Package 'LogisticDx'

July 2, 2014

Type Package

Title Diagnostic tests for logistic regression models

Version 0.1

Date 2014-05-03

Author Chris Dardis

Maintainer Chris Dardis <christopherdardis@gmail.com>

License GPL (>= 2)

Description Diagnostic tests and plots for logistic regression models.

Depends R (>= 2.13.0), multcomp, data.table, pROC, car

Imports gRbase, rms, stats, statmod, graphics, speedglm

LazyLoad yes

Collate 'genLogi.R' 'logiDx.R' 'stukel.R' 'logiGOF.R' 'logiProb.R' 'logiSS.R' 'LogisticDx-internal.R' 'multiPlot.R' 'plotLogiDx.R' 'summaryLogiGOF.R'

Repository CRAN

Repository/R-Forge/Project logisticdx

Repository/R-Forge/Revision 9

Repository/R-Forge/DateTimeStamp 2014-05-14 01:02:09

Date/Publication 2014-05-14 07:28:28

NeedsCompilation no

R topics documented:

`LogisticDx-package` *Diagnostic tests for logistic regression models*

Description

Generate sample data frames for logistic regression. Diagnostic tests and plots for logistic regression models

Details

Package:	LogisticDx
Type:	Package
Version:	0.5
Date:	2014-05-01
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

Christopher Dardis <christopherdardis@gmail.com>

References

Hosmer, D. and Lemeshow, S 2000 *Applied Logistic Regression. Second edition*. John Wiley and Sons, Inc.

genLogi *Generate data for logistic regression*

Description

Generates a `data.frame` or `data.table` with a binary outcome, and a logistic model to describe it.

Usage

```
genLogiDf(b = 2L, f = 2L, c = 1L, n = 20L, nlf = 3L,
  pb = 0.5, rc = 0.8, py = 0.5, asFactor = TRUE,
  model = TRUE, timelim = 5, speedglm = FALSE)

genLogiDt(b = 2L, f = 2L, c = 1L, n = 20L, nlf = 3L,
  pb = 0.5, rc = 0.8, py = 0.5, asFactor = TRUE,
  model = TRUE, timelim = 5, speedglm = FALSE)
```

Arguments

`b`	*binomial predictors*, the number of predictors which are binary, i.e. limited to 0 or 1
`f`	*factors*, the number of predictors which are factors
`c`	*continuous predictors*, the number of predictors which are continuous
`n`	number of observations in the data frame
`nlf`	the no. of levels in a factor
`pb`	*probability for binomnial predictors*: the probability of binomial predictors being $= 1$ e.g. if `pb=0.3`, 30% will be 1s, 70% will be 0s
`rc`	*ratio for continuous variables* the ratio of levels of continuous variables to the total number of observations *n* e.g. if `rc=0.8` and `n=100`, it will be in the range 1-80
`py`	*ratio for y* the ratio of 1s to total observations for the binomial predictors e.g. if `ry=0.5`, 50% will be 1s, 50% will be 0s
`asFactor`	If `asFactor=TRUE` (the default), predictors given as `factors` will be converted to `factors` in the data frame before the model is fit
`model`	If `model=TRUE` will also return a model fitted with `stats::glm` or `speedglm::speedglm`
`timelim`	function will timeout after `timelim` secs. This is present to prevent duplication of rows.
`speedglm`	If `speedglm=TRUE`, return a model fitted with `speedglm` instead of `glm`

Value

If `model=TRUE`: a list with the following values:

`df or dt`	A `data.frame` (for `genLogiDf`) or `data.table` (for `genLogiDt`). Predictors are labelled $x1, x2, ..., xn$. Outcome is y. Rows represent to n observations
`model`	A model fit with `stats::glm` or `speedglm::speedglm`

If `model=FALSE` a `data.frame` or `data.table` as above.

Note

`genLogiDt` is faster and more efficient for larger datasets.

Using `asFactor=TRUE` with factors which have a large number of levels (e.g. `nlf >30`) on large datasets (e.g. $n > 1000$) can cause fitting to be excessively slow.

