

Package ‘CSCNet’

July 21, 2025

Type Package

Title Fitting and Tuning Regularized Cause-Specific Cox Models with Elastic-Net Penalty

Version 0.1.2

Maintainer Shahin Roshani <s.roshani@nki.nl>

Description Flexible tools to fit, tune and obtain absolute risk predictions from regularized cause-specific cox models with elastic-net penalty.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.0

Depends R (>= 2.10), tidyverse (>= 1.3.1), magrittr (>= 2.0.3)

Imports dplyr (>= 1.0.9), purrr (>= 0.3.4), stringr (>= 1.4.0), tibble (>= 3.1.7), survival (>= 3.3-1), prodlim (>= 2019.11.13), riskRegression (>= 2022.03.22), glmnet (>= 4.1-4), caret (>= 6.0-92), recipes (>= 0.2.0), future (>= 1.26.1), furrr (>= 0.3.0)

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Shahin Roshani [aut, cre, cph] (ORCID:
<<https://orcid.org/0000-0001-7518-157X>>)

Repository CRAN

Date/Publication 2022-11-08 18:50:02 UTC

Contents

penCSC	2
predict.penCSC	3
predictRisk.penCSC	5
print.penCSC	6
print.tune_penCSC	7
tune_penCSC	7

Index**13**

penCSC

*penCSC***Description**

Function to fit penalized cause-specific-cox with elastic-net penalty.

Usage

```
penCSC(
  time,
  status,
  vars.list,
  data,
  alpha.list,
  lambda.list,
  standardize = TRUE,
  keep = NULL
)
```

Arguments

<code>time</code>	A character showing the name of the time variable in the data.
<code>status</code>	A character showing the name of the status/event variable in the data.
<code>vars.list</code>	A named list containing the variables to be included in each cause-specific model. Variables can be vectors of variable names or a one sided formula. Names of the list must be the events and exactly the same as values in the status variable. See ‘Examples’ for details.
<code>data</code>	A data frame containing the information of the variables.
<code>alpha.list</code>	A named list containing the single alpha values of each cause-specific model. Names of the list must be the events and exactly the same as values in the status variable. See ‘Examples’ for details.
<code>lambda.list</code>	A named list containing the single lambda values of each cause-specific model. Names of the list must be the events and exactly the same as values in the status variable. See ‘Examples’ for details.
<code>standardize</code>	Logical indicating whether the variables must be standardized or not. Default is TRUE.
<code>keep</code>	A character vector of the names of variables that should not be shrunk. Default is NULL.

Value

A named list containing all the information related to the used data and the fitted models for all causes. Use \$ to explore all the involved information.

Author(s)

Shahin Roshani

References

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." Journal of Statistical Software, 33(1), 1-22. doi:10.18637/jss.v033.i01, <https://www.jstatsoft.org/v33/i01/>.

Therneau T (2022). A Package for Survival Analysis in R. R package version 3.3-1, <https://CRAN.R-project.org/package=survival>.

Wickham H, Averick M, Bryan J, Chang W, McGowan L, François R, et al. Welcome to the tidyverse. J Open Source Softw. 2019 Nov 21;4(43):1686.

Bache S, Wickham H (2022). magrittr: A Forward-Pipe Operator for R. <https://magrittr.tidyverse.org>, <https://github.com/tidyverse/magrittr>.

Examples

```
library(riskRegression)

data(Melanoma)

v1 <- list('1'=c('age','sex','ulcer','thick'),
          '2'=~age+sex+epicel+thick+ici)

a1 <- list('1'=0, '2'=.5)

l1 <- list('1'=.01, '2'=.04)

penCSC(time='time', status='status', vars.list=v1,
       data=Melanoma, alpha.list=a1, lambda.list=l1)
```

predict.penCSC

predict.penCSC

Description

Flexible prediction method for the objects of class 'penCSC' including the absolute risk prediction.

Usage

```
## S3 method for class 'penCSC'
predict(object, newX, event = NULL, time, type = "lp", reference = "zero", ...)
```

Arguments

object	An object of class 'penCSC'.
newX	A data frame containing the information of variables related to new records. Information of variables not included in the model creation will be ignored.
event	A vector of event codes which we want predictions for. This must be the same as values in the status variable of the data that was used to create the models. If NULL, absolute risk will be calculated for all involved events. Default is NULL which returns values for all involved causes.
time	A vector of time horizons which we want absolute risk predictions at. Only applicable when type='absRisk'.
type	Type of the predictions. Valid values are: 'lp' or 'link' for linear predictors, 'risk' or 'response' for exp(lp) and finally 'absRisk' for semi-parametric estimates of absolute risk.
reference	Reference for centering predictions. Valid values are 'zero' and 'sample'. Default is 'zero'. For more information on referencing see details in ?predict.coxph.
...	Additional arguments. Not used by predict.penCSC.

