Package ‘BSGW’

September 21, 2016

Type Package

Title Bayesian Survival Model with Lasso Shrinkage Using Generalized Weibull Regression

Version 0.9.2

Date 2016-09-21

Author Alireza S. Mahani, Mansour T.A. Sharabiani

Maintainer Alireza S. Mahani <alireza.s.mahani@gmail.com>

Description Bayesian survival model using Weibull regression on both scale and shape parameters. Dependence of shape parameter on covariates permits deviation from proportional-hazard assumption, leading to dynamic - i.e. non-constant with time - hazard ratios between subjects. Bayesian Lasso shrinkage in the form of two Laplace priors - one for scale and one for shape coefficients - allows for many covariates to be included. Cross-validation helper functions can be used to tune the shrinkage parameters. Monte Carlo Markov Chain (MCMC) sampling using a Gibbs wrapper around Radford Neal’s univariate slice sampler (R package MfUSampler) is used for coefficient estimation.

License GPL (>= 2)

Imports foreach, doParallel, survival, MfUSampler, methods

NeedsCompilation no

Repository CRAN

Date/Publication 2016-09-21 08:06:29

R topics documented:

bsgw ................................................................. 2
bsgw.crossval ................................................... 6
plot.bsgw ........................................................ 8
predict.bsgw ................................................... 9
summary.bsgw .................................................. 11

Index 13
Bayesian Survival using Generalized Weibull Regression

Description

Bayesian survival model - with stratification and shrinkage - using Weibull regression on both scale and shape parameters, resulting in time-dependent (i.e. dynamic) hazard ratios.

Usage

bsgw(formula, data, formulas=formula, weights, subset, na.action=na.fail, init="survreg" , ordweib=FALSE, scale=0, control=bsgw.control(), print.level=2)
bsgw.control(scale=TRUE, iter=1000, burnin=round(iter/2), sd.thresh=1e-4
, lambda=0.0, lambdas=lambda, nskip=round(iter/10), alpha.min=0.1, alpha.max=10.0
, beta.max=log(20), betas.max=5.0, memlim.gb=8)
## S3 method for class 'bsgw'
print(x,...)

Arguments

- **formula**: Survival formula expressing the time/status variables as well as covariates used in regression on scale parameter. Currently, only right and left censoring is supported. Must include intercept term.
- **data**: Data frame containing the covariates and response variable.
- **formulas**: Formula expressing the covariates used in regression on shape parameter. No left-hand side is necessary since the response variable information is extracted from formula. Default value is formula. Must include intercept term.
- **weights**: Optional vector of case weights. *Not supported yet*
- **subset**: Subset of the observations to be used in the fit. *Not supported yet*
- **na.action**: Missing-data filter function. *Not supported yet (only na.fail behavior works)*
- **init**: Initialization behavior. Currently, three options are supported: 1) If init="survreg", an ordinary Weibull regression is performed and coefficients are used to initialize the bsgw MCMC run. 2) If init is a survreg object, e.g. from a previous Weibull regression fit, the object can be directly passed as parameter. 3) If init is any other value, or if survreg produces error or warning, we simply set all coefficients to zero.
- **ordweib**: If TRUE, a Bayesian ordinary Weibull model is estimated, in which any covariates in formulas are stripped away, and the inverse-logit transformation in the shape-parameter regression is replaced with a simple exponential transformation. If shrinkage parameters are kept at 0, the result is a Bayesian equivalent of an ordinary Weibull regression.
- **scale**: If scale>0, the value of the shape parameter is fixed, i.e. not estimated from data.
- **control**: See bsgw.control for a description of the parameters inside the control list.
print.level Controlling verbosity level.
scalex If TRUE, each covariate vector is centered and scaled before model estimation. The scaling parameters are saved in return object, and used in subsequent calls to predict function. Users are strongly advised against turning this feature off, since the quality of Gibbs sampling MCMC is greatly enhanced by covariate centering and scaling.
iter Number of MCMC samples to draw.
burnin Number of initial MCMC samples to discard before calculating summary statistics.
.sd.thresh Threshold for standard deviation of a covariate (after possible centering/scaling). If below the threshold, the corresponding coefficient is removed from sampling, i.e. its value is clamped to zero.
lambda Bayesian Lasso shrinkage parameter for scale-parameter coefficients.
lambdas Bayesian Lasso shrinkage parameter for shape-parameter coefficients.
nskip Controlling how often to print progress report during MCMC run. For example, if nskip=10, progress will be reported after 10, 20, 30,... samples.
alpha.min Lower bound on the shape parameter.
alpha.max Upper bound on the shape parameter.
beta.max Upper bound on absolute value of coefficients of scale parameter (with the exception of the intercept).
betas.max Upper bound on absolute value of coefficients of shape parameter (with the exception of the intercept).
memlim.gb User-specified limit on total memory (in GB) available during prediction. Hazard, cumulative hazard, and survival prediction objects are all three-dimensional arrays which can quickly grow very large, depending on data length, number of MCMC samples collected, and number of time points along which prediction is made.
x Object of class `bsgw`, usually the result of a call to the bsgw.

