Package 'gldrm'

April 13, 2018

Type Package

Title Generalized Linear Density Ratio Models

Version 1.5

Description Fits a generalized linear density ratio model (GLDRM).
A GLDRM is a semiparametric generalized linear model.
In contrast to a GLM, which assumes a particular exponential family distribution,
the GLDRM uses a semiparametric likelihood to estimate the reference distribution.
The reference distribution may be any discrete, continuous, or mixed exponential
family distribution. The model parameters, which include both the regression
coefficients and the cdf of the unspecified reference distribution, are estimated
by maximizing a semiparametric likelihood. Regression coefficients are estimated
with no loss of efficiency, i.e. the asymptotic variance is the same as if the
true exponential family distribution were known.
Huang (2014) <doi:10.1080/01621459.2013.824892>.
Huang and Rathouz (2012) <doi:10.1093/biomet/asr075>.
Rathouz and Gao (2008) <doi:10.1093/biostatistics/kxn030>.

Depends R (>= 3.2.2)

Imports stats (>= 3.2.2), graphics (>= 3.2.2)

Suggests testthat (>= 1.0.2)

License MIT + file LICENSE

LazyData TRUE

RoxygenNote 6.0.1

NeedsCompilation no

Author Michael Wurm [aut, cre],
Paul Rathouz [aut]

Maintainer Michael Wurm <wurm@uwalumni.com>

Repository CRAN

Date/Publication 2018-04-13 07:58:20 UTC

R topics documented:

`beta.control` *Control arguments for β update algorithm*

Description

This function returns control arguments for the β update algorithm. Each argument has a default value, which will be used unless a different value is provided by the user.

Usage

```
beta.control(eps = 1e-10, maxiter = 1, maxhalf = 10)
```

Arguments

`eps`	Convergence threshold. The update has converged when the relative change in log-likelihood between iterations is less than `eps`. Only applies if `maxiter>1`.
`maxiter`	Maximum number of iterations allowed.
`maxhalf`	Maximum number of half steps allowed per iteration if log-likelihood does not improve.

Value

Object of S3 class "betaControl", which is a list of control arguments.

f0.control — *Control arguments for f0 update algorithm*

Description

This function returns control arguments for the f_0 update algorithm. Each argument has a default value, which will be used unless a different value is provided by the user.

Usage

```
f0.control(eps = 1e-10, maxiter = 1000, maxhalf = 20, maxlogstep = 2)
```

Arguments

`eps`	Convergence threshold. The update has converged when the relative change in log-likelihood between iterations is less than `eps`. absolute change is less than `thesh`.
`maxiter`	Maximum number of iterations allowed.
`maxhalf`	Maximum number of half steps allowed per iteration if log-likelihood does not improve between iterations.
`maxlogstep`	Maximum optimization step size allowed on the `log(f0)` scale.

Value

Object of S3 class "f0Control", which is a list of control arguments.

gldrm — *Fits a generalized linear density ratio model (GLDRM)*

Description

A GLDRM is a semiparametric generalized linear model. In contrast to a GLM, which assumes a particular exponential family distribution, the GLDRM uses a semiparametric likelihood to estimate the reference distribution. The reference distribution may be any discrete, continuous, or mixed exponential family distribution. The model parameters, which include both the regression coefficients and the cdf of the unspecified reference distribution, are estimated by maximizing a semiparametric likelihood. Regression coefficients are estimated with no loss of efficiency, i.e. the asymptotic variance is the same as if the true exponential family distribution were known.

