Package ‘catspec’

March 5, 2012

Version 0.96

Title Special models for categorical variables

Author John Hendrickx <John_Hendrickx@yahoo.com>

Maintainer John Hendrickx <John_Hendrickx@yahoo.com>

Description ‘ctab’ creates (multiway) percentage tables. ‘sqtab’contains a set of functions for estimating models for square tables such as quasi-independence, symmetry, uniform association. Examples show how to use these models in a loglinear model using glm or in a multinomial logistic model using mlogit or clogit

Suggests mlogit,survival

URL http://home.wanadoo.nl/john.hendrickx/statres/

Repository CRAN

Date/Publication 2012-03-05 07:24:31

R topics documented:

ctab *Percentage tables*

Description

Produces one-way, two-way or multi-way percentage tables

Usage

```
ctab(...,dec.places=NULL,digits=NULL,type=NULL,style=NULL,row.vars=NULL,col.vars=NULL,percentages=N
```

Usage

```
ctab(..., dec.places=2, digits=dec.places, type=c("n", "row", "column", "total"),
style="long", row.vars=NULL, col.vars=NULL, percentages=TRUE, addmargins=FALSE)

## S3 method for class 'ctab'
print(x, dec.places=x$dec.places, addmargins=x$addmargins, ...)

## S3 method for class 'ctab'
summary(object,...)
```

Arguments

`...`	either • one or more factors, • a class `table` • a class `ftable` table object, • or a class `ctab` table object
`dec.places`	number of decimal places (default 2)
`digits`	synonym for `dec.places`, for compatability with previous version
`type`	Row, column, total percentages or counts (`type= n`). Multiple values may be specified as a character vector. Partial matchin is used.
`style`	Applicable if more than one percentage type is specified. If `style=long`, percentages are printed underneath each other. If `style=wide`, the percentages are printed side by side
`row.vars`	Same as `ftable`: *"a vector of integers giving the numbers of the variables, or a character vector giving the names of the variables to be used for the rows of the [] table"*
`col.vars`	*"a vector of integers giving the numbers of the variables, or a character vector giving the names of the variables to be used for the columns of the [] table"*
`percentages`	If `FALSE`, proportions rather than percentages are printed
`addmargins`	Use `addmargins=TRUE` to add subtotals to the table
`x`	is a tables object created by `ctab`
`object`	is a tables object created by `ctab`

Details

`ctab` uses `ftable` and `prop.table` to produce one-way frequency tables, two-way crosstables, or multi-way percentage tables. More than one percentage type may be specified, in which case "percentage type" is an unnamed dimension of the table. `row.vars` and `col.vars` can be used to control the layout of multi-way tables using the facilities of `ftable`. Subtotals can be added by specifiying `addmargins=TRUE`.

`CrossTable` in the `gmodels` package also provides an easy method for producing percentage tables, but is restricted to two-way tables.

If `ctab` is specified with no further options and for more than one factor, the output is identical to that of `ftable`. If a single factor is specified, the default is to print the frequencies column-wise with the percentaqes next to them.

Value

An object of class "ctab". `print.ctab` prints the table, `summary.ctab` passes the frequency table on to `summary.table`, which prints the number of cases, number of factors, and a chi-square test of independence.

`table`	A `class(table)` object containing the table counts. Used by `summary.ctab` and by `ctab` itself if a `ctab` object is used as input.
`ctab`	A `class(ftable)` object containing the percentage types specified. This is printed by `print.ctab`.
`row.vars`	The `row.vars` options as numeric vectors
`col.vars`	The `col.vars` options as numeric vectors
`dec.places`	The `dec.places` option
`type`	The `type` option
`style`	The `style` option
`percentages`	The `percentages` option
`addmargins`	The `addmargins` option

Author(s)

John Hendrickx <`<John_Hendrickx@yahoo.com>`>

References

http://home.wanadoo.nl/john.hendrickx/statres/

See Also

`table`, `ftable`, `addmargins`, `prop.table`, `xtabs`, `[gmodels]CrossTable`

Examples

```
ftable(Titanic)
ctab(Titanic) # same output
ctab(Titanic,type="r")
ctab(Titanic,type=c("n","r"),addmargins=TRUE)
ctab(Titanic,type=c("n","c","t","r"),style="w")
mytab<-ftable(Titanic,row.vars=c(1,3),type="r")
mytab
ctab(mytab)
newtab<-ctab(mytab,type="r")
newtab
summary(newtab)

#second example using a data frame rather than table data
library(survival)
data(logan)
attach(logan)
class(logan) #"data.frame"
ctab(occupation)
ctab(occupation,addmargins=TRUE)
ctab(occupation,style="w",type="c")
ctab(occupation,style="l",type="n")
z<-ctab(occupation,addmargins=TRUE,style="l")
z
print(z,addmargins=FALSE,dec.places=5)
summary(z)

t<-ctab(focc,occupation,type=c("n","r","c"))
t
summary(t)
```

