Package: glmxdiag (via r-universe)

September 14, 2024

Version 1.0.0

Date 2022-01-04

Title A Collection of Graphic Tools for GLM Diagnostics and some Extensions

Depends R (>= 3.6.0)

Description Provides diagnostic graphic tools for GLMs, beta-binomial regression model (estimated by 'VGAM' package), beta regression model (estimated by 'betareg' package) and negative binomial regression model (estimated by 'MASS' package). Since most of functions implemented in 'glmxdiag' already exist in other packages, the aim is to provide the user unique functions that work on almost all regression models previously specified. Details about some of the implemented functions can be found in Brown (1992) <doi:10.2307/2347617>, Dunn and Smyth (1996) <doi:10.2307/1390802>, O'Hara Hines and Carter (1993) <doi:10.2307/2347405>, Wang (1985) <doi:10.2307/1269708>.

License GPL (>= 2)

NeedsCompilation no

Author Giuseppe Reale [aut, cre]

Maintainer Giuseppe Reale <giuseppe.reale97@gmail.com>

Imports VGAM

Date/Publication 2022-01-10 08:32:45 UTC

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

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

RemoteRef HEAD

RemoteSha 7f27397c6bbb0f58b2c54a0429e0108c8bfba3ca

Contents

`avplot` *Added Variable Plot*

Description

Constructs added variable plots for generalized linear models and extensions admitted by `glmxdiag`.

Usage

```
avplot(model, variables, type = c("Wang", "Hines-Carter"), label.id,
              n.label.id, xlab, ylab, main, pos, pch, cex, lcol, lwd, lty, ...)
```

Arguments

`model`	a model supported by `glmxdiag`.
`variables`	a vector of characters containing names of regressors; must be included inside the model matrix. If nothing is specified, avplot is applied on all variables.
`type`	default is `"Wang"` developed by Wang (1985); if `"Hines-Carter"` the method developed by Hines and Carter (1993) is used.
`label.id`	labels of observations, should be a vector with n elements
`n.label.id`	number of observations to label in the plot that most influence the "clean" relationship of considered variables. Cook's distance is used as the measure of influence.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`main`	an overall title for the plot.
`pos`	position of observations labels. Values `1, 2, 3, 4` respectively indicate below, left, above, right.
`pch`	type of points.

cex	size of points.
lcol	color of line.
lwd	width of line.
lty	type of line.
...	further arguments passed to plot.

Details

The aim of added variable plot is to isolate the relationship between a specific variable and the response, i.e. fixing all the other variables included in the linear predictor. In the y-axis there are the *working residuals* of the reduced model (i.e. a model where the variable we're interested in, say "x", is excluded) while on the x-axis there are the residuals of regression of x using all the other variables. Values in both axis are weighted: Wang (1985) uses the *working weights* of the reduced model while Hines and Carter (1993) suggest that using the weights of the full model is more reliable in indicating the presence of influential observations.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

References

Wang, P C. (1985) Adding a variable in generalized linear models. Technometrics 27, 273-276.

R. J. O'Hara Hines, & Carter, E. M. (1993). Improved Added Variable and Partial Residual Plots for the Detection of Influential Observations in Generalized Linear Models. Journal of the Royal Statistical Society. Series C (Applied Statistics), 42(1), 3-20.

Examples

```
data(moons)
m <- glm(Moons ~ Mass + Distance + Diameter,
         family = poisson, data = moons)
summary(m)
avplot(m, 'Distance', label.id = moons$Name)
```

`cookDist` *Visualize Cook's distances*

Description

Graphical visualization of Cook's distances for each observation within input model.

Usage

```
cookDist(object, label.id, n.label.id, xlab, ylab, pos, ...)
```

Arguments

`object`	object of class 'influence', see Details.
`label.id`	labels of observations, should be a vector with n elements.
`n.label.id`	number of observations with highest Cook's distance to label in the plot.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`pos`	position of observations labels. Values `1, 2, 3, 4` respectively indicate below, left, above, right.
`...`	further arguments passed to `plot`.

Details

This function only works with objects given as output from `influenceDiag`, it takes Cook's distances and plot them as vertical segments.

