Package 'fastcmprsk'

September 12, 2019

Type Package

Title Fine-Gray Regression via Forward-Backward Scan

Version 1.1.1

Author Eric S. Kawaguchi

Maintainer Eric S. Kawaguchi <ekawaguc@usc.edu>

Description In competing risks regression, the proportional subdistribution hazards (PSH) model is popular for its direct assessment of covariate effects on the cumulative incidence function. This package allows for both penalized and unpenalized PSH regression in linear time using a novel forward-backward scan. Penalties include Ridge, Lease Absolute Shrinkage and Selection Operator (LASSO), Smoothly Clipped Absolute Deviation (SCAD), Minimax Concave Plus (MCP), and elastic net.

Depends R (>= 3.5.0)

Imports dynpred, foreach, survival

Suggests testthat, cmprsk, crrp

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-09-11 23:00:05 UTC

R topics documented:

`AIC.fcrr` *Akaike's An Information Criterion*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrr'
AIC(object, ..., k = 2)
```

Arguments

`object`	`fcrr` object
`...`	Additional arguments. Not implemented.
`k`	Numeric, the penalty per parameter to be used; the default k = 2 is the classical AIC.

Value

A numeric value with the corresponding AIC (or BIC, or ..., depending on `k`).

AIC.fcrrp *Akaike's An Information Criterion*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrrp'
AIC(object, ..., k = 2)
```

Arguments

`object`	`fcrrp` object
`...`	Additional arguments. Not implemented.
`k`	Numeric, the penalty per parameter to be used; the default k = 2 is the classical AIC.

Value

A numeric value with the corresponding AIC (or BIC, or ..., depending on `k`).

coef.fcrr *Extract coefficients from an "fcrr" object.*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrr'
coef(object, ...)
```

Arguments

`object`	`fcrr` object
`...`	Additional arguments. Not implemented.

Value

Coefficients extracted from the model object `object`.

coef.fcrrp *Extract coefficients from an "fcrrp" object.*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrrp'
coef(object, ...)
```

Arguments

`object`	`fcrrp` object
`...`	Additional arguments. Not implemented.

Value

Coefficients extracted from the model object `object`.

confint.fcrr *Confidence Intervals for Model Parameters*

Description

Computes confidence intervals for one or more parameters in a fitted model of class `fcrr`.

Usage

```
## S3 method for class 'fcrr'
confint(object, parm, level = 0.95,
  digits = max(options()$digits - 5, 2), ...)
```

Arguments

`object`	`fcrr` object (output from `fastCrr()`)
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required
`digits`	Number of significant difits to round to.
`...`	Additional arguments. Not implemented.

Details

Prints out table of confidence intervals for the Fine-Gray model.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each coefficient estimate.

`Crisk` *Create a Competing Risk Object*

Description

Create a competing risk object, used as a response variable in the model formula for `fastCrr` and `fastCrrp`. Adapted from the `Surv` object.

Usage

```
Crisk(ftime, fstatus, cencode = 0, failcode = 1, silent = TRUE)
```

Arguments

`ftime`	A vector of event/censoring times.
`fstatus`	A vector with unique code for each event type and a separate code for censored observations.
`cencode`	Integer: code of `fstatus` that denotes censored observations (default is 0)
`failcode`	Integer: code of `fstatus` that event type of interest (default is 1)
`silent`	Logical: print information about coding.

Value

Returns an object, used as a response variable, of class `Crisk`.

`time`	vector of observed event times
`status`	vector of event indicators. 0 = censored, 1 = event of interest, 2 = competing risks

References

Fine J. and Gray R. (1999) A proportional hazards model for the subdistribution of a competing risk. *JASA* 94:496-509.

See Also

Surv

Examples

```
library(fastcmprsk)

set.seed(10)
ftime <- rexp(200)
fstatus <- sample(0:2, 200, replace = TRUE)
obj <- Crisk(ftime, fstatus, silent = FALSE)
```

`fastCrr` *Fast Fine-Gray Model Estimation*

Description

Estimates parameters for the proportional subdistribution hazards model using two-way linear scan approach.

