Package ‘gllm’

November 27, 2018

Version 0.37
Date 2018-11-26
Title Generalised log-Linear Model
Author David Duffy <David.Duffy@qimr.edu.au>
   C code in emgllmfitter by Andreas Borg <borg@imbei.uni-mainz.de>
Maintainer David Duffy <David.Duffy@qimberghofer.edu.au>
Depends R (>= 0.99)
Description Routines for log-linear models of incomplete contingency tables,
   including some latent class models, via EM and Fisher scoring
   approaches. Allows bootstrapping. See Espeland and Hui (1987)
License GPL-2 | GPL-3
URL https://genepli.qimr.edu.au/Staff/davidD/loglin
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-11-27 10:00:03 UTC

R topics documented:

anova.gllm ............................................... 2
boot.gllm .................................................. 3
boot.table .................................................. 5
emgllm .................................................... 6
emgllmfitter ............................................... 8
gllm ....................................................... 9
hildesheim ............................................... 12
ld2 ......................................................... 14
ld2.model ............................................... 15
lsat ....................................................... 16
scatter .................................................... 16
scoregllm ............................................... 17
summary.gllm ............................................ 19
anova.gllm

Summarize multiple results from gllm

Description

Compare likelihood ratio test statistics from multiple calls to gllm.

Usage

## S3 method for class 'gllm'
anova(object, ..., test=c("Chisq","none"))

Arguments

object is an object output from gllm.
... other objects from gllm.
test evaluate LRTS for model, or nothing.

Value

A list with components:

<table>
<thead>
<tr>
<th>Model</th>
<th>name of each object being compared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resid.df</td>
<td>residual degrees of freedom for each model</td>
</tr>
<tr>
<td>Deviance</td>
<td>likelihood ratio test statistic for model versus saturated model</td>
</tr>
<tr>
<td>Pr.Fit</td>
<td>chi-square based P-value for model</td>
</tr>
<tr>
<td>Test</td>
<td>models compared in stepwise testing</td>
</tr>
<tr>
<td>Df</td>
<td>degrees of freedom of model comparison</td>
</tr>
<tr>
<td>LRtest</td>
<td>likelihood ratio test statistic comparing models</td>
</tr>
<tr>
<td>Prob</td>
<td>chi-square based P-value for LRTS</td>
</tr>
</tbody>
</table>

Author(s)

David L Duffy
**Description**

Fits log-linear models for incomplete contingency tables, including some latent class models, via EM and Fisher scoring approaches. Performs a bootstrap for the sampling distribution of the full unobserved table.

**Usage**

```r
boot.gllm(y, s, X, method = "hybrid", em.maxit = 1, tol = 0.00001, strata = NULL, R = 200)
```

**Arguments**

- `y` is the observed contingency table.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`.
- `X` is the design matrix, or a formula.
- `method` chooses the EM, Fisher scoring or a hybrid (EM then scoring) method for fitting the model.
- `em.maxit` is the number of EM iterations.
- `tol` is the convergence criterion for the LR criterion.
- `strata` is a vector identifying the sampling strata.
- `R` is the number of bootstrap replicates.

**Details**

The generalized log-linear model allows for modelling of incomplete contingency tables, that is tables where one or more dimensions have been collapsed over. See `gllm` for details.

Often, functions of the full unobserved table are the main focus of the analysis. For example, in a double sampling design where there is a gold standard measure for one part of the data set and only an unreliable measure for another part, the expected value of the gold standard in the entire dataset is the outcome of interest. The standard error of this statistic may be a complex function of the observed counts and model parameters.

Bootstrapping is one way to estimate such standard errors from a complex sampling design. The bootstrap sampling may be stratified if the design implies this, e.g. product-multinomial.

