Package ‘DESnowball’

January 4, 2014

Type Package

Title Bagging with Distance-based Regression for Differential Gene Expression Analyses

Version 1.0

Date 2014-1-3

Author Yaomin Xu <yaomin.xu@vanderbilt.edu>

Maintainer Yaomin Xu <yaomin.xu@vanderbilt.edu>

Depends R (>= 3.0.0)

Imports clue, combinat, MASS, parallel, cluster

Description This package implements a statistical data mining method to
compare whole genome gene expression profiles, with respect to the presence
of a recurrent genetic disturbance event, to identify the affected target genes.

License GPL-3

URL https://github.com/snowball-project/DESnowball

BugReports https://github.com/snowball-project/DESnowball/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2014-01-04 04:19:06

R topics documented:

`DESnowball-package` *A R package implemented Snowball approach (see references)*

Description

Genome-wide differential gene expression analysis with respect to the presence of a recurrent genetic disturbance (a driver mutation)

Details

The DESnowball package implements a differential gene expression analysis tool that compares the whole genome gene expression profiles on samples relative to the presence of a recurrent genetic disturbance (driver mutation).

The input data for the snowball analysis are the profiling of the whole genome gene expression and the mutation status of a recurrent genetic event on a group of samples. The analysis has been tested on human primary tumor samples and the minimum sample size required per group is three. Snowball does not require a balanced design between groups (see references).

The main function of the package is `snowball`, it requires two input data, named `y` and `X`, where `y` is a binary vector indicating the mutation status of the samples, and `X` is the gene expression profiles with rows corresponding to genes and columns the samples. `y` can be a `numerical`, `character` or `logical` vector. It can also be a factor. The typical format is a character vector with two values indicating the the mutation status of each subject. `X` is expected to be a data.frame with gene names as its row names, and typically it is after the initial filtering and in log scale. A reasonable choice for the initial filtering could be based on the variation of gene expression across all the samples in the study, e.g., using the coefficient of variation of each gene to select the ones with greater values than a given cutoff.

The other functions include `plotJn` for visualizing gene selection, `select.features` for gene ranking and statistical significance assessment, and `toplist` to report the top genes based on the user provided cutoff.

References

Xu, Y. and Sun, J. (2005) PfCluster: a new cluster analysis procedure for gene expression profiles. Presented at a conference on Nonparametric Inference and Probability With Applications to Science honoring Michael Woodroofe; September 24-25, 2005; Ann Arbor, Mich, 2005.

McArdlei, B.H. and Anderson, M.J. (2001) Fitting multivariate models to community data: A comment on distance-based redundancy analysis. Ecology 82(1): 290-297.

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

Guo, X., Xu, Y. and Zhao, Z.. Driver mutation BRAF regulates cell proliferation and apoptosis via MITF in the pathogenesis of melanoma, manuscript.

plotJn *Plot Jn values*

Description

Plot the $J_n(x)$ values output from snowball, with significant genes highlighted. See references for more details.

Usage

```
plotJn(x, fs, pch.nonsig = 21, pch.sig = 19, below.median = T,
  col.above = "red", col.below = "red")
```

Arguments

x	an output from snowball
fs	the corresponding output from select.features
pch.nonsig	pch of the symbols for non-significant genes. See par for more details
pch.sig	pch of the symbols for significant genes.
below.median	a logical value, set to TRUE if the genes blow the median are to be highlighted
col.above	set the highlight color for genes above the median
col.below	set the highlight color for genes below the median

sb.expression *Gene expression data of 14 patients*

Description

A demo dataset containing 6597 gene expression profiles on 14 patients, the corresponding mutation status is provided in sb.mutation

Format

A data.frame with 6597 rows and 14 variables

References

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

Guo, X., Xu, Y. and Zhao, Z.. Driver mutation BRAF regulates cell proliferation and apoptosis via MITF in the pathogenesis of melanoma, manuscript.

`sb.mutation` *Mutation status of 14 patients*

Description

A character vector indicating the mutation status of 14 patients

Format

A character vector of 14 elements

References

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

Guo, X., Xu, Y. and Zhao, Z.. Driver mutation BRAF regulates cell proliferation and apoptosis via MITF in the pathogenesis of melanoma, manuscript.

`select.features` *Compute ranking statistics, RD, and p value for gene selection*

Description

Gene selection based on the statistical significances according to the Snowball approach (see references for more details).

Usage

```
select.features(x, cutoff.p = 0.05, p.adjust.method = "BH")
```

Arguments

`x`	an output from the main function `snowball`
`cutoff.p`	cutoff for top gene list. This is applied on the multiple testing adjusted p values
`p.adjust.method`	specifies a multiple testing adjustment method, see `p.adjust` for more details

Value

a list with two elements - `fullList` and `selectedList`.`fullLIst` is a data.frame that contains `rd`, `pval` and `positive`, corresponding respectively to the RD, p value and an indicator variable of weather the RD value is above or below the median value. `selectedList` is a data.frame that contains the same variables as those in `fullList` with only the top genes that satisfy the significance cutoff specified by `cutoff.p`.

References

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

`snowball` *main function for Snowball analysis*

Description

This is the main function to perform snowball analysis. It requires a minimum input with many default operating parameters set.

