Package ‘catregs’

June 11, 2024

Type Package

Title Post-Estimation Functions for Generalized Linear Mixed Models

Version 0.2.0

Description Several functions for working with mixed effects regression models for limited dependent variables. The functions facilitate post-estimation of model predictions or margins, and comparisons between model predictions for assessing or probing moderation. Additional helper functions facilitate model comparisons and implements simulation-based inference for model predictions of alternative-specific outcome models. See also, Melamed and Doan (2024, ISBN: 978-1032509518).

Depends R (>= 3.5.0)

License GPL-2 | GPL-3

Encoding UTF-8

LazyData true

Imports methods

Suggests tidyverse, ggplot2, MASS, emmeans, pscl, nnet, marginaleffects, ggpubr, knitr, rmarkdown, Epi, dplyr, ordinal, nlme, lme4

VignetteBuilder knitr

NeedsCompilation no

Author David Melamed [aut, cre] (<https://orcid.org/0000-0002-8821-7698>)

Maintainer David Melamed <dmmelamed@gmail.com>

Repository CRAN

Date/Publication 2024-06-11 12:30:10 UTC

Contents

compare.margins 2
count.fit 3
diagn 4
ess 5
essUK 6

first.diff.fitted 8
gss2016 10
LF06art 11
LF06travel 12
list.coef 13
logan 14
lr.test 15
margins.dat 16
margins.dat.clogit 18
margins.des 19
Mize19AH 20
Mize19GSS 22
rubins.rule 23
second.diff.fitted 24
wagepan 26

Index **28**

`compare.margins` *Compares two marginal effects (MEMs or AMEs). Estimate of uncertainty is from a simulated draw from a normal distribution.*

Description

Given two marginal effects (AMEs or MEMs), as estimated via the margins package or via first.diff.fitted, this function simulates draws from the distribution of MEs defined by the estimates and their standard error, and computes the overlap in the two distributions. The p-value refers to proportion of times the two draws overlapped.

Usage

```
compare.margins(margins,margins.ses,seed=1234,rounded=3,nsim=10000)
```

Arguments

`margins`	The two marginal effects that you want to compare.
`margins.ses`	The standard errors for the marginal effects you want to compare.
`seed`	Random number seed so that results are reproducible.
`rounded`	The number of decimal places to round the output. The default is 3.
`nsim`	The number of simulated AMEs to draw from each distribution. The default is 10,000.

Value

`differnce`	The observed difference in the two AMEs.
`p.value`	The p-value associated with the difference. This is the proportion of the simulated sample when the MEs overlapped.

Author(s)

David Melamed

Examples

```
data("essUK")
m1 <- glm(safe ~ religious + minority*female + age,data=essUK,family="binomial")
des<-margins.des(m1,expand.grid(minority=c(0,1),female=c(0,1)))
des
ma1<-suppressWarnings(as.data.frame(marginaleffects::avg_slopes(m1,
variables="female",newdata=data.frame(minority=0,religious=3.6024,age=53.146))))
ma2<-as.data.frame(marginaleffects::avg_slopes(m1,variables="female",
newdata=data.frame(minority=1,religious=3.6024,age=53.146)))
cames <- rbind(ma2,ma1)
compare.margins(margins=cames$estimate,margins.ses=cames$std.error)
```

---

count.fit — *Fits four different count models and compares them.*

Description

Given a Poisson model object, count.fit fits Poisson, negative binomial, zero-inflated Poisson, and zero-inflated negative binomial models to the data. It reports results of Vuong tests between the zero-inflated and non-zer-inflated models, summarizes the information criteria of the four models, summarizes the model output of the four models, creates a ggplot object of coefficient plots for each model, and creates a ggplot object of model residuals.

Usage

```
count.fit(m1,y.range,rounded=3,use.color="yes")
```

Arguments

`m1`	A Poisson regression model, as estimated via the glm function.
`y.range`	The observed response range for the count outcome. For example, if the observed range is 0 to 18, this would be 0:18
`rounded`	The number of decimal places to round the output. The default is 3.
`use.color`	Whether to use color in the ggplot objects. Default is "yes"

Value

`ic`	A data.frame summarizing the information criteria for the four models. Bayesian and Akaike's informaiton criteria are included.
`models`	A summary of the model estimates, including coefficients and standard errors.
`pic`	A ggplot object illustrating model residuals for each type of model.
`models.pic`	A ggplot object of coefficient plots from each type of model.

Author(s)

David Melamed

Examples

```
data("LF06art")
p1 <- glm(art ~ fem + mar + kid5 + phd + ment , family = "poisson", data = LF06art)
table(LF06art$art)
fit<-count.fit(p1,0:19)
names(fit)
```

---

diagn — *Computes diagnostics for generalized linear models.*

Description

Given a glm object, diagn returns case-level diagnostics. For logistic, probit, Poisson, and negative binomial models, it returns Pearson residuals, standardized Pearson residuals, the diagonal of the hat matrix, delta-beta (Cook's D), and deviance residuals. For zero-inflated and hurdle models, it returns the Pearson residual and the observation number.

