Package ‘predict3d’

April 12, 2023

Type Package
Title Draw Three Dimensional Predict Plot Using Package ‘rgl’
Version 0.1.4
URL https://github.com/cardiomoon/predict3d
BugReports https://github.com/cardiomoon/predict3d/issues
Description Draw 2 dimensional and three dimensional plot for multiple regression models using package ‘ggplot2’ and ‘rgl’.
Supports linear models (lm), generalized linear models (glm) and local polynomial regression fittings (loess).
Depends R (>= 3.3.0)
License GPL-2
Encoding UTF-8
Imports ggplot2(>= 3.1.0), rgl(>= 1.0.1), dplyr, ggiraphExtra, modelr, prediction, purrr, rlang, stringr, magrittr, stats, reshape2, plyr, tidyr
RoxygenNote 7.2.3
Suggests moonBook, TH.data, knitr, rmarkdown
VignetteBuilder knitr
NeedsCompilation no
Author Keon-Woong Moon [aut, cre]
Maintainer Keon-Woong Moon <cardiomoon@gmail.com>
Repository CRAN
Date/Publication 2023-04-12 16:20:06 UTC

R topics documented:

add_lines ................................................................. 2
beNumeric ................................................................. 3
calEquation ................................................................. 4
add_lines

Add lines with labels to pre-existing ggplot

Description

Add lines with labels to pre-existing ggplot

Usage

add_lines(
  p,
  df,
  xpos = 0.3,
  add.coord.fixed = TRUE,
  lty = NULL,
  color = NULL,
  size = 0.5,
  add_theme_bw2 = TRUE,
  ...
)

Index

22
beNumeric

Arguments

- **p**: An object of class ggplot
- **df**: A data.frame. Required columns are slope, intercept and label
- **xpos**: A numeric. Relative horizontal position
- **add.coord.fixed**: Logical. Whether or not add coord_fixed() function
- **lty**: line type
- **color**: line color
- **size**: line size
- **add_theme_bw2**: logical Whether or not add theme_bw2()
- **...**: Further arguments to be passed to geom_text

Examples

```r
require(ggplot2)
fit=lm(mpg~wt*hp,data=mtcars)
df=calEquation(fit)
p=ggplot(data=mtcars,aes(x=wt,y=mpg))
add_lines(p,df)
add_lines(p,df,lty=1:3,color=1:3,size=1)
fit=lm(mpg~wt*vs,data=mtcars)
df=calEquation(fit)
p=ggplot(data=mtcars)+geom_point(aes(x=wt,y=mpg))
add_lines(p,df)
add_lines(p,df,lty=1:2,color=1:2,size=1)+theme_bw()
```

beNumeric

Whether a string vector can be converted to numeric

Description

Whether a string vector can be converted to numeric

Usage

beNumeric(x)

Arguments

- **x**: A string vector

Examples

```r
x=c("age","22.5","11/2")
beNumeric(x)
```
calEquation  
\textit{calculated slope and intercept from object of class \texttt{lm}}

\textbf{Description}

calculated slope and intercept from object of class \texttt{lm}

\textbf{Usage}

\begin{verbatim}
calEquation(
  fit, 
  mode = 1, 
  pred = NULL, 
  modx = NULL, 
  modx.values = NULL, 
  label = NULL, 
  maxylev = 6, 
  digits = 2
)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{fit} An object of class \texttt{lm}
  \item \texttt{mode} A numeric
  \item \texttt{pred} name of predictor variable
  \item \texttt{modx} name of modifier variable
  \item \texttt{modx.values} Numeric. Values of modifier variable
  \item \texttt{label} A character string
  \item \texttt{maxylev} maximum length of unique value of variable to be treated as a categorial variable
  \item \texttt{digits} Integer indicating the number of decimal places
\end{itemize}

\textbf{Examples}

\begin{verbatim}
fit=lm(mpg~wt*hp+carb,data=mtcars)
calEquation(fit)
calEquation(fit,pred="hp")
\end{verbatim}
**expand.grid2**

