Package ‘tramnet’

July 19, 2021

Title Penalized Transformation Models
Version 0.0-6
Date 2021-07-19
URL http://ctm.R-forge.R-project.org
Description Partially penalized versions of specific transformation models implemented in package 'mlt'. Available models include a fully parametric version of the Cox model, other parametric survival models (Weibull, etc.), models for binary and ordered categorical variables, normal and transformed-normal (Box-Cox type) linear models, and continuous outcome logistic regression. Hyperparameter tuning is facilitated through model-based optimization functionalities from package 'mlrMBO'. The accompanying vignette describes the methodology used in 'tramnet' in detail. Transformation models and model-based optimization are described in Hothorn et al. (2019) <doi:10.1111/sjos.12291> and Bischl et al. (2016) <arxiv:1703.03373>, respectively.
Depends R (>= 3.5.0), tram (>= 0.3-2), CVXR (>= 0.99-4), mlrMBO (>= 1.1-2)
Imports mlt, basefun, sandwich, ParamHelpers, lhs, mlr, smoof, stats
Suggests penalized, TH.data, survival, knitr, mlbench, colorspace, mvtnorm, glmnet, trtf, Matrix, lasso2, lattice, kableExtra, coin, tbm, DiceKriging
VignetteBuilder knitr
Encoding UTF-8
License GPL-2
NeedsCompilation no
Author Lucas Kook [cre, aut], Balint Tamasi [ctb], Sandra Siegfried [ctb], Samuel Pawel [ctb], Torsten Hothorn [ctb] (<https://orcid.org/0000-0001-8301-0471>)
Maintainer Lucas Kook <lucasheinrich.kook@uzh.ch>
Repository CRAN
Date/Publication 2021-07-19 10:30:02 UTC
R topics documented:

- coef.tramnet
- coef.tramnet_Lm
- cvl_tramnet
- elnet_obj
- estfun.tramnet
- lasso_obj
- logLik.tramnet
- mbo_recommended
- mbo_tramnet
- plot.tramnet
- plot_path
- predict.tramnet
- print.summary.tramnet
- print.tramnet
- prof_alpha
- prof_lambda
- residuals.tramnet
- ridge_obj
- simulate.tramnet
- summary.tramnet
- tramnet

Index

---

| coef.tramnet | coef method for class "tramnet" |

Description

coef method for class "tramnet"

Usage

```r
## S3 method for class 'tramnet'
coef(object, with_baseline = FALSE, tol = 1e-06, ...)
```

Arguments

- `object` object of class "tramnet"
- `with_baseline` If TRUE, also prints coefficients for the baseline transformation
- `tol` tolerance when an estimate should be considered 0 and not returned (default: 1e-6)
- `...` Additional arguments to coef
coef.tramnet_Lm

Value

Numeric vector containing the model shift parameter estimates

Author(s)

Torsten Hothorn, Lucas Kook

table

Description

coef method for class "tramnet_Lm"

Usage

## S3 method for class 'tramnet_Lm'
coef(object, with_baseline = FALSE, tol = 1e-06,
as.lm = FALSE, ...)

Arguments

object object of class "tramnet_Lm"
with_baseline If TRUE, also prints coefficients for the baseline transformation
tol tolerance when an estimate should be considered 0 and not returned (default: 1e-6)

as.lm If TRUE parameters are rescaled to the usual parametrization of lm

... Additional arguments to coef

Value

Numeric vector containing the linear model shift parameter estimates

Author(s)

Torsten Hothorn, Lucas Kook

Examples

data(cars)
m0 <- Lm(dist ~ 1, data = cars)
x <- as.matrix(cars[, "speed", drop = FALSE])
mt <- tramnet(m0, x = x, alpha = 0, lambda = 0, check_dcp = FALSE)
coef(mt)
coef(mt, with_baseline = TRUE)
coef(mt, as.lm = TRUE)
coef(lm(dist ~ speed, data = cars))
cvl_tramnet  

Cross validation for "tramnet" models

Description

k-fold cross validation for "tramnet" objects over a grid of the tuning parameters based on out-of-sample log-likelihood.