`mclgen` *Deprecated*

Description

`mclgen` has been deprecated. Use `mlogit.data` in the `mlogit` package

`mclgen` was used to restructure a data-frame into a *person-choice* file for estimation of a multinomial logit as a conditional logit model. The `mlogit.data` does this as well but has greater flexibility. See examples in the `sqtab` help file.

sqtab *sqtab: models for square tables*

Description

These functions are used to estimate loglinear models for square tables such as quasi-independence, quasi-symmetry. Examples show how these models can be incorporated in a multinomial logistic model with covariates.

Usage

```
mob.qi(rowvar, colvar, constrained = FALSE, print.labels = FALSE)
mob.eqmain(rowvar, colvar, print.labels = FALSE)
mob.symint(rowvar, colvar, print.labels = FALSE)
mob.cp(rowvar,colvar)
mob.unif(rowvar, colvar)
mob.rc1(rowvar, colvar, equal = FALSE, print.labels = FALSE)
fitmacro(object)
check.square(rowvar, colvar, equal = TRUE)
```

Arguments

`rowvar`	Factor representing the row variable
`colvar`	Factor representing the column variable
`print.labels`	If FALSE (default) then numeric values rather than factor values are printed for compact results
`equal`	(`mob.rc1`) If TRUE, a homogeneous row and column effects model 1 with equal scale values for the row and column variables is estimated. Otherwise, a (regular) RC1 model is estimated with different scale values for the row and column variables (`check.square`) If TRUE, the row and column variables must have the same number of categories
`constrained`	(`mob.qi`)If TRUE, a quasi-independence model-constrained is estimated with a single parameter for the diagonal cells
`object`	(`fitmacro`) An object of class `glm` for `family=poisson` and `link=log`

Details

These functions are used to estimate loglinear models for square tables:

mob.qi Quasi-independence

mob.eqmain Equal main effects (Hope's halfway model)

mob.symint Symmetric interaction

mob.cp Crossings-parameter model

mob.unif Uniform association

mob.rc1 Row and columns model 1

fitmacro Calculates BIC and AIC relative to a saturated loglinear model

check.square Internal function to check if row and column variables are both factors with the same number of levels

These functions create part of the design matrix for a loglinear model. With the exception of mob.eqmain the models are an interaction effect between the row and column variable in order to test for a certain pattern of association. As with most things in R, these functions represent one way of doing things.

These restricted models can also be applied in a multinomial logistic regression model. In order to do this, the data have a “long” shape. Rather than one record per subject, the dataset must have *ncat* records per subject, where *ncat* is the number of categories of the dependent variable in the multinomial logistic model. The dataset must also have variables indexing the choices and indicating the choice made by each subject.

The documentation for clogit has an example that shows how the data can be restructured in this way. The mlogit.data function can also restructure the data as required. The models can be subsequently estimated using clogit or mlogit.data.

Value

mob.qi	A factor that will produce coefficients for the diagonal cells of a table, using off-diagonal cells as base category
mob.eqmain	A design matrix with equality contraints on the main effects
mob.symint	A design matrix for an interaction with equality contraints on coefficients on opposite sides of the diagonal
mob.cp	A set of vectors for a crossings-parameter model
mob.unif	A vector for a uniform association model
mob.rc1	A set of vectors for a row and columns model 1
fitmacro	Prints deviance, df, BIC, AIC, number of parameters and N
check.square	Stops function if either the row or column variable is not a factor or if the number of levels is unequal

Author(s)

John Hendrickx <<John_Hendrickx@yahoo.com>>

References

Agresti, Alan. (1990). *Categorical data analysis*. New York: John Wiley & Sons.

Allison, Paul D. and Nicholas Christakis. (1994). Logit models for sets of ranked items. Pp. 199-228 in *Peter V. Marsden (ed.), Sociological Methodology*. Oxford: Basil Blackwell.

Breen, Richard. (1994). Individual Level Models for Mobility Tables and Other Cross- Classifications. *Sociological Methods & Research* 33: 147-173.

