

# Package ‘glm.predict’

November 17, 2020

**Type** Package

**Title** Predicted Values and Discrete Changes for GLM

**Version** 4.0-0

**Date** 2020-11-16

**Author** Benjamin Schlegel [aut,cre]

**Maintainer** Benjamin Schlegel <kontakt@benjaminschlegel.ch>

**Description** Functions to calculate predicted values and the difference between the two cases with confidence interval for `lm()` [linear model], `glm()` [general linear model], `glm.nb()` [negative binomial model], `polr()` [ordinal logistic model], `multinom()` [multinomial model] and `tobit()` [tobit model] using Monte Carlo simulations or bootstrap. Reference: Bennet A. Zelter (2009) <doi:10.1002/smj.783>.

**License** GPL (>= 2)

**Depends** R (>= 3.1.0), stats, MASS, parallel

**Imports** nnet, AER, survival

**Suggests** knitr, rmarkdown, schlegel, ggplot2, devtools

**VignetteBuilder** knitr, rmarkdown

**Additional\_repositories** <https://benjaminschlegel.github.io/drat/>

**URL** <https://benjaminschlegel.ch/r/glm-predict/>

**Language** en-US

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-11-17 09:50:02 UTC

## R topics documented:

<code>glm.predict-package</code> . . . . .	2
<code>basepredict</code> . . . . .	2
<code>basepredict.glm</code> . . . . .	3
<code>basepredict.lm</code> . . . . .	4

basepredict.multinom . . . . .	5
basepredict.polr . . . . .	6
basepredict.tobit . . . . .	8
dc . . . . .	9
dc.glm . . . . .	10
dc.lm . . . . .	11
dc.multinom . . . . .	12
dc.polr . . . . .	14
dc.tobit . . . . .	15
predicts . . . . .	16

<b>Index</b>	<b>19</b>
--------------	-----------

---

glm.predict-package	<i>Predicted Values and Discrete Changes for GLM</i>
---------------------	--

---

## Description

This package provides functions to calculate predicted values and the difference between two cases with confidence interval.

## Author(s)

Benjamin Schlegel

Maintainer: Benjamin Schlegel <kontakt@benjaminschlegel.ch>

---

basepredict	<i>predicted value</i>
-------------	------------------------

---

## Description

The generic function calculates the predicted value with the confidence interval. It can be used for any `lm()`, `glm()`, `glm.nb()`, `polr()`, `tobit()` or `multinom()` model.

## Usage

```
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model Object generated with glm(), glm.nb(), polr() or multinom()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
model1 = glm(Sex ~ Height + Smoke + Pulse, data=MASS::survey, family=binomial(link=logit))
summary(model1)
# predicted probability of a non smoking person with height 150 and average pulse
basepredict(model1, c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))
```

---

basepredict.glm	<i>predicted value</i>
-----------------	------------------------

---

**Description**

The function calculates the predicted value with the confidence interval. It can be used for any glm model.

**Usage**

```
## S3 method for class 'glm'
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model Object generated with glm() or glm.nb()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a 3x3 matrix having in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
model1 = glm(Sex ~ Height + Smoke + Pulse, data=MASS::survey, family=binomial(link=logit))
summary(model1)
# predicted probability of a non smoking person with height 150 and average pulse
basepredict(model1, c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))
```

---

basepredict.lm	<i>predicted value</i>
----------------	------------------------

---

**Description**

The function calculates the predicted value with the confidence interval for a lm model.

**Usage**

```
## S3 method for class 'lm'
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model Object generated with lm()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a 3x3 matrix having in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
model1 = lm(Pulse ~ Height + Smoke, data=MASS::survey)
summary(model1)
# predicted pulse value of a non smoking person with height 150
basepredict(model1, c(1,150,1,0,0))
```

---

basepredict.multinom    *predicted value*

---

**Description**

The function calculates the predicted value with the confidence interval. It can be used for any multinom model.

**Usage**

```
## S3 method for class 'multinom'
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model Object generated with multinom()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
## Not run:
model1 = nnet::multinom(Clap ~ Height + Smoke + Pulse, data=MASS::survey)
summary(model1)
# predicted probability of a non smoking person with height 150 and average pulse
basepredict(model1, c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))

## End(Not run)
```

---

basepredict.polr	<i>predicted value</i>
------------------	------------------------

---

**Description**

The function calculates the predicted value with the confidence interval. It can be used for any polr model.

