Package: glm.predict (via r-universe)

October 22, 2024

Type Package

Title Predicted Values and Discrete Changes for Regression Models

Version 4.3-1.9000

Date 2024-08-23

Author Benjamin E. Schlegel [aut,cre]

Maintainer Benjamin E. 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() [generalized linear model], glm.nb() [negative binomial model], polr() [ordinal logistic model], vglm() [generalized ordinal logistic model], multinom() [multinomial model], tobit() [tobit model], svyglm() [survey-weighted generalised linear models] and lmer() [linear multilevel models] using Monte Carlo simulations or bootstrap. Reference: Bennet A. Zelner (2009) <doi:10.1002/smj.783>.

License GPL (>=2)

Depends R (>= 3.5.0), stats

Imports nnet, AER, survival, MASS, parallel, mlogit, dfidx, survey, lme4, VGAM

Suggests ggplot2, devtools, knitr, rmarkdown

LazyData true

URL https://github.com/benjaminschlegel/glm.predict/

Language en-US

VignetteBuilder knitr

Repository https://benjaminschlegel.r-universe.dev

RemoteUrl https://github.com/benjaminschlegel/glm.predict

RemoteRef HEAD

RemoteSha 81904a7a010e9bf2170b7841fb4f3b627850e314

Contents

`glm.predict-package` *Predicted Values and Discrete Changes for GLM*

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 *predicted value*

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 *predicted value*

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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` *predicted value*

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"), summary = TRUE)
```

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)
`summary`	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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.lmerMod *predicted value*

Description

The function calculates the (average) predicted value with the confidence interval for a lmer model.

Usage

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

Arguments

model	the model Object generated with lmer()
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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 = lme4::lmer(lr_self ~ age + gender + (1 | canton), data=selects2015)
summary(model1)
# predicted left-right position of a 18 year old woman.
basepredict(model1, c(1,18,1))
```

---

basepredict.mlogit *predicted value*

Description

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

Usage

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

Arguments

model	the model Object generated with mlogit()
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	type is ignored as only simulation is implemented
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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:
df_selects_withoutNA = selects2015 |>
  dplyr::filter(vote_choice != "other") |>
  dplyr::mutate(vote_choice = factor(vote_choice)) |>
  dplyr::select(age, gender, vote_choice, starts_with("lr_")) |>
  na.omit()
mlogit_data = dfidx::dfidx(df_selects_withoutNA, varying = 5:11,
                    sep = "_", shape = "wide",
                    choice = "vote_choice")

mlogit_data$distance = abs(mlogit_data$lr - mlogit_data$lr_self)

model1 = mlogit::mlogit(vote_choice ~ distance | lr_self +
                          gender, data = mlogit_data)
summary(model1)
# predicted probability of a left male person with a distance of 2
basepredict(model1, list(1, c(2, 5, 5, 5, 5, 5), 0, 0))

## End(Not run)
```

`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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 *predicted value*

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 *predicted value*

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"), summary = TRUE)
```

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()
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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))
```

---

basepredict.vglm *predicted value*

Description

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

Usage

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

Arguments

model	the model Object generated with vglm()
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). Note: bootstrap is very slow for vglm() models.
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 = VGAM::vglm(Smoke ~ Height + Pulse, data=data,
  family = cumulative(parallel=FALSE ~ Pulse), maxit=1000)
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)
```

---

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"), summary = TRUE)
```

Arguments

model	the model-Object generated with glm(), glm.nb(), polr() or 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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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.lmerMod *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 'lmerMod'
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"), summary = TRUE)
```

Arguments

model	the model-Object generated with lmer()
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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 = lme4::lmer(lr_self ~ age + gender + (1 | canton), data=selects2015)
summary(model1)
# comparing a 20 year old woman with a 20 year old man considering their left-right position
dc(model1, values1 = c(1,20,1),
  values2 = c(1,20,0))
```

---

dc.mlogit *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 mlogit model.

Usage

