Package 'BDWreg'

February 17, 2017

Type Package

Version 1.2.0

Date 2017-02-16

Author Hamed Haselimashhadi <hamedhaseli@gmail.com>

Maintainer Hamed Haselimashhadi <hamedhaseli@gmail.com>

Depends R (>= 3.0)

Description A Bayesian regression model for discrete response, where the conditional distribution is modelled via a discrete Weibull distribution. This package provides an implementation of Metropolis-Hastings and Reversible-Jumps algorithms to draw samples from the posterior. It covers a wide range of regularizations through any two parameter prior. Examples are Laplace (Lasso), Gaussian (ridge), Uniform, Cauchy and customized priors like a mixture of priors. An extensive visual toolbox is included to check the validity of the results as well as several measures of goodness-of-fit.

Title Bayesian Inference for Discrete Weibull Regression

License LGPL (>= 2)

Imports coda, parallel, foreach, doParallel, MASS, methods, graphics, stats, utils, DWreg

URL http://hamedhaseli.webs.com

NeedsCompilation no

Repository CRAN

Date/Publication 2017-02-17 14:37:54

R topics documented:

bdw 2
bdw.mc 5
plot.bdw 7
summary.bdw 8

Index **9**

bdw *Bayesian parameter estimation for discrete Weibull regression*

Description

Bayesian estimation of the parameters for discrete Weibull regression. The conditional distribution of the response given the predictors is assumed to be DW with parameters q and beta dependent on the predictors.

Usage

```
bdw(data, formula = NA, reg.q = FALSE, reg.b = FALSE,
  logit = TRUE, initial = c(.5,.5), iteration = 25000,
  v.scale = 0.1,RJ = FALSE,  dist.q = imp.d, dist.b = imp.d,
  q.par = c(0, 0), b.par = c(0, 0), penalized = FALSE,
  dist.l = imp.d, l.par = c(0, 0), bi.period = 0.25, cov = 1,
  sampling = c("bin"), est = Mode, fixed.l = -1,jeffrey = FALSE,
  ...)
```

Arguments

data	A data frame containing the variables in the model. If not found in data, the variables are taken from the environment (formula).
formula	An object of class "formula". A symbolic description of the model to be fitted.
reg.q	Logical flag. If TRUE, the model includes a dependency of q on the predictors x, as explained in 'logit' argument.
reg.b	Logical flag. If TRUE, the model includes a dependency of beta on the predictors x, given by: $$log(\beta) = \gamma_0 + \gamma_1 X_1 + \ldots + \gamma_p X_p.$$ If FALSE, beta(x) is constant.
logit	Logical flag. If TRUE, the model includes a dependency of q on predictors via a logit transformation as explained below. The conditional distribution of Y (response) given x (predictors) is assumed a DW(q(x),beta(x)). If logit=TRUE & (reg.q=TRUE) $$log(q/(1-q)) = \theta_0 + \theta_1 X_1 + \ldots + \theta_p X_p.$$

If logit=FALSE & (reg.q=TRUE)

$$log(-log(q)) = \theta_0 + \theta_1 X_1 + \ldots + \theta_p X_p.$$

initial	Vector of initial values for parameters. In all cases DW(q,B), DW(regQ,B), DW(q,rebB) and DW(regQ,regB) the first parameters correspond to q or corresponding regression coefficients and next is beta or corresponding regression coefficients. If penalized=TRUE then an extra value must be added for tuning parameter.
iteration	Number of MCMC iterations.
v.scale	The scale of the proposal function. Setting to lower values results in an increase in the acceptance rate of the sampler.
RJ	Logical flag. If TRUE, Reversible-Jump sampling is used to draw samples from the posterior. Otherwise, a Metropolis-Hastings sampler applies.
dist.q	Density function. Prior density for theta. If not set, an improper prior applies. Any two parameter density function is allowed, e.g. dnorm, dlaplace, dunif etc. Any customized density function must support log=TRUE flag.
dist.b	Density function. Prior density for gamma. If not set, an improper prior applies. Any two parameter density function is allowed, e.g. dnorm, dlaplace, dunif etc. Any customized density function must support log=TRUE flag.
q.par	A vector of length two corresponding to the parameters of 'dist.q'. The default is set to c(0,0).
b.par	A vector of length two corresponding to the parameters of 'dist.b'. The default is set to c(0,0).
penalized	logical flag. If TRUE, an hyper-parameter inducing shrinkage is considered. In this case, prior must be set to an informative prior, e.g. Gaussian, Laplace. See also 'l.par' and 'dist.l' below.
dist.l	Density function. Hyper prior for penalty term. If not set, an improper prior is used. Any non-negative two parameter density function is allowed, e.g. dgamma, dbeta,... Any customized density function must support log=TRUE flag.
l.par	A vector of length two corresponding to the parameters of the hyper-prior 'dist.l'. The default is set to c(0,0).
bi.period	A numeric value in (0,1) indicating the burn-in period of the MCMC chain. The default is set to 0.25 meaning that 25% of values remove from the beginning of the output chain. See 'sampling'.
cov	A value in {1,2,3,4}. If set to 1 then the adaptive-MH is performed; 2: an independent uniform proposal; 3: an independent Laplace proposal and 4: an independent Gaussian proposal. The default is 1. If cov=1 then the scale of the proposal is adapted from the data.
sampling	Choose between independent (indp), systematic (syst) and burn-in (bin). If set to indp then the chain is not ordered! The default is 'bin'. Sampling interval for Systematic sampling is calculated from iteration(1-bi.period). Similarly for indp the number of samples is computed from iteration(1-bi.period).