Usage

```
diagn(model)
```

Arguments

`model`	A model object. The model should be regression model for limited dependent variables, such as a logistic regression.

Value

`out`	The output is a dataframe of diagnostic statistics. For logit, probit, Poisson, and negative binomial models, the output includes the Pearson residual (pearsonres), the diagonal of the Hat matrix (h), the standardized Pearson residual (stdpres), the delta-beta statistic (deltabeta), the observation number (obs), and the deviance residual (devres). For zero-inflated and hurdle models, the output includes the Pearson residual (pearsonres), and the observation number (obs).

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age + race2 + race3 +
race4 + income + ed1 + ed2 + ed3 + ed4,
family="binomial",data=Mize19AH)
head(diagn(m1))
```

---

ess *A subset of data from the European Social Survey*

Description

These are data from the European Social Survey used to illustrate mixed effects, multilevel, or hierarchical regression models.

Usage

```
data("ess")
```

Format

A data frame with 49519 observations on the following 37 variables.

`country` a character vector

`can.trust.people` a numeric vector

`people.try.fair` a numeric vector

`say.in.govt` a factor with levels `Not at all` `Very little` `Some` `A lot` `A great deal`

`trust.legal.sys` a numeric vector

`trust.police` a numeric vector

`vote` a factor with levels `Yes` `No` `Not eligible to vote`

`conservative` a numeric vector

`life.satisfaction` a numeric vector

`immigration.good.economy` a numeric vector

`happy` a numeric vector

`important.matters.people` a character vector

`walk.alone.dark` a factor with levels `Very unsafe` `Unsafe` `Safe` `Very safe`

`religious` a numeric vector

`ethnic.minority` a character vector

`num.children` a numeric vector

`ideal.age.parent` a numeric vector

`household.size` a numeric vector

`gender` a character vector

`age` a numeric vector

`marital` a character vector

`education` a numeric vector

`employment` a character vector

`income.decile` a numeric vector

`father.education` a character vector

`father.education.num` a numeric vector

`everyone.job.wanted` a numeric vector

`income.fairness` a factor with levels `Low, extremely unfair` `Low, very unfair` `Low, somewhat unfair` `Low, slightly unfair` `Fair` `High, slightly unfair` `High, somewhat unfair` `High, very unfair` `High, extremely unfair`

`under.over.paid` a factor with levels `Underpaid` `Right amount` `Overpaid`

`income.fairness.num` a numeric vector

`wealth.diff.fair` a factor with levels `Small, extremely unfair` `Small, very unfair` `Small, somewhat unfair` `Small, slightly unfair` `Fair` `Large, slightly unfair` `Large, somewhat unfair` `Large, very unfair` `Large, extremely unfair`

`wealth.differences` a factor with levels `Too little` `Just right` `Too much`

`gdp` a numeric vector

`urban.population` a numeric vector

`unemployment` a numeric vector

`alcolhol.consumption` a numeric vector

`suicide.rate` a numeric vector

Source

European Social Survey.

Examples

```
data(ess)
head(ess)
```

`essUK` *A subset of data from the European Social Survey*

Description

These are data from respondents in the United Kingdom from the European Social Survey. They are used to illustrate regression models for limited dependent variables.

Usage

```
data("essUK")
```

Format

A data frame with 2204 observations on the following 45 variables.

`country` a character vector

`can.trust.people` a numeric vector

`people.try.fair` a numeric vector

`say.in.govt` a factor with levels `Not at all` `Very little` `Some` `A lot` `A great deal`

`trust.legal.sys` a numeric vector

`trust.police` a numeric vector

`vote` a factor with levels `Yes` `No` `Not eligible to vote`

`conservative` a numeric vector

`life.satisfaction` a numeric vector

`immigration.good.economy` a numeric vector

`happy` a numeric vector

`important.matters.people` a character vector

`walk.alone.dark` a factor with levels `Very unsafe` `Unsafe` `Safe` `Very safe`

`religious` a numeric vector

`ethnic.minority` a character vector

`num.children` a numeric vector

`ideal.age.parent` a numeric vector

`household.size` a numeric vector

`gender` a character vector

`age` a numeric vector

`marital` a character vector

`education` a numeric vector

`employment` a character vector

`income.decile` a numeric vector

`father.education` a character vector

`father.education.num` a numeric vector

`everyone.job.wanted` a numeric vector

`income.fairness` a factor with levels `Low, extremely unfair` `Low, very unfair` `Low, somewhat unfair` `Low, slightly unfair` `Fair` `High, slightly unfair` `High, somewhat unfair` `High, very unfair` `High, extremely unfair`

`under.over.paid` a factor with levels `Underpaid` `Right amount` `Overpaid`

`income.fairness.num` a numeric vector

`wealth.diff.fair` a factor with levels `Small, extremely unfair` `Small, very unfair` `Small, somewhat unfair` `Small, slightly unfair` `Fair` `Large, slightly unfair` `Large, somewhat unfair` `Large, very unfair` `Large, extremely unfair`

`wealth.differences` a factor with levels `Too little` `Just right` `Too much` `gdp` a numeric vector

`urban.population` a numeric vector

`unemployment` a numeric vector

`alcolhol.consumption` a numeric vector

`suicide.rate` a numeric vector

`safe` a numeric vector

`minority` a numeric vector

`female` a numeric vector

`divorced` a numeric vector

`married` a numeric vector

`widow` a numeric vector

`highinc` a numeric vector

`age2` a numeric vector

Source

European Social Survey.

Examples

