Package ‘robustbetareg’

October 8, 2022

Version 0.1.0

Title Robust of Beta Regression Model

Author Yuri Maluf [aut,cre], Silvia Ferrari [ths], Francisco Queiroz [cre], Terezinha Ribeiro [cre]

Maintainer Yuri Maluf <yurimaluf@gmail.com>, Francisco Queiroz <ffelipeq@outlook.com> and Terezinha Ribeiro <terezinha.ribeiro@unb.br>

Description Robust estimators for modeling beta-distributed dependent variables as rates and proportions. The implemented estimators are: LSMLE, LMDPDE, SMLE and MDPDE. Robust Wald-type tests are included. Additionally, diagnostic tools such as goodness-of-fit and several kinds of residuals are also available.

Depends R (>= 3.0.0), betareg

Imports Rmpfr, rstudioapi, crayon, pracma, numDeriv, Formula, robustbase, zoo, methods, BBmisc, miscTools, Matrix, parallel

License GPL-3

NeedsCompilation no

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

degb *The EGB of the Second Type*

Description

Density, distribution function and random generation for the exponential generalized beta (EGB) of the second type.

Usage

```
degb(y_star, mu, phi, log = FALSE)

pegb(q, mu, phi)

qegb(p, mu, phi)

regb(n, mu, phi)
```

Arguments

`y_star, q`	vector of quantiles.
`mu`	mu parameter ($\mu \in (0,1)$).
`phi`	phi parameter ($\phi > 0$).
`log`	a logical value. If TRUE, it returns the log of density function.
`p`	vector of probabilities.
`n`	number of observations.

Details

The EGB density function is $f_\theta(y^\star;\mu,\phi) = B^{-1}(\mu\phi,(1-\mu)\phi)\exp\{-y^\star(1-\mu)\phi\}/(1+\exp\{-y^\star\})^\phi$, with $\mu \in (0,1), \phi > 0$ and $y^\star \in R$. The cumulative distribution function is given by $F_\theta(y^\star;\mu,\phi) = \frac{{}_2F_1(\mu\phi, 1-(1-\mu)\phi; \mu\phi+1; (1+e^{-y^\star})^{-1})}{B(\mu\phi;(1-\mu)\phi)\mu\phi(1+e^{-y^\star})^{\mu\phi}}$, where ${}_2F_1(\cdot,\cdot;\cdot;\cdot)$ is the hypergeometric function. The $E(y^\star) = \psi(\mu\phi)-\psi((1-\mu)\phi)$ and $Var(y^\star) = \psi'(\mu\phi)+\psi'((1-\mu)\phi)$ where ψ is the digamma function and ψ' its first derivative. See Kerman and McDonald (2015) for more details.

Value

`degb` gives the density, `pegb` gives the distribution function, `qegb` gives the quantile function, and `regb` generates random sample of EGB variables.

References

Kerman, S. McDonald, J. B. Skewness-kurtosis bounds for EGB1, EGB2, and special cases. Communications in Statistics - Theory and Methods, 44:3857-3864 (2015).

HIC *Health Insurance Coverage (HIC)*

Description

A dataset containing the proportion of the population from several counties of the state of São Paulo in Brazil who have health insurance.

Usage

```
data("HIC", package = "robustbetareg")
```

Format

A data frame with 80 rows and 4 variables:

County the corresponding county of the state of São Paulo, Brazil

Urbanization percentage of the total population who lives in the city's urban zone

GDP_percapita per capita gross domestic product (GDP)

Percent_HIC the health insurance coverage index

plot.robustbetareg *Interactive Plots for Diagnostic of robustbetareg Object Class*

Description

Several types of standard diagnostic plots can be produced interactively, involving various kinds of residuals, influence measures, weights etc.

Usage

```
## S3 method for class 'robustbetareg'
plot(object, ask = TRUE, ...)
```

Arguments

`object`	fitted model object of class `robustbetareg`.
`ask`	logical. If TRUE the user is asked before each plot.
`...`	other parameters to be passed through to plotting functions.