**expand.grid with two data.frames**

**Description**

expand.grid with two data.frames

**Usage**

expand.grid2(df1, df2)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df1</td>
<td>A data.frame</td>
</tr>
<tr>
<td>df2</td>
<td>A data.frame</td>
</tr>
</tbody>
</table>

**fit2newdata**

Make a new data set for prediction

**Description**

Make a new data set for prediction

**Usage**

fit2newdata(
  fit,
  predictors,
  mode = 1,
  pred.values = NULL,
 modx.values = NULL,
  mod2.values = NULL,
  colorn = 3,
  maxylev = 6,
  summarymode = 1
)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit</td>
<td>An object of class &quot;lm&quot;, &quot;glm&quot; or &quot;loess&quot;</td>
</tr>
<tr>
<td>predictors</td>
<td>Names of predictor variables in string</td>
</tr>
<tr>
<td>mode</td>
<td>A numeric. Useful when the variables are numeric. If 1, c(-1,0,1)*sd + mean is used. If 2, the 16th, 50th, 84th percentile values used. If 3 sequence over a the range of a vector used</td>
</tr>
</tbody>
</table>
pred.values For which values of the predictors should be used? Default is NULL. If NULL, 20 seq_range is used.
modx.values For which values of the moderator should lines be plotted? Default is NULL. If NULL, then the customary +/- 1 standard deviation from the mean as well as the mean itself are used for continuous moderators. If the moderator is a factor variable and modx.values is NULL, each level of the factor is included.
mod2.values For which values of the second moderator should lines be plotted? Default is NULL. If NULL, then the customary +/- 1 standard deviation from the mean as well as the mean itself are used for continuous moderators. If the moderator is a factor variable and modx.values is NULL, each level of the factor is included.
colorn The number of regression lines when the modifier variable(s) are numeric.
maxylev An integer indicating the maximum number of levels of numeric variable be treated as a categorical variable
summarymode An integer indicating method of extracting typical value of variables. If 1, typical() is used. If 2, mean() is used.

Examples

```r
fit=lm(mpg~hp*wt*cyl+carb+am,data=mtcars)
fit2newdata(fit,predictors=c("hp","wt","am"))
fit2newdata(fit,predictors=c("hp","wt","cyl"))
fit2newdata(fit,predictors=c("hp"))
fit2newdata(fit,predictors=c("hp","wt"))
fit=loess(mpg~hp*wt*am,data=mtcars)
fit2newdata(fit,predictors=c("hp"))
## Not run:
mtcars$engine=ifelse(mtcars$vs==0,"V-shaped","straight")
fit=lm(mpg~wt*engine,data=mtcars)
fit2newdata(fit,predictors=c("wt","engine"))
fit=lm(mpg~wt*factor(vs),data=mtcars)
fit2newdata(fit,predictors=c("wt","vs"))
fit2newdata(lm(mpg~hp*wt*am),predictors=c("hp","wt"),mode=3,colorn=30)
fit=lm(mpg~hp*log(wt),data=mtcars)
fit2newdata(fit,predictors=c("hp","log(wt)"))
fit=lm(mpg~hp*wt*factor(vs),data=mtcars)
fit2newdata(fit,predictors=c("hp"))
## End(Not run)
require(moonBook)
fit=lm(log(NTAV)~I(age^2)*sex,data=radial)
fit2newdata(fit,predictors=c("I(age^2)","sex"))
```

getAspectRatio

Get aspect information of a ggplot

Description

Get aspect information of a ggplot
getMeans

Usage

getAspectRatio(p)

Arguments

p A ggplot object

description

calculate mean values of two consecutive number

Usage

getMeans(x)

Arguments

x A numeric vector

Examples

x=c(50,60,70)
getMeans(x)

description

Make new formula

Usage

getNewFormula(fit, predictors = NULL)

Arguments

fit An object of class lm or glm
predictors Names of variables to exclude
Examples

```r
fit = lm(mpg ~ factor(cyl) * factor(am) + wt + carb, data = mtcars)
getNewFormula(fit, predictors = c("cyl", "wt"))
fit = lm(Sepal.Length ~ Sepal.Width * Petal.Length + Species, data = iris)
getNewFormula(fit, predictors = c("Petal.Length"))
fit = lm(mpg ~ hp * wt * factor(cyl), data = mtcars)
getNewFormula(fit, predictors = c("hp", "cyl"))
fit = loess(mpg ~ hp * wt, data = mtcars)
getNewFormula(fit, predictors = c("hp", "wt"))
```

---

**ggPredict**

*Visualize predictions from the multiple regression models.*

**Description**

Visualize predictions from the multiple regression models.

