Package ‘tramnet’

March 20, 2020

Title Penalized Transformation Models

Version 0.0-2

Date 2020-03-19

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 here: Hothorn et al. (2019) <doi:10.1007/s11222-019-09870-4>, Horn et al. (2016) <doi:10.1007/s11222-019-09870-4>.

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, tmb

VignetteBuilder knitr

Encoding UTF-8

License GPL-2

NeedsCompilation no

Author Lucas Kook [cre],
Balint Tamasi [aut],
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 2020-03-20 15:10:02 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

---

**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`
Value

Numeric vector containing the model shift parameter estimates

Author(s)

Torsten Hothorn, Lucas Kook

deco.tramnet_Lm  coef method for class "tramnet_Lm"

Description

ccoef 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

```r
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

```r
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)
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))
```
**Elastic net 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 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**
```
elnet_obj(object, minlambda = 0, maxlambda = 16, minalpha = 0, maxalpha = 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 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.

**estfun.tramnet**

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

**Usage**
```
estfun(object, parm = coef(object, with_baseline = TRUE, tol = 0), w = NULL, newdata, ...)
```
lasso_obj

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 compara-
bility 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, ...)```
mbo_tramnet

Model based optimization for regularized transformation models

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, ...)

Arguments

object object return by mbo_tramnet
fold fold for cross validation
n_design results in n_design times the number of tuning parameters rows for the initial
design matrix based on a random latin hypercube design
n_iter number of iterations in the model based optimization procedure
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 compara-
        bility purposes)
learner type of learner used for the optimization (default: "regr.km")
pred.type prediction type of the learner (default: "se")
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**

```r
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**

```r
## 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
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
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**

**Profiling tuning parameters**

**Value**

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

**Author(s)**

Lucas Kook

**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)
```
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
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
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)
fit <- tramnet(y, x, lambda = 0, alpha = 1)
pfl <- prof_lambda(fit)
plot_path(pfl)
residuals.tramnet

residuals method for class "tramnet"

Description
residuals method for class "tramnet"

Usage
## 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
ridge_obj(object, minlambda = 0, maxlambda = 16, folds,
   noisy = FALSE, fold)
**simulate.tramnet**

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>object of class <code>tramnet</code></td>
</tr>
<tr>
<td>minlambda</td>
<td>minimum value for lambda (default: 0)</td>
</tr>
<tr>
<td>maxlambda</td>
<td>maximum value for lambda (default: 16)</td>
</tr>
<tr>
<td>folds</td>
<td>self specified folds for cross validation (mainly for reproducibility and comparability purposes)</td>
</tr>
<tr>
<td>noisy</td>
<td>indicates whether folds for k-fold cross-validation should be random for each iteration, leading to a noisy objective function (default: FALSE)</td>
</tr>
<tr>
<td>fold</td>
<td>fold for cross validation</td>
</tr>
</tbody>
</table>

**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**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>object of class &quot;tramnet&quot;</td>
</tr>
<tr>
<td>nsim</td>
<td>number of simulation</td>
</tr>
<tr>
<td>seed</td>
<td>random number generator seed</td>
</tr>
<tr>
<td>newdata</td>
<td>new data to simulate from</td>
</tr>
<tr>
<td>bysim</td>
<td>see <code>simulate.ctm</code></td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments to <code>simulate.ctm</code></td>
</tr>
</tbody>
</table>

**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 contraints on the regression coefficients and a vector specifying the rhs of the inequality

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
coef(fit)
coef(fit2)
Index

*Topic **model**
  tramnet, 17

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

elnet_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