Package: CPMCGLM (via r-universe)

September 3, 2024

Version 1.2

Date 2017-11-28

Title Correction of the P-Value after Multiple Coding in Generalized Linear Models

Author Jeremie Riou, and Benoit Liquet

Maintainer Jeremie Riou <jeremie.riou@univ-angers.fr>

LazyLoad yes

Description We propose to determine the correction of the significance level after multiple coding of an explanatory variable in Generalized Linear Model. The different methods of correction of the p-value are the Single step Bonferroni procedure, and resampling based methods developed by P.H.Westfall in 1993. Resampling methods are based on the permutation and the parametric bootstrap procedure. If some continuous, and dichotomous transformations are performed this package offers an exact correction of the p-value developed by B.Liquet & D.Commenges in 2005. The naive method with no correction is also available.

License GPL (> 2)

Depends R (>= 2.10.0), mvtnorm, plyr, abind

URL https://cran.r-project.org/

NeedsCompilation no

Date/Publication 2017-12-06 13:47:46 UTC

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

RemoteUrl https://github.com/cran/CPMCGLM

RemoteRef HEAD

RemoteSha 60510f263fa125ea9ea06aae9cece4f0bbb007ef

Contents

`CPMCGLM` *Correction of the significance level after multiple coding of an explanatory variable in generalized linear model.*

Description

We propose to determine the correction of the significance level after multiple coding of an explanatory variable in Generalized Linear Model. The different methods of correction of the p-value are the Single step Bonferroni procedure, and resampling based methods developped by P.H.Westfall in 1993. Resampling methods are based on the permutation and the parametric bootstrap procedure. If some continuous, and dichotomous transformations are performed this package offers an exact correction of the p-value developped by B.Liquet & D.Commenges in 2005. The naive method with no correction is also available.

Usage

```
CPMCGLM(formula, family, link, data, varcod, dicho, nb.dicho, categ,
nb.categ, boxcox, nboxcox, FP , N=1000, cutpoint)
```

Arguments

`formula`	an object of class "formula" : a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
`family`	a description of the error distribution to be used in the model. This should be a character string naming a family function. The possible family functions are: "binomial", "gaussian", and "poisson".
`link`	a description of the link function to be used in the model. This needs to be a character string naming a link function. For the "gaussian" family, you must use the "identity" link. For the "binomial" family function, "logit" and "probit" link are available. And for "Poisson" family function, you must use the "log" link function.
`data`	a data frame containing the variables of the model.
`varcod`	a continuous variable that you want to transform.
`dicho`	a vector with the order of the quantile which are used for computing the cutpoint of each dichotomous transformation. The length of the vector corresponds to the number of transformation. If you specify this argument, "nb.dicho" must not be present.

`nb.dicho` if you do not enter the "dicho" argument, you can enter the number of dichotomous transformations that you want. The strategy of coding is presented in "Details" section.

`categ` a matrix with the order of quantile which are used for computing the categorical cutpoints of each transformation. The details of the "categ" specification are given under "Details". If you specify this argument, "nb.categ" must not be present.

`nb.categ` if you do not enter the "categ" argument, you can enter the number of categorical transformations that you want. The strategy of coding is presented in "Details" section.

`boxcox` a vector of λ parameters corresponding to each BoxCox transformation.

`nboxcox` if you do not enter the "boxcox" argument, you can enter the number of boxcox transformations that you want. The maximum number of transformations that you can enter is 5. For the stategy of coding, it seems natural to try the crude variable (λ1=1) and the log transformation. When two transformations are condidered we propose λ1 =1 and λ2 =0. Further interesting transformations include the square λ3 =2 and the square root λ4 =0.5. Finally, the power λ5 =3/2 may be tried.

`FP` a matrix with powers which are used for computing each fractional polynomial transformations. The details of the "FP" specification are given under "Details" section.

`N` the number of resampling that you want to do.

`cutpoint` a matrix with the different numeric values for the cutpoints. The details of the cutpoint specification are given under 'Details'.

Details

- formula: A typical predictor has the form "response ~ terms" where "response" is the numeric response (possibly binary "0","1") vector and "terms" is a serie of terms which specifies a linear predictor for response.

