Package ‘coefplot’

January 14, 2022

Type Package
Title Plots Coefficients from Fitted Models
Version 1.2.8
Date 2022-01-12
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, dplyr (>= 0.6.0), dygraphs, tibble, magrittr, purrr, plotly
ByteCompile TRUE
Suggests testthat (>= 2.0.0), covr, glmnet, maxLik, xgboost, workflows, parsnip, knitr, rmarkdown
Encoding UTF-8
RoxygenNote 7.1.2
NeedsCompilation no
Repository CRAN
Date/Publication 2022-01-14 09:42:47 UTC

R topics documented:

annotateSeries ............................................................... 2
buildModelCI ................................................................. 3
buildModelCI.default ....................................................... 4
buildPlotting.default ...................................................... 6
buildPlottingPloty.default ................................................. 8
coeffpath .................................................................. 10
annotateSeries

Description

Annotate a series

Usage

annotateSeries(
  dygraph,
  series,
  x = 0,
  text = series,
  tooltip = series,
  width = 50,
Arguments

- **dygraph**: Dygraph to add an annotation to.
- **series**: Series to attach the annotation to. By default, the last series defined using `dySeries`.
- **x**: Either numeric or date value indicating where to place the annotation. For date value, this should be of class POSIXct or convertible to POSIXct.
- **text**: Text to overlay on the chart at the location of `x`.
- **tooltip**: Additional tooltip text to display on mouse hover.
- **width**: Width (in pixels) of the annotation flag.
- **...**: Further arguments passed to `link[dygraphs]{dyAnnotation}`.

Details

A helper function that changes the order of some options for `link[dygraphs]{dyAnnotation}` so it is easier to use with `reduce`.

Author(s)

Jared P. Lander

Description

Construct Confidence Interval Values

Usage

```r
buildModelCI(model, ...)
```

Arguments

- **model**: A Fitted model such as from `lm`, `glm`.
- **...**: Arguments passed onto other methods.

Details

Takes a model and builds a `data.frame` holding the coefficient value and the confidence interval values.
Value
A \texttt{data.frame} listing coefficients and confidence bands.

Author(s)
Jared P. Lander

See Also
\texttt{coefplot\ multiplot}

Examples

\begin{verbatim}
data(diamonds)
model1 <- lm(price ~ carat + cut, data=diamonds)
coefplot:::buildModelCI(model1)
coefplot(model1)
\end{verbatim}

Description
Construct Confidence Interval Values

Usage

\begin{verbatim}
## 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, 
  trans = identity, 
  decreasing = TRUE, 
  name = NULL, 
  interceptName = "(Intercept)",
...
)
\end{verbatim}
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
- **trans**: A transformation function to apply to the values and confidence intervals. identity by default. Use `invlogit` for binary regression.
- **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