```
data(essUK)
head(essUK)
```

`first.diff.fitted` *Computes the first difference in fitted values, or a series of first differences. Inference in supported via the delta method or bootstrapping.*

Description

first.diff.fitted computes first differences between fitted values from a regression model.

Supported models include OLS regression via lm, logistic regression via glm, Poisson regression via glm, negative binomial regression via MASS:glm.nb, ordinal logistic regression via MASS::polr, partial proportional odds models via vgam::vglm, multinomial logistic regression via nnet::multinom, zero-inflated Poisson or negative binomial regression via pscl::zeroinfl, hurdle Poisson or negative binomial regression via pscl::hurdle, mixed effects logistic regression via lme4/lmerTest::glmer, mixed effects Poisson regression via lme4/lmerTest::glmer, mixed effects negative binomial regression via lme4/lmerTest::glmer.nb, and mixed effects ordinal logistic regression via ordinal::clmm.

Usage

```
first.diff.fitted(mod,design.matrix,compare,alpha=.05,rounded=3,
bootstrap="no",num.sample=1000,prop.sample=.9,data,seed=1234,cum.probs="no")
```

Arguments

- `mod` — A model object. The model should be regression model for limited dependent variables, such as a logistic regression.
- `design.matrix` — Design matrix of values for the independent variables in the regression model.
- `compare` — Pairs of rows in the design matrix to use for computing the fitted values. The first difference between the fitted values is then computed. For example, compare=c(4,2) means to compute the difference in the fitted values between predictions for row 4 of the design matrix and row 2 of the design matrix. If more than two rows are provided, the function uses them two at a time and computes multiple first differences.
- `alpha` — The alpha value for confidence intervals. Default is .05.
- `rounded` — The number of decimal places to round the output. The default is 3.
- `bootstrap` — By default, inference is based on the Delta Method, as implemented in the marginaleffects package. Alternatively, inference can be based upon a bootstrapped sampling distirbution. To do so, change this to "yes." Note that bootstrapping is only supported for one first difference at a time.
- `num.sample` — num.sample is the number samples drawn to compute the sampling distibution when using bootstrapping. Default is 1,000
- `prop.sample` — prop.sample is the proportion of the original sample (with replacement) to include in the sampling distibution samples when using bootstrapping. Default is .9
- `data` — For nonparametric inference, provide the data used in the original model statement.
- `seed` — For models using bootstrapped inference. The seed ensures reproducible results across runs. Default is 1234, but may be changed.
- `cum.probs` — For ordinal logistic regression models, including mixed effects models, do you want the first differences to be based on probabilities of the response categories or cumulative probabilities of the response categories. The default is cum.probs=="no" corresponding to non-cumulative probabilities. Change cum.probs to "yes" for cumulative probabilities.

Value

- `out` — If using parametric inference (delta method): output is a dataframe including the first fitted value ("fitted1"), the second fitted value ("fitted2"), the difference in fitted values ("first.diff"), the standard error ("std.error"), the lower limit ("ll"), and upper limit ("ul") of the confidence interval. Of course, ll and ul are based on the alpha level. If using nonparametric inference (bootstrapping): output is a list of objects. obs.diff is the observed difference in the response or fitted values. boot.dist is the sorted bootstrapped distribution of differences in the samples. mean.boot.dist is the average of the differences in the responses or fitted values. sd.boot.dist is the standard deviation of the sampling distribution. ci.95 is the Lower and Upper limits of the confidence interval; despite it's name, the confidence interval is based upon the alpha level. model.class is just the class of the model that was used to generate the fitted values.

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age + race2 +
race3 + race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
des2<-margins.des(m1,expand.grid(woman=c(0,1),parrole=c(0,1)))
des2
first.diff.fitted(m1,des2,compare=c(4,2))
# Pr(Drink | Mothers) - Pr(Drink | Childless Women)

first.diff.fitted(m1,des2,compare=c(3,1))
# Pr(Drink | Fathers) - Pr(Drink | Childless Men)
```

---

gss2016 *Data from the 2016 General Social Survey.*

Description

Limited date from the 2016 General Social Survey on respondent and paternal class and occupational classifications.

Usage

```
data("gss2016")
```

Format

A data frame with 12498 observations on the following 13 variables.

`pclass` a factor with levels `Unskilled Manual` `Skilled Manual` `Self-Employed` `Non-Manual/Service` `Professional, Lower` `Professional, Higher`

`sclass` a factor with levels `Unskilled Manual` `Skilled Manual` `Self-Employed` `Non-Manual/Service` `Professional, Lower` `Professional, Higher`

`educ` a numeric vector

`race` a character vector

`id` a numeric vector

`occ2` a character vector

`occ` a numeric vector

`unskmanual` a numeric vector

`skmanual` a numeric vector

`selfemp` a numeric vector

`service` a numeric vector

`proflow` a numeric vector

`profhigh` a numeric vector

Source

The General Social Survey.

