

Package ‘tlars’

July 15, 2022

Title The T-LARS Algorithm: Early-Terminated Forward Variable Selection

Version 0.0.1

Date 2022-07-11

Description Computes the solution path of the Terminating-LARS (T-LARS) algorithm. The T-LARS algorithm is a major building block of the T-Rex selector (see R package ‘trex’). The package is based on the papers Machkour, Muma, and Palomar (2021) <[arXiv:2110.06048](https://arxiv.org/abs/2110.06048)>, Efron, Hastie, Johnstone, and Tibshirani (2004) <[doi:10.1214/0090536040000000067](https://doi.org/10.1214/0090536040000000067)>, and Tibshirani (1996) <[doi:10.1111/j.2517-6161.1996.tb02080.x](https://doi.org/10.1111/j.2517-6161.1996.tb02080.x)>.

Maintainer Jasin Machkour <jasin.machkour@tu-darmstadt.de>

URL <https://github.com/jasinmachkour/tlars>,
<https://arxiv.org/abs/2110.06048>

BugReports <https://github.com/jasinmachkour/tlars/issues>

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Suggests knitr, rmarkdown, ggplot2, patchwork, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports Rcpp, methods, stats, graphics

Depends R (>= 2.10)

LinkingTo RcppArmadillo, Rcpp

VignetteBuilder knitr

NeedsCompilation yes

Author Jasin Machkour [aut, cre],
Simon Tien [aut],
Daniel P. Palomar [aut],
Michael Muma [aut]

Repository CRAN

Date/Publication 2022-07-15 08:40:02 UTC

R topics documented:

Gauss_data	2
plot.Rcpp_tlars_cpp	3
print.Rcpp_tlars_cpp	4
tlars	5
tlars_cpp	6
tlars_model	8

Index **10**

Gauss_data	<i>Toy data generated from a Gaussian linear model</i>
------------	--

Description

A data set containing a predictor matrix X with $n = 50$ observations and $p = 100$ variables (predictors), and a sparse parameter vector β with associated support vector.

Usage

Gauss_data

Format

A list containing a matrix X and vectors y , β , and support:

X Predictor matrix, $n = 50$, $p = 100$.

y Response vector.

beta Parameter vector.

support support vector.

Examples

```
# Generated as follows:
set.seed(789)
n <- 50
p <- 100
X <- matrix(stats::rnorm(n * p), nrow = n, ncol = p)
beta <- c(rep(5, times = 3), rep(0, times = 97))
support <- beta > 0
y <- X %*% beta + stats::rnorm(n)
Gauss_data <- list(
  X = X,
  y = y,
```

```

    beta = beta,
    support = support
  )

```

plot.Rcpp_tlars_cpp *Plots the T-LARS solution path*

Description

Plots the T-LARS solution path stored in C++ objects of class `tlars_cpp` (see [tlars_cpp](#) for details) if the object is created with `type = "lar"` (no plot for `type = "lasso"`).

Usage

```

## S3 method for class 'Rcpp_tlars_cpp'
plot(
  x,
  xlab = "# Included dummies",
  ylab = "Coefficients",
  include_dummies = TRUE,
  actions = TRUE,
  col_selected = "black",
  col_dummies = "red",
  lty_selected = "solid",
  lty_dummies = "dashed",
  legend_pos = "topleft",
  ...
)

```

Arguments

<code>x</code>	Object of the class <code>tlars_cpp</code> . See tlars_cpp for details.
<code>xlab</code>	Label of the x-axis.
<code>ylab</code>	Label of the y-axis.
<code>include_dummies</code>	Logical. If TRUE solution paths of dummies are added to the plot.
<code>actions</code>	Logical. If TRUE axis above plot with indices of added variables (Dummies represented by 'D') along the solution path is added.
<code>col_selected</code>	Color of lines corresponding to selected variables.
<code>col_dummies</code>	Color of lines corresponding to included dummies.
<code>lty_selected</code>	Line type of lines corresponding to selected variables. See par for more details.
<code>lty_dummies</code>	Line type of lines corresponding to included dummies. See par for more details.
<code>legend_pos</code>	Legend position. See xy.coords for more details.
<code>...</code>	Ignored. Only added to keep structure of generic plot function.

Value

Plots the T-LARS solution path stored in C++ objects of class `tlars_cpp` (no plot for type = "lasso").

See Also

[tlars_cpp](#), [plot](#), [par](#), and [xy.coords](#).

