coefficient can be specified. Use predictors to specify entire factors.
- `strict`: If TRUE then predictors will only be matched to its own coefficients, not its interactions
newNames  Named character vector of new names for coefficients
plot     logical; If the plot should be drawn, if false then a data.frame of the values will be returned
...      Arguments passed on to other functions

Details
A graphical display of the coefficients and standard errors from a fitted model
coeffplot is the S3 generic method for plotting the coefficients from a fitted model.
This method also plots coefficients from glm (using coefplot.lm) and rxLinMod models (through a redirection from coefplot.rxLinMod)

Value
If plot is TRUE then a ggplot object is returned. Otherwise a data.frame listing coefficients and confidence bands is returned.

Author(s)
Jared P. Lander

See Also
lm glm ggplot coefplot plotcoef

Examples

data(diamonds)
head(diamonds)
model1 <- lm(price ~ carat + cut*color, data=diamonds)
model2 <- lm(price ~ carat*color, data=diamonds)
coefplot(model1)
coefplot(model2)
coefplot(model1, predictors="color")
coefplot(model1, predictors="color", strict=TRUE)
coefplot(model1, coefficients=c("(Intercept)", "color.Q"))
**Usage**

```r
## S3 method for class 'glm'
coefplot(...)```

**Arguments**

... All arguments are passed on to `coefplot.default`. Please see that function for argument information.

**Details**

A graphical display of the coefficients and standard errors from a fitted glm model `coefplot` is the S3 generic method for plotting the coefficients from a fitted model. For more information on this function and its arguments see `coefplot.default`.

**Value**

A ggplot object. See `coefplot.lm` for more information.

**Author(s)**

Jared P. Lander

**Examples**

```r
model2 <- glm(price > 10000 ~ carat + cut*color, data=diamonds, family=binomial(link="logit"))
coefplot(model2)
```

---

**Description**

Dotplot for lm coefficients

**Usage**

```r
## S3 method for class 'lm'
coefplot(...)```

**Arguments**

... All arguments are passed on to `coefplot.default`. Please see that function for argument information.
Details

A graphical display of the coefficients and standard errors from a fitted lm model.

`coefplot` is the S3 generic method for plotting the coefficients from a fitted model.

For more information on this function and it's arguments see `coefplot.default`

Value

A ggplot object. See `coefplot.lm` for more information.

Author(s)

Jared P. Lander

Examples

```r
model1 <- lm(price ~ carat + cut*color, data=diamonds)
coefplot(model1)
```

Description

Dotplot for rxGlm coefficients

Usage

```r
## S3 method for class 'rxGlm'
coefplot(...)
```

Arguments

`...` All arguments are passed on to `coefplot.default`. Please see that function for argument information.

Details

A graphical display of the coefficients and standard errors from a fitted rxGlm model.

`coefplot` is the S3 generic method for plotting the coefficients from a fitted model.

For more information on this function and it's arguments see `coefplot.default`

Value

A ggplot object. See `coefplot.lm` for more information.
Author(s)
Jared P. Lander

Examples

## Not run:
mod4 <- rxGlm(price ~ carat + cut + x, data=diamonds)
mod5 <- rxGlm(price > 10000 ~ carat + cut + x, data=diamonds, family="binomial")
coefplot(mod4)
coefplot(mod5)

## End(Not run)

Description
Dotplot for rxLinMod coefficients

Usage

## S3 method for class 'rxLinMod'
coefplot(...)

Arguments

... All arguments are passed on to coefplot.lm. Please see that function for argument information.

Details
A graphical display of the coefficients and standard errors from a fitted rxLinMod model

coefplot is the S3 generic method for plotting the coefficients from a fitted model.

For more information on this function and its arguments see coefplot.lm

Value

A ggplot object. See coefplot.lm for more information.

Author(s)
Jared P. Lander www.jaredlander.com
Examples

```r
## Not run:
data(diamonds)
mod3 <- rxLinMod(price ~ carat + cut + x, data=diamonds)
coefplot(mod3)

## End(Not run)
```

Description

Dotplot for rxLogit coefficients

Usage

```r
## S3 method for class 'rxLogit'
coefplot(...)
```

Arguments

... All arguments are passed on to `coefplot.lm`. Please see that function for argument information.

Details

A graphical display of the coefficients and standard errors from a fitted rxLogit model

`coefplot` is the S3 generic method for plotting the coefficients from a fitted model.