Goodman, Leo A. (1984). *The analysis of cross-classified data having ordered categories*. Cambridge, Mass.: Harvard University Press.

Hendrickx, John. (2000). Special restrictions in multinomial logistic regression. *Stata Technical Bulletin* 56: 18-26.

Hendrickx, John, Ganzeboom, Harry B.G. (1998). Occupational Status Attainment in the Netherlands, 1920-1990. A Multinomial Logistic Analysis. *European Sociological Review* 14: 387-403.

Hout, Michael. (1983). *Mobility Tables.* Sage Publication 07-031.

Logan, John A. (1983). A Multivariate Model for Mobility Tables. *American Journal of Sociology 89*: 324-349.

See Also

`glm`, `[mlogit]mlogit`, `[mlogit]mlogit.data`, `[survival]clogit`, `[survival]coxph`, `[nnet]multinom`

Examples

```
# Examples of loglinear models for square tables,
# from Hout, M. (1983). "Mobility Tables". Sage Publication 07-031

# Table from page 11 of "Mobility Tables"
# Original source: Featherman D.L., R.M. Hauser. (1978) "Opportunity and Change."
# New York: Academic, page 49

Freq <- c(
1414,  521,  302,   643,   40,
 724,  524,  254,   703,   48,
 798,  648,  856,  1676,  108,
 756,  914,  771,  3325,  237,
 409,  357,  441,  1611, 1832)

OccFather<-gl(5,5,labels=c("Upper nonmanual","Lower nonmanual","Upper manual","Lower manual","Farm"))
OccSon<-gl(5,1,labels=c("Upper nonmanual","Lower nonmanual","Upper manual","Lower manual","Farm"))
FHtab <- data.frame(OccFather,OccSon,Freq)

xtabs(Freq~OccFather+OccSon,data=FHtab)

# independence model
indep<-glm(Freq~OccFather+OccSon,family=poisson(),data=FHtab)
summary(indep)
fitmacro(indep)

wt <- as.numeric(OccFather != OccSon)
qi0<-glm(Freq~OccFather+OccSon,weights=wt,family=poisson(),data=FHtab)
# A quasi-independence loglinear model, using structural zeros
# (page 23 of "Mobility Tables").
#  0  1  1  1  1   values of variable "wt"
#  1  0  1  1  1
#  1  1  0  1  1
#  1  1  1  0  1
#  1  1  1  1  0
qi0<-glm(Freq~OccFather+OccSon,weights=wt,family=poisson(),data=FHtab)
summary(qi0)
fitmacro(qi0)
```

```
# Quasi-independence using a "dummy factor" to create the design
# vectors for the diagonal cells (page 23).
#  1  0  0  0  0
#  0  2  0  0  0
#  0  0  3  0  0
#  0  0  0  4  0
#  0  0  0  0  5
glm.qi<-glm(Freq~OccFather+OccSon+mob.qi(OccFather,OccSon),family=poisson(),data=FHtab)
summary(glm.qi)
fitmacro(glm.qi)

# Quasi-independence without using the functions
# Factor labels prevent numeric comparisons, create numeric versions
# of the row and column variables
OccFather_n <- unclass(OccFather)
OccSon_n <- unclass(OccSon)
q1 <- ifelse(OccFather_n==OccSon_n & OccSon_n==1,1,0)
q2 <- ifelse(OccFather_n==OccSon_n & OccSon_n==2,1,0)
q3 <- ifelse(OccFather_n==OccSon_n & OccSon_n==3,1,0)
q4 <- ifelse(OccFather_n==OccSon_n & OccSon_n==4,1,0)
q5 <- ifelse(OccFather_n==OccSon_n & OccSon_n==5,1,0)
glm.qi2<-glm(Freq~OccFather+OccSon+q1+q2+q3+q4+q5,family=poisson(),data=FHtab)
summary(glm.qi2)
fitmacro(glm.qi2)

# Quasi-independence constrained (QPM-C, page 31)
# Single immobility parameter
#  1  0  0  0  0
#  0  1  0  0  0
#  0  0  1  0  0
#  0  0  0  1  0
#  0  0  0  0  1
glm.q0<-glm(Freq~OccFather+OccSon+mob.qi(OccFather,OccSon,constrained=TRUE),family=poisson(),data=FHtab)
# slightly different results than Hout also found in Stata: L2=2567.658, q0=0.964
summary(glm.q0)
fitmacro(glm.q0)

# Quasi-symmetry using the symmetric cross-classification (page 23)
#  0  1  2  3  4   values of variable "sym"
#  1  0  5  6  7
#  2  5  0  8  9
#  3  6  8  0 10
#  4  7  9 10  0  */
glm.qsym<-
glm(Freq~OccFather+OccSon+mob.symint(OccFather,OccSon),family=poisson(),data=FHtab)