Usage

```
gldrm(formula, data = NULL, link = "identity", mu0 = NULL,
  offset = NULL, gldrmControl = gldrm.control(),
  thetaControl = theta.control(), betaControl = beta.control(),
  f0Control = f0.control())
```

Arguments

formula	An object of class "formula".
data	An optional data frame containing the variables in the model.
link	Link function. Can be a character string to be passed to the make.link function in the stats package (e.g. "identity", "logit", or "log"). Alternatively, link can be a list containing three functions named linkfun, linkinv, and mu.eta. The first is the link function. The second is the inverse link function. The third is the derivative of the inverse link function. All three functions must be vectorized.
mu0	Mean of the reference distribution. The reference distribution is not unique unless its mean is restricted to a specific value. This value can be any number within the range of observed values, but values near the boundary may cause numerical instability. This is an optional argument with mean(y) being the default value.
offset	Known component of the linear term. Offset must be passed through this argument - offset terms in the formula will be ignored. value and covariate values. If sampling weights are a function of both the response value and covariates, then sampprobs must be a $n \times q$ matrix, where n is the number of observations and q is the number of unique observed values in the response vector. If sampling weights do not depend on the covariate values, then sampprobs may alternatively be passed as a vector of length n. All values must be nonnegative and are assumed to correspond to the sorted response values in increasing order.
gldrmControl	Optional control arguments. Passed as an object of class "gldrmControl", which is constructed by the gldrm.control function. See gldrm.control documentation for details.
thetaControl	Optional control arguments for the theta update procedure. Passed as an object of class "thetaControl", which is constructed by the theta.control function. See theta.control documentation for details.
betaControl	Optional control arguments for the beta update procedure. Passed as an object of class "betaControl", which is constructed by the beta.control function. See beta.control documentation for details.
f0Control	Optional control arguments for the f0 update procedure. Passed as an object of class "f0Control", which is constructed by the f0.control function. See f0.control documentation for details.

Details

The arguments linkfun, linkinv, and mu.eta mirror the "link-glm" class. Objects of this class can be created with the stats::make.link function.

The "gldrm" class is a list of the following items.

- conv Logical indicator for whether the gldrm algorithm converged within the iteration limit.
- iter Number of iterations used. A single iteration is a beta update, followed by an f0 update.
- llik Semiparametric log-likelihood of the fitted model.
- beta Vector containing the regression coefficient estimates.

- `mu` Vector containing the estimated mean response value for each observation in the training data.
- `eta` Vector containing the estimated linear combination of covariates for each observation.
- `f0` Vector containing the semiparametric estimate of the reference distribution, evaluated at the observed response values. The values of correspond to the support values, sorted in increasing order.
- `spt` Vector containing the unique observed response values, sorted in increasing order.
- `mu0` Mean of the estimated semiparametric reference distribution. The mean of the reference distribution must be fixed at a value in order for the model to be identifiable. It can be fixed at any value within the range of observed response values, but the `gldrm` function assigns `mu0` to be the mean of the observed response values.
- `varbeta` Estimated variance matrix of the regression coefficients.
- `seBeta` Standard errors for $\hat{\beta}$. Equal to `sqrt(diag(varbeta))`.
- `seMu` Standard errors for $\hat{\mu}$ computed from `varbeta`.
- `seEta` Standard errors for $\hat{\eta}$ computed from `varbeta`.
- `theta` Vector containing the estimated tilt parameter for each observation. The tilted density function of the response variable is given by

$$f(y|x_i) = f_0(y)\exp(\theta_i y)/\int f_0(u)\exp(\theta_i u)du.$$

- `bPrime` is a vector containing the mean of the tilted distribution, $b'(\theta_i)$, for each observation. `bPrime` should match `mu`, except in cases where `theta` is capped for numerical stability.

$$b'(\theta_i) = \int u f(u|x_i)du$$

- `bPrime2` is a vector containing the variance of the tilted distribution, $b''(\theta_i)$, for each observation.

$$b''(\theta_i) = \int (u - b'(\theta_i))^2 f(u|x_i)du$$

- `fTilt` is a vector containing the semiparametric fitted probability, $\hat{f}(y_i|x_i)$, for each observation. The semiparametric log-likelihood is equal to

$$\sum_{i=1}^{n} \log \hat{f}(y_i|x_i).$$

- `sampprobs` If sampling probabilities were passed through the `sampprobs` argument, then they are returned here in matrix form. Each row corresponds to an observation.
- `llikNull` Log-likelihood of the null model with no covariates.
- `lr.stat` Likelihood ratio test statistic comparing fitted model to the null model. It is calculated as $2 \times (llik - llik_0)/(p-1)$. The asymptotic distribution is F(p-1, n-p) under the null hypothesis.
- `lr.pval` P-value of the likelihood ratio statistic.