High values are associated to high influence within the model.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

Examples

```
data("mtcars")
mod <- glm(mpg ~ cyl + hp + carb, family = Gamma, data = mtcars)
inf <- influenceDiag(mod)
cookDist(inf, label.id = rownames(mtcars))
```

devAnalysis *Graphical analysis of Deviance*

Description

A graphical tool developed by Brown (1992) to assess the contribution of one or more of the explanatory variables inside a Generalized Linear Model. This plot displays all single variable comparisons and shows how they relate to other terms that are already in the model in terms of deviance. The output plot is divided into three frames, see Details.

Usage

```
devAnalysis(model, cex.pl = .8, cex.vars = 1, cex.vars2 = 1,
            layout.heights = c(1.2,0.8,4), pl.scale = 1,
            xlab, ylab, title)
```

Arguments

`model`	a model supported by `glmxdiag`.
`cex.pl`	dimension of points of linear predictors.
`cex.vars`	dimension of variables names inside lower frame.
`cex.vars2`	dimension of variables names inside medium frame.
`layout.heights`	a vector with three heights for the frames, from upper to lower.
`pl.scale`	proportion given to the sub-frame of the lower frame indicating the linear predictors.
`xlab`	a title for the x axis of lower frame.
`ylab`	a title for the y axis of lower frame.
`title`	an overall title for the plot.

Details

In order to give a clear explanation of the plot, these will be given referred to the output plot of example code. The plot is divide into three frames: upper, medium and lower; we will start by explaining the lower one.

On the y axis we have the residual deviance and each horizontal line corresponds to the (residual) deviance of a particular fitted model. On the right side of each horizontal line there are plus signs: based on the column they're in, they indicate if that specific variable is included in the linear predictor. For example, looking at the first horizontal line (right under the names diam, mass, dist), there are no plus signs on the adjacent columns: that is the null model. The line under the null model has a plus sign in correspondece of dist, i.e. only variable distance is included within that model.

The differences in deviance corresponding to the addition of a term to the various possible models (i.e. models not containing the term in question) are represented by vertical lines connecting the two appropriate horizontal model lines. For example the first vertical line connects the null model to the one that has just diam in the linear predictor. Note that this frame is divided into blocks and each one contains the differences in deviance (i.e. vertical lines) corresponding to addition of a specific variable; in this case there are three blocks for three variables.

The upper frame is just a summary of the lower one; the vertical lines are the same but in a reduced scale. The medium frame indicates the linear predictors of a model before adding the block-specific variable. For example the are no plus signs under the first vertical line; this means that it corresponds to the change in deviance when diam is added to the null model (because we are in the diam block as in the lower frame). The second line corresponds to the difference in deviance in a model containing only mass due to adding diam, and so on.

This graph is really useful to investigate the relationship between variable inside a model. In this case we may notice that adding mass is relevant when there are no variables or only dist; on the other hand its contribution is almost imperceptible when diam is already included within linear predictor.

If the description of arguments `layout.heights` and `pl.scale` is not clear, the best way to understand their role is to run the code a few times changing these parameters: this will do it.

The best advice is to use this function with a maximum of 4-5 variables or else the plot will result confusing.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

References

Brown, D. (1992). A Graphical Analysis of Deviance. Journal of the Royal Statistical Society. Series C (Applied Statistics), 41(1), 55-62.

M. Sciandra & A. Plaia (2018) A graphical model selection tool for mixed models, Communications in Statistics - Simulation and Computation, 47:9, 2624-2638

Examples

```
data(moons)
dist <- moons$Distance
diam <- moons$Diameter
mass <- moons$Mass
mod <- glm(moons$Moons ~ diam + mass + dist, family = poisson)
devAnalysis(mod, cex.vars = 1.3, cex.vars2 = 1.1)
```

DFbeta *Visualize DFbetas*

Description

Graphical visualization of DFbetas for each observation and chosen variables within input model.

Usage

```
DFbeta(object, variables, centered = TRUE, label.id, n.label.id,
              xlab, ylab, main, pos, pch, cex, lcol, lwd, lty, ...)
```

Arguments

`object`	object of class 'influence', see Details.
`variables`	a vector of characters containing names of variables; if nothing is specified, DFbetas for all variables are plotted.
`centered`	logical, if TRUE the plot is centered on the coefficients of the full model and each point corresponds to the coefficient if the i-th observation is excluded. if FALSE, usual DFbetas centered on zero are plotted.
`label.id`	labels of observations, should be a vector with n elements.
`n.label.id`	number of observations with highest DFbetas to label in the plot.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`main`	an overall title for the plot.
`pos`	position of observations labels. Values `1`, `2`, `3`, `4` respectively indicate below, left, above, right.
`pch`	type of points.
`cex`	size of points.
`lcol`	color of line.
`lwd`	width of line.
`lty`	type of line.
`...`	further arguments passed to `plot`

Details

This function only works with objects given as output from `influenceDiag`.