For more information on this function and its arguments see `coefplot.lm`

Value

A ggplot object. See `coefplot.lm` for more information.

Author(s)

Jared P. Lander www.jaredlander.com

Examples

```r
## Not run:
data(diamonds)
mod6 <- rxLogit(price > 10000 ~ carat + cut + x, data=diamonds)
coefplot(mod6)

## End(Not run)
```
doRegex

description
Helper function for matching coefficients

usage
DoRegex(x, matchAgainst, pattern = "(\^[| ]%s($|,|=)\")"

arguments
x Root pattern to search for
matchAgainst Text to search through
pattern Regex pattern to build x into

details
Only used by getCoefsFromPredictorsRevo for finding matches between predictors and coefficients

value
A list of indices of matchAgainst that is matched

Author(s)
Jared P. Lander

extract.coef

description
Extract Coefficient Information from glm Models

usage
extract.coef(model, ...)

arguments
model Model object to extract information from.
... Further arguments
Details

Gets the coefficient values and standard errors, and variable names from a glm model.

Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

```r
## Not run:
require(ggplot2)
data(diamonds)
library(coefplot)
mod1 <- lm(price ~ carat + cut + x, data=diamonds)
mod2 <- glm(price > 10000 ~ carat + cut + x, data=diamonds, family=binomial(link="logit"))
mod3 <- lm(price ~ carat*cut + x, data=diamonds)
extract.coef(mod1)
extract.coef(mod2)
extract.coef(mod3)

mod4 <- rxLinMod(price ~ carat*cut + x, diamonds)

## End(Not run)
```

---

**Description**

Extract Coefficient Information from Models

**Usage**

```r
## S3 method for class 'cv.glmnet'
extract.coef(model, lambda = "lambda.min", ...)
```

**Arguments**

- `model` Model object from which to extract information.
- `lambda` Value of penalty parameter. Can be either a numeric value or one of "lambda.min" or "lambda.1se"
- `...` Further arguments
extract.coef.default

Details

Gets the coefficient values and variable names from a model. Since glmnet does not have standard errors, those will just be NA.

Value

A `data.frame` containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

```r
# Not run:
library(glmnet)
library(ggplot2)
library(useful)
data(diamonds)
diaX <- build.x(price ~ carat + cut + x - 1, data=diamonds, contrasts = TRUE)
diaY <- build.y(price ~ carat + cut + x - 1, data=diamonds)
modG1 <- cv.glmnet(x=diaX, y=diaY, k=5)
extract.coef(modG1)
```

```
extract.coef.default  extract.coef.default
```

Description

Extract Coefficient Information from Models

Usage

```r
# Default S3 method:
extract.coef(model, ...)
```

Arguments

- `model`: Model object to extract information from.
- `...`: Further arguments

Details

Gets the coefficient values and standard errors, and variable names from a model.
Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

```r
## Not run:
require(ggplot2)
library(coefplot)
data(diamonds)
mod1 <- lm(price ~ carat + cut + x, data=diamonds)
extract.coef(mod1)
## End(Not run)
```

Description

Extract Coefficient Information from glm Models

Usage

```r
## S3 method for class 'glm'
extract.coef(model, ...)
```

Arguments

- `model`    Model object to extract information from.
- `...`      Further arguments

Details

Gets the coefficient values and standard errors, and variable names from a glm model.

Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander
Examples

```r
## Not run:
require(ggplot2)
data(diamonds)
library(coefplot)
mod2 <- glm(price > 10000 ~ carat + cut + x, data=diamonds, family=binomial(link="logit"))
extract.coef(mod2)

## End(Not run)
```

Description

Extract Coefficient Information from Models

Usage

```r
## S3 method for class 'glmnet'
extract.coef(model, lambda = stats::median(model$lambda), ...)
```

Arguments

- `model`: Model object from which to extract information.
- `lambda`: Value of penalty parameter
- `...`: Further arguments

Details

Gets the coefficient values and variable names from a model. Since glmnet does not have standard errors, those will just be NA.