- `fTiltMatrix` is a matrix containing the semiparametric density for each observation, i.e. $\hat{f}(y|x_i)$ for each unique `y` value. This is a matrix with nrow equal to the number of observations and ncol equal to the number of unique response values observed. Only returned if `returnfTilt = TRUE` in the gldrmControl arguments.
- `score.logf0` Score function for `log(f0)`. Only returned if `returnf0ScoreInfo = TRUE` in the gldrmControl arguments.
- `info.logf0` Information matrix for `log(f0)`. Only returned if `returnf0ScoreInfo = TRUE` in the gldrmControl arguments.
- `formula` Model formula.
- `data` Model data frame.
- `link` Link function. If a character string was passed to the `link` argument, then this will be an object of class "link-glm". Otherwise, it will be the list of three functions passed to the `link` argument.

Value

An S3 object of class "gldrm". See details.

Examples

```
data(iris, package="datasets")

# Fit a gldrm with log link
fit <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
             data=iris, link="log")
fit

# Fit a gldrm with custom link function
link <- list()
link$linkfun <- function(mu) log(mu)^3
link$linkinv <- function(eta) exp(eta^(1/3))
link$mu.eta <- function(eta) exp(eta^(1/3)) * 1/3 * eta^(-2/3)
fit2 <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
              data=iris, link=link)
fit2
```

`gldrm.control` *Control arguments for* `gldrm` *algorithm*

Description

This function returns control arguments for the `gldrm` algorithm. Each argument has a default value, which will be used unless a different value is provided by the user.

Usage

```
gldrm.control(eps = 1e-10, maxiter = 100, returnfTiltMatrix = TRUE,
  returnf0ScoreInfo = FALSE, print = FALSE, betaStart = NULL,
  f0Start = NULL)
```

Arguments

eps	Convergence threshold. The fitting algorithm has converged when the relative change in log-likelihood between iterations is less than eps. A single iteration consists of a beta update followed by an f0 update.
maxiter	Maximum number of iterations allowed.
returnfTiltMatrix	Logical. Return nonparametric fitted probabilities for each observation. This is a matrix with nrow equal to the number of observations and ncol equal to the number of unique response values observed.
returnf0ScoreInfo	Logical. If TRUE, the score and information for log(f0) are returned as components of the "gldrm" object.
print	Logical. If TRUE, the relative change in the log-likelihood will be printed after each iteration.
betaStart	Optional vector of starting values for beta. If the call to gldrm contains a formula, the values of betaStart should correspond to the columns of the model matrix.
f0Start	Optional vector of starting values for f0. The length of the vector should be the number of unique values in the response, and the vector should correspond to these values sorted in increasing order. The starting values will be scaled to sum to one and tilted to have mean mu0. All values should be strictly positive.

Value

Object of S3 class "gldrmControl", which is a list of control arguments.

---

gldrmCI *Confidence intervals for gldrm coefficients*

Description

Calculates a Wald, likelihood ratio, or score confidence interval for a single gldrm coefficient. Also calculates upper or lower confidence bounds. Wald confidence intervals and bounds are calculated from the standard errors which are available from the gldrm model fit. For likelihood ratio and score intervals and bounds, a bisection search method is used, which takes longer to run.