See Also

`robustbetareg`, `residuals.robustbetareg`

Examples

```
## Not run:
get(data("HIC", package = "robustbetareg")_
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | GDP_percapita,
data = HIC, alpha = 0.06)
plot(hic)
## End(Not run)
```

---

plotenvelope *Simulated Envelope of Residuals*

Description

Plot a Simulated Envelope of Robust Beta Residuals from robustbetareg Object Class.

Usage

```
plotenvelope(object, type = c("sweighted2", "pearson", "weighted", "sweighted",
"sweighted.gamma", "sweighted2.gamma", "combined", "combined.projection", "sweighted3"),
conf = 0.95, n.sim = 100, PrgBar = T, control = robustbetareg.control(...), ...)
```

Arguments

`object`	Fitted model object of class `robustbetareg` (see `robustbetareg`).
`type`	character indicating type of residuals.
`conf`	the confidence level of the envelopes required. The default is to find 95% confidence envelopes.
`n.sim`	the number of simulation sample. Deafault `n.sim=100`.
`PrgBar`	a logical value. If TRUE the progress bar will be shown in the console.
`control`	a list of control arguments specified via `robustbetareg.control`.
`...`	other parameters to be passed through to plotting functions.

Value

Return a simulated envelope graphic.

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.

Espinheira, P.L., Santos, E.G.and Cribari-Neto, F. (2017). On nonlinear beta regression residuals. Biometrical Journal, 59(3), 445-461.

See Also

robustbetareg, robustbetareg.control, residuals.robustbetareg

Examples

```
## Not run:
get(data("HIC", package = "robustbetareg"))
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | GDP_percapita, data = HIC, alpha = 0.06)
plotenvelope(hic, n.sim = 500)

get(data("RiskManagerCost", package = "robustbetareg"))
rmc <- robustbetareg(FIRMCOST ~ INDCOST + SIZELOG | INDCOST + SIZELOG, data = RiskManagerCost)
plotenvelope(rmc, conf = 0.90)
## End(Not run)
```

predict.robustbetareg *Prediction Methods for robustbetareg Objects Class*

Description

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

Usage

```
predict(object, newdata = NULL,
type = c("response", "link", "precision", "variance", "quantile"), at = 0.5)
```

Arguments

object	Fitted model object of class robustbetareg (see robustbetareg).
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 predictions: fitted means of response ("response"), corresponding linear predictor ("link"), fitted precision parameter phi ("precision"), fitted variances of response ("variance"), or fitted quantile(s) of the response distribution ("quantile").
at	numeric vector indicating the level(s) at which quantiles should be predicted (only if type = "quantile"), defaulting to the median at = 0.5.

Value

Return a vector of predicted values.

See Also

robustbetareg

Examples

```
## Not run:
get(data("HIC", package = "robustbetareg"))
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | 1, data = HIC, alpha = 0.04)
cbind(predict(hic, type = "response"), predict(hic, type = "quantile", at = c(0.25, 0.5, 0.75)))
## End(Not run)
```

residuals.robustbetareg

Residuals Method for robustbetareg Object

Description

Extract several types of residuals from robust beta regression models: Pearson residuals (raw residuals scaled by square root of variance function) and different kinds of weighted residuals suggested by Espinheira et al. (2008) and Espinheira et al. (2017).

Usage

```
residuals(object,
type = c("sweighted2", "pearson", "weighted", "sweighted", "sweighted.gamma",
 "sweighted2.gamma", "combined", "combined.projection", "sweighted3"), ...)
```

Arguments

object	fitted model object of class robustbetareg (see robustbetareg).
type	character indicating type of residuals.
...	currently not used.

Details

The definitions of the first four residuals are provided in Espinheira et al. (2008): Equation (2) for "pearson", Equation (6) for "weighted", Equation (7) for "sweighted", and Equation (8) for "sweighted2". For the last four residuals the definitions are described in Espinheira et al. (2017): Last equation of Equation (7) and Equation (10) for "sweighted.gamma" and "sweighted2.gamma" respectively, Equation (9) for "combined", and Equation (11) for "combined.projection".