**Value**

A matrix $R + 1$ by `ncol(X)` containing the initial estimate of the full (unobserved) contingency table, and the $R$ bootstrap replicates of the full table.
References

Examples

```r
# # Fit Hochberg 1977 double sampling data
# # 2x2 table of imprecise measures and 2x2x2x2 reliability data
# # 2x2 table of imprecise measures
# y1 <- c(1196, 13562, 7151, 58175)
a2 <- 2-as.integer(gl(2, 1, 4))
b2 <- 2-as.integer(gl(2, 2, 4))
set1 <- data.frame(y1, a2, b2)
#
# # 2x2x2x2 reliability data
# y2 <- c(17, 3, 10, 258, 3, 4, 4, 25, 16, 3, 25, 197, 100, 13, 107, 1014)
a <- 2-as.integer(gl(2, 1, 16))
a2 <- 2-as.integer(gl(2, 2, 16))
b <- 2-as.integer(gl(2, 4, 16))
b2 <- 2-as.integer(gl(2, 8, 16))
set2 <- data.frame(y2, a, a2, b, b2)
#
# # Combined analysis
# y <- c(y1, y2)
#
# # Map observed table onto underlying 2x2x2x2 table
# s <- c(1, 1, 2, 2, 1, 1, 2, 2, 3, 3, 4, 4, 3, 3, 4, 4,
# 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20)
#
# # Model combining the tables is A*A2*B*B2 + L (dummy study variable)
# a <- 2-as.integer(gl(2, 1, 32))
a2 <- 2-as.integer(gl(2, 2, 32))
b <- 2-as.integer(gl(2, 4, 32))
b2 <- 2-as.integer(gl(2, 8, 32))
l <- 2-as.integer(gl(2, 16, 32))
X <- model.matrix(~ a*a2*b*b2+l)
```
```r
# Table 1 using unreliable measure
#
res1<-glm(y1 ~ a2*b2, family=poisson(),data=set1)
print(summary(res1))

# Table 2 using reliable measure
#
res2a<-glm(y2 ~ a*b, family=poisson(),data=set2)
print(summary(res2a))

# Table 2 demonstrating complex relationship between gold standard and unreliable measure
#
res2b<-glm(y2 ~ a*a*b*b, family=poisson(),data=set2)
print(summary(res2b))

# Combined analysis
#
require(gllm)
res12<-gllm(y,s,X)
print(summary.gllm(res12))

# Bootstrap the collapsed table to get estimated OR for reliable measures
#
# a and b are binary vectors the length of the *full* table
# and define the variables for which the odds ratio is to be estimated, here the reliable measure of injury and seatbelt usage.
#
boot.hochberg <- function (y,s,X,nrep,a,b) {
  z<boot.gllm(y,s,X,R=nrep)
  boot.tab<cbind(apply(z[,a & b],1,sum),
                 apply(z[,!a & b],1,sum),
                 apply(z[,a & !b],1,sum),
                 apply(z[,!a & !b],1,sum))
  oddsr<boot.tab[,1]*boot.tab[,4]/boot.tab[,2]/boot.tab[,3]
  hochberg.tab<-data.frame( c("yes","yes","no","no"),
                           c("yes","no","yes","no"),
                           boot.tab[,1],
                           apply(boot.tab[,1:1+nrep],2,mean))
  colnames(hochberg.tab)<-c("Precise Injury","Precise seatbelt usage",
                           "Estimated Count","Bootstrap S.E.")
  print(hochberg.tab)
  cat("Estimated OR=",oddsr[1],"\n")
  cat("Bias=",mean(oddsr[2:1+nrep]),"\n")
  cat("Bootstrap SE=",sd(oddsr[2:1+nrep]),"\nQuantiles\n"
    quantile(oddsr[2:1+nrep]),c(0.025,0.50,0.975))
}
boot.hochberg(y,s,X,nrep=20,a,b)
```
Description
Given a vector of counts from a contingency table, produce a bootstrap replicate. Sampling zeroes are replaced by 0.5.

Usage
boot.table(y,strata=NULL)

Arguments
y 
is the observed contingency table.
strata 
is a vector defining the strata for a stratified bootstrap.

