Package 'SLCARE'

October 12, 2023

Type Package

Title Semiparametric Latent Class Analysis for Recurrent Event

Version 1.0.1

Maintainer Qi Yu <qi.yu2@emory.edu>

Description An easy-to-use tool for latent class analysis for recurrent events. The modeling framework is based on the semiparametric multiplicative modeling in Zhao et al. (2022) <doi:10.1111/rssb.12499>. Our package provides an alternative method to define initial values in the estimation algorithm based on a joint frailty scale-change model described in Wang et al. (2001) <doi:10.1198/016214501753209031> and K-means. Users are also allowed to specify different initial values by themselves. Our package also provides an alternative algorithm to solving the estimating equation for unobservable latent class membership by fitting a ``pseudo" weighted multinomial regression which speeds up the rate of convergence.

License GPL (>= 3)

Encoding UTF-8

LazyData true

URL https://github.com/qyxxx/SLCARE, http://www.metaknight.us/SLCARE/

BugReports https://github.com/qyxxx/SLCARE/issues

Imports dplyr, tidyr, ggplot2, reReg, nnet

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

NeedsCompilation no

Author Qi Yu [aut, cre],
Limin Peng [aut]

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2023-10-12 03:20:02 UTC

R topics documented:

`entropy` *Calculate relative entropy*

Description

Calculate relative entropy for the selection of individual frailty and number of latent classes

Usage

```
entropy(alpha, beta, d, Z, mu_censor, gamma = 0)
```

Arguments

`alpha`	regression coefficient for multinomial logistic regression model
`beta`	class specific parameters for recurrent model
`d`	a vector of observed recurrent events for subjects of interest
`Z`	a vector of time-independent corvariates
`mu_censor`	a vector of estimated mu(C), where C is a vector of censoring time
`gamma`	individual frailty. 0 represents the frailty equals 1 and k represents the frailty follows gamma(k,k)

Value

a numerical number which measures relative entropy

get_initial *Obtain initial values for estimation procedure.*

Description

Obtain initial value for alpha by: 1. assign class membership to all subjects with Kmeans, then fit the multinomial regression to obtain alpha. Obtain initial value for beta by fitting the multiplicative intensity model studies by Wang et al. (2001) using the reReg() function, stratified by the latent class membership assigned by Kmeans.

Usage

```
get_initial(dat, K)
```

Arguments

dat	a data frame containing the data in the model
K	number of latent classes

Value

A list containing the following components:

ini_alpha	A matrix of initial alpha
ini_beta	A matrix of initial beta

mu_t *estimate mu(t)*

Description

Estimate mu0 with the Nelson-Aalen type estimator under the assumed multiplicative intensity modeling of recurrent events

Usage

```
mu_t(time_long, censor_long, t)
```

Arguments

time_long	long format time - events (excluding censoring time)
censor_long	long format censoring time (longest follow up time)
t	time of interest

Value

estimated mu(t)

`PreprocessData` *data pre-processing*

Description

pre-process data into long and wide formats that fit functions in this R package

Usage

```
PreprocessData(dat = dat)
```

Arguments

`dat`	a data frame containing the data in the model

Value

A list containing the following components:

`id_wide`	A vector of subjects ID in wide format
`d`	A vector of observed recurrent events for subjects of interest
`Z`	A vector of time-independent corvariates in wide format
`censor_wide`	A vector of censoring time (longest follow up time) in wide format
`id_long`	A vector of subjects ID in long format
`time_long`	A vector of long format time for events (excluding censoring time)
`censor_long`	A vector of censoring time (longest follow up time) in long format
`Z_long`	A vector of time-independent corvariates in long format

`p_D` *Estimate P(D|xi, Z, C)*

Description

Estimate P(Di = di | xi = k, Zi, Ci)

Usage

```
p_D(d, beta, Z, mu_censor, gamma = 0)
```

Arguments

d	a vector of observed recurrent events for subjects of interest
beta	class specific parameters for recurrent model
Z	a vector of time-independent corvariates
mu_censor	a vector of estimated mu(C), where C is a vector of censoring time
gamma	individual frailty. 0 represents the frailty equals 1 and k represents the frailty follows gamma(k,k)

Value

a vector of estimated P(Di_di | xi = k, Zi, Ci)

---

p_xi *Estimate P(xi | Z, C)*

Description

estimate P(xi = k | Zi, Ci)

Usage

```
p_xi(alpha, Z)
```

Arguments

alpha	regression coefficient for multinomial logistic regression model (xi)
Z	a vector of time-independent corvariates

Value

a vector of estimated P(xi = k | Zi, Ci)

SLCARE *Semiparametric Latent Class Analysis for Recurrent Event*

Description

Conduct Semiparametric Latent Class Analysis for Recurrent Event.

