Package 'betareg'

September 3, 2009

Version 2.0-0

Date 2009-09-02

Title Beta Regression

Author Achim Zeileis

Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>

Description Beta regression for modeling rates and proportions.

Depends R (>= 2.6.0), stats

Imports graphics, sandwich, lmtest

Suggests car, lmtest, sandwich

License GPL-2

Repository CRAN

Date/Publication 2009-09-03 08:41:04

R topics documented:

betareg 2
betareg.control 4
FoodExpenditure 6
GasolineYield 7
gleverage 8
plot.betareg 9
predict.betareg 10
residuals.betareg 11
summary.betareg 12

Index **15**

betareg *Beta Regression for Rates and Proportions*

Description

Fit beta regression models for rates and proportions via maximum likelihood using a parametrization with mean (depending through a link function on the covariates) and dispersion parameter (called phi).

Usage

```
betareg(formula, data, subset, na.action, weights, offset,
  link = c("logit", "probit", "cloglog"),
  control = betareg.control(...),
  model = TRUE, y = TRUE, x = FALSE, ...)

betareg.fit(x, y, weights = NULL, offset = NULL,
  link = "logit", control = betareg.control())
```

Arguments

`formula`	symbolic description of the model (of type `y ~ x`).
`data, subset, na.action`	arguments controlling formula processing via `model.frame`.
`weights`	optional numeric vector of weights.
`offset`	optional numeric vector with an a priori known component to be included in the linear predictor.
`link`	character specification of link function.
`control`	a list of control arguments specified via `betareg.control`.
`model, y, x`	logicals. If `TRUE` the corresponding components of the fit (model frame, response, model matrix) are returned. For `betareg.fit`, `x` should be a numeric regressor matrix and `y` should be the numeric response vector (with values in (0,1)).
`...`	arguments passed to `betareg.control`.

Details

Beta regression as suggested by Ferrari and Cribari-Neto (2004) is implemented in `betareg`. It is useful in situations where the dependent variable is continuous and restricted to the unit interval (0, 1), e.g., resulting from rates or proportions. It is modeled to be beta-distributed with parametrization using mean and precision/dispersion parameter (called phi). The mean is linked, as in generalized linear models (GLMs), to the responses through a link function and a linear predictor. Estimation is performed by maximum likelihood (ML) via `optim` using analytical gradients and (by default) starting values from an auxiliary linear regression of the transformed response.

The main parameters of interest are the coefficients in the linear predictor and the additional precision/dispersion parameter phi which can either be treated as a full model parameter (default) or as a nuisance parameter. In the latter case the estimation does not change, only the reported information in output from `print`, `summary`, or `coef` (among others) will be different. See also `betareg.control`.

A set of standard extractor functions for fitted model objects is available for objects of class `"betareg"`, including methods to the generic functions `print`, `summary`, `plot`, `coef`, `vcov`, `logLik`, `residuals`, `predict`, `terms`, `model.frame`, `model.matrix`, `cooks.distance` and `hatvalues` (see `influence.measures`), `gleverage` (new generic), `estfun` and `bread` (from the **sandwich** package), and `coeftest` (from the **lmtest** package).

See `predict.betareg`, `residuals.betareg`, `plot.betareg`, and `summary.betareg` for more details on all methods.

Value

`betareg` returns an object of class `"betareg"`, i.e., a list with components as follows. `betareg.fit` returns an unclassed list with components up to `converged`.