Examples

```
set.seed(1)
genLogiDf()
genLogiDt(b=0, c=2, n=100, rc=0.7, model=FALSE)
```

---

logiDx *Diagnostics for logistic regression*

Description

Returns standard diagnostic measures for a logistic regression model by covariate pattern

Usage

```
logiDx(x, round = FALSE, roundTo = 3)
```

Arguments

`x`	A model of class `glm`
`round`	If `round=TRUE`, digits will be ronded to `roundTo` decimal places
`roundTo`	No. decimal places to which to round digits

Value

A `data.table`. There is one row per covariate pattern with at least one observation. These are sorted by `dBhat` (see below).

The initial columns give all combinations of the predictor variables with at least one observation.

Subsequent columns are labelled as follows:

`obs`	Number of observations with this covariate pattern
`prob`	Probability of this covariate pattern
`yhat`	Number of observations of $y = 1$, *predicted* by the model
`y`	*Actual* number of observations of $y = 1$ from the data
`lev`	*Leverage*, the diagonal of the hat matrix used to generate the model; a measure of influence of this covariate pattern
`devR`	*Deviance residual*, calculated by covariate pattern; a measure of influence of this covariate pattern
`PeR`	*Pearson residual*, calculated by covariate pattern; a measure of influence of this covariate pattern. Given by: $$\sqrt{obs}\sqrt{\frac{prob}{(1-prob)}}$$
`sPeR`	*Standardized Pearson residual* calculated by covariate pattern; a measure of influence of this covariate pattern. Given by: $$\frac{PeR}{\sqrt{(1-lev)}}$$
`dBhat`	*Change in Bhat*, the standardized difference between the original maximum likelihood estimates **B** and that estimates with this covariate pattern excluded
`dXsq`	*Change in chi-square*, decrease in the value of Pearson chi-square statistic with this covariate pattern excluded. Given by: $$sPeR^2$$
`dDev`	*Change in deviance* **D** with this covariate pattern excluded. Given by: $$\frac{dev^2}{(1-lev)}$$

Note

Values for the statistics are calculated by *covariate pattern*. Different values may be obtained if calculated for each individual obervation (i.e. row in data frame).

Generally, the values calculated by covariate pattern are preferred, particularly where no. observations are > 5.

See Also

plotLogiDx

Examples

```
d1 <- genLogiDt(model=FALSE)
f1 <- stats::glm(y ~ I(x5^2)*x1 -1, family=binomial("logit"), data=d1)
logiDx(f1)
```

---

logiGOF *Goodness of fit tests for a logistic regression model*

Description

Gives 15 commonly employed measures of goodness of fit for a logistic regression model

Usage

```
logiGOF(x, g = 10)
```

Arguments

x	A model of class glm
g	No. groups (quantiles) into which to split observations for Hosmer-Lemeshow and modified Hosmer-Lemeshow tests.

Value

A list of class logiGOF with the following items:

chiPearCov	Pearsons chi-square, calculated by *covariate group*, with p value and interpretation
chiPearIndiv	Pearsons chi-square, calculated by *individual observation*, with p value and interpretation
chiPearTab	Pearsons chi-square, calculated by *table of covariate patterns by outcome*, with p value and interpretation
OsRo	Osius & Rojek test of the logistic link, with p value and interpretation
chiDevCov	Deviance chi-square, calculated by *covariate group*, with p value and interpretation
chiDevIndiv	Deviance chi-square, calculated by *individual observation*, with p value and interpretation
chiDevTab	Deviance chi-square, calculated by *table of covariate patterns by outcome*, with p value and interpretation
covPatTab	Matrix of covariance patterns, used to calculate above chi-square tests of Pearson residuals and deviance

HosLem	Hosmer & Lemeshow goodness of fit test, with g quantile groups,with p value and interpretation
modHosLem	modified Hosmer & Lemeshow goodness of fit test, with g quantile groups, with p value and interpretation
CesHou	le Cessie, van Houwelingen, Copas & Hosmer unweighted sum of squares test for global goodness of fit, with p value and interpretation
Stuk	Stukels test of the appropriateness of the logistic link, with p value and interpretation
PR2	Pearsons R^2, correlation of observed outcome with predicted
ssR2	Linear regression-like sum-of-squares R^2, using covariate patterns
llR2	Log-likelohood based R^2, calculated by covariate group
ROC	Area under the Receiver Operating Curve, with 95% CI by method of DeLong

Note

A summary method is available

Warning: Will fail if cannot generate a hat matrix for the model using logiDx

Author(s)

Modified Hosmer & Lemeshow goodness of fit test: adapted from existing work by Yongmei Ni

See Also

logiDx

Examples

```
set.seed(1)
m1 <- genLogiDf(n=100)$model
logiGOF(m1)
```

---

logiProb *Logit, odds ratio and probability for coefficients of a logistic regression*

Description

Generate logit, log odds ratio, odds ratio and probability for coefficients in a logistic regression. Can be generalized to all combinations of coefficients.