Usage

```
fastCrr(formula, data, eps = 1e-06, max.iter = 1000,
  getBreslowJumps = TRUE, standardize = TRUE, variance = TRUE,
  var.control = varianceControl(B = 100, useMultipleCores = FALSE),
  returnDataFrame = FALSE)
```

Arguments

`formula`	a formula object, with the response on the left of a ~ operator, and the terms on the right. The response must be a Crisk object as returned by the `Crisk` function.
`data`	a data.frame in which to interpret the variables named in the formula.
`eps`	Numeric: algorithm stops when the relative change in any coefficient is less than `eps` (default is `1E-6`)
`max.iter`	Numeric: maximum iterations to achieve convergence (default is 1000)
`getBreslowJumps`	Logical: Output jumps in Breslow estimator for the cumulative hazard.
`standardize`	Logical: Standardize design matrix.
`variance`	Logical: Get standard error estimates for parameter estimates via bootstrap.
`var.control`	List of options for variance estimation.
`returnDataFrame`	Logical: Return (ordered) data frame.

Details

Fits the 'proportional subdistribution hazards' regression model described in Fine and Gray (1999) using a novel two-way linear scan approach. By default, the `Crisk` object will specify which observations are censored (0), the event of interest (1), or competing risks (2).

Value

Returns a list of class `fcrr`.

`coef`	the estimated regression coefficients
`var`	estimated variance-covariance matrix via bootstrap (if `variance = TRUE`)
`logLik`	log-pseudo likelihood at the estimated regression coefficients
`logLik.null`	log-pseudo likelihood when the regression coefficients are 0
`lrt`	log-pseudo likelihood ratio test statistic for the estimated model vs. the null model.
`iter`	iterations of coordinate descent until convergence
`converged`	logical.
`breslowJump`	Jumps in the Breslow baseline cumulative hazard (used by `predict.fcrr`)
`uftime`	vector of unique failure (event) times
`isVariance`	logical to return if variance is chosen to be estimated
`df`	returned ordered data frame if `returnDataFrame = TRUE`.

References

Fine J. and Gray R. (1999) A proportional hazards model for the subdistribution of a competing risk. *JASA* 94:496-509.

Examples

```
library(fastcmprsk)

set.seed(10)
ftime <- rexp(200)
fstatus <- sample(0:2, 200, replace = TRUE)
cov <- matrix(runif(1000), nrow = 200)
dimnames(cov)[[2]] <- c('x1','x2','x3','x4','x5')
fit <- fastCrr(Crisk(ftime, fstatus) ~ cov, variance = FALSE)

# Not run: How to set up multiple cores for boostrapping
# library(doParallel) #  make sure necessary packages are loaded
# myClust <- makeCluster(2)
# registerDoParallel(myClust)
# fit1 <- fastCrr(Crisk(ftime, fstatus) ~ cov, variance = TRUE,
# var.control = varianceControl(B = 100, useMultipleCores = TRUE))
# stopCluster(myClust)
```

`fastCrrp` *Penalized Fine-Gray Model Estimation via two-way linear scan*

Description

Performs penalized regression for the proportional subdistribution hazards model. Penalties currently include LASSO, MCP, SCAD, and ridge regression. User-specificed weights can be assigned to the penalty for each coefficient (e.g. implementing adaptive LASSO and broken adaptive ridge regerssion).