Value
The function bsgw.control returns a list with elements identical to the input parameters. The function bsgw returns an object of class bsgw, with the following components:
call The matched call.
formula Same as input.
formulas Same as input.
weights Same as input. *Not supported yet*
subset Same as input. *Not supported yet*
aa.action Same as input. *Not supported yet* (current behavior is na.fail)
init Initial values for scale and shape coefficients used in MCMC sampling, either by performing an ordinary Weibull regression or by extracting estimated coefficients from a previously-performed such regression.
4

ordweib  Same as input.
survreg.scale.ref

Value of scale parameter, estimated using ordinary Weibull regression by calling the `survreg` function in the `survival` package.

ordreg  The "survreg" object returned from calling the same function for initialization of coefficients.
scale  Same as input.
control  Same as input.
X  Model matrix used for regression on scale parameter, after potential centering and scaling. The corresponding vector of coefficients is called `beta`.
Xs  Model matrix used for regression on shape parameter, after potential centering and scaling. The corresponding vector of coefficients is called `betas`.
y  Survival response variable (time and status) used in the model.
contrasts  The contrasts used for scale-parameter coefficients.
contrastss  The contrasts used for shape-parameter coefficients.
xlevels  A record of the levels of the factors used in fitting for scale parameter regression.
xlevelss  A record of the levels of the factors used in fitting for shape parameter regression.
terms  The terms object used for scale parameter regression.
terms  The terms object used for shape parameter regression.
colnamesX  Names of columns for X, also names of scale coefficients.
colnamesXs  Names of columns for Xs, also names of shape coefficients.
apply.scale.X  Index of columns of X where scaling has been applied.
apply.scale.Xs  Index of columns of Xs where scaling has been applied.
centerVec.X  Vector of centering parameters for columns of X indicated by `apply.scale.X`.
scaleVec.X  Vector of scaling parameters for columns of X indicated by `apply.scale.X`.
centerVec.Xs  Vector of centering parameters for columns of Xs indicated by `apply.scale.Xs`.
scaleVec.Xs  Vector of scaling parameters for columns of Xs indicated by `apply.scale.Xs`.
idx  Vector of indexes into X for which sampling occurred. All columns of X whose standard deviation falls below `sd.thresh` are excluded from sampling and their corresponding coefficients are clamped to 0.
idxs  Vector of indexes into Xs for which sampling occurred. All columns of Xs whose standard deviation falls below `sd.thresh` are excluded from sampling and their corresponding coefficients are clamped to 0.
median  List of median values, with elements including `beta` (coefficients of scale regression), `betas` (coefficients of shape regression), `survreg.scale` (value of `survreg`-style scale parameter for all training set observations).
smp  List of coefficient samples, with the following elements: 1) `beta` (scale parameter coefficients), 2) `betas` (shape parameter coefficients), 3) `lp` (vector of linear predictor for scale parameter, within-sample), 4) `loglike` (log-likelihood of
bsgw

model), 5) logpost (log-posterior of mode, i.e. log-likelihood plus the shrinkage term). The last two entities are used during within-sample prediction of response, i.e. during a subsequent call to predict. Each parameter has control$iter samples.

km.fit Kaplan-Meyer fit to training data. Used in plot.bsgw method.

tmax Maximum time value in training set. Used in predict.bsgw for automatic selection of the tvec parameter.

Author(s)

Alireza S. Mahani, Mansour T.A. Sharabiani

References


See Also

For calculating median and lower/upper bounds on coefficients, use summary.bsgw.

For prediction, use predict.bsgw.

Examples

```r
## model estimation using 800 samples, printing progress every 100 samples
library("survival")
data(ovarian)
est <- bsgw(Surv(futime, fustat) ~ ecog.ps + rx, ovarian
, control=bsgw.control(iter=400, nskip=100))

## comparing shape of Weibull curves between ordinary Weibull and bsgw
## since in bsgw shape is dependent on covariates, only a population average is meaningful
## Note that survreg-style scale is inverse of bsgw shape parameter, see survreg help page
west <- survreg(Surv(futime, fustat) ~ ecog.ps + rx, ovarian)
cat("constant survreg-style scale parameter: ", west$scale, 
"n")
cat("population average of survreg-style scale parameter from bsgw model: ", mean(est$median$survreg.scale), 
"n")
```
bsgw.crossval

Convenience functions for cross-validation-based selection of shrinkage parameter in the bsgw model.