Value

A `data.frame` containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander
Examples

```r
## Not run:
library(glmnet)
library(ggplot2)
library(useful)
data(diamonds)
diaX <- build.x(price ~ carat + cut + x - 1, data=diamonds, contrasts = TRUE)
diaY <- build.y(price ~ carat + cut + x - 1, data=diamonds)
modGl <- glmnet(x=diaX, y=diaY)
extract.coef(modGl)

## End(Not run)
```

Description

Extract Coefficient Information from lm Models

Usage

```r
## S3 method for class 'lm'
extract.coef(model, ...)
```

Arguments

- `model` Model object to extract information from.
- `...` Further arguments

Details

Gets the coefficient values and standard errors, and variable names from an lm model.

Value

A `data.frame` containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander
Examples

## Not run:
require(ggplot2)
data(diamonds)
library(coefplot)
mod1 <- lm(price ~ carat + cut + x, data=diamonds)
extract.coef(mod1)

## End(Not run)

---

extract.coef.maxLik  

Description

Extract Coefficient Information from Models

Usage

## S3 method for class 'maxLik'
extract.coef(model, ...)

Arguments

model  
Model object from which to extract information.

...  
Further arguments

Details

Gets the coefficient values and variable names from a model.

Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

## Not run:
library(maxLik)
loglik <- function(param) {
  mu <- param[1]
  sigma <- param[2]
  ll <- -0.5*N*log(2*pi) - N*log(sigma) - sum(0.5*(x - mu)^2/sigma^2)
}
extract.coef.rxGlm

Description

Extract Coefficient Information from rxGlm Models

Usage

## S3 method for class 'rxGlm'
extract.coef(model, ...)

Arguments

model

Model object to extract information from.

... Further arguments

Details

Gets the coefficient values and standard errors, and variable names from an rxGlm model.

Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

## Not run:
require(ggplot2)
data(diamonds)
mod4 <- rxGlm(price - carat + cut + x, data=diamonds)
mod5 <- rxGlm(price > 10000 - carat + cut + x, data=diamonds, family="binomial")
extract.coef(mod4)
extract.coef(mod5)

## End(Not run)
Description

Extract Coefficient Information from rxLinMod Models

Usage

```r
## S3 method for class 'rxLinMod'
extract.coef(model, ...)
```

Arguments

- `model`: Model object to extract information from.
- `...`: Further arguments

Details

Gets the coefficient values and standard errors, and variable names from an rxLinMod model.

Value

A `data.frame` containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

```r
## Not run:
require(ggplot2)
data(diamonds)
mod3 <- rxLinMod(price ~ carat + cut + x, data=diamonds)
extract.coef(mod3)

## End(Not run)
```
extract.coef.rxLogit

Description

Extract Coefficient Information from rxLogit Models

Usage

## S3 method for class 'rxLogit'
extract.coef(model, ...)

Arguments

model Model object to extract information from.
...
Further arguments

Details

Gets the coefficient values and standard errors, and variable names from an rxLogit model.

Value

A data.frame containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

## Not run:
require(ggplot2)
data(diamonds)
mod6 <- rxLogit(price > 10000 ~ carat + cut + x, data=diamonds)
exttract.coef(mod6)

## End(Not run)
get.assign

Description
The assignment vector for a model

Usage
get.assign(model, ...)

Arguments
- model: Fitted model
- ...: Further arguments

Details
Gets relative positions of predictors

Value
The assignment vector

Author(s)
Jared P. Lander

glm

Description
The assignment vector for a glm model

Usage
## S3 method for class 'glm'
glm
get.assign(model, ...)

Arguments
- model: Fitted model
- ...: Further arguments
Details

Gets relative positions of predictors

Value

The assignment vector

Author(s)

Jared P. Lander

---

get.assign.lm

Description

The assignment vector for an lm model

Usage

```r
## S3 method for class 'lm'
get.assign(model, ...)
```

Arguments

- `model` Fitted model
- `...` Further arguments

Details

Gets relative positions of predictors

Value

The assignment vector

Author(s)

Jared P. Lander
getCoefsFromPredictors

Description
Generic function for finding which coefficients go with which predictors

Usage
getCoefsFromPredictors(model, predictors, ...)

Arguments
model A fitted model
predictors A character vector of predictors to match against
... further arguments

Details
The user specifies predictors whose coefficients should be included in the coefplot.

