Package ‘mclogit’

Type Package
Title Mixed Conditional Logit Models
Version 0.6.1
Date 2018-09-26
Author Martin Elff
Maintainer Martin Elff <mclogit@elff.eu>
Description Specification and estimation of conditional logit models of binary responses and multinomial counts is provided, with or without random effects. The current implementation of the estimator for random effects variances uses a Laplace approximation (or PQL) approach and thus should be used only if groups sizes are large.
License GPL-2
Depends stats, Matrix
Imports memisc, methods
LazyLoad Yes
BugReports http://github.com/melff/mclogit/issues
RoxygenNote 5.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2018-09-27 07:40:07 UTC

R topics documented:

electors ................................................................. 2
getSummary-methods ......................................... 3
mblogit ............................................................... 4
mclogit ............................................................... 6
mclogit.control ................................................. 8
mclogit.fit ....................................................... 8
Transport .......................................................... 9
Index 10

electors  Class, Party Position, and Electoral Choice

Description

This is an artificial data set on electoral choice as influenced by class and party positions.

Usage

data(electors)

Format

A data frame containing the following variables:

- **class**: class position of voters
- **party**: party that runs for election
- **Freq**: frequency by which each party list is chosen by members of each class
- **time**: time variable, runs from zero to one
- **econ.left**: economic-policy "leftness" of each party
- **welfare**: emphasis of welfare expansion of each party
- **auth**: position on authoritarian issues

Examples

data(electors)

summary(mclogit(
  cbind(Freq,interaction(time,class))~econ.left+welfare+auth,
  data=electors))

summary(mclogit(
  cbind(Freq,interaction(time,class))~econ.left/class+welfare/class+auth/class,
  data=electors))

summary(mclogit(
  cbind(Freq,interaction(time,class))~econ.left/class+welfare/class+auth/class,
  random~-1|party.time,
  data=within(electors,party.time<-interaction(party,time))))

summary(mclogit(
  cbind(Freq,interaction(time,class))~econ.left/(class*time)+welfare/class+auth/class,
  random=-1|party.time,
  data=within(electors,{
    party.time <- interaction(party,time)
    econ.left.sq <- (econ.left-mean(econ.left))^2
  }))))
Description

`getSummary` methods for use by `mtable`.

Usage

```r
## S3 method for class 'mblogit'
getSummary(mb, alpha=.05,
   ...)
## S3 method for class 'mclogit'
getSummary(mc, alpha=.05,
   rearrange=NULL,
   ...)
```

Arguments

- `obj`: an object returned by `mblogit` or `mclogit`
- `alpha`: level of the confidence intervals; their coverage should be 1-alpha/2
- `rearrange`: an optional named list of character vectors. Each element of the list designates a column in the table of estimates, and each element of a character vector refers to a coefficient. Names of list elements become column heads and names of the character vector elements become coefficient labels.
- `...`: further arguments; ignored.

Examples

```r
## Not run:
summary(classd.model <- mclogit(cbind(Freq,choice.set)-
   (econdim.sq+nonmatdim1.sq+nonmatdim2.sq)+
   (econdim1+nonmatdim1+nonmatdim2)+
   (econdim1+nonmatdim1+nonmatdim2):classd,
   data=mvote.int.classd,random=~1|mvoteint/eb,
   subset=classd!="Farmers"))
myGetSummary.classd <- function(x) getSummary.mclogit(x, rearrange=list(  
   "Econ. Left/Right"=c(    
      "Squared effect"="econdim1.sq",
      "Linear effect"="econdim1",
      " x Intermediate/Manual worker"="econdim1:classdIntermediate",
      " x Service class/Manual worker"="econdim1:classdService class",
      " x Self-employed/Manual worker"="econdim1:classdSelf-employed"
   ),
   "Lib./Auth."=c(    
      "Squared effect"="nonmatdim1.sq",
      "Linear effect"="nonmatdim1",
      " x Self-employed/Service class"="nonmatdim1:classdSelf-employed"
   ),
   "Pos./Neg."=c(    
      "Squared effect"="matdim1.sq",
      "Linear effect"="matdim1",
      " x Self-employed/Service class"="matdim1:classdSelf-employed"
   )))
```
"Linear effect"="nonmatdim1",
" x Intermediate/Manual worker"="nonmatdim1:classdIntermediate",
" x Service class/Manual worker"="nonmatdim1:classdService class",
" x Self-employed/Manual worker"="nonmatdim1:classdSelf-employed"
),
"Mod./Trad."="("
"Squared effect"="nonmatdim2.sq",
"Linear effect"="nonmatdim2",
" x Intermediate/Manual worker"="nonmatdim2:classdIntermediate",
" x Service class/Manual worker"="nonmatdim2:classdService class",
" x Self-employed/Manual worker"="nonmatdim2:classdSelf-employed"
)
)