est	Statistic. The statistic that is used to estimate the parameters from marginal densities. Possible values are: mode, mean, median or any customized univariate measure of location. The default is 'mode'.
fixed.l	A positive number. Set to a positive value corresponding to a parameter that does not need estimation. Set to any negative value to disable this option.
jeffrey	A logical flag. Set to a TRUE to use Jeffrey prior. Notice that MCMC based on Jeffrey can take considerably long time and the results in the most cases are worse than using an improper flat (=1) prior.
...	

Value

res	Estimation of the parameters from marginal densities using the statistic specified in 'est' .
chain	A list of values including sampler configurations. chain : Including estimation of marginal densities acceptance.rate: Acceptance rate of sampler RejAccChain : A zero-one vector reporting rejection (0) and acceptance (1) of the samples error : Total number of errors during the sampling procedure minf : Minimum loglikelihood among all iterations minState : Coefficients corresponding to minimum loglikelihood lb : The number of gamma parameters lq : The number of theta parameters model.chain : If RJ=TRUE then this chain contains acceptance (+values) and rejection (-values) of models duration : Time elapsed until completion of the sampling procedure

Author(s)

Hamed Haselimashhadi <hamedhaseli@gmail.com>

References

Haselimashhadi, Vinciotti and Yu (2015), A new Bayesian regression model for counts in medicine.

See Also

plot.bdw, summary.bdw, bdw.mc

Examples

```
set.seed(123)
#========== example 1 - estimating DW parameters under logit transformation ==========
q = .41    # <<< true parameters
b = 1.1    # <<< true parameters
y = BDWreg:::rdw(n = 200,q = q,beta = b) #<<< generating data
```

```
result  = bdw(data = y, v.scale = .10,initial = c(.5,.5),iteration = 8000 )
plot(result)
summary(result)


## Not run:
  #==== example 2 - estimating logit-DW(regQ,beta) parameters using RJ ======
  set.seed(1234)
  n = 500
  x1 = runif(n = n, min = 0, max = 1.5)
  x2 = runif(n = n, min = 0, max = 1.5)

  theta0 = .6   #<<< true parameter
  theta1 =  0   #<<< true parameter
  theta2 = .34  #<<< true parameter

  lq = theta0 +  x1*theta1 + x2*theta2

  q = exp(lq - log(1+exp(lq)) )
  beta = 1.5

  y = c()
  for(i in 1:n){
    y[i] = BDWreg:::rdw(1,q = q[i],beta = beta)
  }

  data = data.frame(x1,x2,y) # <<<- data
  result2 = bdw(data = data              ,
                formula = y~.            ,
                RJ = TRUE                ,
                initial = rep(.5,4)      ,
                iteration = 25000        ,
                reg.b = FALSE,reg.q = TRUE,
                v.scale = .1             ,
                q.par = c(0,1)           ,
                b.par = c(0,1)           ,
                dist.q = dnorm           ,
                dist.b = dnorm
  )
  plot(result2)
  summary(result2)

## End(Not run)
```

---

bdw.mc *Producing several chains from a MCMC object of class 'bdw'*

Description

This function is equipped with multicore options to produce several chains from a MCMC of class 'bdw'

Usage

```
bdw.mc(dw.object, n.repeat = 10, cores = 0)
```

Arguments

dw.object	Object of class 'bdw'.
n.repeat	The number of chains to be generated.
cores	The number of processors. If set to zero then the procedure uses all cores.