Value
A vector of counts with the same total.

Examples
boot.table(c(1,3,4,2))
## 0.5 2.0 5.0 3.0
boot.table(c(1,3,4,2),c(1,2,1,2))
## 2 1 3 4

emgllm 

Generalized log-linear modelling by EM and iterative proportional fitting

Description
Fits log-linear models for incomplete contingency tables, including some latent class models, via an EM approach.

Usage
eemgllm(y,s,X,maxit=1000,tol=0.00001)

Arguments
y 
is the observed contingency table.
s 
is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of y
X 
is the design matrix, or a formula.
maxit 
is the number of EM iterations.
tol 
is the convergence criterion for the LR criterion.
Details

The generalized log-linear model allows for modelling of incomplete contingency tables, that is tables where one or more dimensions have been collapsed over. These include situations where imprecise measures have been calibrated using a "perfect" gold standard, and the true association between imperfectly measured variables is to be estimated; where data is missing for a subsample of the population; latent variable models where latent variables are "errorless" functions of observed variables - eg ML gene frequency estimation from counts of observed phenotypes; specialised measurement models eg where observed counts are mixtures due to perfect measures and error prone measures; standard latent class analysis; symmetry and quasi-symmetry models for square tables.

The general framework underlying these models is summarised by Espeland (1986), and Espeland & Hui (1987), and is originally due to Thompson & Baker (1981). An observed contingency table \( y \), which will be treated as a vector, is modelled as arising from an underlying complete table \( z \), where observed count \( y_j \) is the sum of a number of elements of \( z \), such that each \( z_i \) contributes to no more than one \( y_j \). Therefore one can write \( y = F'z \), where \( F \) is made up of orthogonal columns of ones and zeros.

We then specify a loglinear model for \( z \), so that \( \log(\mathbb{E}(z)) = X'b \), where \( X \) is a design matrix, and \( b \) a vector of loglinear parameters. The loglinear model for \( z \) and thus \( y \), can be fitted via an iterative proportional fitting algorithm for \( b \) and \( z \), with an EM fitting for \( y \), \( z \) and \( b \) (Haber 1984).

The \texttt{emgllm} function is a wrapper for C code implementing the approach in Haber (1984).

Value

A list with components:

- deviance
- observed.values
- fitted.values
- full.table

References


Examples

```r
# latent class analysis: two latent classes
# Data matrix 2x2x2x2x2 table of responses to five binary items
```
emgllmfitter

Fits log-linear models for incomplete contingency tables, via an EM approach.

Usage

emgllmfitter(y, s, X, maxit, tol)

Arguments

- `y` is the observed contingency table.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`.
- `X` is the design matrix.
- `maxit` is the number of EM iterations.
- `tol` is the convergence criterion for the LR criterion.
Details

The call to Andreas Borg’s C code that fits the model by EM/IPF. The algorithm follows the approach in Haber (1984).

Value

A list with components:

- `y` the observed table
- `ji` s, the scatter vector
- `c` the design matrix
- `istop` maximum EM iterations
- `conv` the convergence tolerance
- `e` expected counts for the full (unobserved) table
- `ni` nrow(X)
- `nj` length(y)
- `nk` ncol(X)-1
- `f` expected counts

References


---

gllm

*Generalized log-linear modelling*

Description

Fits log-linear models for incomplete contingency tables, including some latent class models, via EM and Fisher scoring approaches.