Description

bsgw.crossval calculates cross-validation-based, out-of-sample log-likelihood of a bsgw model for a data set, given the supplied folds. bsgw.crossval.wrapper applies bsgw.crossval to a set of combinations of shrinkage parameters (lambda, lambdas) and produces the resulting vector of log-likelihood values as well as the specific combination of shrinkage parameters associated with the maximum log-likelihood. bsgw.generate.folds generates random partitions, while bsgw.generate.folds.eventbalanced generates random partitions with events evenly distributed across partitions. The latter feature is useful for cross-validation of small data sets with low event rates, since it prevents over-accumulation of events in one or two partitions, and lack of events altogether in other partitions.

Usage

bsgw.generate.folds(ntot, nfold=5)
bsgw.generate.folds.eventbalanced(formula, data, nfold=5)
bsgw.crossval(data, folds, all=FALSE, print.level=1
, control=bsgw.control(), ncores=1, ...)
bsgw.crossval.wrapper(data, folds, all=FALSE, print.level=1
, control=bsgw.control(), ncores=1
, lambda.vec=exp(seq(from=log(0.01), to=log(100), length.out = 10)), lambdas.vec=NULL
, lambda2=if (is.null(lambdas.vec)) cbind(lambda=lambda.vec, lambdas=lambda.vec)
  else as.matrix(expand.grid(lambda=lambda.vec, lambdas=lambdas.vec))
, plot=TRUE, ...)

Arguments

ntot Number of observations to create partitions for. It must typically be set to nrow(data).
nfold Number of folds or partitions to generate.
formula Survival formula, used to extract the binary status field from the data. Right-hand side of the formula is ignored, so a formula of the form Surv(time, status)~1 is sufficient.
data Data frame used in model training and prediction.
folds An integer vector of length nrow(data), defining fold/partition membership of each observation. For example, in 5-fold cross-validation for a data set of 200 observations, folds must be a 200-long vector with elements from the set \{1,2,3,4,5\}. Convenience functions bsgw.generate.folds and bsgw.generate.folds.eventbalanced can be used to generate the folds vector for a given survival data frame.
all If TRUE, estimation objects from each cross-validation task is collected and returned for diagnostics purposes.
bsgw.crossval

print.level Verbosity of progress report.
control List of control parameters, usually the output of bsgw.control.
ncores Number of cores for parallel execution of cross-validation code.
lambda.vec Vector of shrinkage parameters to be tested for scale-parameter coefficients.
lambdas.vec Vector of shrinkage parameters to be tested for shape-parameter coefficients.
lambda2 A data frame that enumerates all combinations of lambda and lambdas to be tested. By default, it is constructed from forming all permutations of lambda.vec and lambdas.vec. If lambdas.vec=NULL, it will only try equal values of the two parameters in each combination.
plot If TRUE, and if the lambda and lambdas entries in lambda2 are identical, a plot of loglike as a function of either vector is produced.
... Other arguments to be passed to bsgw.

Value

Functions bsgw.generate.folds and bsgw.generate.folds.eventbalanced produce integer vectors of length ntot or nrow(data) respectively. The output of these functions can be directly passed to bsgw.crossval or bsgw.crossval.wrapper. Function bsgw.crossval returns the log-likelihood of data under the assumed bsgw model, calculated using a cross-validation scheme with the supplied fold parameter. If all=TRUE, the estimation objects for each of the nfold estimation jobs will be returned as the "estobjs" attribute of the returned value. Function bsgw.crossval.wrapper returns a list with elements lambda and lambdas, the optimal shrinkage parameters for scale and shape coefficients, respectively. Additionally, the following attributes are attached:

loglike.vec Vector of log-likelihood values, one for each tested combination of lambda and lambdas.
loglike.opt The maximum log-likelihood value from the loglike.vec.
lambda2 Data frame with columns lambda and lambdas. Each row of this data frame contains one combination of shrinkage parameters that are tested in the wrapper function.
estobjs If all=TRUE, a list of length nrow(lambda2) is returned, with each element being itself a list of nfold estimation objects associated with each call to the bsgw function. This object can be examined by the user for diagnostic purposes, e.g. by applying plot against each object.