DFbetas are calculated as *B - B(-i)* hence they are not standardized; if `centered` is set to TRUE then each plotted point corresponds to B(-i), otherwise to B - B(-i).

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

Examples

```
data("mtcars")
mod <- glm(mpg ~ cyl + hp + carb, family = Gamma, data = mtcars)
inf <- influenceDiag(mod)
DFbeta(inf, label.id = rownames(mtcars))
DFbeta(inf, variables = 'cyl', label.id = rownames(mtcars), pos = 2)

# to visualize all of them in a single plot
par(mfrow = c(2, 2))
DFbeta(inf, label.id = rownames(mtcars))
# press <Enter> four times
par(mfrow = c(1, 1))
```

`diabetes` *Diabetes*

Description

This data comes from a study (Sockett et al, 1987) of the factors affecting patterns of insulin-dependent mellitus in children. The objective is to investigate the dependence of the level of serum C-peptide on the various other factors in order to understand the patterns of residual insulin secretion. The response measurement is the logarithm of C-peptide concentration (pmol/ml) at the diagnosis, and the predictor measurements age and base deficit, a measure of acidity.

Usage

```
data("diabetes")
```

Format

A data frame with 43 observations on the following 3 variables.

`Age`	age of the child.
`Def`	base deficit, a measure of acidity.
`C_pep`	logarithm of C-peptide concentration (pmol/ml).

Source

Generalized Additive Models (p.304) by Hastie & Tibshirani, Chapman & Hall.

funComp *Compatibility of functions*

Description

Not all functions inside glmxdiag are available for quasi-GLM or GLM extensions . The output table helps the user orientate inside the package in order to check which function can be used according to the model under analysis.

Usage

```
funComp()
```

Value

Returns a dataframe. 'X' inside a cell means that the row-function can be used with the specific column-model, otherwise it cannot be used.

Author(s)

Giuseppe Reale

Examples

```
funComp()
```

influenceDiag *Influence Diagnostic Measures*

Description

Calculates or extracts some influence diagnostic measures such as DFbetas, Cook's distance and leverage.

Usage

```
influenceDiag(model, approx = TRUE)
```

Arguments

model	a model supported by glmxdiag.
approx	logical, if TRUE the function is faster but returns approximated results.

Details

Leverage is extracted from the models using `hatvalues` function of each model class.

The elements for GLMs and negative binomial regression models are always approximated as they are extracted from `influence` function.

The argument `approx` can be useful only with models of class *vglm* and *betareg* since `influence` method function is not defined and items are fully calculated. It estimates *n* models where observations are excluded one by one. When `approx = T` one step approximation is used; the *tol* parameter within every fitting model function is set high such that the fitting process stops at the first iteration hence results are not exact but they are supposed to be close.

For betabinomial models, cook's distance is replaced with the quantity (b - b(-i)) Var(b) (b - b(-i)))/p as suggested by *glmtoolbox* package..

The aim of the function is to group these diagnostic measures in one list; one should use `influenceDiag` to calculate the measures and use the output object inside functions `DFbeta`, `cookDist` and `leverage` in order to graphically visualize results.

Value

Returns a list of class "influence" with the following elements:

`DFbeta`	data frame containing dfbetas for all observations and variables.
`cookDist`	a vector containing cook's distances.
`leverage`	hat values, i.e. the diagonal of the hat matrix.
`full.beta`	coefficients of the full model.
`family`	family name

Author(s)

Giuseppe Reale

Examples

```
data("mtcars")
mod <- glm(mpg ~ cyl + hp + carb + wt, family = Gamma, data = mtcars)
inf <- influenceDiag(mod)
cookDist(inf)
leverage(inf)
DFbeta(inf)
```

`leverage` *Visualize Leverages*

Description

Graphical visualization of leverages for each observation within input model. Corresponds to the main diagonal of the hat matrix, i.e *H[i,i]*.