Usage

```
fastCrrp(formula, data, eps = 1e-06, max.iter = 1000,
  getBreslowJumps = TRUE, standardize = TRUE, penalty = c("LASSO",
  "RIDGE", "MCP", "SCAD", "ENET"), lambda = NULL, alpha = 0,
  lambda.min.ratio = 0.001, nlambda = 25, penalty.factor = rep(1,
  ncol(X)), gamma = switch(penalty, scad = 3.7, 2.7))
```

Arguments

`formula`	a formula object, with the response on the left of a ~ operator, and the terms on the right. The response must be a Crisk object as returned by the `Crisk` function.
`data`	a data.frame in which to interpret the variables named in the formula.
`eps`	Numeric: algorithm stops when the relative change in any coefficient is less than `eps` (default is `1E-6`)
`max.iter`	Numeric: maximum iterations to achieve convergence (default is 1000)
`getBreslowJumps`	Logical: Output jumps in Breslow estimator for the cumulative hazard.
`standardize`	Logical: Standardize design matrix.
`penalty`	Character: Penalty to be applied to the model. Options are "lasso", "scad", "ridge", "mcp", and "enet".
`lambda`	A user-specified sequence of `lambda` values for tuning parameters.
`alpha`	L1/L2 weight for elastic net regression.
`lambda.min.ratio`	Smallest value for `lambda`, as a fraction of `lambda.max` (if `lambda` is NULL).
`nlambda`	Number of `lambda` values (default is 25).
`penalty.factor`	A vector of weights applied to the penalty for each coefficient. Vector must be of length equal to the number of columns in `X`.
`gamma`	Tuning parameter for the MCP/SCAD penalty. Default is 2.7 for MCP and 3.7 for SCAD and should be left unchanged.

Details

The `fastCrrp` functions performed penalized Fine-Gray regression. Parameter estimation is performed via cyclic coordinate descent and using a two-way linear scan approach to effiiciently calculate the gradient and Hessian values. Current implementation includes LASSO, SCAD, MCP, and ridge regression.

Value

Returns a list of class `fcrrp`.

`coef`	fitted coefficients matrix with `nlambda` columns and `nvars` columns
`logLik`	vector of log-pseudo likelihood at the estimated regression coefficients
`logLik.null`	log-pseudo likelihood when the regression coefficients are 0
`lambda.path`	sequence of tuning parameter values
`iter`	number of iterations needed until convergence at each tuning parameter value
`converged`	convergence status at each tuning parameter value
`breslowJump`	Jumps in the Breslow baseline cumulative hazard (used by `predict.fcrr`)
`uftime`	vector of unique failure (event) times
`penalty`	same as above
`gamma`	same as above
`above`	same as above

References

Fu, Z., Parikh, C.R., Zhou, B. (2017) Penalized variable selection in competing risks regression. *Lifetime Data Analysis* 23:353-376.

Breheny, P. and Huang, J. (2011) Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Ann. Appl. Statist.*, 5: 232-253.

Fine J. and Gray R. (1999) A proportional hazards model for the subdistribution of a competing risk. *JASA* 94:496-509.

Examples

```
library(fastcmprsk)
set.seed(10)
ftime <- rexp(200)
fstatus <- sample(0:2, 200, replace = TRUE)
cov <- matrix(runif(1000), nrow = 200)
dimnames(cov)[[2]] <- c('x1','x2','x3','x4','x5')
fit <- fastCrrp(Crisk(ftime, fstatus) ~ cov, lambda = 1, penalty = "RIDGE")
fit$coef
```

logLik.fcrr *Extract log-pseudo likelihood from an "fcrr" object.*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrr'
logLik(object, ...)
```

Arguments

`object`	`fcrr` object
`...`	Additional arguments. Not implemented.

Value

Returns the log-pseudo likelihood of object `object`.

logLik.fcrrp *Extract log-pseudo likelihood from an "fcrrp" object.*

Description

Similar functional utility to `coef` methods.

Usage

```
## S3 method for class 'fcrrp'
logLik(object, ...)
```

Arguments

`object`	`fcrrp` object
`...`	Additional arguments. Not implemented.

Value

Returns the log-pseudo likelihood of object `object`.

plot.fcrrp *Plots solution path for penalized methods*

Description

Plots solution path for penalized methods

Usage

