Package 'ldamatch'

August 27, 2015

Title Multivariate Condition Matching by Backwards Elimination Using Linear Discriminant Analysis

Version 0.6.3

Description Performs group matching by backward elimination using linear discriminant analysis.

Depends R (>= 3.0.0)

License MIT + file LICENSE

LazyData true

Suggests knitr

VignetteBuilder knitr

Imports RUnit, data.table, foreach, kSamples, MASS, entropy, iterators, iterpc

NeedsCompilation no

Author Kyle Gorman [aut, cre], Geza Kiss [ctb]

Maintainer Kyle Gorman <kylebgorman@gmail.com>

Repository CRAN

Date/Publication 2015-08-27 01:04:08

R topics documented:

`ad_crit` *Criterion function for ad_halt.*

Description

Criterion function for ad_halt.

Usage

```
ad_crit(covariate, condition)
```

Arguments

`covariate`	A vector containing a covariate to match the conditions on.
`condition`	A factor vector containing condition labels.

`ad_halt` *A univariate halting test using the Anderson-Darling test (kSamples::ad.test).*

Description

A univariate halting test using the Anderson-Darling test (kSamples::ad.test).

Usage

```
ad_halt(condition, covariates, thresh)
```

Arguments

condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
thresh	The statistical threshold to pass onto the aforementioned test.

---

apply_crit *Returns smallest halting_test() / thresh ratio, or zero if less than 1.*

Description

Returns smallest halting_test() / thresh ratio, or zero if less than 1.

Usage

```
apply_crit(covariates, crit, condition, thresh)
```

Arguments

covariates	A vector or columnwise matrix containing covariates to match the conditions on.
crit	The criterion function to use, such as t_crit.
condition	A factor vector containing condition labels.
thresh	The statistical threshold to pass onto the aforementioned test.

---

check_subspaces_for_group_size_setup *Searches over all possible subspaces for specified group size setup.*

Description

Searches over all possible subspaces for specified group size setup.

Usage

```
check_subspaces_for_group_size_setup(best, grpsize_setup, sspace, condition,
  covariates, halting_test, thresh, print_info)
```

Arguments

best	The best matched groups so far together with its p-value / thresh ratio; a list containing ratio and inds (a list of subject index vectors).
grpsize_setup	A set of group sizes as a data.table row (also a list).
sspace	An ordered subject subspace: a list of vectors, with one vector per group containing the corresponding subject indices.
condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
halting_test	A function to apply to ‘covariates‘ (in matrix form) which is TRUE iff the conditions are matched.
thresh	The statistical threshold to pass onto the aforementioned test.
print_info	If TRUE, prints summary information on the input and the results, as well as progress information for the exhaustive search algorithm. Default: TRUE; can be changed using set_param("PRINT_INFO", FALSE).

Value

The best

---

combine_ratios *Combines matched group candidate with current best one.*

Description

Combines matched group candidate with current best one.

Usage

```
combine_ratios(best, candidate)
```

Arguments

best	The best matched groups so far together with its p-value / thresh ratio; a list containing ratio and inds (a list of subject index vectors).
candidate	A list containing ratio (a number) and ind (a vector).

Value

A structure like best containing the ind vectors with the highest ratio.

`create_Cartesian_iterable` *Creates Cartesian product of iterators.*

Description

Creates Cartesian product of iterators.

Usage

```
create_Cartesian_iterable(initializers, get_next, sspace)
```

Arguments

`initializers`	A list of initializer functions (with no arguments) for iterators.
`get_next`	A function for retrieving next item for an iterator argument; it assumes that the iterator returns NULL when finished.
`sspace`	elements to be used (a list of vectors)

Value

A function that returns list of values, and stops with "StopIteration" message when finished, so that it can be used with the iterators::iter() function to create an iterator that works with foreach.

`create_halting_test` *Creates halt test from multiple tests.*

Description

The created halt test function returns the smallest p-value / threshold ratio of the values produced by the supplied halt test functions.

Usage

```
create_halting_test(halting_tests)
```

Arguments

`halting_tests`	A vector of halt test functions with the signature: halting_test(condition, covariates, thresh). For the meaning of the parameters see `ldamatch`.

Value

A function that returns the minimum of all halting test values; the threshold value supplied to it is recycled for the individual functions.

`create_subject_subspace_using_LDA` *Creates ordered subspace of subject candidates for removal, using LDA.*

Description

Creates ordered subspace of subject candidates for removal, using LDA.

Usage

```
create_subject_subspace_using_LDA(condition, covariates)
```

Arguments

`condition`	A factor vector containing condition labels.
`covariates`	A vector or columnwise matrix containing covariates to match the conditions on.

Value

An ordered subject subspace: a list of vectors, with one vector per group containing the corresponding subject indices.

