Package ‘penMSM’

October 14, 2022

Type Package

Title Estimating Regularized Multi-state Models Using L1 Penalties

Version 0.99

Date 2015-01-12

Author Holger Reulen

Maintainer Holger Reulen <hreulen@uni-goettingen.de>

Description
Structured fusion Lasso penalized estimation of multi-state models with the penalty applied to absolute effects and absolute effect differences (i.e., effects on transition-type specific hazard rates).

License GPL (>= 2)

Imports Rcpp (>= 0.11.3)

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-01-12 14:41:14

R topics documented:

`buildrisksets` *Calculation of risksets needed for partial likelihood formulation of multistate models.*

Description

This function calculates the risksets needed to calculate the partial likelihood of a multistate model, and/or it's derivatives.

Usage

```
buildrisksets(entry, exit, trans, event, trace)
```

Arguments

`entry`	vector with entry times.
`exit`	vector with exit times.
`trans`	vector with transition types.
`event`	vector with noncensoring event indicators.
`trace`	logical triggering printout of status information during the fitting process.

Details

This function calculates risksets.

Value

A list of length 2 with elements `Ci` and `Ri`, each vectors of length `n`.

Author(s)

Holger Reulen

`dapproxpenalty` *First derivative of the locally quadratic approximated penalty.*

Description

This function calculates the first derivative of the locally quadratic approximated penalty.

Usage

```
dapproxpenalty(psv, beta, constant)
```

Arguments

`psv`	penalty structure vector that determines the l-th penalty component when multyplied with `beta`.
`beta`	vector of regression coefficients.
`constant`	constant that is needed for the locally (in the neighborhood of 0) quadratical approximation of the absolute value function.

Details

This function calculates the first derivative of the locally quadratic approximated penalty.

Value

The value of the derivative.

Author(s)

Holger Reulen

Examples

```
## Not run: almatrix(psv, beta, constant)
```

`ddlpl` *ddlpl.*

Description

Second partial derivative of the log partial likelihood with respect to the linear predictor.

Usage

```
ddlpl(b, X, Ri, Ci)
```

Arguments

`b`	vector of regression coefficients.
`X`	design matrix.
`Ri`	list of length `n` with vectors as list elements, with the i-th element being the riskset belonging to the i-th spell.
`Ci`	list of length `n` with vectors as list elements, with the i-th element capturing the indexes of risksets in which spell `i` is included.

Details

This function calculates the second partial derivative of the log partial likelihood.

Value

A vector with second gradients.

Author(s)

Holger Reulen

Examples

```
## Not run: ddlpl(b, X, Ri, Ci)
```

`dlpl` *First derivative of the Log Partial Likelihood.*

Description

Calculates the first partial derivative of the log partial likelihood with respect to the linear predictor.

Usage

```
dlpl(event, b, X, Ri, Ci)
```

Arguments

`event`	non-censoring event indicator.
`b`	vector of regression coefficients
`X`	design matrix
`Ri`	list of length `n` with vectors as list elements, with the i-th element being the riskset belonging to the i-th spell.
`Ci`	list of length `n` with vectors as list elements, with the i-th element capturing the indexes of risksets in which spell `i` is included.

Details

This function calculates the first derivative of the log partial likelihood of a Cox type multistate model.

Value

A vector with the values of the partial first derivatives of the log partial likelihood with respect to the regression effects.

Author(s)

Holger Reulen

Examples

```
## Not run: dlpl(event, b, X, Ri, Ci)
```

---

dpenaltyfunction *First derivative of the penalty function.*

Description

This function implements the first derivative of the penalty function.

Usage

```
dpenaltyfunction(psv, beta)
```

Arguments

psv	penalty structure vector.
beta	estimated regression effects.

Details

This function implements the first derivative of the penalty function with respect to the penalty. The term 'penalty function' is described in detail on p. 4 in Oelker, Tutz (2013): A General Family of Penalties for Combining Differing Types of Penalties in Generalized Structured Models.

Value

Value of the first derivative of the penalty function (note: this is always 1, since the penalty fucntion p(xi)=xi is just the identity).

Author(s)

Holger Reulen

Examples

```
## Not run: dpenaltyfunction(psv, beta)
```

`fishercpp` *Fisher information matrix of the log partial likelihood of a multistate model.*

Description

This function provides a fast implementation for the calculation of the Fisher information matrix needed for the estimation of fusion lasso penalized multi-state models in a piece-wise exponential framework.