Author(s)

Alireza S. Mahani, Mansour T.A. Sharabiani

Examples

library("survival")
data(ovarian)
folds <- bsgw.generate.folds.eventbalanced(Surv(futime, fustat) ~ 1, ovarian, 5)
cv <- bsgw.crossval(ovarian, folds, formula=Surv(futime, fustat) ~ ecog.ps + rx
 , control=bsgw.control(iter=50, nskip=10), print.level = 3)
cv2 <- bsgw.crossval.wrapper(ovarian, folds, formula=Surv(futime, fustat) ~ ecog.ps + rx, control=bsgw.control(iter=50, nskip=10), print.level=3, lambda.vec=exp(seq(from=log(0.1), to=log(1), length.out = 3)))

plot.bsgw

Plot diagnostics for a bsgw object

Description

Four sets of MCMC diagnostic plots are currently generated: 1) log-likelihood and log-posterior (including shrinkage effect) as a function of iteration number, 2) coefficient trace plots, 3) coefficient autocorrelation plots, 4) coefficient histograms.

Usage

## S3 method for class 'bsgw'
plot(x, pval=0.05, burnin=round(x$control$iter/2), nrow=2, ncol=3, ...)

Arguments

x A bsgw object, typically the output of bsgw function.
pval The P-value at which lower/upper bounds on coefficients are calculated and overlaid on trace plots and histograms.
burnin Number of samples discarded from the beginning of an MCMC chain, after which parameter quantiles are calculated.
nrow Number of rows of subplots within each figure, applied to plot sets 2-4.
col Number of columns of subplots within each figure, applied to plot sets 2-4.
... Further arguments to be passed to/from other methods.

Author(s)

Alireza S. Mahani, Mansour T.A. Sharabiani

Examples

library("survival")
data(ovarian)
est <- bsgw(Surv(futime, fustat) ~ ecog.ps + rx, ovarian, control=bsgw.control(iter=400, nskip=100))
plot(est)
predict.bsgw

Predict method for bsgw model fits

Description

Calculates log-likelihood and hazard/cumulative hazard/survival functions over a user-supplied vector time values, based on BSGW model object.

Usage

## S3 method for class 'bsgw'
predict(object, newdata=NULL, tvec=NULL, burnin=object$control$burnin, ncores=1, ...)
## S3 method for class 'predict.bsgw'
summary(object, idx=1:length(object$median$survreg.scale), burnin=object$burnin, pval=0.05, popmean=identical(idx,1:length(object$median$survreg.scale)), make.plot=TRUE, ...)

Arguments

- **object**: For `predict.bsgw`, an object of class "bsgw", usually the result of a call to `bsgw`; for `summary.predict.bsgw`, an object of class "predict.bsgw", usually the result of a call to `predict.bsgw`.
- **newdata**: An optional data frame in which to look for variables with which to predict. If omitted, the fitted values (training set) are used.
- **tvec**: An optional vector of time values, along which time-dependent entities (hazard, cumulative hazard, survival) will be predicted. If omitted, only the time-independent entities (currently only log-likelihood) will be calculated. If a single integer is provided for `tvec`, it is interpreted as number of time points, equally spaced from 0 to `object$tmax`: `tvec <- seq(from=0, to=object$tmax, length.out=tvec)`.
- **burnin**: Number of samples to discard from the beginning of each MCMC chain before calculating median value(s) for time-independent entities.
- **ncores**: Number of cores to use for parallel prediction.
- **idx**: Index of observations (rows of `newdata` or training data) for which to generate summary statistics. Default is the entire data.
- **pval**: Desired p-value, based on which lower/upper bounds will be calculated. Default is 0.05.
- **popmean**: Whether population averages must be calculated or not. By default, population averages are only calculated when the entire data is included in prediction.
- **make.plot**: Whether population mean and other plots must be created or not.
Details

The time-dependent predicted objects (except loglike) are three-dimensional arrays of size \((nsmp \times nt \times nobs)\), where \(nsmp\) = number of MCMC samples, \(nt\) = number of time values in \(tvec\), and \(nobs\) = number of rows in \(newdata\). Therefore, even for modest data sizes, these objects can occupy large chunks of memory. For example, for \(nsmp\]1000, \(nt\]100, \(nobs\]1000, the three objects \(h\), \(H\), \(S\) have a total size of 2.2GB. Since applying quantile to these arrays is time-consuming (as needed for calculation of median and lower/upper bounds), we have left such summaries out of the scope of \(predict\) function. Users can instead apply \(summary\) to the prediction object to obtain summary statistics. During cross-validation-based selection of shrinkage parameter \(lambda\), there is no need to supply \(tvec\) since we only the log-likelihood value. This significantly speeds up the parameter-tuning process. The function \(summary.predict.bsgw\) allows the user to calculates summary statistics for a subset (or all of) data, if desired. This approach is in line with the overall philosophy of delaying the data summarization until necessary, to avoid unnecessary loss in accuracy due to premature blending of information contained in individual samples.