`decrease_group_sizes` *Creates all group sizes by reducing one group in all rows of grpsizes.*

Description

Used for generating all group size combinations for one specific total size iteratively, starting from grpsizes with one row containing original group sizes.

Usage

```
decrease_group_sizes(grpsizes, grpnames)
```

Arguments

`grpsizes`	A data.table with the columns containing the group names, and the rows containing a particular setup of group sizes. All rows are expected to have the same sum (not checked).
`grpnames`	The group names (specified because the table can have other columns as well).

Value

A data.table with the same format as grpsizes, containing all possible group setups totaling to one less than the total in grpsizes.

`estimate_exhaustive` *Estimates the maximum number of cases to be checked during exhaustive search.*

Description

Estimates the maximum number of cases to be checked during exhaustive search.

Usage

```
estimate_exhaustive(min_preserved, condition, cases_per_second = 100,
  print_info = get("PRINT_INFO", .ldamatch_globals))
```

Arguments

`min_preserved`	Assumes that at least a total of this many subjects will be preserved.
`condition`	A factor vector containing condition labels.
`cases_per_second`	Assumes that this number of cases are checked out per second, for estimating the time it takes to run the exhaustive search; default: 100.
`print_info`	If TRUE, prints partial calculations as well for the number of cases and estimated time when removing 1, 2, ... subjects.

Value

The maximum number of cases.

Examples

```
estimate_exhaustive(58, as.factor(c(rep('ALN', 25), rep('TD', 44))))
estimate_exhaustive(84, as.factor(c(rep('ASD', 51), rep('TD', 44))))
```

`get_human_readable` *Returns human readable format for number of seconds.*

Description

Returns human readable format for number of seconds.

Usage

```
get_human_readable(seconds, num_decimals = 3)
```

Arguments

seconds	The number of seconds to convert to human-readable form.
num_decimals	The number of decimals to print in the output.

Value

A string containing "<number> seconds/minutes/hours/days/years".

---

get_param *Gets parameter value for ldamatch.*

Description

Gets parameter value for ldamatch.

Usage

```
get_param(name)
```

Arguments

name	The name of the global parameter.

Value

The value of the global parameter.

See Also

set_param for parameter names.

GUtils *ldamatch: Multivariate Condition Matching by Backwards Elimination using Fisher's Linear Discriminant*

Description

Performs group matching by backward elimination using linear discriminant analysis.

ks_crit *Criterion function for ks_halt.*

Description

Criterion function for ks_halt.

Usage

```
ks_crit(covariate, logical_condition)
```

Arguments

covariate	A vector containing a covariate to match the conditions on.
logical_condition	A logical vector separating the groups.

ks_halt *A univariate halting test using the Kolmogorov-Smirnov Test (ks.test).*

Description

Note that the condition must have two levels. The null hypothesis is that they are drawn from the same distribution, so we want this to be *likely*. It is up to you to choose an appropriate threshold.

Usage

```
ks_halt(condition, covariates, thresh)
```

Arguments

condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
thresh	The statistical threshold to pass onto the aforementioned test.

ldamatch *Creates a matched group via backward selection.*

Description

Creates a matched group via backward selection.

Usage

```
ldamatch(condition, covariates, halting_test, thresh = 0.2,
  method = c("heuristic", "montecarlo", "exhaustive"), props = NULL,
  replicates = NULL, print_info = get("PRINT_INFO", .ldamatch_globals))
```

Arguments

condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
halting_test	A function to apply to ‘covariates‘ (in matrix form) which is TRUE iff the conditions are matched.
thresh	The statistical threshold to pass onto the aforementioned test.
method	The choice of search method. The "heuristic" method deploys the table of desired proportions (see below) to structure search. The "montecarlo" method randomly generates subspaces of decreasing size. The "exhaustive" considers all possible subsamples and may be very, very slow.
props	The desired proportions (percentage) of the sample for each condition; if not specified, the (full) sample proportions are used. This is used for the "heuristic" and "exhaustive" methods.
replicates	The maximum number of Monte Carlo replications to be performed. This is only used for the "montecarlo" method.
print_info	If TRUE, prints summary information on the input and the results, as well as progress information for the exhaustive search algorithm. Default: TRUE; can be changed using set_param("PRINT_INFO", FALSE).

Value

A logical vector, TRUE iff row is in the match, or a list of such vectors for the exhaustive search.

The exhaustive search method uses the foreach package to parallelize computation. To take advantage of this, you must register a cluster. For example, to use all but one of the CPU cores: doMC::registerDoMC(max(1, parallel::detectCores() - 1)) To use sequential processing: foreach::registerDoSEQ()

search_exhaustive *Searches the space backwards, prefering more subjects and better ratios. While the search is done in parallel, the search space is enormous and so it can be very slow in the worst case. It is perhaps most useful as a tool to study other matching procedures.*

Description

Searches the space backwards, prefering more subjects and better ratios. While the search is done in parallel, the search space is enormous and so it can be very slow in the worst case. It is perhaps most useful as a tool to study other matching procedures.