- nb.dicho: Dichotomous transformations include only the categorical transformations in two classes. The most natural method is to use a transformation based on the quantile. For one transformation, the median is used as a cutpoint for the dichotomous coding. For two transformations, the first tercile is used for the first dichotomous transformation, and the second tercile for the second one, and so on.

- categ: The categ argument needs to be a matrix. You need to have one line per transformation. Therefore, the dimension of the matrix is nbq $\times$ maxq, where nbq is the number of transformations tried with the categ transformations, and maxq is the maximum of number of quantiles that is used in one quantile transformation.

For example:

[1,]	0.33	0.66	NA	NA
[2,]	0.25	0.5	0.75	NA
[3,]	0.2	0.4	0.6	0.8

In this example, three transformations are performed, so nbq=3. And maxq=4, because the maximum of number of quantiles that we used for the quintiles is 4. The first transformation leads to a categorical transformation in three classes, with cutpoints at the first and the second tercile. The second transformation allows to obtain a categorical variable in four classes with cutpoints at the quartile. And the third one allows to obtain a variable in five classes with the cutpoints at the quintiles.

- nb.categ: This concerns categorical transformations in more than two classes. Considering one of these transformations, the most intuitive method is to use a transformation in three classes at the tercile. For two of such transformations, we added the previous coding and a categorical transformation in four classes based on the quartile, and so on.

- cutpoint: The cutpoint argument needs to be a matrix. The form of this matrix is similar as one of the quantile matrix. The number of rows corresponds to the number of tranformations (nbc) tried with this method, and the number of columns corresponds to the maximum of cutpoints (maxc) that is used in one transformation.

For example:

[1,]	8	16	NA	NA
[2,]	6	12	18	NA
[3,]	5	10	15	20

In this example,one wants to perform three transformations, hence the three rows. The first transformation leads to a categorical variable in three classes, with two cutpoints for the value "8", and the value "16". The second transformation allows to obtain a categorical transformation in four classes, with cutpoints for values: "6","12" and "18". The last transformation tried allows to obtain a categorical transformation in five classes with cutpoint for values: "5","10","15", and "20". Therefore, we used four columns because four is the maximum of cutpoints used, in the third transformation.

- FP: The FP argument needs to be a matrix. The number of rows correspond to the number of transformations tested, and the number of columns is the maximum number of degrees tested for a single transformation.

For example:

[1,]	-2	NA	NA	NA
[2,]	0.5	1	-0.5	2
[3,]	-0.5	1	NA	NA

In this example, the user performs three transformations of the variable of interest. The first is a fractional polynomial transformation with one degree and a power of -2. The second transformation is a fractional polynomial transformation with four degrees and powers of 0.5,1,-0.5,and 2. The third transformation is a fractional polynomial transformation with two degrees and powers of -0.5, and 1.

Value

`call` the code used for the model.

n	the number of subjects in the dataset.
N	the number of resampling.
family	the family function used.
link	the link function used.
nbt	the number of score tests realised.
nbb	the number of score tests realised with BoxCox transformation.
nbq	the number of score tests realised with Quantile transformation.
nbc	the number of score tests realised with Cutpoint transformation.
vq	the vector quantiles' values for the best coding.
adj	the number of adjustment variables.
trans	the method of transformation for each coding.
BC	the method of the best transformation: "Dichotomous", "Categorical", "Boxcox", "Continuous","Cutpoint".
bestcod	the corresponding value of the transformation parameter for the best transformation.
naive.pvalue	the Pvalue of the best association without correction.
exact.pvalue	the adjusted Pvalue of the best association with an exact correction.
bonferroni.adjusted.pvalue	the adjusted Pvalue of the best association with the Bonferroni correction.
parametric.bootstrap.adjusted.pvalue	the adjusted Pvalue of the best association with the parametric bootstrap correction.
permutation.adjusted.pvalue	the adjusted Pvalue of the best association with the permutation correction.

Author(s)

J.Riou, A.Diakite, and B.Liquet

References

Liquet, B. and Commenges, D. (2005). Computation of the p-value of the minimum of score tests in the generalized linear model, application to multiple coding. *Statistics & Probability Letters*, 71:33-38.

Liquet, B. and Commenges, D. (2001). Correction of the p-value after multiple coding of anexplanatory variable in logistic regression. *Statistics in Medicine*, 20:2815-2826.