**Usage**

```r
ggPredict(
  fit,
  pred = NULL,
 modx = NULL,
  mod2 = NULL,
  modx.values = NULL,
  mod2.values = NULL,
  dep = NULL,
  mode = 1,
  colorn = 3,
  maxylev = 6,
  show.point =getOption("ggPredict.show.point", TRUE),
  show.error = FALSE,
  error.color = "red",
  jitter = NULL,
  se = FALSE,
  alpha = 0.1,
  show.text = TRUE,
  add.modx.values = TRUE,
  add.loess = FALSE,
  labels = NULL,
  angle = NULL,
  xpos = NULL,
  vjust = NULL,
  digits = 2,
  facet.modx = FALSE,
  facetbycol = TRUE,
  plot = TRUE,
)```


Arguments

**fit**
An object of class "lm" or "glm"

**pred**
The name of predictor variable

**modx**
Optional. The name of moderator variable

**mod2**
Optional. The name of second moderator variable

**modx.values**
For which values of the moderator should lines be plotted? Default is NULL. If NULL, then the customary +/- 1 standard deviation from the mean as well as the mean itself are used for continuous moderators. If the moderator is a factor variable and modx.values is NULL, each level of the factor is included.

**mod2.values**
For which values of the second moderator should lines be plotted? Default is NULL. If NULL, then the customary +/- 1 standard deviation from the mean as well as the mean itself are used for continuous moderators. If the moderator is a factor variable and modx.values is NULL, each level of the factor is included.

**dep**
Optional. The name of dependent variable

**mode**
A numeric. Useful when the variables are numeric. If 1, c(-1,0,1)*sd + mean is used. If 2, the 14th, 50th, 86th percentile values used. If 3 sequence over a the range of a vector used

**colorn**
The number of regression lines when the modifier variable(s) are numeric.

**maxylev**
An integer indicating the maximum number of levels of numeric variable be treated as a categorical variable

**show.point**
Logical. Whether or not add points

**show.error**
Logical. Whether or not show error

**error.color**
color of error. default value is "red"

**jitter**
logical Whether or not use geom_jitter

**se**
Logical. Whether or not add confidence interval

**alpha**
A numeric. Transparency

**show.text**
Logical. Whether or not add regression equation as label

**add.modx.values**
Logical. Whether or not add moderator values to regression equation

**add.loess**
Logical. Whether or not add loess line

**labels**
labels on regression lines

**angle**
angle of text

**xpos**
x axis position of label

**vjust**
vertical alignment of labels

**digits**
integer indicating the number of decimal places

**facet.modx**
Create separate panels for each level of the moderator? Default is FALSE
Facet by column Logical.

Plot Logical. Should a plot of the results be printed? Default is TRUE.

Summary mode An integer indicating method of extracting typical value of variables. If 1, typical() is used. If 2, mean() is used.

Additional arguments to be passed to geom_text

Examples