Examples

```
data("Gauss_data")
X <- Gauss_data$X
y <- drop(Gauss_data$y)
p <- ncol(X)
n <- nrow(X)
num_dummies <- p
dummies <- matrix(stats::rnorm(n * p), nrow = n, ncol = num_dummies)
XD <- cbind(X, dummies)
mod_tlars <- tlars_model(X = XD, y = y, num_dummies = num_dummies)
tlars(model = mod_tlars, T_stop = 3, early_stop = TRUE)
plot(mod_tlars)
```

`print.Rcpp_tlars_cpp` *Prints a summary of the results stored in a C++ object of class `tlars_cpp`.*

Description

Prints a summary of the results stored in a C++ object of class `tlars_cpp` (see [tlars_cpp](#) for details), i.e., selected variables, computation time, and number of included dummies.

Usage

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

Arguments

`x` Object of the class `tlars_cpp`. See [tlars_cpp](#) for details.
`...` Ignored. Only added to keep structure of generic [print](#) function.

Value

Prints a summary of the results stored in a C++ object of class `tlars_cpp`.

See Also

[tlars_cpp](#).

Examples

```

data("Gauss_data")
X <- Gauss_data$X
y <- drop(Gauss_data$y)
p <- ncol(X)
n <- nrow(X)
num_dummies <- p
dummies <- matrix(stats::rnorm(n * p), nrow = n, ncol = num_dummies)
XD <- cbind(X, dummies)
mod_tlars <- tlars_model(X = XD, y = y, num_dummies = num_dummies)
tlars(model = mod_tlars, T_stop = 3, early_stop = TRUE)
print(mod_tlars)

```

tlars

Executes the Terminating-LARS (T-LARS) algorithm

Description

Modifies the generic `tlars_cpp` model by executing the T-LARS algorithm and including the results in the `tlars_cpp` model.

Usage

```
tlars(model, T_stop = 1, early_stop = TRUE, info = TRUE)
```

Arguments

<code>model</code>	Object of the class <code>tlars_cpp</code> .
<code>T_stop</code>	Number of included dummies after which the random experiments (i.e., forward selection processes) are stopped.
<code>early_stop</code>	Logical. If <code>TRUE</code> , then the forward selection process is stopped after <code>T_stop</code> dummies have been included. Otherwise the entire solution path is computed.
<code>info</code>	If <code>TRUE</code> information about the T-LARS step are printed.

Value

No return value. Executes the T-LARS algorithm and includes the results in the associated object of class `tlars_cpp`.

Examples

```

data("Gauss_data")
X <- Gauss_data$X
y <- drop(Gauss_data$y)
p <- ncol(X)
n <- nrow(X)
num_dummies <- p
dummies <- matrix(stats::rnorm(n * p), nrow = n, ncol = num_dummies)

```

```

XD <- cbind(X, dummies)
mod_tlars <- tlars_model(X = XD, y = y, num_dummies = num_dummies)
tlars(model = mod_tlars, T_stop = 3, early_stop = TRUE)
beta <- mod_tlars$get_beta()
beta

```

tlars_cpp

Exposes the C++ class tlars_cpp to R

Description

Type 'tlars_cpp' in the console to see the constructors, variables, and methods of the class tlars_cpp.

Arguments

X	Real valued predictor matrix.
y	Response vector.
verbose	Logical. If TRUE progress in computations is shown.
intercept	Logical. If TRUE an intercept is included.
standardize	Logical. If TRUE the predictors are standardized and the response is centered.
num_dummies	Number of dummies that are appended to the predictor matrix.
type	Type of used algorithm (currently possible choices: 'lar' or 'lasso').
lars_state	Input list that was extracted from a previous tlars_cpp object using get_all().
T_stop	Number of included dummies after which the random experiments (i.e., forward selection processes) are stopped.
early_stop	Logical. If TRUE, then the forward selection process is stopped after T_stop dummies have been included. Otherwise the entire solution path is computed.

Value

No return value. Exposes the C++ class tlars_cpp to R.

Fields

- Constructor: new - Creates a new object of the class tlars_cpp.
- Constructor: new - Re-creates an object of the class tlars_cpp based on a list of class variables that is obtained via get_all().
- Method: execute_lars_step - Executes LARS steps until a stopping-condition is satisfied.
- Method: get_beta - Returns the estimate of the beta vector.
- Method: get_beta_path - Returns a a matrix with the estimates of the beta vectors at all steps.
- Method: get_num_active - Returns the number of active predictors.
- Method: get_num_active_dummies - Returns the number of dummy variables that have been included.