Westfall, P. H. and Young, S. (1992). Resampling-based multiple testing: examples and methods for pvalue adjustment. Wiley Series in Probability and Mathematical Statistics. *Applied Probability and Statistics*. New York, NY: Wiley. xvii, 340 p.

Yu, K., Liang, F., Ciampa, J., and Chatterjee, N. (2011). Efficient p-value evaluation for resampling-based tests. *Biostatistics*, 12(3):582-593.

See Also

print.CPMCGLM, summary.CPMCGLM

Examples

```
## Not run:

# load data
data(data_sim)
#
#Example of quantile matrix definition

#Linear Gaussian Model

fit1 <- CPMCGLM(formula= Weight~Age+as.factor(Sport)+Desease+Height,
family="gaussian",link="identity",data=data_sim,varcod="Age",N=1000,
boxcox=c(0,1,2,3),nb.dicho=3,nb.categ=4)
### print fit1
fit1
### summary fit1
summary(fit1)

#Loglinear Poisson Model
fit2 <- CPMCGLM(formula= Stroke~Age+as.factor(Sport)+Height+Weight,
family="poisson",link="log",data=data_sim,varcod="Age",N=1000,
boxcox=c(0,1,2,3))

### print fit2
fit2
### summary fit2
summary(fit2)

#Logit Model
FP1 <-  matrix(NA,ncol=4,nrow=3)
FP1[1,1] <- -2
FP1[2,] <- c(0.5,1,-0.5,2)
FP1[3,1:2] <- c(-0.5,1)

fit3 <- CPMCGLM(formula= Parameter~Age+as.factor(Sport)+Height+Weight,
family="binomial",link="logit",data=data_sim,varcod="Age",N=1000,
boxcox=c(0,1,2,3),nb.dicho=3,FP=FP1)
### print fit3
fit3
### summary fit3
summary(fit3)

#Probit Model

fit4 <- CPMCGLM(formula= Parameter~Age+as.factor(Sport)+Height+Weight,
family="binomial",link="probit",data=data_sim,varcod="Age",N=1000,
nboxcox=2,nb.categ=4)
### print fit4
```

```
fit4
### summary fit4
summary(fit4)

## End(Not run)
```

`data_sim` *dataset for CPMCGLM package.*

Description

This simulated dataset contains 100 subjects and 8 variables.

Usage

```
data(data_sim)
```

Format

A data frame with 100 observations on the following 8 variables.

`Height` a numeric vector

`Weight` a numeric vector

`Age` a numeric vector

`Smoke` a binary vector

`Sport` a numeric vector

`Desease` a numeric vector

`Parameter` a binary vector

`Stroke` a categorical vector

Examples

```
data(data_sim)
head(data_sim)
```

`ForInternalUse` *For internal use only ...*

Description

For internal use only ...

print.CPMCGLM *Output of a* CPMCGLM *object*

Description

The function provides the output of a CPMCGLM correction of pvalue.

Usage

```
## S3 method for class 'CPMCGLM'
print(x,...)
```

Arguments

x	an object inheriting from classes CPMCGLM.
...	other parameters.

Author(s)

J.Riou, A.Diakite, and B.Liquet

See Also

CPMCGLM,summary.CPMCGLM

Examples

```
# load data
## Not run:
data(data_sim)

#Linear Gaussian Model
fit1 <- CPMCGLM(formula= Weight~Age+as.factor(Sport)+Desease+Height,
family="gaussian",link="identity",data=data_sim,varcod="Age",N=1000,
boxcox=c(0,1,2,3))
### print fit1
fit1

## End(Not run)
```

summary.CPMCGLM *Short summary of a* CPMCGLM *object*

Description

The function provides the summary of a CPMCGLM correction of pvalue.

Usage

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

Arguments

object	an object inheriting from classes CPMCGLM.
...	other parameters.

Author(s)

J.Riou, A.Diakite, and B.Liquet

See Also

CPMCGLM,print.CPMCGLM

Examples

```
## Not run:
# load data
data(data_sim)

#Linear Gaussian Model
fit1 <- CPMCGLM(formula= Weight~Age+as.factor(Sport)+Desease+Height,
family="gaussian",link="identity",data=data_sim,varcod="Age",N=1000,
boxcox=c(0,1,2,3))
### summary fit1
summary(fit1)

## End(Not run)
```

Index