Package 'SemiPar.depCens'

August 27, 2024

Type Package

Title Copula Based Cox Proportional Hazards Models for Dependent Censoring

Version 0.1.3

Date 2024-08-26

Maintainer Negera Wakgari Deresa <negera.deresa@gmail.com>

Description Copula based Cox proportional hazards models for survival data subject to dependent censoring. This approach does not assume that the parameter defining the copula is known. The dependency parameter is estimated with other finite model parameters by maximizing a Pseudo likelihood function. The cumulative hazard function is estimated via estimating equations derived based on martingale ideas. Available copula functions include Frank, Gumbel and Normal copulas. Only Weibull and lognormal models are allowed for the censoring model, even though any
parametric model that satisfies certain identifiability conditions could be used. Implemented methods are described in the article ``Copula based Cox proportional hazards models for dependent
censoring" by Deresa and Van Keilegom (2024) <doi:10.1080/01621459.2022.2161387>.

Imports survival, copula, stats, foreach, parallel, doParallel, pbivnorm

License GPL-3

Depends R (>= 3.5.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/Nago2020/SemiPar.depCens

BugReports https://github.com/Nago2020/SemiPar.depCens/issues

NeedsCompilation no

Author Negera Wakgari Deresa [aut, cre]
(<https://orcid.org/0000-0002-1302-3725>),
Ingrid Van Keilegom [aut] (<https://orcid.org/0000-0001-8827-7642>)

Repository CRAN

Date/Publication 2024-08-26 23:30:08 UTC

Contents

`boot.fun` *Nonparametric bootstrap approach for the dependent censoring model*

Description

This function estimates the bootstrap standard errors for the finite-dimensional model parameters and for the non-parametric cumulative hazard function. Parallel computing using foreach has been used to speed up the estimation of standard errors.

Usage

```
boot.fun(
  init,
  resData,
  X,
  W,
  lhat,
  cumL,
  dist,
```

```
  k,
  lb,
  ub,
  Obs.time,
  cop,
  n.boot,
  n.iter,
  ncore,
  eps
)
```

Arguments

`init`	Initial values for the finite dimensional parameters obtained from the fit of `fitDepCens`
`resData`	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
`X`	Data matrix with covariates related to T
`W`	Data matrix with covariates related to C. First column of W should be a vector of ones
`lhat`	Initial values for the hazard function obtained from the fit of `fitDepCens` based on the original data.
`cumL`	Initial values for the cumulative hazard function obtained from the fit of `fitDepCens` based on the original data.
`dist`	The distribution to be used for the dependent censoring time C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value `"Weibull"` as the default.
`k`	Dimension of X
`lb`	lower boundary for finite dimensional parameters
`ub`	Upper boundary for finite dimensional parameters
`Obs.time`	Observed survival time, which is the minimum of T, C and A, where A is the administrative censoring time.
`cop`	Which copula should be computed to account for dependency between T and C. This argument can take one of the values from `c("Gumbel", "Frank", "Normal")`.
`n.boot`	Number of bootstraps to use in the estimation of bootstrap standard errors.
`n.iter`	Number of iterations; the default is `n.iter = 20`. The larger the number of iterations, the longer the computational time.
`ncore`	The number of cores to use for parallel computation is configurable, with the default `ncore = 7`.
`eps`	Convergence error. This is set by the user in such away that the desired convergence is met; the default is `eps = 1e-3`

Value

Bootstrap standard errors for parameter estimates and for estimated cumulative hazard function.

`boot.funI` *Nonparametric bootstrap approach for the independent censoring model*

Description

This function estimates the bootstrap standard errors for the finite-dimensional model parameters and for the non-parametric cumulative hazard function under the assumption of independent censoring. Parallel computing using foreach has been used to speed up the computation.