Examples

```
data(gss2016)
head(gss2016)
```

LF06art *Data to replicate Long and Freese's (2006) count models (pp354-414)*

Description

For replication purposes between Stata and R. Long and Freese (2006) analyze these data to illustrate regression models for count dependent variables.

Usage

```
data("LF06art")
```

Format

A data frame with 915 observations on the following 6 variables.

`art` count response

`fem` dummy for sex

`mar` dummy for married

`kid5` number of children under five

`phd` a numeric vector

`ment` a numeric vector

Source

Long, Scott J. and Jeremy Freese. 2006. "Regression Models for Categorical Dependent Variables Using Stata." Austin, TX: Stata Press

Examples

```
data(LF06art)
head(LF06art)
```

LF06travel *Travel time example data for alternative-specific outcomes.*

Description

Example data, also used in Long and Freese (2006), to illustrate conditional or fixed effects logistic regression. Also refered to as alternative-specific outcomes.

Usage

```
data("LF06travel")
```

Format

A data frame with 456 observations on the following 13 variables.

`id` a numeric vector denoting nested units (individuals) or strata

`mode` a numeric vector denoting mode of transit

`train` a dummy variable for selecting the train

`bus` a dummy variable for selecting the bus

`car` a dummy variable for selecting a car

`time` a numeric vector denoting transit time

`invc` a numeric vector denoting invertng cost

`choice` a numeric vector denoting the choice of travel, i.e. the dependent variable

`ttme` a numeric vector

`invt` a numeric vector

`gc` a numeric vector

`hinc` a numeric vector

`psize` a numeric vector

Source

Long, Scott J. and Jeremy Freese. 2006. "Regression Models for Categorical Dependent Variables Using Stata." Austin, TX: Stata Press

Examples

```
data(LF06travel)
head(LF06travel)
```

`list.coef` *Transform glm and mixed model objects into model summaries that include coefficients, standard errors, exponentiated coefficients, confidence intervals and percent change.*

Description

Given a glm model object or a mixed model model object, the function computes and returns: coefficients, standard errors, z-scores, confidence intervals, p-values, exponentiated coefficients, confidence intervals for exponentiated coefficients, and percent change.

Supported models include logistic regression via the glm function, ordinal regression via mass::polr, multinomial regression via nnet:multinom, Poisson regression via the glm function, negative binomial regression via mass::glm.nb, mixed effects models for continuous outcomes with serial correlation via nlme::lme, mixed effects logistic and poisson regression via lme4::glmer, mixed effects negative binomial regression via lme4::glmer.nb, and mixed effects ordinal regression via ordinal::clmm.

Usage

```
list.coef(model,rounded=3,alpha=.05)
```

Arguments

- `model` — A model object. The model should be regression model for limited dependent variables, such as a logistic regression, or a mixed model from nlme or lme4/lmerTest.
- `rounded` — The number of decimal places to round the output. The default is 3.
- `alpha` — The alpha value for confidence intervals. Default is .05.

Value

- `b` — The estimated model coefficients from the model object.
- `S.E.` — The estimated model standard errors from the model object.
- `Z` — The z-statistic corresponding to the coefficient.
- `LL CI` — Given the coefficient, standard error and alpha value (default=.05), the lower limit of the confidence interval around the coefficient is reported.
- `UL CI` — Given the coefficient, standard error and alpha value (default=.05), the upper limit of the confidence interval around the coefficient is reported.
- `p-val` — The p-value associated with the z-statistic.
- `exp(b)` — The exponentiated model coefficients. That is, odds ratios in the case of a logistic regression, or incidence rate ratios in the case of a count model.
- `LL CI for exp(b)` — Given the exponentiated coefficient, standard error and alpha value (default=.05), the lower limit of the confidence interval around the exponentiated coefficient is reported.

`UL CI for exp(b)`
: Given the exponentiated coefficient, standard error and alpha value (default=.05), the upper limit of the confidence interval around the exponentiated coefficient is reported.

`Percent`
: The coefficients in terms of percent change. That is, 100*(exp(coef(model))-1)

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age + race2 +
race3 + race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
list.coef(m1,rounded=4)
list.coef(m1,rounded=4,alpha=.01)
```

`logan` *Replication data for Logan's (1983) application of conditional logistic regression to mobility processes.*

Description

Replication data for Logan's (1983) application of conditional logistic regression to mobility processes.

Usage

```
data("logan")
```

Format

A data frame with 4190 observations on the following 11 variables.

- `occupation` respondent occupation with levels `farm` `operatives` `craftsmen` `sales` `professional`
- `focc` paternal occupation, i.e., father's occupation with levels `farm` `operatives` `craftsmen` `sales` `professional`
- `education` a numeric vector
- `race` a factor with levels `non-black` `black`
- `id` a numeric vector
- `tocc` a factor with levels `farm` `operatives` `craftsmen` `sales` `professional`
- `case` a numeric vector
- `craftsmen` a numeric vector
- `farm` a numeric vector
- `operatives` a numeric vector
- `professional` a numeric vector

References

Logan, John A. 1983. “A Multivariate Model for Mobility Tables.” American Journal of Sociology 89(2):324–349.

Examples