**Usage**

```
## S3 method for class 'polr'
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model Object generated with polr()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
## Not run:
data = MASS::survey
data$Smoke = ordered(data$Smoke, levels = c("Never", "Occas", "Regul", "Heavy"))
model1 = polr(Smoke ~ Height + Pulse, data=data)
summary(model1)
# predicted probability of smoking of a person with height 170 and an average pulse
basepredict(model1, c(170, mean(MASS::survey$Pulse, na.rm=TRUE)))

## End(Not run)
```

---

basepredict.tobit	<i>predicted value</i>
-------------------	------------------------

---

### Description

The function calculates the predicted value with the confidence interval for a tobit model.

### Usage

```
## S3 method for class 'tobit'
basepredict(model, values, sim.count=1000, conf.int=0.95, sigma=NULL, set.seed=NULL,
  type = c("any", "simulation", "bootstrap"))
```

### Arguments

model	the model Object generated with tobit()
values	the values of the case as vector in the order how they appear in the summary(model) Estimate
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
type	OPTIONAL only simulation is supported for tobit()

### Details

The function makes a simulation for the two cases and compares them to each other.

### Value

The output is a 3x3 matrix having in the first column the predicted value, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

### Author(s)

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

### Examples

```
library(AER)
model1 = tobit(Age ~ Height + Pulse, right = 65, data=MASS::survey)
summary(model1)
# Person with a height of 160 and a pulse of 80
basepredict(model1, values = c(1,160,80))
```



---

dc *predicted values and discrete change*

---

### Description

The generic function calculates the predicted values and the difference of two cases with the confidence interval. It can be used for any `lm()`, `glm()`, `glm.nb()`, `polr()`, `tobit()` or `multinom()` model.

### Usage

```
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
    sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
    type = c("any", "simulation", "bootstrap"))
```

### Arguments

<code>model</code>	the model-Object generated with <code>glm()</code> , <code>glm.nb()</code> , <code>polr()</code> or <code>multinom()</code>
<code>values</code>	the values of case 1 and 2 as vector in the order how they appear in the <code>summary(model) Estimate</code> . Values is if <code>values1</code> and <code>values2</code> are specified after each other in the same vector. Either <code>values</code> or <code>values1</code> and <code>values2</code> have to be specified.
<code>sim.count</code>	OPTIONAL numbers of simulations to be done by the function. default: 1000
<code>conf.int</code>	OPTIONAL the confidence interval used by the function. default: 0.95
<code>sigma</code>	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered <code>vcov</code> . default: <code>vcov(model)</code>
<code>set.seed</code>	OPTIONAL set a seed for the random number generator
<code>values1</code>	the values of case 1 as vector in the order how they appear in the <code>summary(model) Estimate</code> . Has to be defined if <code>values</code> is not defined.
<code>values2</code>	the values of case 2 as vector in the order how they appear in the <code>summary(model) Estimate</code> . Has to be defined if <code>values</code> is not defined.
<code>type</code>	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if $n < 1000$ )

### Details

The function makes a simulation for the two cases and compares them to each other.

### Value

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

### Author(s)

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

## Examples

```

model1 = glm(Sex ~ Height + Smoke + Pulse, data=MASS::survey, family=binomial(link=logit))
summary(model1)
# comparing a person with the height 150cm to 151cm
dc(model1, values1 = c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)),
  values2 = c(1,151,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))
# the higher person has a greater probability to be a man
# the difference is significant, because the confidence interval
# does not include the 0

```

---

dc.glm

*predicted values and discrete change*

---

## Description

The function calculates the predicted values and the difference of two cases with the confidence interval. It can be used for any glm model.

## Usage

```

## S3 method for class 'glm'
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
  sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
  type = c("any", "simulation", "bootstrap"))

```

## Arguments

model	the model-Object generated with glm() or glm.nb()
values	the values of case 1 and 2 as vector in the order how they appear in the summary(model) Estimate. Values is if values1 and values2 are specified after each other in the same vector. Either values or values1 and values2 have to be specified.
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
values1	the values of case 1 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
values2	the values of case 2 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
model1 = glm(Sex ~ Height + Smoke + Pulse, data=MASS::survey, family=binomial(link=logit))
summary(model1)
# comparing a person with the height 150cm to 151cm
dc(model1, values1 = c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)),
  values2 = c(1,151,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))
# the higher person has a greater probability to be a man
# the difference is significant, because the confidence interval
# does not include the 0
```

---

dc.lm

*predicted values and discrete change*

---

**Description**

The function calculates the predicted values and the difference of two cases with the confidence interval for a lm model.

**Usage**

```
## S3 method for class 'lm'
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
  sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model-Object generated with lm()
values	the values of case 1 and 2 as vector in the order how they appear in the summary(model) Estimate. Values is if values1 and values2 are specified after each other in the same vector. Either values or values1 and values2 have to be specified.
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95

sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
values1	the values of case 1 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
values2	the values of case 2 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

### Details

The function makes a simulation for the two cases and compares them to each other.