```r
fit = loess(mpg ~ hp*wt*am, data = mtcars)
ggPredict(fit)
ggPredict(fit, hp)
## Not run:
ggPredict(fit, hp, wt)
fit = lm(mpg ~ wt*hp-1, data = mtcars)
ggPredict(fit, xpos = 0.7)
fit = lm(mpg ~ hp*wt, data = mtcars)
ggPredict(fit)
ggPredict(fit, labels = paste0("label", 1:3), xpos = c(0.3, 0.6, 0.4))
ggPredict(fit, se = TRUE)
ggPredict(fit, mode = 3, colorn = 40, show.text = FALSE)
fit = lm(log(mpg) ~ hp*wt, data = mtcars)
ggPredict(fit, dep = mpg)
fit = lm(mpg ~ hp*wt*cyl, data = mtcars)
ggPredict(fit, modx = wt, modx.values = c(2, 3, 4, 5), mod2 = cyl, show.text = FALSE)
ggPredict(fit, hp, wt, show.point = FALSE, se = TRUE, xpos = 0.5)
ggPredict(fit, modx = wt, xpos = 0.3)
mtcars$engine = ifelse(mtcars$vs == 0, "V-shaped", "straight")
fit = lm(mpg ~ wt*engine, data = mtcars)
ggPredict(fit)
require(TH.data)
fit = glm(cens ~ pnodes*horTh, data = GBSG2, family = binomial)
ggPredict(fit, pnodes, horTh, se = TRUE, xpos = c(0.6, 0.3), angle = c(40, 60), vjust = c(2, -0.5))
fit1 = glm(cens ~ pnodes, data = GBSG2, family = binomial)
ggPredict(fit1, vjust = 1.5, angle = 45)
fit3 = glm(cens ~ pnodes*age, data = GBSG2, family = binomial)
ggPredict(fit3, pred = pnodes, modx = age, mode = 3, colorn = 10, show.text = FALSE)
fit2 = glm(cens ~ pnodes*age*horTh, data = GBSG2, family = binomial)
ggPredict(fit2, pred = pnodes, modx = age, mod2 = horTh, mode = 3, colorn = 10, show.text = FALSE)
fit = lm(mpg ~ log(hp)*wt, data = mtcars)
ggPredict(fit, hp, wt)
fit = lm(mpg ~ wt + disp + gear + carb + am, data = mtcars)
ggPredict(fit, disp, gear, am)
library(moonBook)
fit = lm(weight ~ I(height^3) + I(height^2) + height + sex, data = radial)
ggPredict(fit)
predict3d(fit)
```

## End(Not run)
Description

Pick default color

Usage

gg_color_hue(n)

Arguments

n An integer

is.mynumeric Decide whether a vector can be treated as a numeric variable

Description

Decide whether a vector can be treated as a numeric variable

Usage

is.mynumeric(x, maxylev = 6)

Arguments

x A vector

maxylev An integer indicating the maximum number of levels of numeric variable be treated as a categorical variable
myseq

Generate regular sequences of desired length between minimum and maximal values

Usage

myseq(x, length = 20)

Arguments

x a numeric vector
length desired length of the sequence

number2group

Convert a numeric vector into groups

Description

Convert a numeric vector into groups

Usage

number2group(
  x,
  mode = 1,
  values = NULL,
  silent = FALSE,
  label = "label",
  digits = 2,
  colorn = 3
)

Arguments

x A numeric vector
mode A numeric. If 1, mean(x) + c(-1,0,1)*sd(x) are used. If 2, quantile(x,probs=c(0.14,0.5,0.86),type=6) are used. If 3, values are used
values A numeric vector
silent A logical. Whether table of result will be shown
label A character string
digits integer indicating the number of decimal places
colorn The number of regression lines when the modifier variable(s) are numeric
Examples

```r
number2group(iris$Sepal.Length,label="Sepal.Length")
x=number2group(mtcars$wt,label="wt")
x
```

---

**predict3d**

_**Draw 3d predict plot using package 'rgl'**_

**Description**

Draw 3d predict plot using package 'rgl'

**Usage**