Value
A character vector of coefficients listing the coefficients that match the predictor

Author(s)
Jared P. Lander

getCoefsFromPredictors.default

Description
Default function (lm, glm) for matching coefficients with predictors

Usage
## Default S3 method:
getCoefsFromPredictors(model, predictors = NULL,
                         strict = FALSE, ...)

getCoefsFromPredictors

getCoefsFromPredictors
Arguments

model  A fitted model
predictors  A character vector of predictors to match against. Interactions can be explicitly specified by VariableA:VariableB.
strict  Logical specifying if interactions terms should be included (FALSE) or just the main terms (TRUE).
...  further arguments

Details

The user specifies predictors whose coefficients should be included in the coefplot.

Value

A character vector of coefficients listing the coefficients that match the predictor

Author(s)

Jared P. Lander

getCoefsFromPredictors.rxGlm

getCoefsFromPredictors.rxGlm

Description

Function for matching coefficients with predictors for rxGlm

Usage

## S3 method for class 'rxGlm'
getCoefsFromPredictors(model, predictors = NULL, strict = FALSE, ...)

Arguments

model  A fitted model
predictors  A character vector of predictors to match against
strict  Logical specifying if interactions terms should be included (FALSE) or just the main terms (TRUE).
...  further arguments

Details

The user specifies predictors whose coefficients should be included in the coefplot.
**Value**

A character vector of coefficients listing the coefficients that match the predictor

**Author(s)**

Jared P. Lander

```r
## S3 method for class 'rxLinMod'
getCoefsFromPredictors(model, predictors = NULL, strict = FALSE, ...)
```

**Arguments**

- `model`: A fitted model
- `predictors`: A character vector of predictors to match against
- `strict`: Logical specifying if interactions terms should be included (FALSE) or just the main terms (TRUE).
- `...`: further arguments

**Details**

The user specifies predictors whose coefficients should be included in the coefplot.

**Value**

A character vector of coefficients listing the coefficients that match the predictor

**Author(s)**

Jared P. Lander
Function for matching coefficients with predictors for rxLogit

Usage

```r
## S3 method for class 'rxLogit'
getCoefsFromPredictors(model, predictors = NULL,
                        strict = FALSE, ...)
```

Arguments

- `model`: A fitted model
- `predictors`: A character vector of predictors to match against
- `strict`: Logical specifying if interactions terms should be included (FALSE) or just the main terms (TRUE).
- `...`: Further arguments

Details

The user specifies predictors whose coefficients should be included in the coefplot.

Value

A character vector of coefficients listing the coefficients that match the predictor

Author(s)

Jared P. Lander

Function that does the work for Revo models for matching coefficients with predictors

Usage

```r
getCoefsFromPredictorsRevo(model, predictors = NULL, strict = FALSE, ...)
```
**matchCoefs**

**Arguments**
- `model` A fitted model
- `predictors` A character vector of predictors to match against
- `strict` Logical specifying if interactions terms should be included (FALSE) or just the main terms (TRUE).
- ... further arguments

**Details**
The user specifies predictors whose coefficients should be included in the coefplot.

**Value**
A character vector of coefficients listing the coefficients that match the predictor. As of now interactions cannot be explicitly specified.

**Author(s)**
Jared P. Lander
Examples

```r
## Not run:
require(reshape2)
require(plyr)
data("tips", package="reshape2")
mod1 <- lm(tip ~ total_bill * sex + day, tips)
mod2 <- lm(tip ~ total_bill * sex + day - 1, tips)
mod3 <- glm(tip ~ total_bill * sex + day, tips, family=gaussian(link="identity"))
mod4 <- lm(tip ~ (total_bill + sex + day)^3, tips)
mod5 <- lm(tip ~ total_bill * sex + day + I(total_bill^2), tips)
coefplot:::matchCoefs(mod1)
coefplot:::matchCoefs(mod2)
coefplot:::matchCoefs(mod3)
coefplot:::matchCoefs(mod4)
coefplot:::matchCoefs(mod5)
## End(Not run)
```

Description

Match coefficients to predictors

Usage

```r
## Default S3 method:
matchCoefs(model, ...)
```

Arguments

- `model`: Fitted model
- `...`: Further arguments

Details

Matches coefficients to predictors using information from model matrices

Value

A data.frame matching predictors to coefficients

Author(s)