mtable(classd.model.getSummary=myGetSummary.classd)
# Output would look like so:
# ==================================================
#  Econ. Left/Right Lib./Auth. Mod./Trad.  
#  --------------------------------------
#  Squared effect  0.030  0.008  -0.129**
#       (0.081)  (0.041)   (0.047)
#  Linear effect -0.583*** -0.038   0.137**
#       (0.063)  (0.041)   (0.045)
#  x Intermediate/Manual worker  0.632*** -0.029   -0.015
#       (0.026)  (0.020)   (0.019)
#  x Service class/Manual worker  1.158***  0.084**   0.000
#       (0.040)  (0.032)   (0.030)
#  x Self-employed/Manual worker  1.140***  0.363***   0.112***
#       (0.035)  (0.027)   (0.026)
#  Var(mvoteint)  1.000***
#       (0.000)
#  Var(mvoteint x eb)  0.118***
#       (0.000)
#  --------------------------------------
#  Dispersion  1.561
#  Deviance  15007.0
#  N 173445
#  ================

## End(Not run)

---

mblogit  

Multinomial (Baseline) Logit Models for Categorical and Multinomial Responses

Description

The function \texttt{mblogit} fits multinomial logit models for categorical and multinomial count responses with fixed alternatives, where the logits are relative to a baseline category.
Usage

mblogit(formula, data = parent.frame(), random = NULL, subset, weights = NULL, na.action = getOption("na.action"), model = TRUE, x = FALSE, y = TRUE, contrasts = NULL, control = mclogit.control(...), ...)

Arguments

- **formula**: the model formula. The response must be a factor or a matrix of counts.
- **data**: an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`, typically the environment from which `glm` is called.
- **random**: an optional formula that specifies the random-effects structure or NULL.
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
- **na.action**: a function which indicates what should happen when the data contain NAs. The default is set by the `na.action` setting of `options`, and is `na.fail` if that is unset. The ‘factory-fresh’ default is `na.omit`. Another possible value is `NULL`, no action. Value `na.exclude` can be useful.
- **model**: a logical value indicating whether `model frame` should be included as a component of the returned value.
- **x, y**: logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value.
- **contrasts**: an optional list. See the `contrasts.arg` of `model.matrix.default`.
- **control**: a list of parameters for the fitting process. See `mclogit.control`... arguments to be passed to `mclogit.control`

Details

The function `mblogit` internally rearranges the data into a 'long' format and uses `mclogit.fit` to compute estimates. Nevertheless, the 'user data' is unaffected.

Value

`mblogit` returns an object of class "mblogit", which has almost the same structure as an object of class "glm". The difference are the components `coefficients`, `residuals`, `fitted.values`, `linear.predictors`, and `y`, which are matrices with number of columns equal to the number of response categories minus one.
**Description**

mclogit fits conditional logit models and mixed conditional logit models to count data and individual choice data, where the choice set may vary across choice occasions.

Conditional logit models without random effects are fitted by Fisher-scoring/IWLS. Models with random effects (mixed conditional logit models) are estimated via maximum likelihood with a simple Laplace approximation (aka PQL).

**Usage**

```r
mclogit(formula, data=parent.frame(), random=NULL, subet, weights = NULL, offset=NULL, na.action = getOption("na.action"), model = TRUE, x = FALSE, y = TRUE, contrasts=NULL, start=NULL, control=mclogit.control(...), ...)
```

**Arguments**

- **formula**
  
  a model formula: a symbolic description of the model to be fitted. The left-hand side contains is expected to be a two-column matrix. The first column contains the choice counts or choice indicators (alternative is chosen=1, is not chosen=0). The second column contains unique numbers for each choice set.

- **data**

  an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`, typically the environment from which glm is called.

- **random**

  an optional formula that specifies the random-effects structure or NULL.

- **weights**

  an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.

- **offset**

  an optional model offset. Currently only supported for models without random effects.

- **subset**

  an optional vector specifying a subset of observations to be used in the fitting process.
mclogit

na.action a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The ‘factory-fresh’ default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.

start an optional numerical vector of starting values for the conditional logit parameters.

model a logical value indicating whether model frame should be included as a component of the returned value.

x, y logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value.

contrasts an optional list. See the contrasts.arg of model.matrix.default.

control a list of parameters for the fitting process. See mclogit.control

Value

mclogit returns an object of class "mclogit", which has almost the same structure as an object of class "glm".