```
## S3 method for class 'mlogit'
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"), summary = TRUE)
```

Arguments

model	the model-Object generated with mlogit()
values	the values of case 1 and 2 as list 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 list in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined. Can be single values or a vector of values with the length of choices - 1 for condition variables.
values2	the values of case 2 as list in the order how they appear in the summary(model) Estimate. Has to be defined if values is not defined. Can be single values or a vector of values with the length of choices - 1 for condition variables.
type	type is ignored as only simulation is implemented
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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:
df_selects_withoutNA = selects2015 |>
  filter(vote_choice != "other") |>
  mutate(vote_choice = factor(vote_choice)) |>
  select(age, gender, vote_choice, starts_with("lr_")) |>
  na.omit()
mlogit_data = dfidx::dfidx(df_selects_withoutNA, varying = 5:11,
                    sep = "_", shape = "wide",
                    choice = "vote_choice")

mlogit_data$distance = abs(mlogit_data$lr - mlogit_data$lr_self)

model1 = mlogit::mlogit(vote_choice ~ distance | lr_self +
                          gender, data = mlogit_data)
summary(model1)
# predicted probability of a left male person with a distance of 2
dc(model1, list(1, c(2, 5, 5, 5, 5, 5), 0, 0, 1, c(2, 5, 5, 5, 5, 5), 10, 0))

## End(Not run)
```

`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"), summary = TRUE)
```

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)
`summary`	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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"), summary = TRUE)
```

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)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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"), summary = TRUE)
```

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()
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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
```

dc.vglm *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 vglm model.

Usage

```
## S3 method for class 'vglm'
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"), summary = TRUE)
```

Arguments

model	the model-Object generated with vglm()
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 < 500) (boostrap is very slow for vglm models)
summary	OPTIONAL if mean/quantiles should be return or all simulated values (default: TRUE)

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 = VGAM::vglm(Smoke ~ Height + Pulse, data=data,
  family = cumulative(parallel=FALSE ~ Pulse), maxit=1000)
summary(model1)
dc(model1, values1 = c(150,mean(MASS::survey$Pulse,na.rm=TRUE)),
  values2 = c(151,mean(MASS::survey$Pulse,na.rm=TRUE)), type = "simulation")
# all differences are significant as the confidence intervals do not include 0

## End(Not run)
```

---

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 = FALSE,
  type = c("any", "simulation", "bootstrap"))
```

Arguments

model	the model-Object generated with glm(), glm.nb(), polr(), multinom(), mlogit() or tobit()
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: • **"all"**: takes all unique values of that variable • **"mean"**: takes the mean of that variable (can only be used when the variable is numeric) • **"median"**: takes the median of that variable (assumes for factors that they are correctly ordered) • **"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 quantiles)

- **"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); also works for factors and takes the given levels form their position
- **value1,value2,value3,...**: takes the given values (for example: "160,180" –> 160,180); also works for factors and takes the given levels form their position
- **constant,|value1, value2, value3|, constant,...**: for conditional logit models, all levels should be held constant besides one level which must be surounded by || and containing all values to simulate for. (for example: "5,|1,2,3,4,5|,5,5,5,5" for a choice with 7 levels (6 without the base category))
- **value1**: takes the given value (for example: "5.34" –> 5.34); also works for factors and takes the given level form its position
- **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 < 500)

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)
```

---

selects2015 *Swiss Electoral Studies (Selects) 2015 - Post-electoral study*

Description

A simplified dataset of the Selects 2015 data. Selects 2015 was conducted after the elections to the national council in Switzerland on October 2015.

Usage

```
selects2015
```

Format

A data frame with 5337 rows and 15 variables:

gender the gender of the participant

age the age of the participant

canton the canton where the participant lives

education the highest education of the participant

participation Indicates if the participant participated in the national election or not

vote_choice The party the participant mainly voted for: SVP, FDP, CVP, SP, GPS, GLP, BDP, other

political_interest political interest of the participant, self declaration

lr_self left right self placement

lr_SVP left right placement of SVP

lr_FDP left right placement of FDP

lr_CVP left right placement of CVP

lr_SP left right placement of SP

lr_GPS left right placement of GPS

lr_GLP left right placement of GLP

lr_BDP left right placement of BDP

knowscale political knowledge scale between 0 and 4

opinion_social_expenses opinion about social expense

opinion_eu_membership opinion if Switzerland should join the EU

opinion_foreigners_swiss_equal opinion if foreigners should be treated equal to Swiss

opinion_environment_economy opinion if environment or economy is more important

opinion_nuclear_energy opinion on nuclear energy

opinion_high_income_taxes opinion on high income taxes

weight_total A weight to make the survey representative (design weight * turnout * vote_choice)

Source

Selects: Post-electoral study - 2015 [Dataset]. Distributed by FORS, Lausanne, 2016. https://forscenter.ch/projects/selects/ doi:10.23662/FORSDS7265

Index