Usage

```
gldrmCI(gldrmFit, term, test = c("Wald", "LRT", "Score"), level = 0.95,
  type = c("2-sided", "lb", "ub"), eps = 1e-10, maxiter = 100)
```

Arguments

gldrmFit	A gldrm model fit. Must be an S3 object of class "gldrm", returned from the gldrm function.
term	Character string containing the name of the coefficient of interest. The coefficient names are the names of the beta component of the fitted model object. They can also be obtained from the printed model output. Usually the names match the formula syntax, but can be more complicated for categorical variables and interaction terms.
test	Character string for the type confidence interval. Options are "Wald", "LRT" (for likelihood ratio), and "Score".
level	Confidence level of the interval. Should be between zero and one.
type	Character string containing "2-sided" for a two-sided confidence interval, "lb" for a lower bound, or "ub" for an upper bound.
eps	Convergence threshold. Only applies for test = "LRT" and test = "Score". Convergence is reached when likelihood ratio p-value is within eps of the target p-value, based on the level of the test. For example, a two-sided 95% confidence interval has target p-value of 0.025 for both the upper and lower bounds. A 95% confidence bound has target p-value 0.05.
maxiter	The maximum number of bisection method iterations for likelihood ratio intervals or bounds. For two-sided intervals, maxiter iterations are allowed for each bound.

Value

An S3 object of class 'gldrmCI', which is a list of the following items.

- term Coefficient name.
- test Type of interval or bound - Wald or likelihood ratio.
- level Confidence level.
- type Type of interval or bound - two-sided, upper bound, or lower bound.
- cilo/cihi Upper and lower interval bounds. One one of the two applies for confidence bounds.
- iterlo/iterhi Number of bisection iterations used. Only applies for likelihood ratio intervals and bounds.
- pvallo/pvalhi For likelihood ratio intervals and bounds, the p-value at convergence is reported.
- conv Indicator for whether the confidence interval limit or bound converged.

Examples

```
data(iris, package="datasets")

### Fit gldrm with all variables
fit <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
             data=iris, link="log")
```

```
### Wald 95% confidence interval for Sepal.Width
ci <- gldrmCI(fit, "Sepal.Width", test="Wald", level=.95, type="2-sided")
ci
```

---

gldrmLRT *Likelihood ratio test for nested models*

Description

Performs a likelihood ratio F-test between nested gldrm models. The F-statistic is calculated as $2 \times (llik - llik_0)/r$, where r is the difference is the number of parameters between the full and null models. The F-statistic has degrees of freedom r and $n - p$, where n is the number of observations and p is the number of parameters in the full model.

Usage

```
gldrmLRT(gldrmFit, gldrmNull)
```

Arguments

gldrmFit	The full model. Must be an object of S3 class 'gldrm' returned from the gldrm function.
gldrmNull	The sub-model being tested under the null hypotheses. Must be an object of S3 class 'gldrm' returned from the gldrm function.

Value

An S3 object of class 'gldrmLRT', containing numerator and denominator degrees of freedom, an F-statistic, and a p-value.

Examples

```
data(iris, package="datasets")

### Fit gldrm with all variables
fit <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
             data=iris, link="log")

### Fit gldrm without the categorical variable "Species"
fit0 <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width,
              data=iris, link="log")

### Likelihood ratio test for the nested models
lrt <- gldrmLRT(fit, fit0)
lrt
```

gldrmPIT *Confidence intervals for gldrm coefficients*

Description

Plots and returns the randomized probability inverse transform of a fitted gldrm.

Usage

```
gldrmPIT(gldrmFit, nbreaks = 7, cex.main = NULL, cex.lab = NULL,
  cex.axis = NULL)
```

Arguments

gldrmFit	A gldrm model fit. Must be an S3 object of class "gldrm", returned from the gldrm function. The matrix of semiparametric tilted probabilities must be returned, which is done by fitting gldrm with gldrmControl = gldrm.control(returnfTiltMatrix = TR
nbreaks	Number of breaks in the histogram.
cex.main	Text size for main titles.
cex.lab	Text size for axis labels.
cex.axis	Text size for axis numbers.