```
data(logan)
head(logan)
```

`lr.test` *LR test of nested models.*

Description

Given two model objects estimated with a logLik() solution, the function computes an LR test. The order of models, in terms of full or reduced, does not matter. Supported models include OLS regression via lm, logistic regression via glm, Poisson regression via glm, negative binomial regression via MASS:glm.nb, ordinal logistic regression via MASS::polr, partial proportional odds models via vgam::vglm, multinomial logistic regression via nnet::multinom, zero-inflated Poisson or negative binomial regression via pscl::zeroinfl, and hurdle Poisson or negative binomial regression via pscl::hurdle. For mixed models, use base R’s anova function.

Usage

```
lr.test(full.model,reduced.model)
```

Arguments

`full.model`	A model object with unconstrained parameters. The order does not matter however.
`reduced.model`	A model object with constrained parameters. The order does not matter however.

Value

`LL Full`	The solution to the log-likelihood function of the full model.
`LL Reduced`	The solution to the log-likelihood function of the reduced model.
`G2/LR Statistic`	The test statistic. Computed as -2 x (ll(Full) - ll(Reduced).
`DF`	The DF for the test. The number of parameters constrained to zero in the reduced model.
`p-value`	The p-value associated with the LR test. The null hypothesis is that the constrained parameters are jointly equal to zero.

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age +
race2 + race3 + race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
m0 <- glm(alcB ~woman + parrole + age +
race2 + race3 + race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
lr.test(m1,m0)
lr.test(m0,m1) # Same Result
```

`margins.dat` *Add model predictions, standard errors and confidence intervals to a design matrix for a model object.*

Description

Given a model object and a design matrix, this creates a data.frame of the design matrix, with model predictions, standard errors and lower/upper limits of confidence intervals around the predictions. This is a wrap around function for calls to emmeans; it adjusts the emmeans equation to return fitted values on the response scale.

Supported models include OLS regression via lm, logistic regression via glm, Poisson regression via glm, negative binomial regression via MASS:glm.nb, ordinal logistic regression via MASS::polr, partial proportional odds models via vgam::vglm, multinomial logistic regression via nnet::multinom, zero-inflated Poisson or negative binomial regression via pscl::zeroinfl, hurdle Poisson or negative binomial regression via pscl::hurdle, linear mixed effects models with or without serial correlation via nlme::lme, linear mixed effects models via lme4/lmerTest::lmer, mixed effects logistic regression via lme4/lmerTest::glmer, mixed effects Poisson regression via lme4/lmerTest::glmer, mixed effects negative binomial regression via lme4/lmerTest::glmer.nb, and mixed effects ordinal logistic regression via ordinal::clmm.

For mixed effects ordinal logistic regression models, as estimated via the ordinal package, the outcome variable in the regression model (i.e., the clmm function) needs to be named "dv."

Given one of these model objects and an appropiate design matrix, the function detects the model response type and generates fitted values on the response scale. For example, a logistic regression model returns predicted probabilities, and a Poisson model returns the fitted counts. In addition to the fitted values, the function returns the delta method standard error for the fitted value and a confidence interval. The confidence interval is 95 percent by default, but that may be changed by the user.

Usage

```
margins.dat(mod,des,alpha=.05,rounded=3,cumulate="no",
pscl.data=data,num.sample=1000,prop.sample=.9,seed=1234)
```

Arguments

- `mod` A regression model object.
- `des` Design matrix of values for the independent variables in the regression model.
- `alpha` The alpha value for confidence intervals. Default is .05.
- `rounded` The number of decimal places to round the output. The default is 3.
- `cumulate` Whether the fitted values should reflect cumulative probabilities. Default is "no." Intended for predicted probabilities drawn from ordinal logistic regression models (polr), ordinal logistic regression models assuming partial proportional odds (vgam), or mixed effects ordinal logistic regression (clmm).
- `pscl.data` If generating predicted counts from Zero-Inflated models (either Poisson or negative binomial), you need to include the data that was specified in the model statement, i.e., the data in the "mod" object.
- `num.sample` num.sample is the number samples drawn to compute the sampling distibution.
- `prop.sample` prop.sample is the proportion of the original sample to include in the sampling distibution samples. Default is .9
- `seed` For models using bootstrapped inference. The seed ensures reproducible results across runs. Default is 1234, but may be changed.

Value

- `marginsdat` Returns a data.frame containing the design matrix and additional columns for the fitted value on the response scale, the delta method standard error (except zero-truncated models, which are bootstrapped), and the lower/upper limits on confidence intervals around the fitted value.