`coefficients` vector with estimated regression coefficients and dispersion (or precision) parameter phi,

`residuals` a vector of raw residuals (observed - fitted),

`fitted.values` a vector of fitted means,

`optim` output from the `optim` call for maximizing the log-likelihood(s),

`method` the method argument passed to the `optim` call,

`control` the control arguments passed to the `optim` call,

`start` the starting values for the parameters passed to the `optim` call,

`weights` the weights used (if any),

`offset` the offset vector used (if any),

`n` number of observations,

`df.null` residual degrees of freedom for the null model (= `n - 2`),

`df.residual` residual degrees of freedom for fitted model,

`phi` logical indicating whether phi will be treated as a full model parameter or a nuisance parameter in subsequent calls to `print`, `summary`, `coef` etc.,

`loglik` log-likelihood of the fitted model,

`vcov` covariance matrix of all parameters in the model (including phi),

`pseudo.R.squared` pseudo R-squared value (squared correlation of linear predictor and link-transformed response),

`link` link object used,

`converged` logical indicating successful convergence of `optim`,

`call` the original function call,

`formula` the original formula,

terms	the terms object used,
levels	levels of the categorical regressors,
contrasts	contrasts corresponding to levels,
model	the full model frame (if model = TRUE),
y	the response proportion vector (if y = TRUE),
x	the model matrix (if x = TRUE).

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

See Also

summary.betareg, predict.betareg, residuals.betareg

Examples

```
## Section 4 from Ferrari and Cribari-Neto (2004)
data("GasolineYield", package = "betareg")
data("FoodExpenditure", package = "betareg")

## Table 1
gy <- betareg(yield ~ batch + temp, data = GasolineYield)
summary(gy)

## Table 2
fe_lin <- lm(I(food/income) ~ income + persons, data = FoodExpenditure)
library("lmtest")
bptest(fe_lin)
fe_beta <- betareg(I(food/income) ~ income + persons, data = FoodExpenditure)
summary(fe_beta)

## nested model comparisons via Wald and LR tests
fe_beta2 <- betareg(I(food/income) ~ income, data = FoodExpenditure)
lrtest(fe_beta, fe_beta2)
waldtest(fe_beta, fe_beta2)
```

---

betareg.control *Control Parameters for Beta Regression*

Description

Various parameters that control fitting of beta regression models using betareg.

Usage

```
betareg.control(phi = TRUE, method = "BFGS", maxit = 5000,
  hessian = FALSE, trace = FALSE, start = NULL, ...)
```

Arguments

phi	logical indicating whether the precision/dispersion parameter phi should be treated as a full model parameter (TRUE, default) or as a nuisance parameter.
method	characters string specifying the method argument passed to optim.
maxit	integer specifying the maxit argument (maximal number of iterations) passed to optim.
trace	logical or integer controlling whether tracing information on the progress of the optimization should be produced (passed to optim).
hessian	logical. Should the numerical Hessian matrix from the optim output be used for estimation of the covariance matrix? If FALSE (the default), the analytical solution is employed.
start	an optional vector with starting values for all parameters (including phi).
...	arguments passed to optim.

Details

All parameters in betareg are estimated by maximum likelihood using optim with control options set in betareg.control. Most arguments are passed on directly to optim, only start controls how optim is called.

Starting values can be supplied via start or estimated by lm.wfit, using the link-transformed response. Covariances are derived analytically (if hessian = FALSE, the default) or numerically using the Hessian matrix returned by optim.

The main parameters of interest are the coefficients in the linear predictor of the model and the additional precision/dispersion parameter phi which can either be treated as a full model parameter (default) or as a nuisance parameter. In the latter case the estimation does not change, only the reported information in output from print, summary, or coef (among others) will be different. See also examples.

Value

A list with the arguments specified.

See Also

betareg

Examples

```
data("GasolineYield", package = "betareg")

## regression with phi as full model parameter
gy1 <- betareg(yield ~ batch + temp, data = GasolineYield)
gy1

## regression with phi as nuisance parameter
gy2 <- betareg(yield ~ batch + temp, data = GasolineYield, phi = FALSE)
gy2
```

```
## compare reported output
coef(gy1)
coef(gy2)
summary(gy1)
summary(gy2)
```

`FoodExpenditure` *Proportion of Household Income Spent on Food*

Description

Data on proportion of income spent on food for a random sample of 38 households in a large US city.