Details

The probability inverse transform is defined generally as $\hat{F}(y|x)$, which is the fitted conditional cdf of each observation evaluated at the observed response value. In the case of gldrm, the fitted cdf is descrete, so we draw a random value from a uniform distribution on the interval $(\hat{F}(y|x), \hat{F}(y-|x))$, where $y-$ is the next largest observed support less than y (or -Infinity if y is the minimum support value). The output and plots generated by this function will vary slightly each time it is called (unless the random number generator seed is set beforehand).

Value

Randomized robability inverse transform as a vector. Also plots the histogram and uniform QQ plot.

Examples

```
data(iris, package="datasets")

### Fit gldrm and return fTiltMatrix
fit <- gldrm(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
             data=iris, link="log")

# Probability inverse transform plot
gldrmPIT(fit)
```

predict.gldrm *Predict method for a gldrm object*

Description

Obtains predicted probabilities, predicted class, or linear predictors.

Usage

```
## S3 method for class 'gldrm'
predict(object, newdata = NULL, type = c("link", "response",
  "terms", "fTilt"), se.fit = FALSE, offset = NULL, ...)
```

Arguments

object	S3 object of class "gldrm", returned from the gldrm function.
newdata	Optional data frame. If NULL, fitted values will be obtained for the training data.
type	The type of prediction required. Type "link" returns the linear predictor. Type "response" returns the fitted mean. Type "terms" returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. Type "fTilt" returns a matrix containing the fitted nonparametric distribution for each observation. Each row of the matrix corresponds to an observation in newdata, and each column corresponds to a unique response value in the training data.
se.fit	Logical. If TRUE, standard errors are also returned. Does not apply for type = "fTilt".
offset	Optional offset vector. Only used if newdata is not NULL.
...	Not used. Additional predict arguments.

Value

The object returned depends on type.

print.gldrm *Print summary of gldrm fit*

Description

Prints fitted coefficients and standard errors, along with a likelihood ratio test against the null model.

Usage

```
## S3 method for class 'gldrm'
print(x, digits = 3, ...)
```

Arguments

x	S3 object of class "gldrm", returned from the gldrm function.
digits	Number of digits for rounding.
...	Unused. Additional arguments for print method.

---

print.gldrmCI *Print confidence interval*

Description

Print method for gldrmCI objects.

Usage

```
## S3 method for class 'gldrmCI'
print(x, digits = 3, ...)
```

Arguments

x	An S3 object of class 'gldrmCI'.
digits	Number of digits for rounding.
...	Not used. Additional arguments for print method.

---

print.gldrmLRT *Print likelihood ratio test results*

Description

Print method for gldrmLRT objects. Prints results of a likelihood ratio F-test between nested models.

Usage

```
## S3 method for class 'gldrmLRT'
print(x, digits = 3, ...)
```

Arguments

x	S3 object of class 'gldrmLRT', returned from the gldrmLRT function.
digits	Number of digits for rounding.
...	Not used. Additional arguments for print method.

`theta.control` *Control arguments for θ update algorithm*

Description

This function returns control arguments for the θ update algorithm. Each argument has a default value, which will be used unless a different value is provided by the user.

Usage

```
theta.control(eps = 1e-10, maxiter = 100, maxhalf = 20, maxtheta = 500,
  logit = TRUE, logsumexp = FALSE)
```

Arguments

`eps`	Convergence threshold for theta updates. Convergence is evaluated separately for each observation. An observation has converged when the difference between $b'(\theta)$ and μ is less than `epsTheta`.
`maxiter`	Maximum number of iterations.
`maxhalf`	Maximum number of half steps allowed per iteration if the convergence criterion does not improve.
`maxtheta`	Absolute value of theta is not allowed to exceed `maxtheta`.
`logit`	Logical for whether logit transformation should be used. Use of this stabilizing transformation appears to be faster in general. Default is TRUE.
`logsumexp`	Logical argument for whether log-sum-exp trick should be used. This may improve numerical stability at the expense of computational time.

Value

Object of S3 class "thetaControl", which is a list of control arguments.

Index