Package ‘TDbasedUFEadv’

February 24, 2024

Type Package
Title Advanced package of tensor decomposition based unsupervised feature extraction
Version 1.2.0
Language en-US
Description This is an advanced version of TDbasedUFE, which is a comprehensive package to perform Tensor decomposition based unsupervised feature extraction. In contrast to TDbasedUFE which can perform simple the feature selection and the multiomics analyses, this package can perform more complicated and advanced features, but they are not so popularly required. Only users who require more specific features can make use of its functionality.

biocViews GeneExpression, FeatureExtraction, MethylationArray, SingleCell, Software
License GPL-3
Encoding UTF-8
Imports TDbasedUFE, Biobase, GenomicRanges, utils, rTensor, methods, graphics, RTCGA, stats, enrichplot, DOSE, STRINGdb, enrichR, hash, shiny

RoxygenNote 7.2.3
BugReports https://github.com/tagtag/TDbasedUFEadv/issues
URL https://github.com/tagtag/TDbasedUFEadv
Suggests knitr, rmarkdown, testthat (>= 3.0.0), RTCGA.rnaseq, RTCGA.clinical, BiocStyle, MOFAdata

VignetteBuilder knitr
Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/TDbasedUFEadv
git_branch RELEASE_3_18
git_last_commit 266b4f5
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Description

This is an advanced version of TDbasedUFE, which is a comprehensive package to perform Tensor decomposition based unsupervised feature extraction. In contrast to TDbasedUFE which can perform simple the feature selection and the multiomics analyses, this package can perform more complicated and advanced features, but they are not so popularly required. Only users who require more specific features can make use of its functionality.

Author(s)

Maintainer: Y-h. Taguchi <tag@granular.com> (ORCID)

See Also

Useful links:

- https://github.com/tagtag/TDbasedUFEadv
- Report bugs at https://github.com/tagtag/TDbasedUFEadv/issues
### computeSVD

**Title** Perform SVD toward reduced matrix generated from a tensor with partial summation

**Usage**

```r
computeSVD(matrix1, matrix2, dim = 10L, scale = TRUE)
```

**Arguments**

- `matrix1`: The first original matrix that generates a tensor
- `matrix2`: The second original matrix that generates a tensor
- `dim`: The number of singular value vectors to be computed
- `scale`: If matrix should be scaled or not

**Value**

Singular value vectors attributed to two sets of objects associated with singular value vectors attributed to features, by multiplying

**Examples**

```r
matrix1 <- matrix(runif(200),20)
matrix2 <- matrix(runif(400),20)
SVD <- computeSVD(matrix1, matrix2)
```

---

### prepareCondDrugandDisease

**Description**

Prepare condition matrix for expDrug

**Usage**

```r
prepareCondDrugandDisease(expDrug)
```

**Arguments**

- `expDrug`: input gene expression profile
prepareCondTCGA

Value
Condition matrix for expDrug

Examples
library(RTCGA.rnaseq)
Cancer_cell_lines <- list(ACC.rnaseq,BLCA.rnaseq,BRCA.rnaseq)
Drug_and_Disease <- prepareexpDrugandDisease(Cancer_cell_lines)
Cond <- prepareCondDrugandDisease(Drug_and_Disease$expDrug)

prepareCondTCGA Prepare Sample label for TCGA data

Description
Prepare Sample label for TCGA data

Usage
prepareCondTCGA(
  Multi_sample,
  Clinical,
  ID_column_of_Multi_sample,
  ID_column_of_Clinical
)

Arguments
Multi_sample list of sample ids
Clinical List of clinical data matrix from RTCGA.clinical
ID_column_of_Multi_sample Column numbers used for conditions
ID_column_of_Clinical Column numbers that include corresponding sample ids in clinical data

Value
list of sample labels

Examples
library(RTCGA.clinical)
library(RTCGA.rnaseq)
Clinical <- list(BLCA.clinical, BRCA.clinical, CESC.clinical, COAD.clinical)
Multi_sample <- list(
  BLCA.rnaseq[seq_len(100), 1, drop = FALSE],
  BRCA.rnaseq[seq_len(100), 1, drop = FALSE],
  CESC.clinical$ID,
  COAD.clinical$ID
)
prepareexpDrugandDisease

Generating gene expression of drug treated cell lines and a disease cell line

Description

Generating gene expression of drug treated cell lines and a disease cell line

Usage

prepareexpDrugandDisease(Cancer_cell_lines)

Arguments

Cancer_cell_lines

<- list(ACC.rnaseq, BLCA.rnaseq, BRCA.rnaseq) list that includes individual data set from RTCGA.rnaseq

Value

list of expDrug and expDisease

Examples

library(RTCGA.rnaseq)
Cancer_cell_lines <- list(ACC.rnaseq, BLCA.rnaseq, BRCA.rnaseq)
Drug_and_Disease <- prepareexpDrugandDisease(Cancer_cell_lines)
prepareTensorfromList  Prepare tensor from a list that includes multiple profiles

Description

Prepare tensor from a list that includes multiple profiles

Usage

prepareTensorfromList(Multi, proj_dim)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi</td>
<td>a list that includes multiple profiles</td>
</tr>
<tr>
<td>proj_dim</td>
<td>the number of projection dimensions</td>
</tr>
</tbody>
</table>