Usage

```
data("FoodExpenditure")
```

Format

A data frame containing 38 observations on 3 variables.

food household expenditures for food.

income household income.

persons number of persons living in household.

Source

Taken from Griffiths et al. (1993, Table 15.4).

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

Griffiths, W.E., Hill, R.C., and Judge, G.G. (199). *Learning and Practicing Econometrics* New York: John Wiley and Sons.

See Also

`betareg`

Examples

```
data("FoodExpenditure", package = "betareg")

## Ferrari and Cribari-Neto (2004)
## Section 4
fe_lin <- lm(I(food/income) ~ income + persons, data = FoodExpenditure)
library("lmtest")
bptest(fe_lin)

## Table 2
fe_beta <- betareg(I(food/income) ~ income + persons, data = FoodExpenditure)
summary(fe_beta)
```

GasolineYield *Estimation of Gasoline Yields from Crude Oil*

Description

Operational data of the proportion of crude oil converted to gasoline after distilation and fractionation.

Usage

```
data("GasolineYield")
```

Format

A data frame containing 32 observations on 6 variables.

yield proportion of crude oil converted to gasoline after distilation and fractionation.

gravity crude oil gravity (degrees API).

pressure vapor pressure of crude oil (lbf/in2).

temp10 temperature (degrees F) at which 10 percent of crude oil has vaporized.

temp temperature (degrees F) at which all gasoline has vaporized.

batch factor indicating unique batch of conditions `gravity`, `pressure`, and `temp10`.

Details

This dataset was collected by Prater (1956), its dependent variable is the proportion of crude oil after distilation and fractionation. This dataset was analyzed by Atkinson (1985), who used the linear regression model and noted that there is "indication that the error distribution is not quite symmetrical, giving rise to some unduly large and small residuals" (p. 60).

The dataset contains 32 observations on the response and on the independent variables. It has been noted (Daniel and Wood, 1971, Chapter 8) that there are only ten sets of values of the first three explanatory variables which correspond to ten different crudes and were subjected to experimentally controlled distilation conditions. These conditions are captured in variable `batch` and the data were ordered according to the ascending order of `temp10`.

Source

Taken from Prater (1956).

References

Atkinson, A.C. (1985). *Plots, Transformations and Regression: An Introduction to Graphical Methods of Diagnostic Regression Analysis*. New York: Oxford University Press.

Daniel, C., and Wood, F.S. (1971). *Fitting Equations to Data*. New York: John Wiley and Sons.

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

Prater, N.H. (1956). *Estimate Gasoline Yields from Crudes*. New York: Springer-Verlag.

See Also

betareg

Examples

```
data("GasolineYield", package = "betareg")

gy1 <- betareg(yield ~ gravity + pressure + temp10 + temp, data = GasolineYield)
summary(gy1)

## Ferrari and Cribari-Neto (2004)
gy2 <- betareg(yield ~ batch + temp, data = GasolineYield)
## Table 1
summary(gy2)
## Figure 2
par(mfrow = c(3, 2))
plot(gy2, which = 1:4)
plot(gy2, which = 5, type = "deviance", sub.caption = "")
plot(gy2, which = 1, type = "deviance", sub.caption = "")

## exclude 4th observation
gy2a <- update(gy2, subset = -4)
gy2a
summary(gy2a)
```

---

gleverage *Generalized Leverage Values*

Description

Compute the generalized leverages values for fitted models.

Usage

```
gleverage(model, ...)
```

Arguments

model	a model object.
...	further arguments passed to methods.

Value

gleverage is a new generic for computing generalized leverage values as suggested by Wei, Hu, and Fung (1998). Currently, there is only a method for betareg models, implementing the formulas from Ferrari and Cribari-Neto (2004). Currently, this requires computations and storage of order $n \times n$.

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

Wei, B.-C., Hu, Y.-Q., and Fung, W.-K. (1998). Generalized Leverage and Its Applications. *Scandinavian Journal of Statistics*, **25**, 25–37.