Usage

cvl_tramnet(object, fold = 2, lambda = 0, alpha = 0, folds = NULL, fit_opt = FALSE)

Arguments

object  
object of class "tramnet"

fold  
number of folds for cross validation

lambda  
values for lambda to iterate over

alpha  
values for alpha to iterate over

folds  
manually specify folds for comparison with other methods

fit_opt  
If TRUE, returns the full model evaluated at optimal hyper parameters

Value

Returns out-of-sample logLik and coefficient estimates for corresponding folds and values of the hyperparameters as an object of class "cvl_tramnet"

Author(s)

Lucas Kook

Examples

set.seed(241068)
library(survival)
data("GBSG2", package = "TH.data")
X <- 1 * matrix(GBSG2$horTh == "yes", ncol = 1)
colnames(X) <- "horThyes"
GBSG2$surv <- with(GBSG2, Surv(time, cens))
m <- Coxph(surv ~ 1, data = GBSG2, log_first = TRUE)
mt <- tramnet(model = m, x = X, lambda = 0, alpha = 0)
mc <- Coxph(surv ~ horTh, data = GBSG2)
cvl_tramnet(mt, fold = 2, lambda = c(0, 1), alpha = c(0, 1))
Description

This function generates an objective function for model-based optimization based on the cross-
validated log-likelihood of a tramnet model with an elastic net penalty. It is not intended to be
called by the user directly, instead it will be given as an argument to mbo_tramnet.

Usage

```r
elnnet_obj(object, minlambda = 0, maxlambda = 16, minalpha = 0,
malpha = 1, folds, noisy = FALSE, fold)
```

Arguments

- `object`: object of class `tramnet`
- `minlambda`: minimum value for lambda (default: 0)
- `maxlambda`: maximum value for lambda (default: 16)
- `minalpha`: minimum value for alpha (default: 0)
- `maxalpha`: maximum value for alpha (default: 1)
- `folds`: self specified folds for cross validation (mainly for reproducibility and compa-
rability purposes)
- `noisy`: indicates whether folds for k-fold cross-validation should be random for each
iteration, leading to a noisy objective function (default: FALSE)
- `fold`: fold for cross validation

Value

Single objective function for model based optimization.

---

Description

estfun method for class "tramnet" which computes the score contributions w.r.t. each model
parameter.

Usage

```r
## S3 method for class 'tramnet'
estfun(object, parm = coef(object, with_baseline =
    TRUE, tol = 0), w = NULL, newdata, ...)
```
Arguments

object  object of class "tramnet"
parm  parameters for evaluating the score
w  weights
newdata  data on which to compute the score contributions
...  additional arguments to estfun

Value

Matrix of score contributions w.r.t. model parameters evaluated at parm

Author(s)

Lucas Kook

lasso_obj  Lasso objective function for model based optimization

Description

This function generates an objective function for model-based optimization based on the cross-validated log-likelihood of a tramnet model with a lasso penalty only. It is not intended to be called by the user directly, instead it will be given as an argument to mbo_tramnet.

Usage

lasso_obj(object, minlambda = 0, maxlambda = 16, folds, noisy = FALSE, fold)

Arguments

object  object of class tramnet
minlambda  minimum value for lambda (default: 0)
maxlambda  maximum value for lambda (default: 16)
folds  self specified folds for cross validation (mainly for reproducibility and comparability purposes)
noisy  indicates whether folds for k-fold cross-validation should be random for each iteration, leading to a noisy objective function (default: FALSE)
fold  fold for cross validation

Value

Single objective function for model based optimization.
**logLik.tramnet**

**Description**

logLik method for class "tramnet"

**Usage**

```r
## S3 method for class 'tramnet'
logLik(object, parm = coef(object, tol = 0,
   with_baseline = TRUE), w = NULL, newdata, ...)
```

**Arguments**

- `object`: object of class "tramnet"
- `parm`: parameters to evaluate the log likelihood at
- `w`: weights
- `newdata`: data to evaluate the log likelihood at
- `...`: Additional arguments to logLik

**Value**

returns potentially weighted (w) log-likelihood based on object evaluated at parameters parm and data newdata

**Author(s)**

Lucas Kook, Torsten Hothorn

---

**mbo_recommended**

*Fit recommended regularized tram based on model based optimization output*

**Description**

Extracts the "optimal" tuning parameters from an object of class "MBOSingleObjResult" and fits the corresponding tramnet model

**Usage**

```r
mbo_recommended(mbo_obj, m0, x, ...)
```
Arguments

- mbo_obj: object return by mbo_tramnet
- m0: null model of class "tram"
- x: matrix of covariables
- ...: additional arguments to tramnet()

Value

Object of class "tramnet"

Description

Uses model based optimization to find the optimal tuning parameter(s) in a regularized transformation model based on cross-validated log-likelihoods. Here the tramnet package makes use of the mlrMBO interface for Bayesian Optimization in machine learning problems to maximize the cv-logLik as a black-box function of the tuning parameters alpha and lambda.