```
## S3 method for class 'fcrrp'
plot(x, ...)
```

Arguments

x	fcrrp object (output from fastCrrp())
...	additional arguments to plot()

Details

Plots solution path for penalized methods. x-axis: log tuning parameter values. y-axis: coeffcient estimates.

Value

A plot of the solution path for the chosen penalized method.

plot.predict.fcrr *Plots predicted cumulative incidence function*

Description

Plots predicted cumulative incidence function

Usage

```
## S3 method for class 'predict.fcrr'
plot(x, ...)
```

Arguments

x	predict.fcrr object (output from predict(fcrr x))
...	additional arguments to plot()

Value

A plot of the estimated cumulative incidence function.

predict.fcrr *Cumulative Incidence Function Estimation*

Description

Predicts cumulative incidence function from a `fcrr` object.

Usage

```
## S3 method for class 'fcrr'
predict(object, newdata, getBootstrapVariance = TRUE,
  var.control = varianceControl(B = 100, useMultipleCores = FALSE),
  type = "none", alpha = 0.05, tL = NULL, tU = NULL, ...)
```

Arguments

`object`	Output from `fcrr` object.
`newdata`	A set of covariate values to predict the CIF.
`getBootstrapVariance`	Logical: Calculate variance for CIF via bootstrap.
`var.control`	List of variance parameters from `varianceControl()`.
`type`	Confidence intervals or confidence bands.
`alpha`	Significance level to compute intervals or bands
`tL`	Lower time for band estimation.
`tU`	Upper time for band estimation.
`...`	additional arguments affecting the fastCrr procedure.
`B`	Number of bootstrap samples for variance estimation.

Details

Calculates the CIF using `fcrr` output conditional on `newdata`.

Value

Returns a list of class `predict.fcrr`.

`ftime`	Unique observed failure times
`CIF`	predicted CIF at time `ftime`
`lower`	lower interval/band limit
`upper`	upper interval/band limit
`type`	same as original argument

References

Fine J. and Gray R. (1999) A proportional hazards model for the subdistribution of a competing risk. *JASA* 94:496-509.

Examples

```
library(fastcmprsk)
set.seed(10)
ftime <- rexp(200)
fstatus <- sample(0:2, 200, replace = TRUE)
cov <- matrix(runif(1000), nrow = 200)
dimnames(cov)[[2]] <- c('x1','x2','x3','x4','x5')
fit <- fastCrr(Crisk(ftime, fstatus) ~ cov, returnDataFrame = TRUE)
cov2 <- rnorm(5)
predict(fit, newdata = cov2)
```

---

print.summary.fcrr *Prints summary of a fcrr x*

Description

Prints summary statistics of a fcrr x

Usage

```
## S3 method for class 'summary.fcrr'
print(x, digits = max(options()$digits - 4, 3),
  ...)
```

Arguments

x	output from `fastCrr()`.
digits	digits for rounding.
...	additional arguments to `print()`.

Details

Prints the convergence status, log-pseudo likelihood, the estimated coefficients, the estimated standard errors, and the two-sided p-values for the test of the individual coefficients equal to 0.

Value

Prints the convergence status, log-pseudo likelihood, the estimated coefficients, the estimated standard errors, and the two-sided p-values for the test of the individual coefficients equal to 0.

`simulateTwoCauseFineGrayModel` *Simulate data from the Fine-Gray Model*

Description

Simulate data from the model proposed in Fine and Gray (1999) for two causes. Cause 1 is assumed to be of primary importance.

Usage