Value

residuals returns a vector of selected residuals type.

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.

Espinheira, P.L., Santos, E.G.and Cribari-Neto, F. (2017). On nonlinear beta regression residuals. Biometrical Journal, 59(3), 445-461.

See Also

robustbetareg

Examples

```
## Not run:
get(data("HIC”, package = "robustbetareg”))
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | 1, data = HIC, alpha = 0.04)
res <- residuals(hic, type = "sweighted2”)
plot(res)
abline(h = 0)
## End(Not run)
```

RiskManagerCost *Risk Managers Cost Effectiveness*

Description

The dataset are from a questionnaire that was sent to risk managers of large U.S.-based organizations.

Usage

```
data("RiskManagerCost”, package = "robustbetareg”)
```

Format

A data frame with 73 rows and 7 variables:

FIRMCOST The measure of the firm’s risk management cost effectiveness, defined as total property and casualty premiums and uninsured losses as a percentage of total assets

ASSUME Per occurrence retention amount as a percentage of total assets

CAP Indicates that the firm owns a captive insurance company

SIZELOG Logarithm of total assets

INDCOST A measure of the firm’s industry risk

CENTRAL A measure of the importance of the local managers in choosing the amount of risk to be retained

SOPH A measure of the degree of importance in using analytical tools ...

Source

Schmit and Roth (1990)

References

Schmit, Joan T., and Kendall Roth. “Cost Effectiveness of Risk Management Practices.” The Journal of Risk and Insurance, vol. 57, no. 3, 1990, pp. 455–70. JSTOR

robustbetareg *Robust Beta Regression*

Description

Fit robust beta regression models for rates and proportions via LSMLE, LMDPDE, SMLE and MDPDE using a parametrization with mean (depending through a link function on the covariates) and precision parameter (called phi).

Usage

```
robustbetareg(formula, data, alpha, type = c("LSMLE", "LMDPDE", "SMLE", "MDPDE"),
link = c("logit", "probit", "cloglog", "cauchit", "loglog"),
link.phi = NULL, control = robustbetareg.control(...), model = TRUE, ...)

LSMLE.fit(y, x, z, alpha = NULL, link = "logit",
link.phi = "log", control = robustbetareg.control(...), ...)

LMDPDE.fit(y, x, z, alpha = NULL, link = "logit",
link.phi = "log", control = robustbetareg.control(...), ...)

SMLE.fit(y, x, z, alpha = NULL, link = "logit",
link.phi = "log", control = robustbetareg.control(...), ...)

MDPDE.fit(y, x, z, alpha = NULL, link = "logit",
link.phi = "log", control = robustbetareg.control(...), ...)
```

Arguments

`formula` symbolic description of the model (of type y ~ x or y ~ x | z).

`data` arguments controlling formula.

`alpha` the tuning value (α) within [0,1) for robust estimation. When alpha is equal to zero (α=0) is equivalent to MLE. If this argument is suppressed, the tuning parameter should be selected automatically through the data-driven algorithm proposed by Ribeiro and Ferrari (2022).

`type` character specification of the type of estimator. Currently, LSMLE (default), LMDPDE, SMLE and MDPDE.

`link` character specification of the link function in the mean model. Currently, "logit", "probit", "cloglog", "cauchit", "loglog" are supported

`link.phi` character specification of the link function in the precision model (phi). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type y ~ x where the default is "identity"

`control` a list of control arguments specified via `robustbetareg.control`.

`model` logicals for `robustbetareg`. If TRUE the corresponding components of the fit (model frame, response, model matrix) are returned. For LSMLE.fit, LMDPDE.fit, SMLE.fit and MDPDE.fit `y` must be a numeric response vector within (0,1).

`y, x, z` `y` should be a numeric response vector ($y \in (0, 1)$), `x` should be a numeric regresor matrix and `z` should be a regressor matrix for the precision model, where there is the intercept only.