Note

Covariates that are constant within choice sets are automatically dropped from the model formula specified by the formula argument of mclogit.

If the model contains random effects, these should

- either vary within choice sets (e.g. the levels of a factor that defines the choice sets should not be nested within the levels of factor)
- or be random coefficients of covariates that vary within choice sets.

In earlier versions of the package (prior to 0.6) it will lead to a failure of the model fitting algorithm if these conditions are not satisfied. Since version 0.6 of the package, the function mclogit will complain about such model a misspecification explicitly.

Examples

```r
data(Transport)

summary(mclogit(
    cbind(resp,suburb)~distance+cost,
    data=Transport
))
```

```r
data(electors)

summary(mclogit(
    cbind(Freq,interaction(time,class))~econ.left/class+welfare/class+auth/class,
    random=~1|party.time,
    data=within(electors,party.time<-interaction(party,time)))
)```
**mclogit.control**  
*Control Parameters for the Fitting Process*

**Description**

mclogit.control returns a list of default parameters that control the fitting process of mclogit.

**Usage**

mclogit.control(epsilon = 1e-08,
maxit = 25, trace=TRUE)

**Arguments**

- **epsilon**: positive convergence tolerance \( \epsilon \); the iterations converge when \( |dev - dev_{old}| / (|dev| + 0.1) < \epsilon \).
- **maxit**: integer giving the maximal number of IWLS or PQL iterations.
- **trace**: logical indicating if output should be produced for each iteration.

**Value**

A list.

---

**mclogit.fit**  
*Internal functions used for model fit.*

**Description**

These functions are exported and documented for use by other packages. They are not intended for end users.

**Usage**

mclogit.fit(y, s, w, X, start = NULL, offset = NULL, control = mclogit.control())

mmclogit.fitPQL(y, s, w, X, Z, G, groups, start,
offset = NULL, control = mclogit.control())
Arguments

- **y**: a response vector. Should be binary.
- **s**: a vector identifying individuals or covariate strata.
- **w**: a vector with observation weights.
- **X**: a model matrix; required.
- **Z**: the random effects design matrix.
- **G**: a list of design matrices for the (co-)variance parameters.
- **groups**: a list of grouping factors.
- **start**: an optional numerical vector of starting values for the coefficients.
- **offset**: an optional model offset. Currently only supported for models without random effects.
- **control**: a list of parameters for the fitting process. See `mclogit.control`.

Value

A list with components describing the fitted model.

---

### Transport

**Choice of Means of Transport**

Description

This is an artificial data set on choice of means of transport based on cost and walking distance.

Usage

```r
data(Transport)
```

Format

A data frame containing the following variables:

- **transport**: means of transportation that can be chosen.
- **suburb**: identifying number for each suburb
- **distance**: walking distance to bus or train station
- **cost**: cost of each means of transportation
- **working**: size of working population of each suburb
- **prop.true**: true choice probabilities
- **resp**: choice frequencies of means of transportation
Index

*Topic datasets
electors, 2
Transport, 9

*Topic models
mclogit, 6

*Topic regression
mclogit, 6

AIC.mclogit (mclogit), 6
anova.mclogit (mclogit), 6
as.data.frame, 5, 6

BIC.mclogit (mclogit), 6
deviance.mclogit (mclogit), 6
electors, 2
fitted.mblogit (mblogit), 4
fitted.mclogit (mclogit), 6

getSummary, 3
getSummary-methods, 3
getSummary.mblogit
  (getSummary-methods), 3
getSummary.mclogit
  (getSummary-methods), 3
glm, 5, 7

logLik.mclogit (mclogit), 6

mblogit, 3, 4
mclogit, 3, 6
mclogit.control, 5, 7, 8, 9
mclogit.fit, 5, 8
mmclogit.fitPQL (mclogit.fit), 8
mtable, 3

na.exclude, 5, 7
na.fail, 5, 7
na.omit, 5, 7

options, 5, 7
predict.mblogit (mblogit), 4
predict.mclogit (mclogit), 6
print.mblogit (mblogit), 4
print.mclogit (mclogit), 6
print.mmblogit (mblogit), 4
print.summary.mblogit (mblogit), 4
print.summary.mclogit (mclogit), 6
print.summary.mmblogit (mblogit), 4

residuals.mclogit (mclogit), 6

summary.mblogit (mblogit), 4
summary.mclogit (mclogit), 6
summary.mmblogit (mblogit), 4

Transport, 9

vcov.mclogit (mclogit), 6