Usage

```
boot.funI(
  init,
  resData,
  X,
  W,
  lhat,
  cumL,
  dist,
  k,
  lb,
  ub,
  Obs.time,
  n.boot,
  n.iter,
  ncore,
  eps
)
```

Arguments

`init`	Initial values for the finite dimensional parameters obtained from the fit of `fitIndepCens`
`resData`	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
`X`	Data matrix with covariates related to T
`W`	Data matrix with covariates related to C. First column of W should be a vector of ones
`lhat`	Initial values for the hazard function obtained from the fit of `fitIndepCens` based on the original data
`cumL`	Initial values for the cumulative hazard function obtained from the fit of `fitIndepCens` based on the original data
`dist`	The distribution to be used for the dependent censoring time C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value `"Weibull"` as the default

k	Dimension of X
lb	lower boundary for finite dimensional parameters
ub	Upper boundary for finite dimensional parameters
Obs.time	Observed survival time, which is the minimum of T, C and A, where A is the administrative censoring time.
n.boot	Number of bootstraps to use in the estimation of bootstrap standard errors.
n.iter	Number of iterations; the default is n.iter = 20. The larger the number of iterations, the longer the computational time
ncore	The number of cores to use for parallel computation is configurable, with the default ncore = 7.
eps	Convergence error. This is set by the user in such away that the desired convergence is met; the default is eps = 1e-3

Value

Bootstrap standard errors for parameter estimates and for estimated cumulative hazard function.

CompC *Compute phi function*

Description

This function estimates phi function at fixed time point t

Usage

```
CompC(theta, t, X, W, ld, cop, dist)
```

Arguments

theta	Estimated parameter values/initial values for finite dimensional parameters
t	A fixed time point
X	Data matrix with covariates related to T
W	Data matrix with covariates related to C. First column of W should be ones
ld	Output of SolveL function at a fixed time t
cop	Which copula should be computed to account for dependency between T and C. This argument can take one of the values from c("Gumbel", "Frank", "Normal"). The default copula model is "Frank".
dist	The distribution to be used for the dependent censoring C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value "Weibull" as the default.

Distance *Distance between vectors*

Description

This function computes distance between two vectors based on L2-norm

Usage

```
Distance(a, b)
```

Arguments

a	First vector
b	Second vector

fitDepCens *Fit Dependent Censoring Models*

Description

This function allows to estimate the dependency parameter along all other model parameters. First, estimates the cumulative hazard function, and then at the second stage it estimates other model parameters assuming that the cumulative hazard function is known. The details for implementing the dependent censoring methodology can be found in Deresa and Van Keilegom (2024).

Usage

```
fitDepCens(
  resData,
  X,
  W,
  cop = c("Frank", "Gumbel", "Normal"),
  dist = c("Weibull", "lognormal"),
  start = NULL,
  n.iter = 50,
  bootstrap = TRUE,
  n.boot = 150,
  ncore = 7,
  eps = 1e-04
)
```

Arguments

resData	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
X	Data matrix with covariates related to T.
W	Data matrix with covariates related to C. First column of W should be a vector of ones.
cop	Which copula should be computed to account for dependency between T and C. This argument can take one of the values from c("Gumbel", "Frank", "Normal").
dist	The distribution to be used for the censoring time C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value "Weibull" as the default.
start	Initial values for the finite dimensional parameters. If start is NULL, the initial values will be obtained by fitting a Cox model for survival time T and a Weibull model for dependent censoring C.
n.iter	Number of iterations; the default is n.iter = 50. The larger the number of iterations, the longer the computational time.
bootstrap	A boolean indicating whether to compute bootstrap standard errors for making inferences.
n.boot	Number of bootstrap samples to use in the estimation of bootstrap standard errors if bootstrap = TRUE. The default is n.boot = 150. But, higher values of n.boot are recommended for obtaining good estimates of bootstrap standard errors.
ncore	The number of cores to use for parallel computation in bootstrapping, with the default ncore = 7.
eps	Convergence error. This is set by the user in such away that the desired convergence is met; the default is eps = 1e-4.

Value

This function returns a fit of dependent censoring model; parameter estimates, estimate of the cumulative hazard function, bootstrap standard errors for finite-dimensional parameters, the nonparametric cumulative hazard function, etc.

References

Deresa and Van Keilegom (2024). Copula based Cox proportional hazards models for dependent censoring, Journal of the American Statistical Association, 119:1044-1054.