Usage

```
leverage(object, label.id, n.label.id, xlab, ylab, pos,
         hline, lcol, lwd, lty, ...)
```

Arguments

`object`	object of class 'influence', see Details.
`label.id`	labels of observations, should be a vector with n elements.
`n.label.id`	number of observations with highest leverage to label in the plot.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`pos`	position of observations labels. Values `1, 2, 3, 4` respectively indicate below, left, above, right.
`hline`	numeric, where to position the horizontal line; useful to individuate points that exceed a specific treshold. Defaults to 2*p/n.
`lcol`	color of line.
`lwd`	width of line.
`lty`	type of line.
`...`	further arguments passed to `plot`

Details

This function only works with objects given as output from `influenceDiag`.

A rule of thumbs says that an observations has high leverage if H[i,i] > 2*p/n where p is the number of coefficients and n the sample size.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

Examples

```
data("mtcars")
mod <- glm(mpg ~ cyl + hp + carb, family = Gamma, data = mtcars)
inf <- influenceDiag(mod)
leverage(inf, label.id = rownames(mtcars))
```

---

linkLin *Checking linearity of link function*

Description

According to Lovison (2014), if the link function is correctly specified then there is a linear relationship between the working response variable *z* and the linear predictor *eta*. This plot suggests if the link function is appropriate.

Usage

```
linkLin(model, smooth = TRUE, xlab, ylab, main, pch, lcol, lwd, ...)
```

Arguments

`model`	a model supported by `glmxdiag`.
`smooth`	logical, whether to plot a smoothing spline.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`main`	an overall title for the plot.
`pch`	type of points.
`lcol`	color of the smoothing line.
`lwd`	size of the smoothing line.
`...`	further arguments passed to `plot`

Details

The assumption behind the output plot is that the model is correctly specified. In the y-axis there is the working response variable while on the x-axis there is the linear predictor: the more their relationship is linear the more appropriate the link function should be.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

Examples

```
## Simulate the data


set.seed(5)
n.obs <- 100
x <- rnorm(n.obs)
shape <- 25
mu <- exp(1 + .5* x)
y <- rgamma(n.obs, rate = shape / mu, shape = shape)

true.mod <- glm(y ~ x, family = Gamma(link = 'log'))
wrong1 <- glm(y ~ x, family = Gamma(link = 'inverse'))
wrong2 <- glm(y ~ x, family = Gamma(link = 'identity'))

par(mfrow = c(2, 2))
linkLin(true.mod)
linkLin(wrong1)
linkLin(wrong2)
par(mfrow = c(1, 1))
```

`moons` *Moons of the 13 planets of the Solar System*

Description

This dataset contains informations about dimensions, number of moons and distance from the Sun of the eight planets and the five dwarf planets of the Solar System. The objective is to find a proper relationship between the number of moons and other variables. All variables are normalized in respect to the Earth, so that it has always value 1.

Usage

```
data("moons")
```

Format

A data frame with 13 observations on the following 5 variables.

`Name` name of the planet.

`Distance` distance from the sun.

`Diameter` diameter of the planet.

`Mass` mass of the planet.

`Moons` number of moons.

Source

https://en.wikipedia.org/wiki/Planet

`Qresiduals` *Graphical normality testing for quantile residuals*

Description

Derives quantile residuals of the input model and creates two plots: the first compares density of a standard normal distribution to the residuals' empirical density while the other one is a *QQplot*.

Usage

```
Qresiduals(model, plot.it = TRUE)
```

Arguments

`model`	a model supported by `glmxdiag`.
`plot.it`	logical, whether to plot the results or not.

Details

Quantile residuals are defined on a continuous cumulative distribution function; for binomial, beta-binomial, poisson and negative binomial regression models the *Randomized quantile residuals* are used.

Residuals have an important role inside the global diagnostic of a regression model. Any departure from the standard normal distribution can be considered as a warning that one or more aspects of the model are misspecified. Quantile residuals are particullary useful with models that do not have a continuous response variable, such as a binomial or poisson models; see examples.

Value

Called for side effects but invisibly returns the calculated quantile residuals.

Author(s)

Giuseppe Reale

References

Peter K. Dunn and Gordon K. Smyth (1996). Randomized Quantile Residuals. Journal of Computational and Graphical Statistics.