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age +
race2 + race3 + race4 + income + ed1 +
ed2 + ed3 + ed4,family="binomial",data=Mize19AH)
des2<-margins.des(m1,expand.grid(woman=c(0,1),parrole=c(0,1)))
margins.dat(m1,des2,rounded=5)
des1 <- margins.des(m1,expand.grid(parrole=1,woman=1))
margins.dat(m1,des1,rounded=5)
des3 <- margins.des(m1,expand.grid(age=seq(20,75,5),parrole=c(0,1)))
a<- margins.dat(m1,des3,rounded=5)
a # Then plot a using ggplot
```

margins.dat.clogit *Computes predicted probabilities for conditional and rank-order/exploded logistic regression models. Inference is based upon simulation techniques (requires the MASS package). Alternatively, bootstrapping is an option for conditional logistic regression models.*

Description

Given a model object and a design matrix, this creates a data.frame of the design matrix, with predicted probabilities for each response category. Inferential information about the predict probabilities is supported with simulation. Bootstrapping may be added as an option for conditional logistic regression models.

Usage

```
margins.dat.clogit(mod,design.matrix,run.boot="no",num.sample=1000,
prop.sample=.9,alpha=.05,seed=1234,rounded=3)
```

Arguments

`mod`	A conditional logistic regression model as estimated in the Epi package or an Exploded logistic regression model as estimated in the mlogit package.
`design.matrix`	Design matrix of values for the independent variables in the regression model. Unlike the design matrices in the margins.des function, the design matrix for a conditional logistic regression entails multiple rows, corresponding to the number of response options.
`run.boot`	Whether to compute confidence intervals around the predicted probabilities using bootstrapping. Defaul is "no."
`num.sample`	num.sample is the number samples drawn to compute the sampling distibution.
`prop.sample`	prop.sample is the proportion of the original sample to include in the sampling distibution samples. Default is .9
`alpha`	The alpha value for confidence intervals. Default is .05.
`seed`	Sets a seed so that random results are reprodicible.
`rounded`	How many decimal places to show in the output.

Value

`des`	Returns a data.frame containing the design matrix and additional columns for the predicted probabilities.
`boot.dist`	The full bootstrapped distribution for the probabilities.

Author(s)

David Melamed

Examples

```
data("LF06travel")
m1 <- Epi::clogistic(choice ~ train + bus +
time  + invc, strata=id, data=LF06travel)
design <- data.frame(train=c(0,0,1),bus=c(0,1,0),time=200,invc=20)
design
margins.dat.clogit(m1,design)
```

`margins.des` *Creates a design matrix of idealized data for illustrating model predictions.*

Description

Create a data frame of idealized data for making model predictions/predicted margins that will be used with margins.dat for generating/plotting model predictions. Given a model object (generalized linear model or generalized linear mixed model), a grid of independent variable values, and a list of any variables (factor variables in particular) to exclude from the design matrix, the function returns the design matrix as a data.frame object. All covariates are set to their means in the data used to estimate the model object. If there are factors in the model, they need to be excluded using the "excl" option.

Supported models include OLS via the lm function, logistic and Poisson regression via the glm function, negative binomial regression via MASS::glm.nb, ordinal logistic regression via MASS::polr, multinomial logistic regression via nnet::multinom, partial proportional odds models via vgam::vglm, linear mixed effects models with or without serial correlation specified via nlme::lme, mixed effects logistic regression via lme4::glmer, mixed effects Poisson regression via lme4::glmer, mixed effects negative binomial regression via lme4::glmer.nb, and mixed effects ordinal logistic regression via ordinal::clmm. Zero-inflated and hurdle models are also supported via pscl::zeroinfl and pscl::hurdle, respectively.

For multinomial regression model, as estimated via the nnet package, you need to provide the data used in the nnet function that defined the model. For partial proportional odds models, as estimated via the vgam package, you need to specify an ordinal model via MASS::polr and provide that model to the margins.des function (the data for the model are not part of a vgam object.) For mixed effects logistic, Poisson, or negative binomial regression models, as estimated by lme4/lmerTest, you need to provide the data used in the glmer function that defined the model. For mixed effects ordinal logistic regression models, as estimated via the ordinal package, the outcome variable in the regression model (i.e., the clmm function) needs to be named "dv."

Usage

```
margins.des(mod,ivs,excl="nonE",data)
```

Arguments

`mod` A model object. The model should be regression model for limited dependent variables, such as a logistic regression. Specifically, supported models include lm, glm, polr, multinom, vgam, zeroinf (pscl), amd hurdle (pscl). vgam is supported for partial proportional odds models, not models for count outcomes. zerotrunc is only supported with bootstrapped inference, and may take a while.

ivs	This should be an 'expand.grid' statement including all desired variables and their corresponding levels in the design matrix.
excl	If you want to exclude covariates from the design matrix, you can list them here. This is designed to exclude factor variables from the design matrix, as they do not have means, but may be useful in other specialized cases. Default is "nonE," corresponding to excluding none of the variables.
data	If the model is a multinomial model, you also need to provide the data. This is because nnet objects do not include the relevant information for computing the means of covariates.

Value

design	Returns a data.frame containing the design matrix for model predictions.

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age + race2 +
race3 + race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
des1 <- margins.des(m1,expand.grid(parrole=1,woman=1))
des1
des2<-margins.des(m1,expand.grid(woman=c(0,1),parrole=c(0,1)))
des2
des3 <- margins.des(m1,expand.grid(age=seq(20,75,5),parrole=c(0,1)))
des3
```

Mize19AH *Add-Health Data analzed in Mize (2019)*

Description

Mize (2019) illustrates how to establish moderation in the context of regression models for limited dependent variables. He illustrates using AddHealth data and provides Stata code to replicate the results. Catregs functions can replicate these results in R.

