Package ‘survivalsvm’

October 14, 2022

Type Package
Title Survival Support Vector Analysis
Version 0.0.5
Date 2018-01-05
Author Cesaire J. K. Fouodo
Maintainer Cesaire Fouodo <fouodo@imbs.uni-luebeck.de>
Description Performs support vectors analysis for data sets with survival outcome. Three approaches are available in the package: The regression approach takes censoring into account when formulating the inequality constraints of the support vector problem. In the ranking approach, the inequality constraints set the objective to maximize the concordance index for comparable pairs of observations. The hybrid approach combines the regression and ranking constraints in the same model.
Imports pracma, quadprog, kernlab, Matrix, stats, Hmisc
Suggests testthat
Depends survival
License GPL
RoxygenNote 6.0.1
URL https://github.com/imbs-hl/survivalsvm
BugReports https://github.com/imbs-hl/survivalsvm/issues
NeedsCompilation no
Repository CRAN
Date/Publication 2018-02-05 07:01:13 UTC

R topics documented:

  predict.survivalsvm .............................................. 2
  print.survivalsvm ............................................... 3
  print.survivalsvmprediction ..................................... 3
  survivalsvm ...................................................... 4

Index 8
predict.survivalsvm    Survivalsvm predictions

Description
Predictions of objects of class survivalsvm.

Usage
## S3 method for class 'survivalsvm'
predict(object, newdata, subset = NULL, ...)

Arguments
object [survivalsvm(1)]
Object of class survivalsvm, fitted with survivalsvm.
newdata [data.frame(1)]
Data frame of observations.
subset [vector(1)]
Indexes of data points of used to make the prediction.
... [any]
Further arguments passed to or from other methods.

Value
survivalsvmprediction(1) Object of class survivalsvmprediction, with elements:
  typeofsurvivalsvm  Type of survivalsvm object that is fitted in the model,
  typeofkernel      type of kernel used to fit the model,
  parameterofkernel Kernel parameters used to fit the model,
  opt.meth          solver used to fit the model,
  predicted         values predicted.

Author(s)
Cesaire J. K. Fouodo

See Also
survivalsvm

Examples
require(survival)
set.seed(123)
n <- nrow(veteran)
train.index <- sample(1:n, 0.7*n, replace = FALSE)
test.index <- setdiff(1:n, train.index)
survs.van.reg <- survivalsvm(Surv(veteran$diagtime, veteran$status) ~ .,
                           subset = train.index, data = veteran,
                           type = "regression", gamma.mu = 1,
                           opt.meth = "quadprog", kernel = "add_kernel")
pred.survs.van.reg <- predict(object = survvs.van.reg, newdata = veteran, subset = test.index)
print(pred.survs.van.reg)

---

Prints object of class `survivalsvm`.

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

- **x** [survivalsvm(1)]
  - Object of class `survivalsvm` to be printed.
- **...** [any]
  - Further arguments passed to or from other methods.

### Author(s)

Cesaire J. K. Fouodo

---

Print objects of class `survivalsvm`.

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

- **x** [survivalsvmprediction(1)]
  - Object of class `survivalsvmprediction` to be printed.
- **...** [any]
  - Further arguments passed to or from other methods.

### Author(s)

Cesaire J. K. Fouodo
Arguments

- `x` [survivalsvm(1)]
  Object survivalsvm to be printed.
- `...` [any]
  Further arguments passed to or from other methods.

Author(s)

Cesaire J. K. Fouodo

survivalsvm survivalsvm

Description

Performs support vectors analysis for data sets with survival outcome. Three approaches are available in the package: The regression approach takes censoring into account when formulating the inequality constraints of the support vector problem. In the ranking approach, the inequality constraints set the objective to maximize the concordance index for comparable pairs of observations. The hybrid approach combines the regression and ranking constraints in the same model.

Usage

```r
survivalsvm(formula = NULL, data = NULL, subset = NULL, 
  type = "regression", diff.meth = NULL, gamma.mu = NULL, 
  opt.meth = "quadprog", kernel = "lin_kernel", kernel.pars = NULL, 
  time.variable.name = NULL, status.variable.name = NULL, 
  sgf.sv = 5, sigf = 7, maxiter = 20, margin = 0.05, bound = 10, 
  eig.tol = 1e-06, conv.tol = 1e-07, posd.tol = 1e-08)
```

Arguments

- `formula` [formula(1)]
  Object of class formula. See `formula` for more details.
- `data` [data.frame(1)]
  Object of class data.frame containing data points that will be used to fit the model.
- `subset` [vector(1)]
  An index vector specifying the cases to be used in the training sample.
- `type` [character(1)]
  String indicating which type of survival support vectors model is desired. This must be one of the following strings: 'regression', 'vanbelle1', 'vanbelle2' or 'hybrid'.
- `diff.meth` [character(1)]
  String indicating which of 'makediff1', 'makediff2' or 'makediff3' is used in case of 'vanbelle1', 'vanbelle2' and 'hybrid'.