Values are calculated for a change in the value of the coeffient for the predictor from 0 to 1. (For continuous predictors changes of more than one unit may have more practical significance).

Usage

```
logiProb(x, usePrim = FALSE, all = FALSE)
```

Arguments

x	A logistic regression model of class glm
all	If all=FALSE (the default) return values for all coefficients in model, considered together. If all=TRUE return values for all *combinations* of coefficients in model.
usePrim	If usePrim=FALSE (the default) use utils::combn to generate combinations. If usePrim=TRUE use gRbase::combnPrim instead (faster).

Value

If all=TRUE, a data.table giving, for each *combination* of coefficients:

coef	Combination of coefficients
logit	The logit for a given combination of coefficients
lnOR	Natural log of Odds Ratio
OR	Odds Ratio
p	probability

This is sorted by OR (low to high).

If all=FALSE, a data.frame giving the above values for all predictors.

Note

To use gRbase::combnPrim the following dependencies may be necessary. Install as follows:
source("http://bioconductor.org/biocLite.R")
biocLite("graph")
biocLite("BiocGenerics")
biocLite("RBGL")

Examples

```
set.seed(1)
f1 <- genLogiDf(n=50)$model
logiProb(f1)
d1 <- genLogiDt(n=50, model=FALSE)
logiProb(glm(y ~ x1 + x3 -1, data=d1, family=binomial()), all=TRUE)
```

`logiSS` *Sample size for given coefficient and events per covariate for model*

Description

Gives sample size necessary to demonstrate that coefficient in model for given predictor is equal to its given value (rather than equal to zero) for a given level of power and significance.

Also number of events (smaller of outcome $y = 0$ and outcome $y = 1$) per predictor.

Uses different methods depending on whether model has one binomial, one continuous or multiple predictors.

Usage

```
logiSS(x, alpha = 0.05, beta = 0.8, coeff = "x1")
```

Arguments

`x`	A logistic regression model of class `glm`
`alpha`	significance level α for null-hypothesis significance test
`beta`	power β for null-hypothesis significance test
`coeff`	Name of predictor (coefficient) in model to be tested

Value

A list of:

`res`	Result: Sample size required to show coefficient for predictor is as given in the model rather than 0
`epc`	Events per covariate; should be >10 to make meaningful statements about coefficients obtained

Examples

```
set.seed(1)
### one coefficient, which is binomial
f1 <- genLogiDf(b=1, c=0, n=50)$model
logiSS(f1)
###
### one coefficient, which is continuous
f1 <- genLogiDf(f=0, b=0, c=1, n=50)$model
logiSS(f1, coeff="x1")
###
### binomial coefficient
f1 <- genLogiDf(f=0, b=1, c=1, n=50)$model
logiSS(f1, coeff="x1")
```

```
###
### continuous coefficient
f1 <- genLogiDf(f=0, b=1, c=1, n=50)$model
logiSS(f1, coeff="x2")
```

---

plotLogiDx *Diagnostic plots for a logistic regression*

Description

Common diagnostic plots for a logistic regression model

Usage

```
plotLogiDx(x, noPerPage = 6,
  cols = c("deepskyblue", "dodgerblue"), cex = 2,
  pch = 21, inches = 0.25, identify = FALSE,
  extras = FALSE, width = NULL, height = NULL)
```