- `coefplot`
- `multiplot`
Examples

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

---

**buildPlotting.default  Coefplot plotting**

**Description**

Build ggplot object for coefplot

**Usage**

```r
buildPlotting.default(
  modelCI,
  title = "Coefficient Plot",
  xlab = "Value",
  ylab = "Coefficient",
  lwdInner = 1 + interactive * 2,
  lwdOuter = if (interactive) 1 else unname((Sys.info()
    [\"sysname\"] != "Windows") * 0.5),
  pointSize = 3 + interactive * 5,
  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
interactive If TRUE an interactive plot is generated instead of [ggplot2]
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

coeffplot.default coefplot multiplot

Examples

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

Description

Builds the plotting structure for interactive coefplots

Usage

buildPlottingPloty.default(
  modelCI,
  title = "Coefficient Plot",
  xlab = "Value",
  ylab = "Coefficient",
  lwdInner = 3,
  lwdOuter = 1,
  color = "blue",
  shape = "circle",
  pointSize = 8
)
Arguments

modelCI An object created by buildModelCI

Arguments

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
color The color of the points and lines
shape The shape of the points
pointSize Size of coefficient point

Details

Uses plotly to make an interactive version of coefplot. Still uses modelCI.

Value

a ggplot graph object

Author(s)

Jared P. Lander

See Also

coefplot.default coefplot buildPlotting.default

Examples

data(diamonds)
mod1 <- lm(price ~ carat + cut, data=diamonds)
theCI1 <- coefplot:::buildModelCI(mod1)
coefplot:::buildPlottingPloty.default(theCI1)
coefplot(mod1, interactive=TRUE)

mod2 <- lm(mpg ~ cyl + qsec - 1, data=mtcars)
mod3 <- lm(mpg ~ cyl + qsec + disp - 1, data=mtcars)
theCI2 <- coefplot:::buildModelCI(mod2)
theCI3 <- coefplot:::buildModelCI(mod3)
coefplot:::buildPlottingPloty.default(theCI2)
coefplot:::buildPlottingPloty.default(theCI3)
coefplot(mod2, interactive=TRUE)
coefplot(mod3, interactive=TRUE)

mod4 <- glmnet::glmnet(
x=as.matrix(diamonds[, c('carat', 'x', 'y', 'z')]),
y=diamonds$price
)
Description

Visualize the coefficient path resulting from the elastic net

Usage

coeffpath(model, ...)  
  # S3 method for class 'glmnet'
coeffpath(
    model,
    xlab = "Log Lambda",
    ylab = "Coefficients",
    showLegend = c("onmouseover", "auto", "always", "follow", "never"),
    annotate = TRUE,
    elementID = NULL,
    ...
  )
  
  # S3 method for class 'cv.glmnet'
coeffpath(
    model,
    xlab = "Log Lambda",
    ylab = "Coefficients",
    showLegend = c("onmouseover", "auto", "always", "follow", "never"),
    annotate = TRUE,
    colorMin = "black",
    strokePatternMin = "dotted",
    labelMin = "lambda.min",
    locMin = c("bottom", "top"),
    color1se = "black",
    strokePattern1se = "dotted",
    label1se = "lambda.1se",
    loc1se = c("bottom", "top"),
    ...
  )

Arguments

model            A glmnet model
...

Arguments passed on to extractPath
coefpath

- `xlab`: x-axis label
- `ylab`: y-axis label
- `showLegend`: When to display the legend. Specify "always" to always show the legend. Specify "onmouseover" to only display it when a user mouses over the chart. Specify "follow" to have the legend show as overlay to the chart which follows the mouse. The default behavior is "auto", which results in "always" when more than one series is plotted and "onmouseover" when only a single series is plotted.
- `annotate`: If TRUE (default) plot the name of the series
- `elementID`: Unique identified for dygraph, if NULL it will be randomly generated
- `colorMin`: Color for line showing lambda.min
- `strokePatternMin`: Stroke pattern for line showing lambda.min
- `labelMin`: Label for line showing lambda.min
- `locMin`: Location for line showing lambda.min, can be 'bottom' or 'top'
- `color1se`: Color for line showing lambda.1se
- `strokePattern1se`: Stroke pattern for line showing lambda.1se
- `label1se`: Label for line showing lambda.1se
- `loc1se`: Location for line showing lambda.1se, can be 'bottom' or 'top'

**Details**

This is a replacement plot for visualizing the coefficient path resulting from the elastic net. This allows for interactively inspecting the plot so it is easier to disambiguate the coefficients.

**Value**

A dygraphs object

**Author(s)**

Jared P. Lander

**Examples**