Usage

```r
gllm(y, s, X, method="hybrid", em.maxit=1, tol=0.00001)
```

Arguments

- `y` is the observed contingency table.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`
- `X` is the design matrix, or a formula.
- `method` chooses the EM, Fisher scoring or a hybrid (EM then scoring) method for fitting the model.
- `em.maxit` is the number of EM iterations.
- `tol` is the convergence criterion for the LR criterion.
Details

The generalized log-linear model allows for modelling of incomplete contingency tables, that is tables where one or more dimensions have been collapsed over. These include situations where imprecise measures have been calibrated using a "perfect" gold standard, and the true association between imperfectly measured variables is to be estimated; where data is missing for a subsample of the population; latent variable models where latent variables are "errorless" functions of observed variables - eg ML gene frequency estimation from counts of observed phenotypes; specialised measurement models eg where observed counts are mixtures due to perfect measures and error prone measures; standard latent class analysis; symmetry and quasi-symmetry models for square tables.

The general framework underlying these models is summarised by Espeland (1986), and Espeland & Hui (1987), and is originally due to Thompson & Baker (1981). An observed contingency table $y$, which will be treated as a vector, is modelled as arising from an underlying complete table $z$, where observed count $y_j$ is the sum of a number of elements of $z$, such that each $z_i$ contributes to no more than one $y_j$. Therefore one can write $y = F'z$, where $F$ is made up of orthogonal columns of ones and zeros.

We then specify a loglinear model for $z$, so that $\log(E(z)) = X'\beta$, where $X$ is a design matrix, and $\beta$ a vector of loglinear parameters. The loglinear model for $z$ and thus $y$, can be fitted using two methods, both of which are available in gllm. The first was presented as AS207 by Michael Haber (1984) and combines an iterative proportional fitting algorithm for $\beta$ and $z$, with an EM fitting for $y$, $z$ and $\beta$. The second is a Fisher scoring approach, presented in Espeland (1986).

The gllm function is actually a simple wrapper for scoregllm().

Value

A list with components:

- `iter` the number of scoring iterations until convergence
- `deviance` the final model deviance (-2 log likelihood)
- `df` the model degrees of freedom
- `coefficients` the model parameter estimates
- `se` the standard errors for the model parameter estimates
- `v` the variance-covariance matrix for the model parameter estimates
- `observed.values` the observed counts in $y$
- `fitted.values` the expected counts under the fitted model
- `residuals` Pearsonian residuals under the fitted model
- `full.table` the expected counts for the full (unobserved) table.

References


**Examples**

```r
# latent class analysis: two latent classes
# Data matrix 2x2x2x2 table of responses to five binary items
# (items 11-15 of sections 6-7 of the Law School Admission Test)
# y <- c( 3, 6, 2, 11, 1, 1, 3, 4,
#        1, 8, 0, 16, 0, 3, 2, 15,
#        10, 29, 14, 81, 3, 28, 15, 80,
#        16, 56, 21, 173, 11, 61, 28, 298)
# Scatter matrix: full table is 2x2x2x2x2
# s <- c(1:32,1:32)
# Design matrix: x is the latent variable (2 levels),
# a-e are the observed variables
# x <- as.integer(gl(2,32,64)) - 1
# a <- as.integer(gl(2,16,64)) - 1
# b <- as.integer(gl(2,8,64)) - 1
# c <- as.integer(gl(2,8,64)) - 1
# d <- as.integer(gl(2,8,64)) - 1
# e <- as.integer(gl(2,1,64)) - 1
res1 <- gllm(y, s, ~ x*(a+b+c+d+e), method="em", tol=0.01)
res1
# An example of model fitting: gametic association between two diallelic loci
# Data matrix
# y <- c(187,386,156,
#       352,310,20,
#       136,0,10)
# Scatter matrix
# s <- c(1, 2, 2, 3,
#        4, 5, 5, 6,
#        4, 5, 5, 6,
#        7, 8, 8, 9)
# Design matrix
# X <- matrix(c(1,0,0,0,0,0,1,
#              1,0,1,0,0,0,0,
```

---

**Notes:**

- The example code demonstrates how to use the `gllm` function from the `gllm` package in R for latent class analysis and gametic association studies.
- The data matrix for latent class analysis is a 2x2x2x2 table of responses to five binary items.
- The scatter matrix and design matrix are also shown, with variables and levels specified for each.