Jared P. Lander
multiplot

Plot multiple coefplots

Description

Plot the coefficients from multiple models

Usage

multiplot(..., title = "Coefficient Plot", xlab = "Value", ylab = "Coefficient", innerCI = 1, outerCI = 2, lwdInner = 1, lwdOuter = 0, pointSize = 3, dodgeHeight = 1, color = "blue", shape = 16, linetype = 1, cex = 0.8, textAngle = 0, numberAngle = 90, zeroColor = "grey", zeroLWD = 1, zeroType = 2, single = TRUE, scales = "fixed", ncol = length(unique(modelCI$Model)), sort = c("natural", "normal", "magnitude", "size", "alphabetical"), decreasing = FALSE, names = NULL, numeric = FALSE, fillColor = "grey", alpha = 1/2, horizontal = FALSE, factors = NULL, only = NULL, shorten = TRUE, intercept = TRUE, interceptName = "(Intercept)", coefficients = NULL, predictors = NULL, strict = FALSE, newNames = NULL, plot = TRUE, drop = FALSE, by = c("Coefficient", "Model"), plot.shapes = FALSE, plot.linetypes = FALSE, legend.position = c("right", "left", "bottom", "top", "none"), secret.weapon = FALSE, legend.reverse = FALSE)

Arguments

title The name of the plot, if NULL then no name is given
xlab The x label
ylab The y label
innerCI How wide the inner confidence interval should be, normally 1 standard deviation. If 0, then there will be no inner confidence interval.
outerCI How wide the outer confidence interval should be, normally 2 standard deviations. If 0, then there will be no outer confidence interval.
lwdInner The thickness of the inner confidence interval
lwdOuter The thickness of the outer confidence interval
pointSize Size of coefficient point
dodgeHeight Amount of vertical dodging
color The color of the points and lines
shape The shape of the points
linetype The type of line drawn for the standard errors
cex The text size multiplier, currently not used
textAngle The angle for the coefficient labels, 0 is horizontal
numberAngle

The angle for the value labels, 0 is horizontal

zeroColor

The color of the line indicating 0

zeroLWD

The thickness of the 0 line

zeroType

The type of 0 line, 0 will mean no line

single

logical; If TRUE there will be one plot with the points and bars stacked, otherwise the models will be displayed in separate facets

scales

The way the axes should be treated in a faceted plot. Can be c("fixed", "free", "free_x", "free_y")

ncol

The number of columns that the models should be plotted in

sort

Determines the sort order of the coefficients. Possible values are c("natural", "magnitude", "alphabetical")

decreasing

logical; Whether the coefficients should be ascending or descending

names

Names for models, if NULL then they will be named after their inputs

numeric

logical; If true and factors has exactly one value, then it is displayed in a horizontal graph with continuous confidence bounds.

fillColor

The color of the confidence bounds for a numeric factor

alpha

The transparency level of the numeric factor’s confidence bound

horizontal

logical; If the plot should be displayed horizontally

factors

Vector of factor variables that will be the only ones shown

only

logical; If factors has a value this determines how interactions are treated. True means just that variable will be shown and not its interactions. False means interactions will be included.

shorten

logical or character; If FALSE then coefficients for factor levels will include their variable name. If TRUE coefficients for factor levels will be stripped of their variable names. If a character vector of variables only coefficients for factor levels associated with those variables will the variable names stripped.

intercept

logical; Whether the Intercept coefficient should be plotted

interceptName

Specifies name of intercept it case it is not the default of "(Intercept)".

coefficients

A character vector specifying which factor coefficients to keep. It will keep all levels and any interactions, even if those are not listed.

predictors

A character vector specifying which coefficients to keep. Each individual coefficient can be specified. Use predictors to specify entire factors

strict

If TRUE then predictors will only be matched to its own coefficients, not its interactions

newNames

Named character vector of new names for coefficients

plot

logical; If the plot should be drawn, if false then a data.frame of the values will be returned

drop

logical; if TRUE then models without valid coefficients to show will not be plotted
**multiplot**

by If "Coefficient" then anormal multiplot is plotted, if "Model" then the coefficients are plotted along the axis with one for each model. If plotting by model only one coefficient at a time can be selected. This is called the secret weapon by Andy Gelman.

plot.shapes If TRUE points will have different shapes for different models

plot.linetypes If TRUE lines will have different shapes for different models

legend.position position of legend, one of "left", "right", "bottom", "top", "none"

secret.weapon If this is TRUE and exactly one coefficient is listed in coefficients then Andy Gelman’s secret weapon is plotted.

legend.reverse Setting to reverse the legend in a multiplot so that it matches the order they are drawn in the plot

... Models to be plotted

**Details**

Plots a graph similar to coefplot but for multiple plots at once.