Usage

```
data("Mize19AH")
```

Format

A data frame with 4307 observations on the following 29 variables.

- `AID` a numeric vector
- `race` a numeric vector
- `age` a numeric vector
- `educ` a numeric vector
- `degree` a numeric vector
- `college` a numeric vector
- `health` a numeric vector
- `role` a numeric vector
- `workrole` a numeric vector
- `parrole` a numeric vector
- `income` a numeric vector
- `wages` a numeric vector
- `logwages` a numeric vector
- `depB` a numeric vector
- `alcB` a numeric vector
- `woman` a numeric vector
- `edyrs` a numeric vector
- `whiteB` a numeric vector
- `X_est_prno` a numeric vector
- `X_est_prpar` a numeric vector
- `X_est_alcedmod` a numeric vector
- `X_est_alcmod` a numeric vector
- `race2` a numeric vector
- `race3` a numeric vector
- `race4` a numeric vector
- `ed1` a numeric vector
- `ed2` a numeric vector
- `ed3` a numeric vector
- `ed4` a numeric vector

Source

Mize, Trenton D. 2019. "Best Practices for Estimating, Interpreting, and Presenting Nonlinear Interaction Effects" Sociological Science 6: 81-117.

Examples

```
data(Mize19AH)
head(Mize19AH)
```

`Mize19GSS` *General Social Survey Data analzed in Mize (2019)*

Description

Mize (2019) illustrates how to establish nonlinear moderation in the context of regression models. He illustrates using General Social Survey (GSS) data and provides Stata code to replicate the results. Catregs functions can replicate these results in R.

Usage

```
data("Mize19GSS")
```

Format

A data frame with 19337 observations on the following 42 variables.

`nosameB` a numeric vector

`sameokB` a numeric vector

`polviews` a character vector

`age` a numeric vector

`age10` a numeric vector

`year` a numeric vector

`id` a numeric vector

`degree` a numeric vector

`race` a numeric vector

`partyid` a character vector

`natspac` a character vector

`natenvir` a character vector

`natheal` a character vector

`natcity` a character vector

`natcrime` a character vector

`natdrug` a character vector

`nateduc` a character vector

`natrace` a character vector

`natarms` a character vector

`nataid` a character vector

`natfare` a character vector

`health` a character vector

`helpnotB` a character vector

`conserv` a character vector

`polviews3` a character vector

`employed` a numeric vector

`male` a numeric vector

`woman` a numeric vector

`white` a numeric vector

`college` a numeric vector

`married` a numeric vector

`parent` a character vector

`edyrs` a numeric vector

`income` a numeric vector

`hrswork` a character vector

`parttime` a character vector

`wages` a numeric vector

`conviewSS` a numeric vector

`year2` a numeric vector

`yearcat` a numeric vector

`year1976` a numeric vector

`year1976.2` a numeric vector

Source

Mize, Trenton D. 2019. "Best Practices for Estimating, Interpreting, and Presenting Nonlinear Interaction Effects" Sociological Science 6: 81-117.

Examples

```
data(Mize19GSS)
head(Mize19GSS)
```

`rubins.rule` *Aggregate Standard Errors using Rubin's Rule.*

Description

The function takes a vector of standard error estimates and it pools them using Rubin's rule.

Usage

```
rubins.rule(std.errors)
```

Arguments

`std.errors` A vector of standard errors to be aggregated using Rubin's rule.

Value

`r.r.std.error` The aggregated standard error.

Author(s)

David Melamed

References

Rubin, Donald B. 2004. Multiple Imputation for Nonresponse in Surveys. Vol. 81. John Wiley & Sons.

`second.diff.fitted` *Computes the second difference in fitted values. Inference in supported via the delta method or bootstrapping.*

Description

second.diff.fitted computes the second differences between fitted values, that is, the difference between two first differences, from a regression model.

Supported models include OLS regression via lm, logistic regression via glm, Poisson regression via glm, negative binomial regression via MASS:glm.nb, ordinal logistic regression via MASS::polr, partial proportional odds models via vgam::vglm, multinomial logistic regression via nnet::multinom, zero-inflated Poisson or negative binomial regression via pscl::zeroinfl, hurdle Poisson or negative binomial regression via pscl::hurdle, mixed effects logistic regression via lme4/lmerTest::glmer, mixed effects Poisson regression via lme4/lmerTest::glmer, mixed effects negative binomial regression via lme4/lmerTest::glmer.nb, and mixed effects ordinal logistic regression via ordinal::clmm.

Usage

```
second.diff.fitted(mod,design.matrix,compare,alpha=.05,rounded=3,
bootstrap="no",num.sample=1000,prop.sample=.9,
data,seed=1234,cum.probs="no")
```

Arguments

`mod` A model object. The model should be regression model for limited dependent variables, such as a logistic regression.