Usage

```
SLCARE(
  alpha = NULL,
  beta = NULL,
  dat,
  K = NULL,
  gamma = 0,
  max_epoches = 500,
  conv_threshold = 0.01,
  boot = NULL
)
```

Arguments

alpha	initial values for alpha for estimation procedure. This should be NULL or a numberic matirx. NULL means obtain initial value with k-means.
beta	initial value for beta for estimation procedure. This should be NULL or a numberic matirx. NULL means obtain initial value with k-means.
dat	a data frame containing the data in the model
K	number of latent classes
gamma	individual frailty. 0 represents the frailty equals 1 and k represents the frailty follows gamma(k,k)
max_epoches	maximum iteration epoches for estimation procedure
conv_threshold	converge threshold for estimation procedure
boot	bootstrap sample size

Value

A list containing the following components:

alpha Point estimates for alpha

beta Point estimates for beta

convergeloss Converge loss in estimation procedure

PosteriorPrediction Posterior prediction for observed events for subjects of interest

EstimatedTau Posterior probability of latent class membership

ModelChecking Plot for model checking

Estimated_mu0t Plot for estimated mu0(t)

est_mu0() A function allows to calculate mu0(t) for specific time points

Estimated_Mean_Function Plot of estimated mean functions

RelativeEntropy Relative entropy

InitialAlpha Initial alpha for estimation procedure

InitialBeta Initial beta for estimation procedure

If argument 'boot' is non-NULL, then SLCARE returns two additional components:

alpha_bootse Bootstrap standard error for alpha

beta_bootse Bootstrap standard error for beta

Examples

```
data(SLCARE_simdat)
# Example 1: number of latent classes k = 2,
# By default, generate initial values in estimation procedure with K-means
model1 <- SLCARE(dat = SLCARE_simdat, K=2)
# contents of output
names(model1)
# point estimates
model1$alpha
model1$beta
# converge loss in estimation procedure
model1$convergeloss
# Posterior prediction
model1$PosteriorPrediction
# Posterior probability of latent class membership
model1$EstimatedTau
# model checking plot
model1$ModelChecking
# Plot of estimated \eqn(\mu_0 (t)) for all observed time
model1$Estimated_mu0t
# Estimated \eqn(\mu_0 (t))
# You may input multiple time points of interest
model1$est_mu0(c(100, 1000, 5000))
# Plot of estimated mean function
model1$Estimated_Mean_Function
# Relative entropy
model1$RelativeEntropy
# Initial values for estimation procedure
model1$InitialAlpha
model1$InitialBeta
# You can select initial value in estimation procedure manually
# alpha <- matrix(c(0, 0, 0.5, -2, 2, -4),
#                 nrow = 3, ncol = 2, byrow = TRUE)
# beta <- matrix(c(2.5, -0.5, -0.3, 1.5, -0.2, -0.5,
#                   2.5,  0.1, 0.2), nrow = 3 , ncol = 2+1 , byrow = TRUE)
# model2 <- SLCARE(alpha, beta, dat = SLCARE_simdat)
# You can define individual frailty with gamma(p,p).
# Below is an example with manually defined initial value and frailty gamma(3,3)
```

```
# model3 <- SLCARE(alpha, beta, dat = SLCARE_simdat, gamma = 3)
# You can use bootstrap for bootstrap standard error.
# Bootstrap sample size = 100 (time consuming procedure)
# model4 <- SLCARE(alpha, beta, dat = SLCARE_simdat, boot = 100)
# SLCARE() with "boot" argument will return to two additional contents:
# "alpha_bootse", "beta_bootse" which are Bootsrap standard errors.
# model4$alpha_bootse
# model4$beta_bootse
```

SLCARE_simdat *Simulated dataset*

Description

A dataset simulated from a real world dataset

id subjects identification

time time recored including event and longest followup time (censoring)

event recurrent event indicator; 1 if a recurrent event is recorded

x1 a dummy baseline covariate

x2 a continuous baseline covariate range from 0 to 1

Usage

```
data(SLCARE_simdat)
```

Format

A data frame with 478 rows and 5 variables.

SLCA_predict *Posterior prediction for model checking*

Description

Predict numbers of recurrent events.

Usage

```
SLCA_predict(alpha, beta, d, Z, mu_censor, gamma = 0)
```

Arguments

alpha	estimated alpha - multinomial regression coefficients for latent class membership
beta	estimated beta - class
d	a vector of observed recurrent events for subjects of interest
Z	a vector of time-independent corvariates
mu_censor	a vector of estimated mu(C), where C is a vector of censoring time
gamma	individual frailty. 0 represents the frailty equals 1 and k reprsents the frailty follows gamma(k,k)

Value

A list containing the following components:

PosteriorPredict	A vector of posterior prediction for observed events for subjects of interest
tauhat	A matrix of posterior probability of latent class membership

---

update_alpha *Estimation algorithm - updating alpha*

Description

Updating alpha in estimation procedure. Updating alpha by fitting a weighted multinomial regression.

Usage

```
update_alpha(alpha, beta, d, Z, mu_censor, gamma = 0)
```

Arguments

alpha	a matrix of alpha before updating - regression coefficient for multinomial logistic regression model
beta	a matrix of beta before updating - class specific parameters for recurrent model
d	a vector of observed recurrent events for subjects of interest
Z	a vector of time-independent corvariates
mu_censor	a vector of estimated mu(C), where C is a vector of censoring time
gamma	individual frailty. 0 represents the frailty equals 1 and k represents the frailty follows gamma(k,k)

Value

a matrix of updated alpha - regression coefficient for multinomial logistic regression model

update_beta *Estimation algorithm - updating beta*

Description

Updating beta in estimation procedure. Updating beta by fitting "pseudo" weighted Poisson regression model

Usage

```
update_beta(alpha, beta, d, Z, mu_censor, gamma = 0)
```

Arguments

alpha	a matrix of alpha before updating - regression coefficient for multinomial logistic regression model
beta	a matrix of beta before updating - class specific parameters for recurrent model
d	a vector of observed recurrent events for subjects of interest
Z	a vector of time-independent corvariates
mu_censor	a vector of estimated mu(C), where C is a vector of censoring time
gamma	individual frailty. 0 represents the frailty equals 1 and k represents the frailty follows gamma(k,k)

Value

a matrix of updated beta - class specific parameters for recurrent model

Index