Usage

```
fishercpp(Xcpp, mucpp)
```

Arguments

`Xcpp`	...
`mucpp`	...

Details

...

Value

...

Author(s)

Holger Reulen

Examples

```
## Not run: fishercpp(Xcpp, mucpp)
```

`fisherinfo` *Fisher information matrix of the log partial likelihood of a multistate model.*

Description

This function calculates the Fisher information matrix needed for the estimation of multistate models using the Fisher scoring algorithm.

Usage

```
fisherinfo(beta, X, risksetlist, event)
```

Arguments

`beta`	vector of regression coefficients.
`X`	design matrix.
`risksetlist`	list of length `n` with vectors as list elements, with the i-th element being the riskset belonging to the i-th spell.
`event`	non-censoring event indicator.

Details

This function implements the Fisher scoring matrix (i.e., the second partial derivative of the log partial likelihood with respect to the components of the regression effect vector `beta`).

Value

Fisher information matrix `info`.

Author(s)

Holger Reulen

Examples

```
## Not run: fisherinfo(beta, X, risksetlist, event)
```

fisherinfoP *Fisher information matrix of the Poisson log likelihood.*

Description

This function calculates the Fisher information matrix needed for the estimation of multistate models using the Fisher scoring algorithm.

Usage

```
fisherinfoP(mu, X)
```

Arguments

mu	mu.
X	design matrix.

Details

This function implements the Fisher scoring matrix (i.e., the second partial derivative of the log partial likelihood with respect to the components of the regression effect vector `beta`).

Value

Fisher information matrix `info`.

Author(s)

Holger Reulen

Examples

```
## Not run: fisherinfo(mu, X)
```

llP *Log Likelihood for Poisson Regression.*

Description

Calculates the log likelihood for poisson regression.

Usage

```
llP(beta, X, event, offset)
```

Arguments

`beta`	vector of regression coefficients.
`X`	design matrix.
`event`	non-censoring event indicator.
`offset`	offset.

Details

This function calculates the Poisson log likelihood.

Value

The values of the Poisson log likelihood.

Author(s)

Holger Reulen

Examples

```
## Not run: llP(beta, X, event, offset)
```

`lpl` *Log Partial Likelihood.*

Description

Calculates the log partial likelihood.

Usage

```
lpl(beta, X, risksetlist, event)
```

Arguments

`beta`	vector of regression coefficients.
`X`	design matrix.
`risksetlist`	list of length `n` with vectors as list elements, with the i-th element being the riskset belonging to the i-th spell.
`event`	non-censoring event indicator.

Details

This function calculates the log partial likelihood of a Cox-type multistate model.

Value

The values of the spell-specific log partial likelihood contributions.

Author(s)

Holger Reulen

Examples

```
## Not run: lpl(beta, X, risksetlist, event)
```

---

penaltymatrix — *Penalty matrix for L1 penalized estimation of multistate models.*

Description

This builds up a penalty matrix needed for the penalized estimation of multistate models.

Usage

```
penaltymatrix(lambda, PSM, beta, w, constant)
```

Arguments

`lambda`	vector with penalty parameters for the respective penalty components.
`PSM`	penalty structure matrix containing the penalty structure vectors `psv` as rows.
`beta`	vector of regression coefficients.
`w`	vector containing weights for the respective penalty components.
`constant`	constat that is needed for the locally (in the neighborhood of 0) quadratical approximation of the absolute value function.

Details

This function calculates the penalty matrix needed for the penalized estimation of multistate models.

Value

A penalty matrix `plambda`.

Author(s)

Holger Reulen

Examples

```
## Not run: penaltymatrix(lambda, PSM, beta, w, constant)
```

penMSM *penMSM.*

Description

L1 penalized estimation of multistate models.

Usage

```
penMSM(type = "fused", d, X, PSM1, PSM2, lambda1, lambda2, w, betastart, nu = 0.5,
tol = 1e-10, max.iter = 50, trace = TRUE, diagnostics = TRUE, family = "coxph",
poissonresponse = NULL, poissonoffset = NULL, constant.approx = 1e-8)
```