`...` argument to be passed to `robustbetareg.control`

Details

Robust beta regression estimators LSMLE and LMDPDE are introduced by Maluf, Ferrari and Queiroz (2022), SMLE is sugested by Ribeiro and Ferrari (2022) and MDPDE by Ghosh (2019). All those estimators are implemented in `robustbetareg` routine. Those robust estimators are useful alternative for maximum likelihood estimator (MLE) since it may be highly influenced by the presence of outlying observation in the dataset. In Ribeiro and Ferrari (2022), the tuning parameter is referred as `q`. For the sake of simplicity, here the tuning parameter is treated by `alpha= 1 -q`. The `robustbetareg` offers a robust version of `betareg` function in **betareg** package. It is noteworthy that depending on the tuning value in some situations SMLE and MDPDE are not applicable, since these estimators are restricted to a subset of parameter space. This may cause a numerical instabilities or no convergence. In these cases, we recomend to select LSMLE or LMDPDE to get rid of those problems. For more details see Maluf, Ferrari and Queiroz (2022).

Value

`robustbetareg` returns an object of class `"robustbetareg"` with a list of the following components. The functions `LSMLE.fit`, `LMDPDE.fit`, `SMLE.fit` and `MDPDE.fit` return an unclassed list with itens up to `method`:

`coefficients` A numeric vector of parameter estimates,

`vcov` the covariance matrix of all parameters in the model,

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

`fitted.values` the vector of predicted values,

`start` the starting values for the parameters estimator,

`weights` the weights generated by robust estimator for each obs,

`Tuning` the selected tuning parameter,

`residuals` a vector of standardized weighted residual 2 (see Espinheira et al (2008)),

`n` number of observations,

`link` mean link function applied,

`link.phi` precision link function applied,

`Optimal.Tuning` logical indicating whether the auto selecting tuning algorithm was selected,

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

`control` the control arguments passed to auto selecting tuning algorithm and `optim` call,

`std.error` the standard error of all parameters,

`method` estimator type,

`call` the original function call,

`formula` the original formula,

`model` the full model frame,

`terms` a list with elements "`mean`", "`precision`" and "`full`" including the terms objects for the respective models,

`y` the response proportion vector,

`data` the dataset used.

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.

Ghosh, A. Robust inference under the beta regression model with application to health care studies. Statistical Methods in Medical Research, 28:271-888 (2019).

Maluf, Y. S., Ferrari, S. L., & Queiroz, F. F. (2022). Robust beta regression through the logit transformation. arXiv

Ribeiro, K. A. T. Ferrari, S. L. P. Robust estimation in beta regression via maximum Lq-likelihood. Statistical Papers (2022).

See Also

`robustbetareg.control`

Examples

```
## Not run:
#Table 1
data("HIC", package = "robustbetareg")
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | 1, data = HIC, type = "LMDPDE")
summary(hic)
```

```
#Table 2
data("RiskManagerCost", package = "robustbetareg")
rmc <- robustbetareg(FIRMCOST ~ INDCOST + SIZELOG | INDCOST + SIZELOG, data = RiskManagerCost, alpha = 0.04)
summary(rmc)
## End(Not run)
```

robustbetareg.control *Control Parameter for Robust Beta Regression*

Description

Several parameters that control fitting of robust beta regression models using robustbetareg.

Usage

```
robustbetareg.control(start = NULL, alpha.optimal = TRUE, tolerance = 1e-3,
maxit = 5000, L = 0.02, M = 3, ...)
```

Arguments

start	a numeric vector as an initial guess of parameter estimation.
alpha.optimal	a logical value. If TRUE the tuning parameter should be selected automatically.
tolerance	numeric tolerance for convergence.
maxit	integer specifying the maxit argument of iterations used by the Newton-Raphson algorithm.
L	a parameter of auto selecting algorithm of tuning parameter.
M	a integer parameter value of auto selecting algorithm of tuning parameter.
...	currently not used.

Details

The arguments L and M set the parameters of the data-driven algorithm for selecting the α tuning parameter of robust estimator. See Ribeiro and Ferrari (2022) for more details.

Value

A list with the arguments specified.