Value

A tibble containing the predictions based on the input arguments.

Author(s)

Shahin Roshani

References

- Pfeiffer, R. M., & Gail, M. M. (2017). Absolute risk: Methods and applications in clinical management and public health.
- Aalen, O.O. (1978) Nonparametric Inference for a Family of Counting Processes. The Annals of Statistics, 6, 701-726. doi:10.1214/aos/1176344247.
- Wickham H, Averick M, Bryan J, Chang W, McGowan L, François R, et al. Welcome to the tidyverse. J Open Source Softw. 2019 Nov 21;4(43):1686.
- Bache S, Wickham H (2022). magrittr: A Forward-Pipe Operator for R. <https://magrittr.tidyverse.org>, <https://github.com/tidyverse/magrittr>.
- Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." Journal of Statistical Software, 33(1), 1-22. doi:10.18637/jss.v033.i01, <https://www.jstatsoft.org/v33/i01/>.

Examples

```
library(riskRegression)

data(Melanoma)

v1 <- list('1'=c('age', 'sex', 'ulcer', 'thick'),
```

```

      '2'~age+sex+epicel+thick+ici)

a1 <- list('1'=0, '2'=.5)

l1 <- list('1'=.01, '2'=.04)

penfit <- penCSC(time='time', status='status', vars.list=v1,
                 data=Melanoma, alpha.list=a1, lambda.list=l1)

predict(penfit, Melanoma[1:5,], type='lp')

predict(penfit, Melanoma[1:5,], type='response')

predict(penfit, Melanoma[1:5,], type='absRisk', event=1:2, time=1825*(1:2))

```

predictRisk.penCSC	<i>predictRisk.penCSC</i>
--------------------	---------------------------

Description

predictRisk method for absolute risk prediction. This is mainly for compatibility of 'CSCNet' with functions of 'riskRegression' package.

Usage

```

## S3 method for class 'penCSC'
predictRisk(object, newdata, times, cause, ...)

```

Arguments

object	An object of class 'penCSC'.
newdata	A data frame containing the variable information of new records.
times	A vector of time horizons which we want the absolute risk predictions at.
cause	A single value indicating the event of interest which we want the absolute risk predictions for. This value should be one of the values in the status variable of the data.
...	Additional arguments. Not used by predictRisk.penCSC.

Value

A matrix with columns of absolute risk predictions of individuals for each requested time horizon.

Author(s)

Shahin Roshani

References

Wickham H, Averick M, Bryan J, Chang W, McGowan L, François R, et al. Welcome to the tidyverse. J Open Source Softw. 2019 Nov 21;4(43):1686.

Bache S, Wickham H (2022). magrittr: A Forward-Pipe Operator for R. <https://magrittr.tidyverse.org>, <https://github.com/tidyverse/magrittr>.

See Also

<https://www.rdocumentation.org/packages/riskRegression/versions/1.3.7/topics/predictRisk>

Details in: <https://rdr.io/cran/riskRegression/man/Score.html>

Examples

```
library(riskRegression)

data(Melanoma)

v1 <- list('1'=c('age','sex','ulcer','thick'),
           '2'=~age+sex+epicel+thick+ici)

a1 <- list('1'=0, '2'=.5)

l1 <- list('1'=.01, '2'=.04)

penfit <- penCSC(time='time', status='status', vars.list=v1,
                data=Melanoma, alpha.list=a1, lambda.list=l1)

predictRisk(penfit, Melanoma[1:5,], times=1825*(1:2), cause=1)
```

`print.penCSC`

print.penCSC

Description

Internal method for printing the objects of class penCSC.

Usage

```
## S3 method for class 'penCSC'
print(x, ...)
```

Arguments

`x` An object of class penCSC.

`...` Other arguments. Not used by `print.penCSC`.

Value

A modified print of penCSC objects.

Author(s)

Shahin Roshani

<code>print.tune_penCSC</code>	<code><i>print.tune_penCSC</i></code>
--------------------------------	---------------------------------------

Description

Internal method for printing the objects of class `tune_penCSC`.

Usage

```
## S3 method for class 'tune_penCSC'  
print(x, ...)
```

Arguments

<code>x</code>	An object of class <code>tune_penCSC</code> .
<code>...</code>	Other arguments. Not used by <code>print.tune_penCSC</code> .

Value

A modified print of `tune_penCSC` objects.

Author(s)

Shahin Roshani

<code>tune_penCSC</code>	<code><i>tune_penCSC</i></code>
--------------------------	---------------------------------

Description

A flexible function for tuning the involved hyper-parameters of a penalized cause-specific-cox model with elastic net penalty using the linking idea.