See Also

betareg

Examples

```
data("GasolineYield", package = "betareg")
gy <- betareg(yield ~ batch + temp, data = GasolineYield)
gleverage(gy)
```

---

plot.betareg *Diagnostic Plots for betareg Objects*

Description

Various types of standard diagnostic plots can be produced, involving various types of residuals, influence measures etc. Half-normal plots can be produced using a simulation approach.

Usage

```
## S3 method for class 'betareg':
plot(x, which = 1:4,
  caption = c("Residuals vs indices of obs.", "Cook's distance plot",
    "Generalized leverage vs predicted values", "Residuals vs linear predictor",
    "Half-normal plot of residuals"),
    sub.caption = paste(deparse(x$call), collapse = "\n"), main = "",
    ask = prod(par("mfcol")) < length(which) && dev.interactive(),
    ..., type = "deviance", nsim = 100, level = 0.9)
```

Arguments

x	fitted model object of class "betareg".
which	if a subset of the plots is required, specify a subset of the numbers 1:8.
caption	captions to appear above the plots.
sub.caption	common title-above figures if there are multiple.
main	title to each plot-in addition to the above caption.
ask	logical. If TRUE, the user is asked before each plot.
...	other parameters to be passed through to plotting functions.
type	character indicating type of residual to be used, see residuals.betareg.
nsim	number of simulations in half-normal plots.
level	confidence level in half-normal plots.

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

See Also

betareg

Examples

```
data("GasolineYield", package = "betareg")

gy <- betareg(yield ~ gravity + pressure + temp10 + temp, data = GasolineYield)

par(mfrow = c(3, 2))
plot(gy, which = 1:5)
```

---

predict.betareg *Prediction Method for betareg Objects*

Description

Extract various types of predictions from beta regression models: either on the scale of responses in (0, 1) or the scale of the linear predictor.

Usage

```
## S3 method for class 'betareg':
predict(object, newdata = NULL,
  type = c("response", "link"), na.action = na.pass, ...)
```

Arguments

`object`	fitted model object of class `"betareg"`.
`newdata`	optionally, a data frame in which to look for variables with which to predict. If omitted, the original observations are used.
`type`	character indicating type of residuals.
`na.action`	function determining what should be done with missing values in `newdata`. The default is to predict `NA`.
`...`	currently not used.

Examples

```
data("GasolineYield", package = "betareg")

gy <- betareg(yield ~ gravity + pressure + temp10 + temp, data = GasolineYield)

cbind(
  predict(gy, type = "response"),
  predict(gy, type = "link")
)
```

`residuals.betareg` *Residuals Method for betareg Objects*

Description

Extract various types of residuals from beta regression models: raw response residuals (observed - fitted), Pearson residuals (raw residuals scaled by square root of variance function), deviance residuals (scaled log-likelihood contributions), and different kinds of weighted residuals suggested by Espinheira et al. (2008).

Usage

```
## S3 method for class 'betareg':
residuals(object,
  type = c("deviance", "pearson", "response", "weighted", "sweighted", "sweighted2"
  ...)
```

Arguments

`object`	fitted model object of class `"betareg"`.
`type`	character indicating type of residuals.
`...`	currently not used.

Details

The definitions of all residuals are provided in Espinheira et al. (2008): Equation 2 for `"pearson"`, last equation on page 409 for `"deviance"`, Equation 6 for `"weighted"`, Equation 7 for `"sweighted"`, and Equation 8 for `"sweighted2"`.

Espinheira et al. (2008) recommend to use `"sweighted2"`.

References

Espinheira, P.L., Ferrari, S.L.P., and Cribari-Neto, F. (2008). On Beta Regression Residuals. *Journal of Applied Statistics*, **35**(4), 407–419.