Examples

```
# Toy data example to illustrate implementation
n = 300
beta = c(0.5)
lambd = 0.35
eta = c(0.9,0.4)
X = cbind(rbinom(n,1,0.5))
```

```
W = cbind(rep(1,n),rbinom(n,1,0.5))
# generate dependency structure from Frank
frank.cop <- copula::frankCopula(param = 5,dim = 2)
U = copula::rCopula(n,frank.cop)
T1 = (-log(1-U[,1]))/(lambd*exp(X*beta))           # Survival time
T2 = (-log(1-U[,2]))^(1.1)*exp(W%*%eta)            # Censoring time
A = runif(n,0,15)                                  # administrative censoring time
Z = pmin(T1,T2,A)
d1 = as.numeric(Z==T1)
d2 = as.numeric(Z==T2)
resData = data.frame("Z" = Z,"d1" = d1, "d2" = d2)   # should be data frame
colnames(W) <- c("ones","cov1")
colnames(X) <- "cov.surv"

# Fit dependent censoring model

fit <- fitDepCens(resData = resData, X = X, W = W, bootstrap = FALSE)

# parameter estimates

fit$parameterEstimates

# summary fit results
summary(fit)

# plot cumulative hazard vs time

plot(fit$observedTime, fit$cumhazardFunction, type = "l",xlab = "Time",
ylab = "Estimated cumulative hazard function")
```

`fitIndepCens` *Fit Independent Censoring Models*

Description

This function allows to estimate all model parameters under the assumption of independent censoring. First, estimates the cumulative hazard function, and then at the second stage it estimates other model parameters assuming that the cumulative hazard is known.

Usage

```
fitIndepCens(
  resData,
  X,
  W,
  dist = c("Weibull", "lognormal"),
  start = NULL,
```

```
  n.iter = 50,
  bootstrap = TRUE,
  n.boot = 150,
  ncore = 7,
  eps = 1e-04
)
```

Arguments

resData	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
X	Data matrix with covariates related to T.
W	Data matrix with covariates related to C. First column of W should be ones.
dist	The distribution to be used for the censoring time C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value "Weibull" as the default.
start	Initial values for the finite dimensional parameters. If start is NULL, the initial values will be obtained by fitting a Cox model for survival time T and a Weibull model for censoring time C.
n.iter	Number of iterations; the default is n.iter = 50. The larger the number of iterations, the longer the computational time.
bootstrap	A boolean indicating whether to compute bootstrap standard errors for making inferences.
n.boot	Number of bootstrap samples to use in the estimation of bootstrap standard errors if bootstrap = TRUE. The default is n.boot = 150. But, higher values of n.boot are recommended for obtaining good estimates of bootstrap standard errors.
ncore	The number of cores to use for parallel computation is configurable, with the default ncore = 7.
eps	Convergence error. This is set by the user in such away that the desired convergence is met; the default is eps = 1e-4.

Value

This function returns a fit of independent censoring model; parameter estimates, estimate of the cumulative hazard function, bootstrap standard errors for finite-dimensional parameters, the nonparametric cumulative hazard function, etc.

Examples

```
# Toy data example to illustrate implementation
n = 300
beta = c(0.5)
lambd = 0.35
eta = c(0.9,0.4)
X = cbind(rbinom(n,1,0.5))
```

```
W = cbind(rep(1,n),rbinom(n,1,0.5))
# generate dependency structure from Frank
frank.cop <- copula::frankCopula(param = 5,dim = 2)
U = copula::rCopula(n,frank.cop)
T1 = (-log(1-U[,1]))/(lambd*exp(X*beta))                  # Survival time'
T2 = (-log(1-U[,2]))^(1.1)*exp(W%*%eta)                   # Censoring time
A = runif(n,0,15)                                        # administrative censoring time
Z = pmin(T1,T2,A)
d1 = as.numeric(Z==T1)
d2 = as.numeric(Z==T2)
resData = data.frame("Z" = Z,"d1" = d1, "d2" = d2)     # should be data frame

colnames(W) <- c("ones","cov1")
colnames(X) <- "cov.surv"

# Fit independent censoring model

fitI <- fitIndepCens(resData = resData, X = X, W = W, bootstrap = FALSE)

# parameter estimates

fitI$parameterEstimates

# summary fit results
summary(fitI)

# plot cumulative hazard vs time

 plot(fitI$observedTime, fitI$cumhazardFunction, type = "l",xlab = "Time",
 ylab = "Estimated cumulative hazard function")
```