### Value

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

### Author(s)

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

### Examples

```
model1 = lm(Pulse ~ Height + Smoke, data=MASS::survey)
summary(model1)
# comparing a person with the height 150cm to 151cm
dc(model1, values1 = c(1,150,1,0,0),
  values2 = c(1,151,1,0,0))
# the difference is not significant, because the confidence interval
# includes the 0
```

---

dc.multinom

*predicted values and discrete change*

---

### Description

The function calculates the predicted values and the difference of two cases with the confidence interval. It can be used for a multinom model.

### Usage

```
## S3 method for class 'multinom'
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
  sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

model	the model-Object generated with multinom()
values	the values of case 1 and 2 as vector in the order how they appear in the summary(model) Estimate. Values is if values1 and values2 are specified after each other in the same vector. Either values or values1 and values2 have to be specified.
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
values1	the values of case 1 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
values2	the values of case 2 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
## Not run:
model1 = nnet::multinom(Clap ~ Height + Smoke + Pulse, data=MASS::survey)
summary(model1)
dc(model1, values1 = c(1,150,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)),
  values2 = c(1,151,1,0,0,mean(MASS::survey$Pulse,na.rm=TRUE)))
# the higher person has a greater probability to be left clapping
# the difference is significant, because the confidence interval
# does not include the 0

## End(Not run)
```

---

dc.polr *predicted values and discrete change*

---

### Description

The function calculates the predicted values and the difference of two cases with the confidence interval. It can be used for a polr model.

### Usage

```
## S3 method for class 'polr'
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
    sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
    type = c("any", "simulation", "bootstrap"))
```

### Arguments

model	the model-Object generated with polr()
values	the values of case 1 and 2 as vector in the order how they appear in the summary(model) Estimate. Values is if values1 and values2 are specified after each other in the same vector. Either values or values1 and values2 have to be specified.
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
values1	the values of case 1 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
values2	the values of case 2 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

### Details

The function makes a simulation for the two cases and compares them to each other.

### Value

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

### Author(s)

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

## Examples

```
## Not run:
data = MASS::survey
data$Smoke = ordered(data$Smoke, levels = c("Never", "Occas", "Regul", "Heavy"))
model1 = polr(Smoke ~ Height + Pulse, data=data)
summary(model1)
dc(model1, values1 = c(150,mean(MASS::survey$Pulse,na.rm=TRUE)),
  values2 = c(151,mean(MASS::survey$Pulse,na.rm=TRUE)))
# all differences are significant as the confidence intervals do not include 0

## End(Not run)
```

---

 dc.tobit

*predicted values and discrete change*


---

## Description

The function calculates the predicted values and the difference of two cases with the confidence interval for a tobit model.

## Usage

```
## S3 method for class 'tobit'
dc(model, values = NULL, sim.count = 1000, conf.int = 0.95,
  sigma = NULL, set.seed = NULL, values1 = NULL, values2 = NULL,
  type = c("any", "simulation", "bootstrap"))
```

## Arguments

model	the model-Object generated with tobit()
values	the values of case 1 and 2 as vector in the order how they appear in the summary(model) Estimate. Values is if values1 and values2 are specified after each other in the same vector. Either values or values1 and values2 have to be specified.
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
values1	the values of case 1 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
values2	the values of case 2 as vector in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined.
type	OPTIONAL only simulation is supported for tobit()

**Details**

The function makes a simulation for the two cases and compares them to each other.

**Value**

The output is a matrix have in the first column the predicted values, in the second column the lower value of the confidence interval and in the third column the upper value of the confidence interval.