```r
library(glmnet)
library(ggplot2)
library(useful)
data(diamonds)
diaX <- useful::build.x(price ~ carat + cut + x - 1, data=diamonds, contrasts = TRUE)
diaY <- useful::build.y(price ~ carat + cut + x - 1, data=diamonds)
modG1 <- glmnet(x=diaX, y=diaY)
coefpath(modG1)
```
modG2 <- cv.glmnet(x=diaX, y=diaY, nfolds=5)
coefpath(modG2)

x <- matrix(rnorm(100*20),100,20)
y <- rnorm(100)
fit1 <- glmnet(x, y)
coefpath(fit1)

## 12  coefplot
 coefplot  
 Plotting Model Coefficients

### 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.

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.

Coefplot method for workflow objects
Coefplot method for parsnip objects

### Usage

coefplot(model, ...)

## Default S3 method:
coefplot(
  model,
  title = "Coefficient Plot",
  xlab = "Value",
  ylab = "Coefficient",
  innerCI = 1,
  outerCI = 2,
  lwdInner = 1 + interactive * 2,
  lwdOuter = if (interactive) 1 else unname((Sys.info()["sysname"] != "Windows") * 0.5),
  pointSize = 3 + interactive * 5,
  color = "blue",
  shape = 16,
  cex = 0.8,
  textAngle = 0,
  numberAngle = 0,
  zeroColor = "grey",
  zeroLWD = 1,
  zeroType = 2,
coefplot

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,
trans = identity,
interactive = FALSE,
newNames = NULL,
plot = TRUE,
"

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

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

## S3 method for class 'workflow'
coefplot(model, ...)

## S3 method for class 'model_fit'
coefplot(model, ...)

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

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

## S3 method for class 'rxLogit'
coefplot(...)
ment information.

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
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", "magnitude", "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.
coefplot

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 coef-
cient 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
trans A transformation function to apply to the values and confidence intervals. identity
by default. Use invlogit for binary regression.
interactive If TRUE an interactive plot is generated instead of ggplot2
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

Details
Currently, methods are available for lm, glm and rxLinMod objects.
For more information on this function and it's arguments see coefplot.default
Pulls model element out of workflow object then calls coefplot.
Pulls model element out of parsnip object then calls coefplot.

Value
A ggplot2 object or data.frame. See details in coefplot.lm for more information
If plot is TRUE then a ggplot object is returned. Otherwise a data.frame listing coefficients and
confidence bands is returned.
A ggplot object. See coefplot.lm for more information.
A ggplot object. See coefplot.lm for more information.
A ggplot object. See coefplot.lm for more information.

Methods (by class)
- default: Default method
- lm: lm
- glm: glm
- workflow: tidymodels workflows
- model_fit: parsnip
- rxGlm: rxGlm
- rxLinMod: rxLinMod
- rxLogit: rxLogit
Author(s)

Jared P. Lander

See Also

coeffplot.lm  coefplot.data.frame
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)
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", "cut^4=""Fourth")
coefplot(model1, predictors=c("(Intercept)", "carat"), newNames=c(carat="Size"))
coefplot(model1, predictors=c("(Intercept)", "carat"),
newNames=c(carat="Size", "(Intercept)"="Constant"))

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"))

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

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

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

Dotplot for coefficients

Usage

```r
## S3 method for class 'data.frame'
coefplot(
  model,
  title = "Coefficient Plot",
  xlab = "Value",
  ylab = "Coefficient",
  interactive = FALSE,
  lwdInner = 1 + interactive * 2,
  lwdOuter = if (interactive) 1 else unname((Sys.info()["sysname"] != "Windows") * 0.5),
  pointSize = 3 + interactive * 5,
  color = "blue",
  cex = 0.8,
  textAngle = 0,
  numberAngle = 0,
  shape = 16,
  linetype = 1,
  outerCI = 2,
  innerCI = 1,
  multi = FALSE,
)```

## Not run:
```r
data(diamonds)
mod4 <- rxLinMod(price ~ carat + cut + x, data=diamonds)
coefplot(mod4)

## Not run:
```
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
interactive | If ‘TRUE’ an interactive plot is generated instead of ‘ggplot2’
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

See Also

ccoefplot

Examples

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)
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:
library(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
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

library(glmnet)
library(ggplot2)
library(useful)
data(diamonds)
diaX <- useful::build.x(price ~ carat + cut + x - 1, data=diamonds, contrasts=FALSE)
diaY <- useful::build.y(price ~ carat + cut + x - 1, data=diamonds)
modG1 <- cv.glmnet(x=diaX, y=diaY, k=5)
extract.coef(modG1)
Description

Extract Coefficient Information from glm Models

Usage

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

Description

Extract Coefficient Information from Models

Usage

```r
## S3 method for class 'glmnet'
extact.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
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

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

## End(Not run)
```
**Description**

Extract Coefficient Information from Models

**Usage**

```r
## 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

---

**Description**

Extract Coefficient Information from `rxGlm` Models

**Usage**

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

**Arguments**

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

Details

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

Value

A \texttt{data.frame} containing the coefficient, the standard error and the variable name.

Author(s)

Jared P. Lander

Examples