Usage

mbo_tramnet(object, fold = 2, n_design = 5, n_iter = 5, 
minlambda = 0, maxlambda = 16, minalpha = 0, maxalpha = 1, 
folds = NULL, learner = "regr.km", pred.type = "se", 
opt_crit = makeMBOInfillCritEI(), noisy = FALSE, 
obj_type = c("lasso", "ridge", "elnet"), verbose = TRUE, ...)
plot.tramnet

opt_crit  optimization criterion, default: expected improvement
noisy    indicates whether folds for k-fold cross-validation should be random for each
         iteration, leading to a noisy objective function (default: FALSE)
obj_type objective type, one of "lasso", "ridge" or "elnet"
verbose  toggle for a verbose output (default: TRUE)
...      additional arguments are ignored

Value

returns an object of class "MBOSingleObjResult" which is documented in mbo

Examples

set.seed(24101968)
data("Prostate", package = "lasso2")
x <- scale(model.matrix(lpsa ~ . - 1, data = Prostate))
y <- Prostate$lpsa
m0 <- BoxCox(lpsa ~ 1, data = Prostate)
mt <- tramnet(m0, x = x, lambda = 0, alpha = 1, check_dcp = FALSE)
(run_lasso <- mbo_tramnet(object = mt, n_design = 2, n_iter = 2,
                         obj_type = "lasso"))

Value

returns an object of class "MBOSingleObjResult" which is documented in mbo

Examples

set.seed(24101968)
data("Prostate", package = "lasso2")
x <- scale(model.matrix(lpsa ~ . - 1, data = Prostate))
y <- Prostate$lpsa
m0 <- BoxCox(lpsa ~ 1, data = Prostate)
mt <- tramnet(m0, x = x, lambda = 0, alpha = 1, check_dcp = FALSE)
(run_lasso <- mbo_tramnet(object = mt, n_design = 2, n_iter = 2,
                         obj_type = "lasso"))

plot.tramnet  plot method for class "tramnet"

Description

plot method for class "tramnet"

Usage

## S3 method for class 'tramnet'
plot(x, newdata, type = c("distribution", "survivor", 
                        "density", "logdensity", "hazard", "loghazard", "cumhazard", "quantile", 
                        "trafo"), q = NULL, prob = 1:(K - 1)/K, K = 50, col = rgb(0.1, 
                        0.1, 0.1, 0.1), lty = 1, add = FALSE, ...)

Arguments

x            object of class "tramnet"
newdata      data used to predict and plot
type         type of plot produced
q             vector of quantiles
prob          vector of probabilities
plot_path

K  number of data points to plot
col  see plot
lty  see plot
add  see plot
...  additional options to plot

Value
None

Author(s)
Lucas Kook

---

plot_path  Plot regularization paths for "prof_*" classes

Description
Plot regularization paths and optionally log-likelihood trajectories of objects of class "prof_alpha" and "prof_lambda". Coefficient names are automatically added to the plot.

Usage
plot_path(object, plot_logLik = FALSE, ...)

Arguments
object  object of class "prof_alpha" or "prof_lambda"
plot_logLik  Whether logLik trajectory should be plotted (default: FALSE)
...  additional arguments to plot

Value
None

Author(s)
Lucas Kook
Examples

```r
library("tramnet")
library("survival")

data("nki70", package = "penalized")
nki70$resp <- with(nki70, Surv(time, event))
x <- scale(model.matrix(~ 0 + DIAPH3 + NUSAP1 + TSPYL5 + C20orf46, data = nki70))
y <- Coxph(resp ~ 1, data = nki70, order = 10, log_first = TRUE)
fit1 <- tramnet(y, x, lambda = 0, alpha = 1)
pfl <- prof_lambda(fit1)
plot_path(pfl)
fit2 <- tramnet(y, x, lambda = 1, alpha = 1)
pfa <- prof_alpha(fit2)
plot_path(pfa)
```