```r
predict3d(
  fit,
  pred = NULL,
 modx = NULL,
  mod2 = NULL,
  dep = NULL,
  xlab = NULL,
  ylab = NULL,
  zlab = NULL,
  width = 640,
  colorn = 20,
  maxylev = 6,
  se = FALSE,
  show.summary = FALSE,
  overlay = NULL,
  show.error = FALSE,
  show.legend = FALSE,
  bg = NULL,
  type = "s",
  radius = NULL,
  palette = NULL,
  palette.reverse = TRUE,
  color = "red",
  show.subtitle = TRUE,
  show.plane = TRUE,
  plane.color = "steelblue",
  plane.alpha = 0.5,
  summarymode = 1,
  ...
)
```
Arguments

- **fit**: A model object for which prediction is desired.
- **pred**: The name of predictor variable.
- **modx**: Optional. The name of moderator variable.
- **mod2**: Optional. The name of second moderator variable.
- **dep**: Optional. The name of dependent variable.
- **xlab**: x-axis label.
- **ylab**: y-axis label.
- **zlab**: z-axis label.
- **width**: the width of device.
- **colorn**: An integer giving the desired number of intervals. Non-integer values are rounded down.
- **maxylev**: Maximal length of unique values of y axis variable to be treated as a categorical variable.
- **se**: Logical. Whether or not show se. Only effective when the y-axis variable is a categorical one.
- **show.summary**: Logical. Whether or not show statistical summary.
- **overlay**: Logical. Whether or not overlay plots.
- **show.error**: Logical. Whether or not show error.
- **show.legend**: Logical. Whether or not show legend.
- **bg**: Character. Background color of plot.
- **type**: For the default method, a single character indicating the type of item to plot. Supported types are: 'p' for points, 's' for spheres, 'l' for lines, 'h' for line segments from z = 0, and 'n' for nothing. For the mesh3d method, one of 'shade', 'wire', or 'dots'. Partial matching is used.
- **radius**: The size of sphere.
- **palette**: Name of color palette.
- **palette.reverse**: Logical. Whether or not reverse the palette order.
- **color**: Default color. Color is used when the palette is NULL.
- **show.subtitle**: Logical. If true, show regression call as subtitle.
- **show.plane**: Logical. If true, show regression plane.
- **plane.color**: Name of color of regression plane.
- **plane.alpha**: Transparency scale of regression plane.
- **summarymode**: An integer indicating method of extracting typical value of variables. If 1, typical() is used. If 2, mean() is used.
- **...**: additional parameters which will be passed to plot3d.
Examples

```r
fit = lm(mpg ~ hp*wt, data = mtcars)
predict3d(fit, show.error = TRUE)
```

```r
fit = lm(log(mpg) ~ hp*wt, data = mtcars)
predict3d(fit, dep = mpg)
```

## Not run:

```r
fit = lm(Sepal.Length ~ Sepal.Width*Species, data = iris)
predict3d(fit)
```

```r
require(TH.data)
fit = glm(cens ~ pnodes*age*horTh, data = GBSG2, family = binomial)
predict3d(fit)
```

```r
mtcars$engine = ifelse(mtcars$vs == 0, "V-shaped", "straight")
fit = lm(mpg ~ wt*engine, data = mtcars)
predict3d(fit)
```

```r
fit = loess(mpg ~ hp*wt, data = mtcars)
predict3d(fit, radius = 4)
```

```r
states <- as.data.frame(state.x77[, c("Murder", "Population", "Illiteracy", "Income", "Frost")])
fit = lm(Murder ~ Population + Illiteracy, data = states)
predict3d(fit)
predict3d(fit, radius = 200)
```

```r
fit = lm(mpg ~ cyl + hp + am, data = mtcars)
predict3d(fit)
```

## End(Not run)

---

**rank2colors**

*Rank a numeric vector using proportional table and returns character vector of names of color using palette*

### Description

Rank a numeric vector using proportional table and returns character vector of names of color using palette

### Usage

```r
rank2colors(x, palette = "Blues", reverse = TRUE, color = "red")
```

### Arguments

- `x`: A numeric vector
- `palette`: Name of the color palette
- `reverse`: Logical. Whether or not reverse the order of the color palette
- `color`: Default color when palette is NULL

### Examples