---

**References:**

Invasive Cervical Cancer v exposure to Herpes Simplex Virus

Description

The case-control study of Hildesheim et al (1991) has been reanalysed by several authors (Carroll et al 1993; Spiegelhalter et al 1999; Prescott et al 2002). Exposure to Herpes Simplex Virus in cases suffering from invasive cervical cancer and in unaffected controls was assessed by Western Blot in all cases and controls and by a gold-standard refined Western blotting in a subset of 115 subjects.

Usage

data(hildesheim)

Format

A data frame table.

Source

hildesheim

References


Examples

data(hildesheim)
ftable(xtabs(Freq ~ case+HSV.inac+HSV.gold, hildesheim))
fisher.test(xtabs(Freq ~ case+HSV.inac, hildesheim))
fisher.test(xtabs(Freq ~ case+HSV.gold, hildesheim, subset=HSV.gold!="?"))

# Combined analysis (ordered as incomplete then complete data)
#
y<-hildesheim$Freq[c(3,9,6,12,1,2,7,8,4,5,10,11)]
#
# Map observed table onto underlying 2x2x2x2 table
#
s <-c(1, 1, 2, 3, 3, 4, 4,
      5, 6, 7, 8, 9, 10, 11, 12)
#
substudy  <- 2-as.integer(gl(2,8,16))
HSV.inac  <- 2-as.integer(gl(2,4,16))
HSV.gold  <- 2-as.integer(gl(2,2,16))
cancer   <- 2-as.integer(gl(2,1,16))

require(gllm)
res<-gllm(y,s, ~substudy+HSV.inac+HSV.gold*cancer)
print(summary.gllm(res))
#
# Bootstrap the collapsed table to get estimated OR for reliable measures
#
# a and b are binary vectors the length of the *full* table
# and define the variables for which the odds ratio is to be
# estimated, here the reliable measure of HSV exposure and Ca Cx
#
boot.hildesheim <- function (y,s,X,nrep,a,b) {
  z<-boot.gllm(y,s,X,R=nrep)
  boot.tab<-cbind(apply(z[,a & b],1,sum),
                  apply(z[,!a & b],1,sum),
                  apply(z,[a & !b],1,sum),
                  apply(z,[!a & !b],1,sum))
  odds<-boot.tab[,1]*boot.tab[,4]/boot.tab[,2]/boot.tab[,3]
  hildesheim.tab<-data.frame( c("yes","yes","no","no"),
                              c("yes","no","yes","no"),
                              boot.tab[1,],
                              apply(boot.tab[2:(1+nrep),],2,sd))
colnames(hildesheim.tab)<-c("Precise HSV","Cervical Cancer", "Estimated Count","Bootstrap S.E.")

print(hildesheim.tab)
cat("Estimated OR=",oddsr[1],"\n")
cat(" Bias=",oddsr[1]-mean(oddsr[2:(1+nrep)]),"\n")
cat("Bootstrap SE=" ,sd(oddsr[2:(1+nrep)]),"\n\n"quantiles\n")
print(quantile(oddsr[2:(1+nrep)],c(0.025,0.50,0.975)))

b=-mean(log(oddsr[2:(1+nrep)]))
se=-sd(log(oddsr[2:(1+nrep)]))
ztest=b/se
cat(" Estimated log(OR)=",log(oddsr[1]),"\n",
 " Bootstrap mean log(OR)=",b,"\n",
 " Bootstrap SE=" ,se,"\n",
 " Wald Z=" ,ztest," (P=" ,2*pnorm(ztest,lower=FALSE)," )\n")

boot.hildesheim(y,s,-substudy+hsv.inac*hsv.gold*cancer,nrep=50,cancer,hsv.gold)

1d2

Estimate linkage disequilibrium between two codominant autosomal loci

Description

Fits a log-linear model for allelic association between two codominant autosomal loci. Measures of LD are odds ratios.