References

Ribeiro, K. A. T. Ferrari, S. L. P. Robust estimation in beta regression via maximum Lq -likelihood. Statistical Papers (2022).

summary.robustbetareg *Methods for robustbetareg Objects*

Description

Methods for extracting information from fitted robust beta regression model objects of class "robustbetareg"

Usage

```
## S3 method for class 'robustbetareg'
summary(object, type = "sweighted2", ...)

## S3 method for class 'robustbetareg'
coef(object, model = c("full", "mean", "precision"))
```

Arguments

`object, x`	fitted model of class `robustbetareg`.
`type`	character specifying type of residuals to be included in the summary output, see `residuals.robustbetareg`.
`...`	currently not used.
`model`	character specifying for which component of the model coefficients/covariance should be extracted.

Details

A set of standard extractor functions for fitted model objects is available for objects of class "robustbetareg", including methods to the generic functions print and summary which print the estimated coefficients along with some further information.

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.

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

Espinheira, P.L., Santos, E.G.and Cribari-Neto, F. (2017). On nonlinear beta regression residuals. Biometrical Journal, 59(3), 445-461.

Ghosh, A. Robust inference under the beta regression model with application to health care studies. Statistical Methods in Medical Research, 28:271-888 (2019).

Maluf, Y. S., Ferrari, S. L., & Queiroz, F. F. (2022). Robust beta regression through the logit transformation. arXiv

Ribeiro, K. A. T. Ferrari, S. L. P. Robust estimation in beta regression via maximum Lq-likelihood. Statistical Papers (2022).

See Also

robustbetareg

Examples

```
## Not run:
data("HIC", package = "robustbetareg")
hic <- robustbetareg(Percent_HIC ~ Urbanization + GDP_percapita | GDP_percapita, data = HIC)
summary(hic)
coef(hic)
## End(Not run)
```

---

waldtypetest *Robust Wald-type Tests*

Description

Wald-type tests for both simple and composite hypothesis for independent but non-homogeneous observations, based on LSMLE, LMDPDE, SMLE and MDPDE estimators.

Usage

```
waldtypetest(object, FUN, ...)
```

Arguments

`object`	fitted model object of class `robustbetareg` (see robustbetareg).
`FUN`	the function representing the null hypothesis to be tested.
`...`	Further arguments to be passed.

References

Basu, A., Ghosh, A., Martin, N. et al. Robust Wald-type tests for non-homogeneous observations based on the minimum density power divergence estimator. Metrika 81, 493–522 (2018)

Maluf, Y. S., Ferrari, S. L., & Queiroz, F. F. (2022). Robust beta regression through the logit transformation. arXiv

Ribeiro, K. A. T. Ferrari, S. L. P. Robust estimation in beta regression via maximum Lq-likelihood. Statistical Papers (2022).

See Also

robustbetareg

Examples

```
## Not run:
set.seed(2022)
N <- 40 #Sample Size
beta.coef <- c(-1,-2) #Arbitrary Beta Coefficients
gamma.coef <- c(5) #Arbitrary Gamma Coefficient
X <- cbind(rep(1,N), x <- runif(N)) #Design Matrix
mu <- exp(X%*%beta.coef)/(1+exp(X%*%beta.coef)) #Inverse Logit Link Function
phi <- exp(gamma.coef) #Inverse Log Link Function
y <- rbeta(N, mu*phi, (1-mu)*phi) #Sampling Response Variable
y[26] <- rbeta(1,((1 + mu[26])/2)*phi, (1-((1 + mu[26])/2))*phi) #Contaminated data point
SimData <- as.data.frame(cbind(y,x))
colnames(SimData) <- c("y","x")
mle <- robustbetareg(y ~ x | 1, data = SimData, alpha = 0) #Non-Robust Estimator
lsmle <- robustbetareg(y ~ x | 1, data = SimData) #Robust Estimator
h0 <- function(theta, B){theta[1:2] - B} #Hiphothesis to be tested
waldtypetest(mle, h0, B = beta.coef) #Testing beta.1=-1 and beta.2=-2
waldtypetest(lsmle, h0, B = beta.coef) #Testing beta.1=-1 and beta.2=-2
## End(Not run)
```

Index