---

**predict.tramnet** predict method for class "tramnet"

**Description**

predict method for class "tramnet"

**Usage**

```r
## S3 method for class 'tramnet'
predict(object, newdata = .get_tramnet_data(object),
        ...)  
```

**Arguments**

- `object` object of class "tramnet"
- `newdata` data used for prediction
- `...` Additional arguments to `predict.ctm`

**Value**

Vector of predictions based on object evaluated at each row of newdata

**Author(s)**

Lucas Kook
print.summary.tramnet  print summary method for class "tramnet"

Description

print summary method for class "tramnet"

Usage

## S3 method for class 'summary.tramnet'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

Arguments

x object of class "tramnet"
digits number of digits to print
... additional arguments

Value

prints textual summary in the console and returns an invisible copy of the "tramnet" object

Author(s)

Lucas Kook

print.tramnet  print method for class "tramnet"

Description

print method for class "tramnet"

Usage

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

Arguments

x object of class "tramnet"
... additional arguments to summary
prof_alpha

**Value**

prints textual summary in the console and returns an invisible copy of the "tramnet" object

**Author(s)**

Lucas Kook

---

### prof_alpha

**Profiling tuning parameters**

---

**Description**

Computes the regularization path of all coefficients for a single tuning, alpha, parameter over a sequence of values.

**Usage**

```r
prof_alpha(model, min_alpha = 0, max_alpha = 1, nprof = 5,
           as.lm = FALSE)
```

**Arguments**

- `model`: model of class tramnet
- `min_alpha`: minimal value of alpha (default = 0)
- `max_alpha`: maximal value of alpha (default = 15)
- `nprof`: number of profiling steps (default = 5)
- `as.lm`: return scaled coefficients for class "tramnet_Lm"

**Value**

Object of class "prof_alpha" which contains the regularization path of all coefficients and the log-likelihood over the mixing parameter alpha

**Author(s)**

Lucas Kook

**Examples**

```r
library("tramnet")
library("survival")
data("nki70", package = "penalized")
nki70$resp <- with(nki70, Surv(time, event))
x <- scale(model.matrix(~ 0 + DIAPH3 + NUSAP1 + TSPYL5 + C20orf46, data = nki70))
y <- Coxph(resp ~ 1, data = nki70, order = 10, log_first = TRUE)
```
```r
fit <- tramnet(y, x, lambda = 1, alpha = 1)
pfa <- prof_alpha(fit)
plot_path(pfa)
```

---

### prof_lambda

Profiling tuning parameters

#### Description

Computes the regularization path of all coefficients for a single tuning parameter, lambda, over a sequence of values.

#### Usage

```r
prof_lambda(model, min_lambda = 0, max_lambda = 15, nprof = 5, 
as.lm = FALSE)
```

#### Arguments

- **model**: model of class "tramnet"
- **min_lambda**: minimal value of lambda (default = 0)
- **max_lambda**: maximal value of lambda (default = 15)
- **nprof**: number of profiling steps (default = 5)
- **as.lm**: return scaled coefficients for class "tramnet_Lm"

#### Value

Object of class "prof_lambda" which contains the regularization path of all coefficients and the log-likelihood over the penalty parameter lambda

#### Author(s)

Lucas Kook

#### Examples

```r
library("tramnet")
library("survival")

data("nk70", package = "penalized")
nk70$resp <- with(nk70, Surv(time, event))
x <- scale(model.matrix(~ 0 + DIAPH3 + NUSAP1 + TSPYL5 + C20orf46, data = nk70))
y <- Coxph(resp ~ 1, data = nk70, order = 10, log_first = TRUE)
fit <- tramnet(y, x, lambda = 0, alpha = 1)
pfl <- prof_lambda(fit)
plot_path(pfl)
```
residuals.tramnet

Description

residuals method for class "tramnet"

Usage

```r
## S3 method for class 'tramnet'
residuals(object, parm = coef(object, tol = 0,
   with_baseline = TRUE), w = NULL, newdata, ...)
```

Arguments

- `object`: object of class "tramnet"
- `parm`: parameters to evaluate score at
- `w`: weights
- `newdata`: data to evaluate score at
- `...`: additional arguments to residuals

Value

Returns a numeric vector of residuals for each row in newdata

Author(s)

Lucas Kook

ridge_obj

Ridge objective function for model based optimization

Description

This function generates an objective function for model-based optimization based on the cross-validated log-likelihood of a tramnet model with a ridge penalty only. It is not intended to be called by the user directly, instead it will be given as an argument to mbo_tramnet.