Method: `get_num_dummies` - Returns the number of dummy predictors.

Method: `get_actions` - Returns the indices of added/removed variables along the solution path.

Method: `get_df` - Returns the degrees of freedom at each step which is given by number of active variables (+1 if intercept is true).

Method: `get_R2` - Returns the R^2 statistic at each step.

Method: `get_RSS` - Returns the residual sum of squares at each step.

Method: `get_Cp` - Returns the Cp-statistic at each step.

Method: `get_lambda` - Returns the lambda-values (penalty parameters) at each step along the solution path.

Method: `get_entry` - Returns the first entry/selection steps of the predictors along the solution path.

Method: `get_norm_X` - Returns the L2-norm of the predictors.

Method: `get_mean_X` - Returns the sample means of the predictors.

Method: `get_mean_y` - Returns the sample mean of the response y .

Method: `get_all` - Returns all class variables: This list can be used as an input to the constructor to re-create an object of class `tlars_cpp`.

Examples

```
data("Gauss_data")
X <- Gauss_data$X
y <- drop(Gauss_data$y)
p <- ncol(X)
n <- nrow(X)
dummies <- matrix(stats::rnorm(n * p), nrow = n, ncol = p)
XD <- cbind(X, dummies)
mod_tlars <- tlars_model(X = XD, y = y, num_dummies = ncol(dummies))
tlars(model = mod_tlars, T_stop = 3, early_stop = TRUE)

mod_tlars$get_beta()

# mod_tlars$get_beta_path()
# mod_tlars$get_num_active()
# mod_tlars$get_num_active_dummies()
# mod_tlars$get_num_dummies()
# mod_tlars$get_actions()
# mod_tlars$get_df()
# mod_tlars$get_R2()
# mod_tlars$get_RSS()
# mod_tlars$get_Cp()
# mod_tlars$get_lambda()
# mod_tlars$get_entry()
# mod_tlars$get_norm_X()
# mod_tlars$get_mean_X()
# mod_tlars$get_mean_y()
# mod_tlars$get_all()
```

tlars_model	<i>Creates a Terminating-LARS (T-LARS) object</i>
-------------	---

Description

Creates an object of the class `tlars_cpp`.

Usage

```
tlars_model(
  lars_state,
  X,
  y,
  num_dummies,
  verbose = FALSE,
  intercept = FALSE,
  standardize = TRUE,
  type = "lar",
  info = TRUE
)
```

Arguments

<code>lars_state</code>	List of variables associated with previous T-LARS step (necessary to restart the forward selection process exactly where it was previously terminated). The <code>lars_state</code> is extracted from an object of class <code>tlars_cpp</code> via <code>get_all()</code> and is only required when the object (or its pointer) of class <code>tlars_cpp</code> is deleted or got lost in another R session (e.g., in parallel processing).
<code>X</code>	Real valued predictor matrix.
<code>y</code>	Response vector.
<code>num_dummies</code>	Number of dummies that are appended to the predictor matrix.
<code>verbose</code>	Logical. If TRUE progress in computations is shown when performing T-LARS steps on the created model.
<code>intercept</code>	Logical. If TRUE an intercept is included.
<code>standardize</code>	Logical. If TRUE the predictors are standardized and the response is centered.
<code>type</code>	'lar' for 'LARS' and 'lasso' for Lasso.
<code>info</code>	Logical. If TRUE and object is not recreated from previous T-LARS state, then information about the created object is printed.

Value

Object of the class `tlars_cpp`.

Examples

```
data("Gauss_data")
X <- Gauss_data$X
y <- drop(Gauss_data$y)
p <- ncol(X)
n <- nrow(X)
num_dummies <- p
dummies <- matrix(stats::rnorm(n * p), nrow = n, ncol = num_dummies)
XD <- cbind(X, dummies)
mod_tlars <- tlars_model(X = XD, y = y, num_dummies = num_dummies)
mod_tlars
```

Index

* datasets

Gauss_data, 2

Gauss_data, 2

par, 3, 4

plot, 3, 4

plot.Rcpp_tlars_cpp, 3

print, 4

print.Rcpp_tlars_cpp, 4

tlars, 5

tlars_cpp, 3, 4, 6

tlars_model, 8

xy.coords, 3, 4