```r
rank2colors(mtcars$wt, palette = "Blues")
```
rank2group2  

_Rank a numeric vector using proportional table and returns a new ordinal vector_

**Description**

Rank a numeric vector using proportional table and returns a new ordinal vector.

**Usage**

`rank2group2(x, k = 4)`

**Arguments**

- **x**: a numeric vector
- **k**: a integer specifies how many groups you want to classify. default value is 4

---

**restoreData**  

_Restore factors in data.frame as numeric_

**Description**

Restore factors in data.frame as numeric.

**Usage**

`restoreData(data)`

**Arguments**

- **data**: A data.frame

**Examples**

```r
fit=lm(mpg~factor(cyl)*factor(am),data=mtcars)
fit=lm(mpg~wt*factor(am),data=mtcars)
fit=lm(mpg~wt*hp,data=mtcars)
restoreData(fit$model)
```
**restoreData2**  
restore data column with I() function

**Description**  
restore data column with I() function

**Usage**  
`restoreData2(df)`

**Arguments**  
df A data.frame

**Examples**  
```r  
fit=lm(mpg~I(cyl^(1/2))*am, data=mtcars)  
restoreData2(fit$model)  
fit=lm(mpg~sqrt(hp)*log(wt)*am, data=mtcars)  
restoreData2(fit$model)  
```

---

**restoreData3**  
Restore data from arithmetic operator

**Description**  
Restore data from arithmetic operator

**Usage**  
`restoreData3(df, changeLabel = FALSE)`

**Arguments**  
df A data.frame  
changeLabel logical

**Examples**  
```r  
fit=lm(2^mpg~hp*wt, data=mtcars)  
summary(fit)  
restoreData3(fit$model)  
```
## restoreNames

### Description

Restore factors in variable name as numeric

### Usage

```r
restoreNames(x)
```

### Arguments

- `x`: character vector

### Examples

```r
restoreNames(c("factor(cyl)","am"))
restoreNames(c("I(age^2)","am","100/mpg","cyl^1/2","mpg2","sex + 0.5"))
```

## revOperator

### Description

Get opposite arithmetic operator

### Usage

```r
revOperator(operator)
```

### Arguments

- `operator`: A character
seekNamesDf

Find variable names in data.frame

Description
Find variable names in data.frame

Usage
seekNamesDf(vars, df)

Arguments
vars variable names to find
df A data.frame

Value
A character vector

slope2angle
Make angle data with slope data

Description
Make angle data with slope data

Usage
slope2angle(
    df,
    fit,
    ytransform = 0,
    predc,
    temppredc,
    modxc,
    yvar,
    p,
    method = "lm",
    xpos = NULL,
    vjust = NULL,
    digits = 3,
    facetno = NULL,
    add.modx.values = TRUE
)
Arguments

df A data.frame
fit An object of class "lm" or "glm"
ytransform Numeric. If 1, log transformation of dependent variable, If -1, exponential transformation
predc Name of predictor variable
temppredc Name of predictor variable in regression equation
modx Name of moderator variable
yvar Name of dependent variable
p An object of class ggplot
method String. Choices are one of "lm" and "glm".
xpos The relative x-axis position of labels. Should be within 0 to 1
vjust vjust
digits integer indicating the number of decimal places
facetno The number of facets
add.modx.values Whether add name of moderator variable

string2pattern change string to pattern

Description
change string to pattern

Usage

string2pattern(string)

Arguments

string A character vector

Examples

string=c("I(age^2)","factor(cyl)","log(mpg)")
string2pattern(string)
**theme_bw2**

---

**theme_bw2**  
*theme_bw with no grid*

---

**Description**

theme_bw with no grid

**Usage**

theme_bw2()
Index

add_lines, 2
beNumeric, 3
calEquation, 4
expand.grid2, 5
fit2newdata, 5
getAddressRatio, 6
getMeans, 7
getNewFormula, 7
gg_color_hue, 11
ggPredict, 8
is.mynumeric, 11
myseq, 12
number2group, 12
predict3d, 13
rank2colors, 15
rank2group2, 16
restoreData, 16
restoreData2, 17
restoreData3, 17
restoreNames, 18
revOperator, 18
seekNamesDf, 19
slope2angle, 19
string2pattern, 20
theme_bw2, 21