---

follic *Follicular Cell Lymphoma*

Description

Competing risk data set involving follicular cell lymphoma as given in the book of Pintilie (2006). The data consist of 541 patients with early disease stage (I or II) and treated with radiation alone (RT) or with radiation and chemotherapy (CMT). The endpoints of interest are what comes first: relapse of the disease or death in remission.

Format

A data frame with 541 rows and 7 variables:

- age: age
- clinstg: clinical stage: 1=stage I, 2=stage II

- ch: chemotherapy
- rt: radiotherapy
- hgb: hemoglobin level in g/l
- time: first failure time
- status: censoring status; 0=censored, 1=relapse, 2=death

References

Pintilie, M. (2006), Competing Risks: A Practical Perspective, Chichester: Wiley.

`genData` *Simulate a toy example for testing*

Description

Simulate a toy example for testing

Usage

```
genData(n)
```

Arguments

`n`	sample size

`LikCopInd` *Loglikehood function under independent censoring*

Description

Loglikehood function under independent censoring

Usage

```
LikCopInd(theta, resData, X, W, lhat, cumL, dist)
```

Arguments

theta	Estimated parameter values/initial values for finite dimensional parameters
resData	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
X	Data matrix with covariates related to T
W	Data matrix with covariates related to C. First column of W should be ones
lhat	The estimated hazard function obtained from the output of SolveLI.
cumL	The estimated cumulative hazard function from the output of SolveLI.
dist	The distribution to be used for the dependent censoring C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value "Weibull" as the default. @importFrom stats nlminb pnorm qnorm sd

Value

Maximized log-likelihood value

Longfun *Long format*

Description

Change hazard and cumulative hazard to long format

Usage

```
Longfun(Z, T1, lhat, Lhat)
```

Arguments

Z	Observed survival time, which is the minimum of T, C and A, where A is the administrative censoring time.
T1	Distinct observed survival time
lhat	Hazard function estimate
Lhat	Cumulative hazard function estimate

PseudoL *Likelihood function under dependent censoring*

Description

The PseudoL function is maximized in order to estimate the finite dimensional model parameters, including the dependency parameter. This function assumes that the cumulative hazard function is known.

Usage

```
PseudoL(theta, resData, X, W, lhat, cumL, cop, dist)
```

Arguments

theta	Estimated parameter values/initial values for finite dimensional parameters
resData	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
X	Data matrix with covariates related to T
W	Data matrix with covariates related to C. First column of W should be ones
lhat	The estimated hazard function obtained from the output of SolveL.
cumL	The estimated cumulative hazard function from the output of SolveL.
cop	Which copula should be computed to account for dependency between T and C. This argument can take one of the values from c("Gumbel", "Frank", "Normal"). The default copula model is "Frank".
dist	The distribution to be used for the dependent censoring C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value "Weibull" as the default.

Value

maximized log-likelihood value

SearchIndicate *Search function*

Description

Function to indicate position of t in observed survival time

Usage

```
SearchIndicate(t, T1)
```

Arguments

t	fixed time t
T1	distinct observed survival time

SolveL *Cumulative hazard function of survival time under dependent censoring*

Description

This function estimates the cumulative hazard function of survival time (T) under dependent censoring (C). The estimation makes use of the estimating equations derived based on martingale ideas.