Arguments

`type`	character defining the type of penalty, either `fused` or `lasso`.
`d`	data set with variables (mandatory) `entry`, `exit`, `trans`, and `event`.
`X`	design matrix.
`PSM1`	penalty structure matrix containing the penalty structure vectors `psv` as rows (lasso part).
`PSM2`	penalty structure matrix containing the penalty structure vectors `psv` as rows (fusion part).
`lambda1`	vector with penalty parameters for the respective penalty components (lasso part).
`lambda2`	vector with penalty parameters for the respective penalty components (fusion part).
`w`	vector containing weights for the respective penalty components.
`betastart`	vector containing starting values for beta.
`nu`	numeric value denoting the weight, i.e. a value between 0 and 1, of the Fisher scoring updates.
`tol`	relative update tolerance for stopping of the estimation algorithm.
`max.iter`	number of maximum iterations if tlerance is not reached.
`trace`	logical triggering printout of status information during the fitting process. .
`diagnostics`	logical triggering that Fisher matrix, score vector, and approximated penalty matrix are returned with the results.
`family`	character defining the likelihood to be used.
`poissonresponse`	response values for poisson likelihood (if used).
`poissonoffset`	offset values for poisson likelihood (if used).
`constant.approx`	constant for locally squared approximation of the absolute value penalty function.

Details

This function is the core function of this package. It implements L1 penalized estimation of multistate models, with the penalty applied to absolute effects and absolute effect differences on transition-type specific hazard rates.

Value

A list with elements `B` (matrix with estimated effects), `aic` (Akaike Information Criterion), `gcv` (GCV criterion), `df` (degrees of freedom), and (if `diagnostics` are requested) `F` (Fisher matrix), `s` (score vector), and `A` (approximated penalty matrix).

Author(s)

Holger Reulen

Examples

```
## Not run: penMSMtype = "fused", d, X, PSM1, PSM2, lambda1, lambda2, w,
betastart, nu = 0.5, tol = 1e-10, max.iter = 50, trace = TRUE,
diagnostics = TRUE, family = "coxph", poissonresponse = NULL,
poissonoffset = NULL, constant.approx = 1e-8)
## End(Not run)
```

---

plmatrix *plmatrix.*

Description

This function establishes the single vectors that set up the penalty matrix in function `penaltymatrix`.

Usage

```
plmatrix(psv, beta, constant)
```

Arguments

`psv`	index vector that determines the l-th penalty component when multiplied with `beta`.
`beta`	vector of regression coefficients.
`constant`	constant that is needed for the locally (in the neighborhood of 0) quadratical approximation of the absolute value function.

Details

This function calculates the value of the l-th penalty component, which is a locally (in the neighborhood of 0) quadratical approximation of the absolute value of a regression coefficient, or the difference between two coefficients, respectively.

Value

The object `result` takes the value of the l-th penalty component.

Author(s)

Holger Reulen

Examples

```
## Not run: plmatrix(psv, beta, constant)
```

---

scorevector — *Score vector of the log partial likelihood of a multistate model.*

Description

This function calculates the score vector needed for the estimation of multistate models using the Fisher scoring algorithm.

Usage

```
scorevector(beta, X, risksetlist, event)
```

Arguments

`beta`	vector of regression coefficients.
`X`	design matrix.
`risksetlist`	list of length `n` with vectors as list elements, with the i-th element being the riskset belonging to the i-th spell.
`event`	non-censoring event indicator.

Details

This function implements the score vector (i.e., the first partial derivative of the log partial likelihood with respect to the components of the regression effect vector `beta`).

Value

Score vector `scorevector`.

Author(s)

Holger Reulen

Examples

```
## Not run: scorevector(beta, X, risksetlist, event)
```

`scorevectorP` *Score vector of the Poisson log likelihood.*

Description

This function calculates the score vector needed for the estimation of multistate models using the Fisher scoring algorithm.

Usage

```
scorevectorP(mu, X, event)
```

Arguments

`mu`	mu.
`X`	design matrix.
`event`	non-censoring event indicator.

Details

This function implements the score vector (i.e., the first partial derivative of the Poisson log likelihood with respect to the components of the regression effect vector `beta`).

Value

Score vector `scorevector`.

Author(s)

Holger Reulen

Examples

```
## Not run: scorevectorP(beta, X, event)
```

`sF` *Score vector and Fisher information matrix of the Poisson log likelihood.*

Description

This function calculates the score vector and the Fisher information matrix needed for the estimation of multistate models using the Fisher scoring algorithm.

Usage

```
sF(mu, X, event)
```

Arguments

mu	mu.
X	design matrix.
event	non-censoring event indicator.

Details

This function implements the score vector and Fisher information matrix.

Value

s and F.

Author(s)

Holger Reulen

Examples

```
## Not run: sF(mu, X, event)
```

Index