Value

The function \(predict.bsgw\) returns as object of class "predict.bsgw" with the following fields:

- \(tvec\) Actual vector of time values (if any) used for prediction.
- \(burnin\) Same as input.
- \(median\) List of median values for predicted entities. Currently, only loglike and survreg.scale median is produced. See 'Details' for explanation.
- \(smp\) List of MCMC samples for predicted entities. Elements include \(h\) (hazard function), \(H\) (cumulative hazard function), \(S\) (survival function), survreg.scale (inverse of shape parameter in \(rweibull\)), and loglike (model log-likelihood). All functions are evaluated over time values specified in \(tvec\).
- \(km.fit\) Kaplan-Meyer fit of the data used for prediction (if data contains response fields).

The function \(summary.predict.bsgw\) returns a list with the following fields:

- \(lower\) A list of lower-bound values for \(h\), \(H\), \(S\), \(hr\) (hazard ratio of \(idx[2]\) to \(idx[1]\) observation), and \(S.dif\) (survival probability of \(idx[2]\) minus \(idx[1]\)). The last two are only included if \(length(idx)==2\).
- \(median\) List of median values for same entities described in \(lower\).
- \(upper\) List of upper-bound values for same entities described in \(lower\).
- \(popmean\) Lower-bound/median/upper-bound values for population average of survival probability.
- \(km.fit\) Kaplan-Meyer fit associated with the prediction object (if available).

Author(s)

Alireza S. Mahani, Mansour T.A. Sharabiani
Examples

```r
library("survival")
data(ovarian)
est <- bsgw(Surv(futime, fustat) ~ ecog.ps + rx, ovarian
  , control=bsgw.control(iter=400, nskip=100))
pred <- predict(est, tvec=100)
predsumm <- summary(pred, idx=1:10)
```

summary.bsgw

| Summary | Summarizing Bayesian Survival Generalized Weibull (BSGW) model fits |

Description

summary method for class "bsgw".

Usage

```r
## S3 method for class 'bsgw'
summary(object, pval = 0.05, burnin = object$control$burnin, ...)
## S3 method for class 'summary.bsgw'
print(x, ...)
```

Arguments

- **object**: An object of class "bsgw", usually the result of a call to bsgw.
- **x**: An object of class "summary.bsgw", usually the result of a call to summary.bsgw.
- **pval**: Desired p-value, based on which lower/upper bounds will be calculated. Default is 0.05.
- **burnin**: Number of samples to discard from the beginning of each MCMC chain before calculating median and lower/upper bounds.
- ... Further arguments to be passed to/from other methods.

Value

The function summary.bsgw calculates median as well as lower/upper bounds for all model coefficients, given the supplied p-value. It also calculates the p-value for coefficients being significant smaller/larger than zero. It contains returns an object of class "summary.bsgw" with the following elements:

- **call**: The matched call.
- **pval**: Same as input.
- **burnin**: Same as input.
- **coefficients**: A p x 4 matrix with columns for the estimated coefficient median, its lower and upper bounds given the user-supplied p-value, and the p-value for being smaller/larger than zero.
- **survreg.scale**: List of lower, median, and upper values of the survreg-style scale parameter (i.e. inverse of shape parameter in rweibull) for the training-set population.
Author(s)

Alireza S. Mahani, Mansour T.A. Sharabiani

See Also

See summary for a description of the generic method.

The model fitting function is bsgw.

Examples

```r
library("survival")
data(ovarian)
est <- bsgw(Surv(futime, fustat) ~ ecog.ps + rx, ovarian
    , control=bsgw.control(iter=400, nskip=100))
summary(est, pval=0.1)
```
Index

BSGW (bsgw), 2
bsgw, 2, 7–9, 11, 12
bsgw.control, 7
bsgw.crossval, 6
bsgw.generate.folds (bsgw.crossval), 6
plot.bsgw, 5, 8
predict.bsgw, 5, 9
print.bsgw (bsgw), 2
print.summary.bsgw (summary.bsgw), 11

summary, 12
summary.bsgw, 5, 11
summary.predict.bsgw (predict.bsgw), 9