Usage

```
SolveL(
  theta,
  resData,
  X,
  W,
  cop = c("Frank", "Gumbel", "Normal"),
  dist = c("Weibull", "lognormal")
)
```

Arguments

theta	Estimated parameter values/initial values for finite dimensional parameters
resData	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
X	Data matrix with covariates related to T
W	Data matrix with covariates related to C. First column of W should be ones
cop	Which copula should be computed to account for dependency between T and C. This argument can take one of the values from `c("Gumbel", "Frank", "Normal")`. The default copula model is "Frank".
dist	The distribution to be used for the dependent censoring C. Only two distributions are allowed, i.e, Weibull and lognormal distributions. With the value `"Weibull"` as the default.

Value

This function returns an estimated hazard function, cumulative hazard function and distinct observed survival times;

Examples

```
n = 200
beta = c(0.5)
lambd = 0.35
eta = c(0.9,0.4)
X = cbind(rbinom(n,1,0.5))
W = cbind(rep(1,n),rbinom(n,1,0.5))
frank.cop <- copula::frankCopula(param = 5,dim = 2)
U = copula::rCopula(n,frank.cop)
T1 = (-log(1-U[,1]))/(lambd*exp(X*beta))         # Survival time'
T2 = (-log(1-U[,2]))^(1.1)*exp(W%*%eta)          # Censoring time
A = runif(n,0,15)                                # administrative censoring time
Z = pmin(T1,T2,A)
d1 = as.numeric(Z==T1)
d2 = as.numeric(Z==T2)
resData = data.frame("Z" = Z,"d1" = d1, "d2" = d2)
theta = c(0.3,1,0.3,1,2)

# Estimate cumulative hazard function
cumFit <- SolveL(theta, resData,X,W)
cumhaz = cumFit$cumhaz
time = cumFit$times

# plot hazard vs time

plot(time, cumhaz, type = "l",xlab = "Time",
ylab = "Estimated cumulative hazard function")
```

`SolveLI` *Cumulative hazard function of survival time under independent censoring*

Description

This function estimates the cumulative hazard function of survival time (T) under the assumption of independent censoring. The estimating equation is derived based on martingale ideas.

Usage

```
SolveLI(theta, resData, X)
```

Arguments

`theta`	Estimated parameter values/initial values for finite dimensional parameters
`resData`	Data matrix with three columns; Z = the observed survival time, d1 = the censoring indicator of T and d2 = the censoring indicator of C.
`X`	Data matrix with covariates related to T

Value

This function returns an estimated hazard function, cumulative hazard function and distinct observed survival times;

Examples

```
n = 200
beta = c(0.5)
lambd = 0.35
eta = c(0.9,0.4)
X = cbind(rbinom(n,1,0.5))
W = cbind(rep(1,n),rbinom(n,1,0.5))
frank.cop <- copula::frankCopula(param = 5,dim = 2)
U = copula::rCopula(n,frank.cop)
T1 = (-log(1-U[,1]))/(lambd*exp(X*beta))           # Survival time'
T2 = (-log(1-U[,2]))^(1.1)*exp(W%*%eta)            # Censoring time
A = runif(n,0,15)                                  # administrative censoring time
Z = pmin(T1,T2,A)
d1 = as.numeric(Z==T1)
d2 = as.numeric(Z==T2)
resData = data.frame("Z" = Z,"d1" = d1, "d2" = d2)
theta = c(0.3,1,0.3,1)

# Estimate cumulative hazard function

cumFit_ind <- SolveLI(theta, resData,X)

cumhaz = cumFit_ind$cumhaz
time = cumFit_ind$times

# plot hazard vs time

plot(time, cumhaz, type = "l",xlab = "Time",
ylab = "Estimated cumulative hazard function")
```

---

summary.depFit *Summary of* depCensoringFit *object*

Description

Summary of depCensoringFit object

Usage

```
## S3 method for class 'depFit'
summary(object, ...)
```

Arguments

`object`	Output of `fitDepCens` function
`...`	Further arguments

Value

Summary of dependent censoring model fit in the form of table

---

`summary.indepFit` *Summary of* `indepCensoringFit` *object*

Description

Summary of `indepCensoringFit` object

Usage

```
## S3 method for class 'indepFit'
summary(object, ...)
```

Arguments

`object`	Output of `fitIndepCens` function
`...`	Further arguments

Value

Summary of independent censoring model fit in the form of table

Index