gamma.mu [numeric(1)|vector(1)]
Parameters of regularization. Note that a vector with two parameters is required in case of hybrid approach. Just one value is required in case of regression, vanbelle1 or vanbelle2.

opt.meth [character(1)]
Program used to solve the quadratic optimization problem. Either "quadprog" or "ipop".

kernel [Kernel(1)]
Kernel used to fit the model: linear kern ('lin_kernel'), additive kernel ('add_kernel'), radial basis kernels ('rbf_kernel') and the polynomial kernel ('poly_kernel').

kernel.pars [vector(1)]
Parameters of kernel, when required.

time.variable.name [character]
Name of the survival time variable in data, when given in argument.

status.variable.name [character(1)]
Name of the status variable in data.

sgf.sv [character(1)]
Number of decimal digits in the solution of the quadratic optimization problem.

sigf [numeric(1)]
Used by ipop. See ipop for details.

maxiter [integer(1)]
Used by ipop. See ipop for details.

margin [numeric(1)]
Used by ipop. See ipop for details.

bound [numeric(1)]
Used by ipop. See ipop for details.

eig.tol [numeric(1)]
Used by nearPD for adjusting positive definiteness. See nearPD for detail.

conv.tol [numeric(1)]
Used by nearPD for adjusting positive definiteness. See nearPD for detail.

posd.tol [numeric(1)]
Used by nearPD for adjusting positive definiteness. See nearPD for detail.

Details

The following denotations are used for the models implemented:

- 'regression' referring to the regression approach, named SVCR in Van Belle et al. (2011b),
- 'vanbelle1' according to the first version of survival support vector machines based on ranking constraints, named RANKSVMC by Van Belle et al. (2011b),
- 'vanbelle2' according to the second version of survival support vector machines based on ranking constraints like presented in model1 by Van Belle et al. (2011b) and
• 'hybrid' combines simultaneously the regression and ranking constraints in the same model. Hybrid model is labeled model2 by Van Belle et al. (2011b).

The argument 'type' of the function survivalsvm is used to set the type of model to be fitted. For the models vanbelle1, vanbelle2 and hybrid, differences between comparable pairs of observations are required. Each observation is compared with its nearest neighbor according to the survival time, and the three possible comparison approaches makediff1, makediff2 and makediff3 are offered to compute the differences between comparable neighbors.

The current version of survivalsvm uses the solvers ipop and quadprog to solve the dual optimization problems deduced from the support vector formulations of the models presented above. Notice that for using quadprog the kernel matrix needs to be symmetric and positive definite. Therefore when the conditions are not met, the kernel matrix needs be slightly perturbed to obtain the nearest positive definite kernel matrix. The alternative to quadprog is ipop, that can also handle a non-negative definite kernel matrix, however more time may be required to solve the quadratic optimization dual problem. The argument opt.meth is used to select the solver.

The survivalsvm command can be called giving a formula, in which the survival time and the status are grouped into a two column matrix using the command Surv from the package survival. An alternative is to pass the data frame of training data points as an argument using data, to mention the name of the survival time variable and the name of the status variable as illustrated in the third example below.

Value

survivalsvm Object of class survivalsvm, with elements:

- call
- typeofsurvivalsvm
- model.fit
- var.names

Note

This implementation is in part inspired by the Matlab toolbox Survlab (A Survival Analysis Toolbox).

Author(s)

Cesaire J. K. Fouodo

References

survivalsvm

See Also

predict.survivalsvm

Examples

survivalsvm(Surv(time, status) ~ ., veteran, gamma.mu = 0.1)

survsvm.reg <- survivalsvm(formula = Surv(diagtime, status) ~ ., data = veteran,
                           type = "regression", gamma.mu = 0.1,
                           opt.meth = "ipop", kernel = "add_kernel")

survsvm.vb2 <- survivalsvm(data = veteran, time.variable.name = "diagtime",
                           status.variable.name = "status",
                           type = "vanbelle2", gamma.mu = 0.1,
                           opt.meth = "quadprog", diff.meth = "makediff3",
                           kernel = "lin_kernel",
                           sgf.sv = 5, sigf = 7, maxiter = 20,
                           margin = 0.05, bound = 10)
Index

formula, 4
ipop, 5, 6
Kernel, 5
makediff1, 6
makediff2, 6
makediff3, 6
nearPD, 5
predict.survivalsvm, 2, 7
print.survivalsvm, 3
print.survivalsvmprediction, 3
quadprog, 5, 6
Surv, 6
survivalsvm, 2, 4