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

See Also

`betareg`

Examples

```
data("GasolineYield", package = "betareg")

gy <- betareg(yield ~ gravity + pressure + temp10 + temp, data = GasolineYield)

gy_res <- cbind(
  residuals(gy, type = "pearson"),
  residuals(gy, type = "deviance"),
  residuals(gy, type = "response"),
  residuals(gy, type = "weighted"),
  residuals(gy, type = "sweighted"),
  residuals(gy, type = "sweighted2")
)
colnames(gy_res) <- c("pearson", "deviance", "response",
  "weighted", "sweighted", "sweighted2")
pairs(gy_res)
```

---

summary.betareg *Methods for betareg Objects*

Description

Methods for extracting information from fitted beta regression model objects of class `"betareg"`.

Usage

```
## S3 method for class 'betareg':
summary(object, phi = NULL, ...)

## S3 method for class 'betareg':
coef(object, phi = NULL, ...)
## S3 method for class 'betareg':
vcov(object, phi = NULL, ...)
## S3 method for class 'betareg':
bread(x, phi = NULL, ...)
## S3 method for class 'betareg':
estfun(x, phi = NULL, ...)
```

Arguments

`object, x`	fitted model object of class `"betareg"`.
`phi`	logical indicating whether the precision/dispersion parameter phi should be reported as a full model parameter (`TRUE`) or a nuisance parameter (`FALSE`). The default is taken from `object$phi`.
`...`	currently not used.

Details

A set of standard extractor functions for fitted model objects is available for objects of class `"betareg"`, including methods to the generic functions `print` and `summary` which print the estimated coefficients along with some further information. The `summary` in particular supplies partial Wald tests based on the coefficients and the covariance matrix. As usual, the `summary` method returns an object of class `"summary.betareg"` containing the relevant summary statistics which can subsequently be printed using the associated `print` method.

A `logLik` method is provided, hence `AIC` can be called to compute information criteria.

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. *Journal of Applied Statistics*, **31**(7), 799–815.

See Also

`betareg`

Examples

```
data("GasolineYield", package = "betareg")

gy <- betareg(yield ~ gravity + pressure + temp10 + temp, data = GasolineYield)

summary(gy)
coef(gy)
vcov(gy)
```

```
logLik(gy)
AIC(gy)
```

Index

∗Topic **datasets**
FoodExpenditure, 6
GasolineYield, 7
∗Topic **regression**
betareg, 1
betareg.control, 4
gleverage, 8
plot.betareg, 9
predict.betareg, 10
residuals.betareg, 11
summary.betareg, 12

AIC, *13*

betareg, 1, *4–6*, *8–10*, *12*, *13*
betareg.control, *2*, 4, *5*
bread, *2*
bread.betareg *(summary.betareg)*, 12

coef, *2*
coef.betareg *(summary.betareg)*, 12
coeftest, *2*
coeftest.betareg *(summary.betareg)*, 12
cooks.distance.betareg *(summary.betareg)*, 12

estfun, *2*
estfun.betareg *(summary.betareg)*, 12

FoodExpenditure, 6

GasolineYield, 7
gleverage, *2*, 8

hatvalues.betareg *(summary.betareg)*, 12

influence.measures, *2*

lm.wfit, *5*
logLik, *2*, *13*
logLik.betareg *(summary.betareg)*, 12

model.frame, *2*
model.frame.betareg *(summary.betareg)*, 12
model.matrix, *2*
model.matrix.betareg *(summary.betareg)*, 12

optim, *2*, *4*, *5*

plot, *2*
plot.betareg, *2*, 9
predict, *2*
predict.betareg, *2*, *4*, 10
print, *2*, *13*
print.betareg *(summary.betareg)*, 12
print.summary.betareg *(summary.betareg)*, 12

residuals, *2*
residuals.betareg, *2*, *4*, *9*, 11

summary, *2*, *13*
summary.betareg, *2*, *4*, 12

terms, *2*
terms.betareg *(summary.betareg)*, 12

vcov, *2*
vcov.betareg *(summary.betareg)*, 12