```r
## Not run:
library(ggplot2)
data(diamonds)
mod4 <- rxGlm(price ~ carat + cut + x, data=diamonds)
mod5 <- rxGlm(price > 10000 ~ carat + cut + x, data=diamonds, family="binomial")
extra.coef(mod4)
extra.coef(mod5)
## End(Not run)
```

extract.coef.rxLinMod

Description

Extract Coefficient Information from rxLinMod Models

Usage

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

Arguments

- \texttt{model} Model object to extract information from.
- \texttt{...} Further arguments

Details

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

Value

A \texttt{data.frame} containing the coefficient, the standard error and the variable name.
**Author(s)**

Jared P. Lander

**Examples**

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

## End(Not run)
```

**Description**

Extract Coefficient Information from rxLogit Models

**Usage**

```r
## S3 method for class 'rxLogit'
eextract.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
extract.coef.xgb.Booster

Examples

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

## End(Not run)

desc_coef.xgb.Booster

Description

Extract Coefficient Information from Models

Usage

## S3 method for class 'xgb.Booster'
extract.coef(
  model, 
  feature_names = NULL, 
  removeNonSelected = TRUE, 
  zero_threshold = 0.001, 
  ...
)

Arguments

model Model object from which to extract information.
feature_names Names of coefficients
removeNonSelected If TRUE (default) do not return the non-selected (0) coefficients
zero_threshold Since coefficients from xgboost are not exactly zero, this is the threshold under which a coefficient is considered zero
... Further arguments

Details

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

Value

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

```r
library(xgboost)
data(diamonds, package='ggplot2')
diaX <- useful::build.x(price ~ carat + cut + x, data=diamonds, contrasts=FALSE)
diaY <- useful::build.y(price ~ carat + cut + x, data=diamonds)
xg1 <- xgb.train(data=xgb.DMatrix(data=diaX, label=diaY),
booster='gblinear',
objective='reg:squarederror', eval_metric='rmse',
nrounds=50)
extract.coef(xg1)
extract.coef(xg1, zero_threshold=0)
extract.coef(xg1, feature_names=colnames(diaX))
```

Description

Extracts the coefficient path of the elastic net

Usage

```r
extractPath(model, ...)
```

## S3 method for class 'glmnet'
```
extractPath(model, intercept = FALSE, ...)
```

## S3 method for class 'cv.glmnet'
```
extractPath(model, ...)
```

Arguments

- **model**: A `glmnet` model
- **...**: Further arguments
- **intercept**: If `FALSE` (the default), no intercept will be provided

Details

This is a replacement plot for visualizing the coefficient path resulting from the elastic net.
**get.assign**

**Value**

A link[tibble][tibble] holding the coefficients for various lambdas

**Author(s)**

Jared P. Lander

**Examples**

```r
library(glmnet)
data(diamonds, package='ggplot2')
diaX <- useful::build.x(price ~ carat + cut + x - 1, data=diamonds, contrasts = TRUE)
diaY <- useful::build.y(price ~ carat + cut + x - 1, data=diamonds)
modG1 <- glmnet(x=diaX, y=diaY)
extPath(modG1)

modG2 <- cv.glmnet(x=diaX, y=diaY, nfolds=5)
extPath(modG2)
```

**Description**

The assignment vector for a model

**Usage**

```r
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.glm

Description
The assignment vector for a glm model

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

Arguments
model Fitted model
... Further arguments
getCoefsFromPredictors

Details

Gets relative positions of predictors

Value

The assignment vector

Author(s)

Jared P. Lander

getCoefsFromPredictors

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**
```r
## Default S3 method:
getcfsFromPredictors(model, predictors = NULL, strict = FALSE, ...)
```