Usage

ld2(locus1, locus2)

Arguments

locus1 is a character vector containing the genotypes at the first locus, or a RxC contingency table of genotype counts.

locus2 is a character vector containing the genotypes at the second locus.

Value

m0 base model
m1 estimating LD coefficient(s) assuming HWE
m2 testing HWE at locus 1
m3 testing HWE at locus 2
m4 estimating LD and HWD coefficient(s)
**ld2.model**

Write design and filter matrices for log-linear model of linkage disequilibrium between two codominant autosomal loci.

**Description**

Write design and filter matrices for log-linear model of linkage disequilibrium between two codominant autosomal loci.

**Usage**

```
ld2.model(nall1, nall2, formula="~a1+a2+p1+p2+d")
```

**Arguments**

- `nall1` is number of alleles at first codominant locus.
- `nall2` is number of alleles at first codominant locus.
- `formula` is character string listing terms to be included in model, where `a1` denotes allele frequencies for locus 1, `p1` the deviation from Hardy-Weinberg expectations for locus 1, and `d` the intragametic allelic association parameters.

**Value**

A list with components:

- `geno` is a dummy contingency table showing the expected order.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`
- `X` is the design matrix.
lsat

*Five dichotomous items from the Law School Admission Test (LSAT)*

**Description**

Small dataset (items 11-15 from sections 6 and 7 of the Law School Admission Test) used by Bock and Lieberman (1970), Christoffersson (1975) and Joreskog and Sorbom (1986) to test methods for factor analysis of binary data.

**Source**


---

scatter

*Create a filter matrix from a summary array of indices*

**Description**

Create a filter matrix that multiplying the vector of counts from a complete contingency table, gives a collapsed contingency table.

**Usage**

`scatter(y,s)`

**Arguments**

- `y` is the observed contingency table. Provides a target length only.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`

**Value**

`S` A matrix of orthogonal columns of 1s and 0s

**Author(s)**

David L Duffy

**Examples**

```r
y <- double(3)
z <- 1:5
z %*% scatter(y, c(1, 2, 3, 3))
## 1+2, 3, 4+5
```
**scoregllm**  
*Generalized log-linear modelling via Fisher scoring*

**Description**

Fits log-linear models for incomplete contingency tables, including some latent class models, via Fisher scoring approaches.

**Usage**

```r
scoregllm(y,s,X,m,tol=1e-5)
```

**Arguments**

- `y` is the observed contingency table.
- `s` is a vector of indices, one for each cell of the full (unobserved) contingency table, representing the appropriate cell of `y`.
- `X` is the design matrix or a formula.
- `m` is a vector of starting values for the full (unobserved) contingency table.
- `tol` is the convergence criterion for the LR criterion.

**Details**

The generalized log-linear model allows for modelling of incomplete contingency tables, that is tables where one or more dimensions have been collapsed over. These include situations where imprecise measures have been calibrated using a "perfect" gold standard, and the true association between imperfectly measured variables is to be estimated; where data is missing for a subsample of the population; latent variable models where latent variables are "errorless" functions of observed variables - eg ML gene frequency estimation from counts of observed phenotypes; specialised measurement models eg where observed counts are mixtures due to perfect measures and error prone measures; standard latent class analysis; symmetry and quasi-symmetry models for square tables.

The general framework underlying these models is summarised by Espeland (1986), and Espeland & Hui (1987), and is originally due to Thompson & Baker (1981). An observed contingency table `y`, which will be treated as a vector, is modelled as arising from an underlying complete table `z`, where observed count `y_j` is the sum of a number of elements of `z`, such that each `z_i` contributes to no more than one `y_j`. Therefore one can write `y = F'z`, where `F` is made up of orthogonal columns of ones and zeros.

We then specify a loglinear model for `z`, so that `log(E(z)) = X'b`, where `X` is a design matrix, and `b` a vector of loglinear parameters. The loglinear model for `z` and thus `y`, can be fitted by a Fisher scoring approach, presented in Espeland (1986).