summary(glm.qsym)
fitmacro(glm.qsym)

symmetry<-glm(Freq~mob.eqmain(OccFather,OccSon)
+mob.symint(OccFather,OccSon),family=poisson(),data=FHtab)
summary(symmetry)
fitmacro(symmetry)
```

```
# Crossings parameter model (page 35)
#  0  v1 v1 v1 v1 |  0  0  v2 v2 v2 |  0  0  0  v3 v3 |  0  0  0  0 v4
#  v1 0  0  0  0  |  0  0  v2 v2 v2 |  0  0  0  v3 v3 |  0  0  0  0 v4
#  v1 0  0  0  0  |  v2 v2 0  0  0  |  0  0  0  v3 v3 |  0  0  0  0 v4
#  v1 0  0  0  0  |  v2 v2 0  0  0  |  v3 v3 v3 0  0  |  0  0  0  0 v4
#  v1 0  0  0  0  |  v2 v2 0  0  0  |  v3 v3 v3 0  0  |  v4 v4 v4 v4 0
glm.cp<-glm(Freq~OccFather+OccSon+mob.cp(OccFather,OccSon),family=poisson(),data=FHtab)
summary(glm.cp)
fitmacro(glm.cp)

# Uniform association model: linear by linear association (page 58)
glm.unif<-glm(Freq~OccFather+OccSon+mob.unif(OccFather,OccSon),family=poisson(),data=FHtab)
summary(glm.unif)
fitmacro(glm.unif)

# RC model 1 (unequal row and column effects, page 58)
# Fits a uniform association parameter and row and column effect
# parameters. Row and column effect parameters have the
# restriction that the first and last categories are zero.
glm.rc1<-glm(Freq~OccFather+OccSon+mob.rc1(OccFather,OccSon),family=poisson(),data=FHtab)
summary(glm.rc1)
fitmacro(glm.rc1)

# Homogeneous row and column effects model 1 (page 58)
# An equality restriction is placed on the row and column effects
glm.hrc1<-glm(Freq~OccFather+OccSon+mob.rc1(OccFather,OccSon,equal=TRUE),family=poisson(),data=FHtab)
# Results differ from those in Hout, replicated by other programs
summary(glm.hrc1)
fitmacro(glm.hrc1)

#-------------------------------------------------------------------------------
# Examples on using these models in multinomial logistic regression
#-------------------------------------------------------------------------------
# Data from the 1972-78 GSS used by Logan (1983)
library(survival)
data(logan)

# Restructure the data in 'long' format using mlogit.data
library(mlogit)
pc <- mlogit.data(logan, shape = "wide", choice = "occupation")
head(pc,10)
# pc$alt indexes choices, pc$chid indexes respondents
# pc$occupation is a boolean variable that is TRUE where pc$alt corresponds
# with the actual choice made
class(logan$occupation) #"factor"
class(pc$alt) #"character"
# pc$alt needs to be transformed into a factor for use in the some models
# for square tables which assume ordered categories
pc$alt <- factor(pc$alt,levels=c("farm", "operatives", "craftsmen", "sales", "professional"))

# A regular multinomial logit model
m1<-mlogit(occupation ~ 0 | education+race, data=pc, reflevel = "farm")
```

```
summary(m1)

# Estimate a "quasi-uniform association" loglinear model for "focc" and "occupation"
# with "education" and "race" as covariates at the respondent level
# The quasi-uniform association model contains alternative-specific effects,
# education and race are individual specific effects
m2 <- mlogit(occupation ~ mob.qi(focc,alt)+mob.unif(focc,alt) | education+race,data=pc, reflevel = "farm")
summary(m2)

# Same model using survival::clogit
# First create a design matrix for the alternatives that drops the first category
occ.X<-model.matrix(~pc$alt)[,-1]
head(occ.X,10)
# The clogit output exceeds the default 80 characters
options(width=256)
# pc$chid which indexes respondents must be used in strata()
# Individual specific effecs are modelled as interactions between the
# design matrix for pc$alt and the covariates
cl.qu<-clogit(occupation~occ.X+occ.X:education+occ.X:race+
  mob.qi(focc,alt)+mob.unif(focc,alt)+strata(chid),data=pc)
summary(cl.qu)
```

Index