**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

**Description**
Function for matching coefficients with predictors for rxGlm

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

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

getCoefsFromPredictors.rxLinMod

getCoefsFromPredictors.rxLinMod

Description

Function for matching coefficients with predictors for rxLinMod

Usage

```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.
getCoefsFromPredictors.rxLogit

Value

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

Author(s)

Jared P. Lander

Description

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
getCoefsFromPredictorsRevo

**Description**

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

**Usage**

getCoefsFromPredictorsRevo(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. As of now interactions cannot be explicitly specified.

**Author(s)**

Jared P. Lander

---

invlogit

**Description**

Calculates the inverse logit

**Usage**

invlogit(x)
Arguments
x Vector of numbers

Details
Maps the real line to [0, 1]

Value
x mapped to [0, 1]

Author(s)
Jared P. Lander

Examples
invlogit(3)
invlogit(-6:6)
invlogit(c(-1, 1, 2))

Description
Match coefficients to predictors

Usage
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
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 = (Sys.info()$"sysname" != "Windows") * 0.5,
    pointSize = 3,
    dodgeHeight = 1,
    color = "blue",
    shape = 16,
    linetype = 1,
    cex = 0.8,
    textAngle = 0,
    numberAngle = 90,
    zeroColor = "grey",
    zeroLWD = 1,
    zeroType = 2,
    single = TRUE,
    scales = "fixed",
    ncol = length(unique(modelCI$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("bottom", "right", "left", "top", "none"),
secret.weapon = FALSE,
legend.reverse = FALSE,
trans = identity)

Arguments

... Models to be plotted
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")
col 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
by If "Coefficient" then a normal 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
trans A transformation function to apply to the values and confidence intervals. identity by default. Use invlogit for binary regression.
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)
multiplot(mod1, mod2, mod3, mod7, single=TRUE, scales="free_x", plot.shapes=TRUE, plot.linetypes=TRUE)
multiplot(mod1, mod2, mod3, mod7, single=TRUE, scales="free_x", plot.shapes=FALSE, plot.linetypes=TRUE, legend.position="bottom")
# the secret weapon
multiplot(mod1, mod2, mod3, mod7, coefficients="total_bill", secret.weapon=TRUE)
# horizontal secret weapon
multiplot(mod1, mod2, mod3, mod7, coefficients="total_bill", by="Model", horizontal=FALSE)
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(width = 0.9))

# Similarly with error bars:
p + geom_errorbar(aes(ymin = y-1, ymax = y+1), width = 0.2,
  position = "dodge")
p + geom_errorbar(aes(ymin = y-1, ymax = y+1, height = 0.2),
  position = position_dodge(width = 0.90))
Index

* coefficients
  coefplot, 12
* coefficient
  coefplot, 12
* coefplot
  coefplot, 12
* dotplot
  coefplot, 12
* glm
  coefplot, 12
* linear
  coefplot, 12
* lm
  coefplot, 12
* model
  coefplot, 12
* position adjustments
  position_dodgev, 44
* rxLinMod
  coefplot, 12
annotateSeries, 2
buildModelCI, 3, 7, 9
buildModelCI.default, 4
buildPlotting.default, 6, 9
buildPlottingPloty.default, 8
coefpath, 10
ccoefplot, 4, 5, 8, 9, 12, 16, 19, 43
ccoefplot-package (coefplot), 12
ccoefplot.data.frame, 16, 17
ccoefplot.default, 8, 9, 15
ccoefplot.default (coefplot), 12
ccoefplot.glm (coefplot), 12
ccoefplot.lm, 8, 13, 15, 16
ccoefplot.lm (coefplot), 12
ccoefplot.model_fit (coefplot), 12
ccoefplot.rxGlm (coefplot), 12
ccoefplot.rxLinMod (coefplot), 12
ccoefplot.rxLogit (coefplot), 12
ccoefplot.workflow (coefplot), 12
data.frame, 4, 5, 15, 21–29
doRegex, 20
dySeries, 3
extract.coef, 20
extract.coef.cv.glmnet, 21
extract.coef.default, 22
extract.coef.glm, 23
extract.coef.glmnet, 24
extract.coef.lm, 25
extract.coef.maxLik, 26
extract.coef.rxGlm, 26
extract.coef.rxLinMod, 27
extract.coef.rxLogit, 28
extract.coef.xgb.Booster, 29
extractPath, 10, 30
get.assign, 31
get.assign.glm, 32
get.assign.lm, 32
getCoefsFromPredictors, 33
getCoefsFromPredictors.default, 34
getCoefsFromPredictors.rxGlm, 34
getCoefsFromPredictors.rxLinMod, 35
getCoefsFromPredictors.rxLogit, 36
getCoefsFromPredictorsRevo, 20, 37
ggplot, 15, 16
glm, 16
glmnet, 10, 30
invlogit, 37
lm, 16
matchCoefs, 38
matchCoefs.default, 39
multiplot, 4, 5, 7, 8, 18, 40
plotcoef, 16
plotcoef (coefplot), 12
position_dodgev, 44
reduce, 3
xgboost, 29