Usage

```
tune_penCSC(
  time,
  status,
  vars.list,
  data,
  horizons,
  event,
  rhs = ~1,
  method = "cv",
  k = 10,
  times = 25,
  p = 0.7,
  strat.var = NULL,
  metrics = "Brier",
  final.metric = NULL,
  alpha.grid = NULL,
  lambda.grid = NULL,
  nlambdas.list = NULL,
  grow.by = 0.01,
  standardize = TRUE,
  keep = NULL,
  preProc.fun = function(x) x,
  preProc.fun.test = NULL,
  parallel = FALSE,
  preProc.pkgs = NULL,
  preProc.globals = NULL,
  core.num = future::availableCores()/2
)
```

Arguments

<code>time</code>	A character showing the name of the time variable in the data.
<code>status</code>	A character showing the name of the status/event variable in the data.
<code>vars.list</code>	A named list containing the variables to be included in each cause-specific model. Variables can be vectors of variable names or a one sided formula. Names of the list must be the events and exactly the same as values in the status variable. See ‘Examples’ for details.
<code>data</code>	A data frame containing the information of the variables.
<code>horizons</code>	A vector of time horizons which we want the absolute risk predictions to be evaluated at.
<code>event</code>	The value for event of interest which we want the absolute risk predictions to be evaluated for. This must be one of the values in the status variable of the data.
<code>rhs</code>	A right hand sided formula indicating the variables to be used in estimating the inverse probability of censoring weighting (IPCW) model. Default is <code>~1</code> .

method	Resampling method to be used for hyper-parameter tuning. Values can be: 'cv' for cross validation, 'repcv' for repeated cross validation, 'lgocv' for monte-carlo cross validation, 'loocv' for leave one out cross validation and 'boot' for bootstrap. Default is 'cv'.
k	Number of folds. Only applicable for method='cv' and method='repcv'. Default is 10.
times	Repeat number of the resampling process. Only applicable for method='repcv', method='lgocv' and method='boot'. Default is 25.
p	The fraction of data to be used as the training set during resampling. Only applicable for method='lgocv'. Default is 0.7.
strat.var	A single character indicating name of the strata variable to be used to create resamples. If numerical, groups will be specified based on percentiles. Default is NULL which considers status variable as a factor and creates the resamples based on different levels of it.
metrics	Evaluation metric (loss function) to be used. Values can be 'Brier' for IPCW brier score, 'AUC' for IPCW AUC or a vector of both. Default is 'Brier'.
final.metric	The evaluation metric to decide the best hyper-parameters set for the final fits on the whole data. When NULL which is the default value, it takes the value from metrics. If both 'Brier' and 'AUC' were specified in metrics and final.metric is NULL, 'Brier' will be used.
alpha.grid	A named list containing a sequence of alpha values to be evaluated for each cause-specific model. Names of the list must be the events and exactly the same as values in the status variable. Default is NULL which orders the function to set $\text{seq}(0, 1, .5)$ for all cause-specific models. See 'Details' for more information.
lambda.grid	A named list containing a sequence of lambda values to be evaluated for each cause-specific model. Names of the list must be the events and exactly the same as values in the status variable. Default is NULL which orders the function to calculate exclusive lambda sequences for all causes. See 'Details' for more information.
nlambdas.list	A names list of single integers indicating the length of lambda sequences which are calculated automatically by the function for each cause. Only applicable when lambda.grid=NULL. Default is NULL which sets all lengths to 5. See 'Details' for more information.
grow.by	Difference between the values in the growing sequence of lambda values to find the maximum value that makes the null model. Only applicable when lambda.grid=NULL. Default is 0.01. See 'Details' for more information.
standardize	Logical indicating whether the variables must be standardized or not during model fitting procedures. Default is TRUE.
keep	A character vector of the names of variables that should not be shrunk in all model fitting procedures. Default is NULL.
preProc.fun	A function that accepts a data and returns a modified version of it that has gone through the user's desired pre-processing steps. All modifications from this function will be done during the resampling procedures to avoid data leakage. It will modify all training and test set(s) during the validation unless

other argument `preProc.fun.test` is specified by user and then it only affects the training set(s). Default is `function(x) x`. Also see the description of `preProc.fun.test` argument.