**Author(s)**

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
library(AER)
model1 = tobit(Age ~ Height + Pulse, right = 65, data=MASS::survey)
summary(model1)
# comparing a person with the height 150cm to 151cm
dc(model1, values1 = c(1,160,80),
  values2 = c(1,170,80))
# the difference is not significant, because the confidence interval
# includes the 0
```

---

predicts

*predicted values and discrete change*

---

**Description**

The function calculates the predicted values and the difference of a range of cases with the confidence interval. It can be used for any glm, polr or multinom model.

**Usage**

```
predicts(model, values, position = NULL, sim.count = 1000, conf.int = 0.95,
  sigma = NULL, set.seed = NULL, doPar = TRUE,
  type = c("any", "simulation", "bootstrap"))
```

**Arguments**

- |        |   |
|--------|---|
| model  | the model-Object generated with glm(), glm.nb(), polr() or multinom()   |
| values | The values of cases as character in the order how they appear in the summary(model) Estimate. The values must be in the following way: "value1;value2;value3;...". Each one of the values can be one of the following: <ul style="list-style-type: none"> <li>• <b>"mean"</b>: takes the mean of that variable (can only be used when the variable is numeric)</li> <li>• <b>"median"</b>: takes the median of that variable (can only be used when the variable is numeric)</li> </ul> |



- **"mode"**: takes the mode of that variable
- **"Q4"**: takes the quartiles (0,0.25,0.5,0.75,1) of that variable (other number for other quartiles)
- **"min"**: takes the minimum of that variable
- **"max"**: takes the maximum of that variable
- **from-to,by**: takes all values from "from" to "to" with the distance "by" (for example: "160-180,5" -> 160,165,170,175,180)
- **from-to**: same as from-to,by with by=1 (for example: "2-8" -> 2,3,4,5,6,7,8)
- **value1,value2,value3,...**: takes the given values (for example: "160,180" -> 160,180)
- **value1**: takes the given value (for example: "5.34" -> 5.34)
- **log(from-to,by)**: takes the log of all values from "from" to "to" with the distance "by" (for example: "160-180,5" -> 160,165,170,175,180)
- **log(from-to)**: same as log(from-to,by) with by=1 (for example: "2-8" -> 2,3,4,5,6,7,8)
- **log(value1,value2,value3,...)**: takes the log of the given values (for example: "160,180" -> 160,180)
- **log(value1)**: takes the log of the given value (for example: "5.34" -> 5.34)
- **"F"**: takes all values of a factor/character
- **"F(1,4,7)"**: takes the first, fourth and seventh level of a factor/character
- **"F(2)"**: takes the second level of a factor/character

position	OPTIONAL which variable should be taken for the discrete change, the variable must have at least two values. default: only predicted probabilities
sim.count	OPTIONAL numbers of simulations to be done by the function. default: 1000
conf.int	OPTIONAL the confidence interval used by the function. default: 0.95
sigma	OPTIONAL the variance-covariance matrix, can be changed when having for example robust or clustered vcov. default: vcov(model)
set.seed	OPTIONAL set a seed for the random number generator
doPar	OPTIONAL if the code should run parallel if more than 2 cores are detected
type	OPTIONAL choose between simulation and bootstrap, "any" chooses between those two according to the number of cases (bootstrap if n < 1000)

### Details

The function makes a simulation for the all combination of cases and compares them to each other.

### Value

The output is a data.frame with the predicted values and discrete changes.

### Author(s)

Benjamin Schlegel, <kontakt@benjaminschlegel.ch>

**Examples**

```
## Not run:
model1 = glm(Sex ~ Height + Smoke + Pulse, data=MASS::survey, family=binomial(link=logit))
summary(model1)
# comparing person with hight 150 to 160, 160 to 170, 170 to 180, 180 to 190
# with all combination of(non-)smokers and a median of pulse
predicts(model1, "150-190,10;F;median", position = 1, doPar = FALSE)

## End(Not run)
```

# Index

## \* **models**

- basepredict, 2
- basepredict.glm, 3
- basepredict.lm, 4
- basepredict.multinom, 5
- basepredict.polr, 6
- basepredict.tobit, 8
- dc, 9
- dc.glm, 10
- dc.lm, 11
- dc.multinom, 12
- dc.polr, 14
- dc.tobit, 15
- predicts, 16

## \* **package**

- glm.predict-package, 2

- basepredict, 2
- basepredict.glm, 3
- basepredict.lm, 4
- basepredict.multinom, 5
- basepredict.polr, 6
- basepredict.tobit, 8

- dc, 9
- dc.glm, 10
- dc.lm, 11
- dc.multinom, 12
- dc.polr, 14
- dc.tobit, 15

- glm.predict (glm.predict-package), 2
- glm.predict-package, 2

- predicts, 16