Value

An object of class 'bdw'. All chains are combined into a list that is stored in an object named 'all'. The output of this function can be passed to plot() and summary().

Author(s)

Hamed Haselimashhadi <hamedhaseli@gmail.com>

See Also

bdw, plot.bdw, summary.bdw

Examples

```
## Not run:
 #==== multicore example - estimating logit-DW(regQ,B) parameters using RJ and 5 chains ======
 #==== Two variables and four coefficients, including intercepts, are simulated and analysed
  set.seed(1234)
  n = 500
  x1 = runif(n = n, min = 0, max = 1.5)
  x2 = runif(n = n, min = 0, max = 1.5)

  theta0 = .6   #<<< true parameter
  theta1 =  0   #<<< true parameter
  theta2 = .34  #<<< true parameter

  lq = theta0 +  x1*theta1 + x2*theta2

  q = exp(lq - log(1+exp(lq)) )
  beta = 1.5

  y = c()
  for(i in 1:n){
    y[i] = BDWreg:::rdw(1,q = q[i],beta = beta)
  }

  data = data.frame(x1,x2,y) # <<<- data
  result = bdw(data = data              ,
               formula = y~.            ,
               RJ = TRUE                ,
               initial = rep(.5,4)      ,
```

```
                iteration = 25000         ,
                reg.b = FALSE,reg.q = TRUE,
                v.scale = .1              ,
                q.par = c(0,1)            ,
                b.par = c(0,1)            ,
                dist.q = dnorm            ,
                dist.b = dnorm
  )
  result2 = bdw.mc(result,5) # <<<- multicore
  plot(result2)
  summary(result2)

## End(Not run)
```

---

plot.bdw *Plot a MCMC object of class 'bdw'*

Description

This function generates several diagnostics plots from a MCMC object of class 'bdw'

Usage

```
## S3 method for class 'bdw'
plot(x, est = Mode, prob = 0.95, adj = 2, r.outliers = TRUE,
density = FALSE, exc.tun = FALSE, ...)
```

Arguments

x	The object containing the MCMC results of class 'bdw'.
est	The statistic that is used to estimate parameters from marginal densities. The default is 'mode'.
prob	A numerical value in (0 , 1). Corresponding probability for Highest Posterior Density (HPD) interval.
adj	A positive value. Measure of smoothness for densities. A higher value results in smoother density plots.
r.outliers	Logical flag. If TRUE, a preprocessing procedure removes the outliers before showing the results.
density	Logical flag. If TRUE, a density plot accompanies the HPD intervals.
exc.tun	Logical flag. If TRUE and penalized=TRUE, the penalty parameter is excluded from the plots.
...	

Author(s)

Hamed Haselimashhadi <hamedhaseli@gmail.com>

See Also

bdw, summary.bdw, bdw.mc

Examples

```
example(bdw)
```

---

summary.bdw *Summary for a MCMC object of class 'bdw'*

Description

This function produces result summaries from a MCMC object of class 'bdw'

Usage

```
## S3 method for class 'bdw'
summary(object, est = Mode, prob = 0.95, samp = TRUE, ...)
```

Arguments

object	The object containing the MCMC results of class 'bdw'.
est	The statistic that is used to estimate parameters from marginal densities. The default is 'mode'.
prob	A numerical value in (0 , 1). Corresponding probability for Highest Posterior Density (HPD) interval. If either RJ=TRUE or penalized=TRUE, coefficients are marked as zero if corresponding prob% HPD intervals contain zero.
samp	Logical flag. If TRUE, analyse a sample instead of whole MCMC chain to save time. * enable if object is created by 'bdw.mc' function.
...	

Author(s)

Hamed Haselimashhadi <hamedhaseli@gmail.com>

See Also

bdw, plot.bdw, bdw.mc

Examples

```
example(bdw)
```

Index

∗Topic **Discrete Weibull regression**
bdw, 2
∗Topic **HPD interval**
plot.bdw, 7
∗Topic **Metropolise-Hasings**
bdw, 2
∗Topic **Reversible-Jumps**
bdw, 2
∗Topic **multicore**
bdw.mc, 5

bdw, 2, *6*, *8*
bdw.mc, *4*, 5, *8*

plot.bdw, *4*, *6*, 7, *8*

summary.bdw, *4*, *6*, *8*, 8