Examples

```
# Simulate the data

set.seed(10)
n <- 100
x1 <- rt(n, df = 4)
x2 <- rnorm(n, sd = 2)
```

```
b <- c(1, .5, .5)
eta <- b[1] + b[2] * x1 + b[3] * x2
prob <- exp(eta)/(1 + exp(eta))
y <- rbinom(n, size = 1, prob = prob)

# The model is correctly specified
mod <- glm(y ~ x1 + x2, family = binomial)

res.p <- residuals(mod, type = 'pearson')
res.d <- residuals(mod, type = 'deviance')
qres <- Qresiduals(mod, plot.it = FALSE)

shapiro.test(res.p) # not normal
shapiro.test(res.d) # not normal
shapiro.test(qres) # normal

y.hat <- fitted(mod)

plot(y.hat, res.p) # uninformative
plot(y.hat, res.d) # uninformative
plot(y.hat, qres)

Qresiduals(mod)
```

`stopping` *Stopping*

Description

This data set concerns the stopping distances for cars travelling at different speeds. There are replicate values for some speeds. The objective is to find a model for these data.

Usage

```
data("stopping")
```

Format

A data frame with 63 observations on the following 2 variables.

`Speed` travelling speed (miles per hour).

`Distance` stopping distance (feet).

Source

Snee, R.D. (1986) An alternative approach to fitting models when re-expression of the response is useful. *Journal of Quality Technology*, 18,211-225.

A Handbook of Small Data Sets (p.354) by Hand, Daly, McConway, Lunn, Ostrowski. Chapman & Hall.

varCheck *Checking adequacy of variance function*

Description

According to GLM theory, the true variance function can be estimated as (y-mu)^2/phi. Comparing this quantity with the variance function assumed by the model suggests if the chosen one is appropriate. If the model is correct, points should be arranged near the bisector; otherwise this plot suggests the direction of the error made by chosing the variance function.

Usage

```
varCheck(model, xlab, ylab, pch, col, lcol, ...)
```

Arguments

`model`	a model supported by `glmxdiag`.
`xlab`	title for the x axis.
`ylab`	title for the y axis.
`pch`	type of points.
`col`	color of the points.
`lcol`	color of the bisector.
`...`	further arguments passed to `plot`.

Value

Doesn't return a value, called for side effects.

Author(s)

Giuseppe Reale

Examples

```
data(stopping, package = 'glmxdiag')
mod <- glm(Distance ~ Speed, family = Gamma(link = 'sqrt'), data = stopping)
varCheck(mod)
```

variableOut *Excluding a variable from the model*

Description

Graphically compares models in terms of Information Criterion, each one corresponds to a model where a specific variable is deleted from the linear predictor.

Usage

```
variableOut(model, k = 2, update.it = FALSE, xlab, ylab, pch,
                   col, lty, ylim, ...)
```

Arguments

`model`	a model supported by `glmxdiag`.
`k`	numeric, the penalty per parameter to be used; the default `k = 2` is the classical AIC.
`update.it`	logical; if TRUE, model without the variable corresponding to the lowest score is returned.
`xlab`	title for the x axis.
`ylab`	title for the y axis, by default is 'AIC'; should be changed if a different k is chosen.
`pch`	type of points.
`col`	color of points and segments.
`lty`	type of horizontal line.
`ylim`	y limits of the plot.
`...`	further arguments passed to `plot`.

Details

Each plotted point corresponds to the score of the model where the variable indicated on x axis is excluded. A dashed line is drawn in correspondence of the full model score.

Points and segments corresponding to variables whose deletion lead to a increment of the scored are black, those who lead to a decrement are red.

The output plot can be seen as a graphic version of the first step of `stepAIC` function inside `MASS` package.

Theory about Information Criterion suggests that if the minimum score doesn't belong to the full model, then the linear predictor may not be appropriate.

Value

Called for side effects, but if `update.it` is set to TRUE returns a model without the variable corresponding to the lowest score.

Author(s)

Giuseppe Reale

Examples

```
data("moons")
model <- glm(Moons ~ Diameter * Mass + Distance, family = poisson, data = moons)
variableOut(model)

n <- nobs(model)
new.model <- variableOut(model, k = log(n), ylab = 'BIC', update.it = TRUE)
```

Index