Usage

```r
ridge_obj(object, minlambda = 0, maxlambda = 16, folds,
   noisy = FALSE, fold)
```
Arguments

- **object**: object of class `tramnet`
- **minlambda**: minimum value for lambda (default: 0)
- **maxlambda**: maximum value for lambda (default: 16)
- **folds**: self specified folds for cross validation (mainly for reproducibility and comparability purposes)
- **noisy**: indicates whether folds for k-fold cross-validation should be random for each iteration, leading to a noisy objective function (default: FALSE)
- **fold**: fold for cross validation

Value

Single objective function for model based optimization.

---

**simulate.tramnet**  
*simulate method for class "tramnet"*

Description

simulate method for class "tramnet"

Usage

```r
## S3 method for class 'tramnet'
simulate(object, nsim = 1, seed = NULL,
         newdata = .get_tramnet_data(object), bysim = TRUE, ...)
```

Arguments

- **object**: object of class "tramnet"
- **nsim**: number of simulation
- **seed**: random number generator seed
- **newdata**: new data to simulate from
- **bysim**: see `simulate.ctm`
- **...**: Additional arguments to `simulate.ctm`

Value

returns a list of data.frames containing parametric bootstrap samples based on the data supplied in newdata

Author(s)

Lucas Kook
summary.tramnet

Summary method for class "tramnet"

Description

summary method for class "tramnet"

Usage

## S3 method for class 'tramnet'
summary(object, ...)

Arguments

object object of class "tramnet"

... additional arguments

Value

Returns an object of class "summary.tramnet" containing information about the model, optimization status, sparsity and tuning parameters

Author(s)

Lucas Kook

tramnet

Regularised Transformation Models

Description

Partially penalized and constrained transformation models, including Cox models and continuous outcome logistic regression. The methodology is described in the tramnet vignette accompanying this package.

Usage

tramnet(model, x, lambda, alpha, constraints = NULL, ...)
Arguments

model an object of class "tram" as returned by any of the modelling functions from package tram.
x a numeric matrix, where each row corresponds to the same row in the data argument used to fit model
lambda a positive penalty parameter for the whole penalty function
alpha a mixing parameter (between zero and one) defining the fraction between absolute and quadratic penalty terms
constraints an optional list containing a matrix of linear inequality constraints on the regression coefficients and a vector specifying the rhs of the inequality
...
additional parameters to solve

Value

An object of class "tramnet" with coef, logLik, summary, simulate, residuals and plot methods

Author(s)

Lucas Kook, Balint Tamasi, Sandra Sigfried

Examples

library("penalized")
library("survival")
## --- Comparison with penalized
data("nki70", package = "penalized")
nki70$resp <- with(nki70, Surv(time, event))
x <- scale(model.matrix(~ 0 + DIAPH3 + NUSAP1 + TSPYL5 + C20orf46,
data = nki70))
fit <- penalized(response = resp, penalized = x, lambda1 = 1, lambda2 = 0,
standardize = FALSE, data = nki70)
y <- Coxph(resp ~ 1, data = nki70, order = 10, log_first = TRUE)
fit2 <- tramnet(y, x, lambda = 1, alpha = 1) ## L1 only
coeff(fit)
coeff(fit2)
Index

* model
  tramnet, 17

coef.tramnet, 2
coef.tramnet_Lm, 3
cvl_tramnet, 4

e1net_obj, 5
estfun.tramnet, 5

lasso_obj, 6
logLik.tramnet, 7

mbo, 9
mbo_recommended, 7
mbo_tramnet, 5, 6, 8, 15

plot, 10
plot.tramnet, 9
plot_path, 10
predict.tramnet, 11
print.summary.tramnet, 12
print.tramnet, 12
prof_alpha, 13
prof_lambda, 14

residuals.tramnet, 15
ridge_obj, 15

simulate.ctm, 16
simulate.tramnet, 16
solve, 18
summary, 12
summary.tramnet, 17

tramnet, 17