The `gllm` function is actually a simple wrapper for `scoregllm()`.
Value

A list with components:

- **iter**  
  the number of scoring iterations until convergence
- **deviance**  
  the final model deviance (-2 log likelihood)
- **df**  
  the model degrees of freedom
- **coefficients**  
  the model parameter estimates
- **se**  
  the standard errors for the model parameter estimates
- **V**  
  the variance-covariance matrix for the model parameter estimates
- **observed.values**  
  the observed counts in \( y \)
- **fitted.values**  
  the expected counts under the fitted model
- **residuals**  
  Pearsonian residuals under the fitted model
- **full.table**  
  the expected counts for the full (unobserved) table.

References


Examples

```r
# An example of model fitting: gametic association between two diallelic loci
# Data matrix
#
# y<-c( 187,386,156,
#      352,310,20,
#      136,0 ,0)
#
# Scatter matrix
#
# s<- c( 1, 2, 2, 3,
#       4, 5, 5, 6,
#       4, 5, 5, 6,
#       7, 8, 8, 9)
#
# Design matrix
#
# X<- matrix(c( 1,0,0,0,0,0,1,
#              1,0,1,0,0,0,0,
#              1,0,1,0,0,0,0,
#              1,0,0,1,0,0,0,
#              1,0,0,0,1,0,0,
#              1,0,0,0,0,1,0,
#              1,0,0,0,0,0,1),
#            nrow=7,ncol=7)
```
summary.gllm  

1,0,2,0,1,0,0,  
1,1,0,0,0,0,0,  
1,1,1,0,1,0,0,  
1,1,1,0,0,0,1,  
1,1,2,0,1,1,1,  
1,1,0,0,0,0,0,  
1,1,1,0,0,0,1,  
1,1,1,0,1,0,0,  
1,1,2,0,1,1,1,  
1,2,0,1,0,0,0,  
1,2,1,1,0,1,1,  
1,2,1,1,0,1,1,  
1,2,2,1,1,2,2), byrow=TRUE, ncol=7)

colnames(X)<-c("Intercept", "A", "B", "P1", "P2", "Delta", "Epsilon")
res<-scoregllm(y,s,X[,c(1:6)],
c(255,176,121,164,37,113,25,164,113,37,25,90,20,20,5))
summary(res)
#

summary.gllm  Summarize results of gllm

Description

Summarizes contents of result of call to gllm. The print method pretty prints the summary object.

Usage

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

Arguments

object is the object output from gllm.
... other arguments

Value

A list with components:

nobs the number of cells in the observed table
nfull the number of cells in the full table
mean.cell the mean cell count in the observed table
deviance the final model deviance (-2 log likelihood)
model.df the model degrees of freedom
coefficients the model parameter estimates, standard errors
residuals Observed and fitted counts, plus Pearsonian residuals
Author(s)

David L. Duffy
Index

*Topic **category**
  boot.gllm, 3
  boot.table, 6
  emgllm, 6
  emgllmfitter, 8
  gllm, 9
  ld2, 14
  ld2.model, 15
  scoregllm, 17
*Topic **datasets**
  hildesheim, 12
  lsat, 16
*Topic **manip**
  boot.table, 6
  scatter, 16
*Topic **models**
  boot.gllm, 3
  emgllm, 6
  emgllmfitter, 8
  gllm, 9
  ld2, 14
  ld2.model, 15
  scoregllm, 17
*Topic **print**
  anova.gllm, 2
  summary.gllm, 19

anova.gllm, 2
boot.gllm, 3
boot.table, 5
emgllm, 6
emgllmfitter, 8
gllm, 9
hildesheim, 12
ld2, 14
ld2.model, 15

lsat, 16
print.ld2(ld2), 14
print.summary.gllm(summary.gllm), 19
scatter, 16
scoregllm, 17
summary.gllm, 19

21