Usage

```
search_exhaustive(sspace, condition, covariates, halting_test, thresh, props,
  print_info)
```

Arguments

sspace	An ordered subject subspace: a list of vectors, with one vector per group containing the corresponding subject indices.
condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
halting_test	A function to apply to ‘covariates‘ (in matrix form) which is TRUE iff the conditions are matched.
thresh	The statistical threshold to pass onto the aforementioned test.
props	The desired proportions (percentage) of the sample for each condition; if not specified, the (full) sample proportions are used. This is used for the "heuristic" and "exhaustive" methods.
print_info	If TRUE, prints summary information on the input and the results, as well as progress information for the exhaustive search algorithm. Default: TRUE; can be changed using set_param("PRINT_INFO", FALSE).

Value

A list of logical vector(s) for best set(s) of subjects to be kept.

search_heuristic *At each vertex of the search graph, this takes a step which moves the proportions of conditions in the subspace closer to the desired (or sample) proportions.*

Description

At each vertex of the search graph, this takes a step which moves the proportions of conditions in the subspace closer to the desired (or sample) proportions.

Usage

```
search_heuristic(sspace, condition, covariates, halting_test, thresh, props)
```

Arguments

sspace	An ordered subject subspace: a list of vectors, with one vector per group containing the corresponding subject indices.
condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
halting_test	A function to apply to ‘covariates‘ (in matrix form) which is TRUE iff the conditions are matched.
thresh	The statistical threshold to pass onto the aforementioned test.
props	The desired proportions (percentage) of the sample for each condition; if not specified, the (full) sample proportions are used. This is used for the "heuristic" and "exhaustive" methods.

search_montecarlo *Searches for a subset satisfying the test by randomly selecting subspaces of decreasing size.*

Description

Searches for a subset satisfying the test by randomly selecting subspaces of decreasing size.

Usage

```
search_montecarlo(sspace, condition, covariates, halting_test, thresh,
  replicates)
```

Arguments

sspace	An ordered subject subspace: a list of vectors, with one vector per group containing the corresponding subject indices.
condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
halting_test	A function to apply to ‘covariates‘ (in matrix form) which is TRUE iff the conditions are matched.
thresh	The statistical threshold to pass onto the aforementioned test.
replicates	The maximum number of Monte Carlo replications to be performed. This is only used for the "montecarlo" method.

---

set_param *Sets parameters for ldamatch.*

Description

Sets parameters for ldamatch.

Usage

```
set_param(name, value)
```

Arguments

name	The name of the global parameter.
value	The new value of the global parameter.

Details

The names of the available parameters:

- MC_DEFAULT_REPLICATES: Monte Carlo search: default number of replicates
- Anderson-Darling test parameters; see kSamples::ad.test for explanation
  - AD_METHOD: the method parameter for ad.test; default: asymptotic
  - AD_NSIM: the Nsim parameter for ad.test; default: 10000
  - AD_VERSION: 1 or 2 for the two versions of the test statistic; default: 1
- PRINT_INFO: print summary information, and progress information for the exhaustive search algorithm

Value

The previous value of the global parameter.

t_crit *Criterion function for ad_halt.*

Description

Criterion function for ad_halt.

Usage

```
t_crit(covariate, condition)
```

Arguments

covariate	A vector containing a covariate to match the conditions on.
condition	A factor vector containing condition labels.

t_halt *A univariate halting test using the t-test (t.test).*

Description

A univariate halting test using the t-test (t.test).

Usage

```
t_halt(condition, covariates, thresh)
```

Arguments

condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
thresh	The statistical threshold to pass onto the aforementioned test.

U_crit *Criterion function for U_halt.*

Description

Criterion function for U_halt.

Usage

```
U_crit(covariate, condition)
```

Arguments

covariate	A vector containing a covariate to match the conditions on.
condition	A factor vector containing condition labels.

U_halt *A univariate halting test using the Wilcoxon test (wilcox.test).*

Description

A univariate halting test using the Wilcoxon test (wilcox.test).

Usage

```
U_halt(condition, covariates, thresh)
```

Arguments

condition	A factor vector containing condition labels.
covariates	A vector or columnwise matrix containing covariates to match the conditions on.
thresh	The statistical threshold to pass onto the aforementioned test.

`wilks_halt` *A multivariate halting test (appropriate when nlevels(condition) > 2)).*

Description

A multivariate halting test (appropriate when nlevels(condition) > 2)).

Usage

```
wilks_halt(condition, covariates, thresh)
```

Arguments

`condition`	A factor vector containing condition labels.
`covariates`	A vector or columnwise matrix containing covariates to match the conditions on.
`thresh`	The statistical threshold to pass onto the aforementioned test.

Index