Arguments

x	A logistic regression model of class `glm`
noPerPage	Number of plots per page (for initial plots). Will be used as *guidance* and optimised for ease of display
cols	Colours. As used by `graphics::points`
cex	Cex **C**haracter **ex**pansion. See `?graphics::plot.default`
pch	**P**lotting **ch**aracter. See `?graphics::points`
inches	Width of circles for bubble plot. See `?graphics::symbols`
identify	If `TRUE` will give option to identify individual points on a number of the plots produced. The number which appears next to the point corresponds to the relevant row as given by `logiGOF`
extras	If `TRUE` produces additional plots, detailed below
width	Width of screen(display device) in pixels
height	Height of screen(display device) in pixels

Value

The following are plotted, for each covariate group:

`p_X_lev`	Probability of $y = 1$ for this group by leverage (diagonal of hat matrix, a measure of influence)
`p_X_dXsq`	Probability as above by *dXsq* change in Pearson chi-square statistic with deletion of this group
`p_X_dBhat`	Probability by *dBhat* change in Bhat; the difference in the maximum likelihood estimators **Beta** for model coefficients with all subjects included vs those with this group, standardized by the estimated covariance matrix of **Beta**
`p_X_dDev`	Probability by *dDev*, the change in deviance when this group is excluded
`bubbleplot`	Probability by dXsq, with area of circle proportional to dBhat
`lev_X_dXsq`	Leverage by *dXsq*, the change in the Pearson chi-square statistic when this group is excluded
`lev_X_dBhat`	Leverage by *dBhat*, the difference in the maximum likelihood estimators **Beta** for model coefficients with all subjects included vs those when this group is excluded. This is standardized by the estimated covariance matrix of **Beta**
`lev_X_dDev`	Leverage by *dDev*, the change in deviance when this group is excluded
`ROC`	Receiver Operator Curve

Additional plots are given when `extras=TRUE`:

`influenceplot`	See `?car::influencePlot`
`sr_X_hat`	*Studentized residual* by hat values. Studentized residual = residual / estimate of standard deviation of residual
`slp`	Spread-level plot. See from `?car::spreadLevelPlot`
`qqPlot`	quantile-quantile plot vs Normal for residuals. See `?stats::qqplot`
`iip`	Influence-index plot. Gives Cooks distance, studentized residual and hat values for each observation
`pairs`	Pairs plot for the measures of influence dBhat, dXsq and dDev. See `?graphics::pairs`
`crPlots`	Component + residual plots. See `?car::crPlots`
`avPlots`	Added-variable plots. See `?car::avPlots`
`mmps`	Marginal model plots. These require that the `data.frame` used to fit the model be present in the current environment. See `?car::mmps`

Note

Different colors can be found with e.g.
`grDevices::colours()[grep("blue",grDevices::colours())]`

Examples

```
set.seed(1)
### generate up to 8x covariate patterns
mod1 <- genLogiDf(b=3, f=0, c=0, n=50)$model
plotLogiDx(mod1, cex=8, noPerPage=1)
plotLogiDx(mod1, cex=3, noPerPage=6, extras=TRUE)
df1 <- genLogiDf(b=0,f=0,c=2,n=50, model=FALSE)
g1 <- glm(y ~ ., family=binomial("logit"), data=df1)
plotLogiDx(g1)
```

---

stukel *Stukels test of the logistic link*

Description

Calculates Stukels test for a logistic regression model.

This determines the appropriateness of the logistic link.

Two new covariates, z1 and z2 are generated such that

$$z1 = 0.5 * logit^2 * I(pi >= 0.5), z2 = -0.5 * logit^2 * I(pi <= 0.5)$$

where $I(arg) = 1$ where arg is true and $= 0$ where false.

Tests if significant change occurs in the model with the addition of these coefficients.

Usage

```
stukel(object,
  alternative = c("both", "alpha1", "alpha2"))
```

Arguments

`object`	An object of class **glm**
`alternative`	add both $z1$ and $z2$ to model or just one of the above

Value

A list with the following elemens:

`statistic`	value of statistic
`parameter`	degrees of freedom
`p.value`	if < 0.05 suggests should accept null hypothesis that logistic model is incorrect for the data
`alternative`	alternative
`method`	method
`data.name`	name of object
`allstat`	statistics for all tests
`allpar`	degrees of freedom
`allpval`	all p values

Author(s)

Brett Presnell

References

University of Florida

Examples

```
set.seed(1)
m1 <- genLogiDf(b=3,f=0,c=0,n=50)$model
stukel(m1)
```

Index