`design.matrix` Design matrix of values for the independent variables in the regression model.

compare	A set of four rows in the design matrix to use for computing the fitted values that are used in the calculation of second differences. For example, compare(a,b,c,d) results in computing the fitted values for rows a, b, c, and d of the design matrix, respectively, and then computing the following second difference: (a - b) - (c - d). Only four rows may be compared at a time.
alpha	The alpha value for confidence intervals. Default is .05.
rounded	The number of decimal places to round the output. The default is 3.
bootstrap	By default, inference is based on the Delta Method, as implemented in the marginaleffects package. Alternatively, inference can be based upon a bootstrapped sampling distirbution. To do so, change this to "yes"
num.sample	num.sample is the number samples drawn to compute the sampling distibution when using bootstrapping. Default is 1,000
prop.sample	prop.sample is the proportion of the original sample to include in the sampling distibution samples when using bootstrapping. Default is .9
data	For nonparametric inference, provide the data used in the original model statement.
seed	For models using bootstrapped inference. The seed ensures reproducible results across runs. Default is 1234, but may be changed.
cum.probs	For ordinal logistic regression models, including mixed effects models, do you want the first differences to be based on probabilities of the response categories or cumulative probabilities of the response categories. The default is cum.probs=="no" corresponding to non-cumulative probabilities. Change cum.probs to "yes" for cumulative probabilities.

Value

out	If using parametric inference (delta method): output is a dataframe including the second difference in fitted values ("est"), the standard error ("std.error"), the lower limit ("ll"), and upper limit ("ul") of the confidence interval. Of course, ll and ul are based on the alpha level. If using nonparametric inference (bootstrapping): output is a list of objects. obs.diff is the observed second difference in the response or fitted values. boot.dist is the sorted bootstrapped distribution of second differences in the samples. mean.boot.dist is the average of the second differences in the responses or fitted values. sd.boot.dist is the standard deviation of the sampling distribution. ci.95 is the Lower and Upper limits of the confidence interval; despite it's name, the confidence interval is based upon the alpha level. model.class is just the class of the model that was used to generate the fitted values.

Author(s)

David Melamed

Examples

```
data("Mize19AH")
m1 <- glm(alcB ~woman*parrole + age + race2 + race3 +
```

```
race4 + income + ed1 + ed2 + ed3 +
ed4,family="binomial",data=Mize19AH)
des2<-margins.des(m1,expand.grid(woman=c(0,1),parrole=c(0,1)))
des2
second.diff.fitted(m1,des2,compare=c(4,2,3,1),rounded=5)
# [Pr(Drink | Mothers) - Pr(Drink | Childless Women)] -
# [Pr(Drink | Fathers) - Pr(Drink | Childless Men)]

# Note that this is reported as the "Second Difference" in
# Table 3 of Mize (2019: 104, "Best Practices for Estimating,
# Interpreting, and Presenting Nonlinear Interaction Effect.
# Sociological Science. 6(4): 81-117.")
```

`wagepan` *Data to illustrate mixed effects regression models with serial correlation.*

Description

Replication data illustrating serial correlation specifications to adjust for correlated residuals.

Usage

```
data("wagepan")
```

Format

A data frame with 4360 observations on the following 51 variables.

`nr` a numeric vector

`year` a numeric vector

`agric` a numeric vector

`black` a numeric vector

`bus` a numeric vector

`construc` a numeric vector

`ent` a numeric vector

`exper` a numeric vector

`fin` a numeric vector

`hisp` a numeric vector

`poorhlth` a numeric vector

`hours` a numeric vector

`manuf` a numeric vector

`married` a numeric vector

`min` a numeric vector

nrthcen a numeric vector

nrtheast a numeric vector

occ1 a numeric vector

occ2 a numeric vector

occ3 a numeric vector

occ4 a numeric vector

occ5 a numeric vector

occ6 a numeric vector

occ7 a numeric vector

occ8 a numeric vector

occ9 a numeric vector

per a numeric vector

pro a numeric vector

pub a numeric vector

rur a numeric vector

south a numeric vector

educ a numeric vector

tra a numeric vector

trad a numeric vector

union a numeric vector

lwage a numeric vector

d81 a numeric vector

d82 a numeric vector

d83 a numeric vector

d84 a numeric vector

d85 a numeric vector

d86 a numeric vector

d87 a numeric vector

expersq a numeric vector

r a numeric vector

num a numeric vector

number a numeric vector

mn_lwage a numeric vector

yeart a numeric vector

yeart2 a numeric vector

yeart3 a numeric vector

Examples

```
data(wagepan)
head(wagepan)
```

Index

∗ **datasets**
ess, 5
essUK, 6
gss2016, 10
LF06art, 11
LF06travel, 12
logan, 14
Mize19AH, 20
Mize19GSS, 22
wagepan, 26

compare.margins, 2
count.fit, 3

diagn, 4

ess, 5
essUK, 6

first.diff.fitted, 8

gss2016, 10

LF06art, 11
LF06travel, 12
list.coef, 13
logan, 14
lr.test, 15

margins.dat, 16
margins.dat.clogit, 18
margins.des, 19
Mize19AH, 20
Mize19GSS, 22

rubins.rule, 23

second.diff.fitted, 24

wagepan, 26