For now, if names is provided the plots will appear in alphabetical order of the names. This will be adjusted in future iterations. When setting by to "Model" and specifying exactly one variable in variables that one coefficient will be plotted repeatedly with the axis labeled by model. This is Andy Gelman’s secret weapon.

**Value**

A ggplot object

**See Also**

link{coefplot}

**Examples**

data(diamonds)
model1 <- lm(price ~ carat + cut, data=diamonds)
model2 <- lm(price ~ carat + cut + color, data=diamonds)
model3 <- lm(price ~ carat + color, data=diamonds)

multiplot(model1, model2, model3)
multiplot(model1, model2, model3, single=FALSE)
multiplot(model1, model2, model3, plot=FALSE)
require(reshape2)
data(tips, package="reshape2")
mod1 <- lm(tip ~ total_bill + sex, data=tips)
mod2 <- lm(tip ~ total_bill * sex, data=tips)
mod3 <- lm(tip ~ total_bill * sex * day, data=tips)
mod7 <- lm(tip ~ total_bill + day + time, data=tips)
multiplot(mod1, mod2, mod3, mod7, single=FALSE, scales="free_x")
multiplot(mod1, mod2, mod3, mod7, single=FALSE, scales="free_x")
multiplot(mod1, mod2, mod3, mod7, single=FALSE, scales="free_x", plot.shapes=TRUE)
position_dodgev

Adjust position by dodging overlaps to the side.

Description

Adjust position by dodging overlaps to the side.

Usage

position_dodgev(height = NULL)

Arguments

height Dodging height, when different to the height of the individual elements. This is useful when you want to align narrow geoms with wider geoms. See the examples for a use case.

Examples

ggplot(mtcars, aes(factor(cyl), fill = factor(vs))) + geom_bar(position = "dodge")

ggplot(diamonds, aes(price, fill = cut)) + geom_histogram(position="dodge")

# see ?geom_boxplot and ?geom_bar for more examples

# To dodge items with different heights, you need to be explicit
df <- data.frame(x=c("a","a","b","b"), y=2:5, g = rep(1:2, 2))
p <- ggplot(df, aes(x, y, group = g)) + geom_bar(
  stat = "identity", position = "dodge",
  fill = "grey50", colour = "black"
)

p

# A line range has no height:
p + geom_linerange(aes(ymin = y-1, ymax = y+1), position = "dodge")

# You need to explicitly specify the height for dodging
p + geom_linerange(aes(ymin = y-1, ymax = y+1),
position = position_dodge(height = 0.9))

# Similarly with error bars:
p + geom_errorbar(aes(ymin = y-1, ymax = y+1), height = 0.2,
                   position = "dodge")
p + geom_errorbar(aes(ymin = y-1, ymax = y+1, height = 0.2),
                   position = position_dodge(height = 0.9))
Index

*Topic coefficients
  coefplot, 7
*Topic coefficient
  coefplot, 7
*Topic coefplot
  coefplot, 7
*Topic dotplot
  coefplot, 7
*Topic glm
  coefplot, 7
*Topic linear
  coefplot, 7
*Topic lm
  coefplot, 7
*Topic model
  coefplot, 7
*Topic rxLinMod
  coefplot, 7

evaluatecoef, 2
buildModelCI, 2, 5
buildModelCI.default, 3
buildPlotting.default, 5

dotplot, 12–16, 37
dotplot-package (coefplot), 7
dotplot.data.frame, 8
dotplot.default, 6, 10, 13, 14
dotplot.lm, 12
dotplot.rxGlm, 14
dotplot.rxLinMod, 15
dotplot.rxLogit, 16
data.frame, 3, 4, 12, 18–26
doRegex, 17

evaluatecoef, 17
evaluatecoef.cv.glmmnet, 18
evaluatecoef.default, 19
evaluatecoef.lm, 20

evaluatecoef.glmmnet, 21
evaluatecoef.lm, 22
evaluatecoef.maxLik, 23
evaluatecoef.rxGlm, 24
evaluatecoef.rxLinMod, 25
evaluatecoef.rxLogit, 26

glm, 12

glm, 12

matchCoefs, 33
matchCoefs.default, 34

multiplot, 3–6, 9, 34

plotcoef, 12
plotcoef(coefplot), 7

position_dodgev, 38