```
simulateTwoCauseFineGrayModel(nobs, beta1, beta2, X = NULL, u.min = 0,
  u.max, p = 0.5, returnX = FALSE)
```

Arguments

`nobs`	Integer: Number of observations in simulated dataset.
`beta1`	A vector of effect sizes for cause 1 of length ncovs
`beta2`	A vector of effect sizes for cause 2 of length ncovs
`X`	A matrix of fixed covariates (nobs x ncovs). If `X` is NULL (default) then `X` will be simulated from MVN(O, I) with n = nobs and p = length(beta1).
`u.min`	Numeric: controls lower bound of censoring distribution where C ~ U(u.min, u.max)
`u.max`	Numeric: controls upper bound of censoring distribution where C ~ U(u.min, u.max)
`p`	Numeric: value between 0 and 1 which controls the mixture probability.
`returnX`	Logical: Whether to return `X` or not. Default is TRUE. Recommended if `X` is NULL.

Details

The function simulates data according to the setup by Fine and Gray (1999). See their paper for more information.

Value

Returns a list with the following:

`ftime`	vector of `nobs` simulated event times
`ftime`	vector of `nobs` simulated event indicators (0/1/2)
`X`	design matrix if `returnX = TRUE`. (simulated design matrix if `X = NULL`.)

References

Fine J. and Gray R. (1999) A proportional hazards model for the subdistribution of a competing risk. *JASA* 94:496-509.

Examples

```
set.seed(2019)
nobs <- 500
beta1 <- c(0.40, -0.40,  0, -0.50,  0,  0.60,  0.75,  0,  0, -0.80)
beta2 <- -beta1
Z <- matrix(rnorm(nobs * length(beta1)), nrow = nobs)
dat <- simulateTwoCauseFineGrayModel(nobs, beta1, beta2, Z, u.min = 0, u.max = 1, p = 0.5)
```

`summary.fcrr` *Summary method for fastCrr*

Description

Generate and print summaries of `fastCrr` output.

Usage

```
## S3 method for class 'fcrr'
summary(object, conf.int = TRUE, alpha = 0.05,
  digits = max(options()$digits - 5, 2), ...)
```

Arguments

`object`	`fcrr` x (output from `fastCrr()`)
`conf.int`	Logical. Whether or not to outut confidence intervals.
`alpha`	Significance level of the confidence intervals.
`digits`	Numer of significant difits to round to.
`...`	additional arguments to `print()`

Details

The summary method produces an ANOVA table for the coefficient estimates of the Fine-Gray model.

Value

The form of the value returned by `summary` depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

varianceControl *Controls for Variance Calculation*

Description

Controls for variance calculation for the fastcmprsk package.

Usage

```
varianceControl(B = 100L, seed = 1991L, useMultipleCores = FALSE)
```

Arguments

B	Integer: Number of bootstrap samples needed for variance estimation.
seed	Integer: Seed value for bootstrapping. Results may differ if parallelized.
useMultipleCores	Logical: Set to TRUE if parallelizing. (Default is FALSE).

Details

Variance-covariance estimation is done via bootstrap. Independent bootstrap runs can be performed both in serial and parallel. Parallelization is done via the doParallel package.

Value

Returns a list for variance options inputted into fastCrr.

B	same as what is defined in function.
seed	same as what is defined in function.
useMultipleCores	same as what is defined in function.

Examples

```
library(fastcmprsk)
set.seed(10)
ftime <- rexp(200)
fstatus <- sample(0:2, 200, replace = TRUE)
cov <- matrix(runif(1000), nrow = 200)
dimnames(cov)[[2]] <- c('x1','x2','x3','x4','x5')
vc <- varianceControl(B = 100, seed = 2019, useMultipleCores = FALSE)
fit1 <- fastCrr(Crisk(ftime, fstatus) ~ cov, variance = TRUE, var.control = vc)
fit1$var # Estimated covariance matrix via bootstrap
```

vcov.fcrr *Extract variance-covariance matrix from an "fcrr" object.*

Description

Similar functional utility to vcov methods.

Usage

```
## S3 method for class 'fcrr'
vcov(object, ...)
```

Arguments

object	fcrr object.
...	Additional arguments. Not implemented.

Value

Returns the estimated variance-covariance matrix (via bootstrap) from object object.

Index