<code>preProc.fun.test</code>	A function the exact same characteristics and description as <code>preProc.fun</code> argument. If user specifies a separate function for <code>preProc.fun.test</code> , it will only affect test set(s) during validation while the function from <code>preProc.fun</code> will affect the training set(s). Default is <code>NULL</code> which means function from <code>preProc.fun</code> will be used on both training and test set(s) during validation. Also see the description of <code>preProc.fun</code> argument.
<code>parallel</code>	Logical indicating whether the tuning process should be performed in parallel or not. Default is <code>FALSE</code> .
<code>preProc.pkgs</code>	A character vector containing the names of packages that was used in creating user's <code>preProc.fun</code> while using parallel computation. Only applicable if <code>parallel=T</code> and <code>preProc.fun</code> is a user specified function using functions from other packages. See 'Examples' for details.
<code>preProc.globals</code>	A character vector containing names of objects included in <code>preProc.fun</code> to be considered as global objects while using parallel computation. The most frequent ones are the names of the user specified pre processing function or functions within this function. Only applicable if <code>parallel=T</code> and <code>preProc.fun</code> is a user specified function. See 'Examples' for details.
<code>core.num</code>	Number of CPU cores to be used for parallel computation. Only applicable if <code>parallel=T</code> . Default is <code>future::availableCores()/2</code> .

Details

`tune_penCSC` has the ability to automatically determine the candidate sequences of `alpha` & `lambda` values. Setting any of `alpha.grid` & `lambda.grid` to `NULL` will order the function to calculate them automatically. The process of determining the `lambda` values automatically is by:

1. Starting from `lambda=0`, the algorithm fits LASSO models until finding a `lambda` value that creates a `NULL` model where all variables were shrunk to be exactly zero.
2. The obtained `lambda` value will be used as the maximum value of a sequence starting from 0. The length of this sequence is controlled by values in `nlambdas.list`.

This will be done for each cause-specific model to create exclusive sequences of `lambdas` for each of them.

Value

A list containing the detailed information of the hyper-parameter tuning and the validation process, best combination of hyper-parameters and the final fits based on the whole data using the best obtained hyper-parameters. Use `$` to explore all the involved information.

Author(s)

Shahin Roshani

References

- Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, 33(1), 1-22. doi:10.18637/jss.v033.i01, <https://www.jstatsoft.org/v33/i01/>.
- Saadati, M, Beyersmann, J, Kopp-Schneider, A, Benner, A. Prediction accuracy and variable selection for penalized cause-specific hazards models. *Biometrical Journal*. 2018; 60: 288– 306. doi:10.1002/bimj.201600242.
- Gerds TA, Kattan MW (2021). *Medical Risk Prediction Models: With Ties to Machine Learning* (1st ed.). Chapman and Hall/CRC. doi:10.1201/9781138384484
- Pfeiffer, R. M., & Gail, M. M. (2017). *Absolute risk: Methods and applications in clinical management and public health*.
- Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, 28(5), 1–26. doi:10.18637/jss.v028.i05.
- Bengtsson H (2021). "A Unifying Framework for Parallel and Distributed Processing in R using Futures." *The R Journal*, 13(2), 208–227. doi:10.32614/RJ2021048.
- Vaughan D, Dancho M (2022). furr: Apply Mapping Functions in Parallel using Futures. <https://github.com/DavisVaughan/furr>, <https://furr.futureverse.org/>.
- Therneau T (2022). A Package for Survival Analysis in R. R package version 3.3-1, <https://CRAN.R-project.org/package=survival>.
- Wickham H, Averick M, Bryan J, Chang W, McGowan L, François R, et al. Welcome to the tidyverse. *J Open Source Softw*. 2019 Nov 21;4(43):1686.
- Bache S, Wickham H (2022). magrittr: A Forward-Pipe Operator for R. <https://magrittr.tidyverse.org>, <https://github.com/tidyverse/magrittr>.

Examples

```
library(riskRegression)

data(Melanoma)

v1 <- list('1'~age+sex+epicel+ici,
           '2'=c('age','ulcer','thick','invasion'))

a1 <- list('1'=0, '2'=c(.5,1))

#External standardization function with data frame as its input and output

library(recipes)

std.fun <- function(data){
  cont_vars <- data %>% select(where(~is.numeric(.))) %>% names
  cont_vars <- cont_vars[-which(cont_vars %in% c('time','status'))]
```

```
#External functions from recipes package are being used
recipe(~.,data=data) %>%
  step_center(all_of(cont_vars)) %>%
  step_scale(all_of(cont_vars)) %>%
  prep(training=data) %>% juice
}

set.seed(233)

test <- tune_penCSC(time='time',status='status',vars.list=v1,data=Melanoma,horizons=1825,
  event=1,method='cv',k=5,metrics='AUC',alpha.grid=al,standardize=FALSE,
  preProc.fun=std.fun,parallel=TRUE,preProc.pkgs='recipes')

test
```

Index

penCSC, [2](#)
predict.penCSC, [3](#)
predictRisk.penCSC, [5](#)
print.penCSC, [6](#)
print.tune_penCSC, [7](#)

tune_penCSC, [7](#)