Usage

```
snowball(y, X, ncore = 1, d = 300, B = 10000, B.i = 2000,
  sample.n = 100, resample.method = c("sample", "none", "combn"),
  mode.resample = c("count.class", "flat", "percent.class"), k.resample = 1)
```

Arguments

`y`	a factor variable for mutation status
`X`	data.frame containing gene expression data. The columns of `X` should be aligned with `y` on samples
`ncore`	number of processors to use for parallel computation. Set `ncore = 1` or `NULL` for non-parallel computation mode
`d`	the size of gene subset for gene level resampling. See references on d in X_d^x
`B`	bootstrap size, which is B in $J_n(x)$, defining the total number of gene subsets used to estimate J_n, $$J_n(x) = \frac{1}{B}\sum_{i=1}^{B}(\frac{1}{K}\sum_{j=1}^{K}\phi_n(g(X_{i,j}),\kappa))$$
`B.i`	bootstrap size deployed on each child job in parallel mode
`sample.n`	number of samples drawn from the subject level resampling, denoted as K in $J_n(x)$. It is ignored if `resample.method="none"` or `"combn"`
`resample.method`	this defines how the subject level resampling is performed. The possible values are `"sample"`, `"none"` and `"combn"`. Let `resample.method = "sample"` for random sampling with replacement, `"none"` for no resampling on subject dimension, and `"combn"` for all combinations by permuting the subjects in each group. See Note for more information.

`mode.resample`	this specifies how the subjects are counted for subject level leave-k-out random sampling, and whether the stratification by group is applied. The possible input values are `"count.class"`, `"percent.class"` or `"no"`. `"no"` implies that no stratification is applied and the resampling is performed on all subjects pooled together from the both groups. `"count.class"` implies the resampling leaves out a subset of subjects based on the number provided, and `"percent.class"` implies the number of subjects left out was calculated based on the percentage of the total subjects in each group. See Note for more information.
`k.resample`	A numerical value specifies the number of subjects left out during the subject level resampling. It is an integer number if `mode.resample = "count.class"` and a numerical number between 0 and 1 if `mode.resample` = "percent.class". See Note for more information.

Value

A data.frame containing two variables: `weights` and `positives`. `weights` are the $J_n(x)$ values for all genes and positives are indicators to whether a specific $J_n(x)$ is above or below the median of all $J_n(x)$'s.

Note

The resampling is applied on two dimensions (see references): gene level resamping and subject level resampling. The gene level resampling is straightforward - each time it takes `d` number of genes randomly from all the genes in `X`. The subject level resampling is specified by the combination of values given in `sample.n`, `resample.method`, `mode.resample` and `k.resample`. The flat resampling on all subjects regardless of grouping, specified by letting `resample.method="none"`, is simply a leave-k-out random sampling, where `k` is given by `k.resample`. In more complex cases, the subject level resampling can be stratified based on the groups defined on `y`, in which case, `resample.method` takes the value of either `"sample"` or `"combn"`. When `resample.method = "sample"`, it applies a leave-k-out random sampling within each group and finally only `sample.n` samples are generated from the resampling. When `resample.method = "combn"`, all possible combinations after conditioning on the restrictions given by `mode.resample` and `k.resample` are included. In this case, the total number of resampled samples varies depending on the sample size of the study. `mode.resample="count.class"` or `"percent.class"` defines two ways to calculate the number of subjects to be left out in the random sampling. The value of "count.class" indicates the exact number to be left out and "percent.class" indicates the percentage of total subjects to be left out. In all cases, `k.resample` specifies the number of subjects left out in the leave-k-out sampling. If `k.resample` is only a scalar integer number, the subjects will be sampled with exactly `k.resample` subjects left out, either across all the subjects in the case of flat sampling, or within each group in the case of stratified resampling by group. Instead, if `k.resample` a vector with two integer numbers, the sampling will leave out the number of subjects from the two groups based on the two numbers provided. The order of which number is taken for which group is based on that the first number is assigned to the first factor level and the second number is assigned to the second factor level of `factor(y)`. Check `factor(y)` to see how the two numbers in `k.resample` would be assigned to the two groups. A vector with two values for `k.resample` produces error if `mode.resample = "flat"`. This flexible way of defining the sampling scheme allows easy specification for balanced sample size between groups. See references for more details.

References

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

Examples

```
require(DESnowball)
data(snowball.demoData)
# check the demo dataset
print(sb.mutation)
head(sb.expression)
## A test run
Bn <- 10000
ncore <-4
# call Snowball
## Not run:
sb <- snowball(y=sb.mutation,X=sb.expression,
          ncore=ncore,d=100,B=Bn,
          sample.n=1)
# process the gene ranking and selection
sb.sel <- select.features(sb)
# plot the Jn values
plotJn(sb, sb.sel)
# get the significant gene list
top.genes <- toplist(sb.sel)

## End(Not run)
```

---

toplist *select the top list of genes*

Description

Report the top list based on p values.

Usage

```
toplist(fs)
```

Arguments

fs	an object output from function `select.features`

Value

a data.frame with two columns `RD` and `pvalue` (see references for details)

References

Xu, Y., Guo, X., Sun, J. and Zhao. Z. Snowball: resampling combined with distance-based regression to discover transcriptional consequences of driver mutation, manuscript.

Index