Value

a tensor as a bundle of singular value vectors obtained by applying SVD to individual omics

Examples

library(MOFAdata)
data("CLL_data")
data("CLL_covariates")
Z <- prepareTensorfromList(CLL_data,10L)

prepareTensorfromMatrix  Generate tensor from two matrices

Description

Generate tensor from two matrices

Usage

prepareTensorfromMatrix(matrix1, matrix2)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matrix1</td>
<td>the first input matrix</td>
</tr>
<tr>
<td>matrix2</td>
<td>the second input matrix</td>
</tr>
</tbody>
</table>
**Value**

A tensor generated from the first and second matrices

**Examples**

```
Z <- prepareTensorFromMatrix(matrix(runif(100),10),matrix(runif(100),10))
```

---

**prepareTensorRect**

Prepare tensor generated from two matrices that share samples

**Description**

Prepare tensor generated from two matrices that share samples

**Usage**

```
prepareTensorRect(
  sample,
  feature,
  value,
  featureRange = GRanges(NULL),
  sampleData = list(NULL)
)
```

**Arguments**

- **sample**: Character vector of sample names
- **feature**: list of features from two matrices
- **value**: array, contents of
- **featureRange**: Genomic Ranges to be associated with features
- **sampleData**: List of conditional labeling associated with samples

**Value**

Tensor generated from two matrices that share samples

**Examples**

```
matrix1 <- matrix(runif(1000),200)  # row features, column samples
matrix2 <- matrix(runif(2000),400)  # row features, column samples
Z <- prepareTensorFromMatrix(t(matrix1),t(matrix2))
Z <- prepareTensorRect(sample=as.character(seq_len(50)),
  feature=list(as.character(seq_len(200)),as.character(seq_len(400))),
  sampleData=list(rep(seq_len(2),each=25)),value=Z)
```
selectFeatureProj

Select feature when projection strategy is employed for the case where features are shared with multiple omics profiles

Description

Select feature when projection strategy is employed for the case where features are shared with multiple omics profiles

Usage

selectFeatureProj(
  HOSVD,
  Multi,
  cond,
  de = 1e-04,
  p0 = 0.01,
  breaks = 100L,
  input_all = NULL
)

Arguments

HOSVD  
HOSVD

Multi  
list of omics profiles, row: sample, column: feature

cond  
list of conditions for individual omics profiles

de  
initial value for optimization of standard deviation

p0  
Threshold P-value

breaks  
The number of bins of histogram of P-values

input_all  
The number of selected feature. if null, interactive mode is activated

Value

list composed of logical vector that represent which features are selected and p-values

Examples

library(TDbasedUFE)
Multi <- list(matrix(runif(1000),10),matrix(runif(1000),10),
  matrix(runif(1000),10),matrix(runif(1000),10))
Z <- prepareTensorfromList(Multi,10L)
Z <- aperm(Z,c(2,1,3))
Z <- PrepareSummarizedExperimentTensor(feature =as.character(1:10),
  sample=array(“",1),value=Z)
HOSVD <- computeHosvd(Z)
cond <- rep(list(rep(1:2,each=5)),4)
index <- selectFeatureProj(HOSVD,Multi,cond,de=0.1,input_all=2)
**selectFeatureRect**  
*Select features through the selection of singular value vectors*

**Description**
Select features through the selection of singular value vectors

**Usage**

```r
selectFeatureRect(
  SVD,
  cond,
  de = rep(1e-04, 2),
  p0 = 0.01,
  breaks = 100L,
  input_all = NULL
)
```

**Arguments**
- **SVD**: SVD computed from matrix generated by partial summation of a tensor
- **cond**: Condition to select singular value vectors
- **de**: Initial values to be used for optimization of standard deviation
- **p0**: Threshold value for the significance
- **breaks**: Number of bins of histogram of P-values
- **input_all**: The ID of selected singular value vectors. If it is null, interactive mode is activated.

**Value**
List of lists that includes P-values as well as if individual features selected.

**Examples**
```r
set.seed(0)
matrix1 <- matrix(runif(2000), 200)
matrix2 <- matrix(runif(4000), 200)
SVD <- computeSVD(matrix1, matrix2)
index_all <- selectFeatureRect(SVD,
  list(NULL, rep(seq_len(2), each=5), rep(seq_len(2), each=10)),
  de=rep(0.5, 2),
  input_all=1)
```
selectFeatureTransRect

Select features for a tensor generated from two matrices that share samples.

Description

Select features for a tensor generated from two matrices that share samples.

Usage

```r
selectFeatureTransRect(
  HOSVD,
  cond,
  de = rep(1e-04, 2),
  p0 = 0.01,
  breaks = 100L,
  input_all = NULL
)
```

Arguments

- **HOSVD**: HOSVD
- **cond**: list of conditions
- **de**: initial values for optimization of standard deviation
- **p0**: threshold value for the significance
- **breaks**: number of bins of the histogram of P-values
- **input_all**: The selected singular value vectors attributed to samples. if NULL, interactive mode

Value

list of logical vector that represent if the individual features are selected and P-values.

Examples

```r
library(TDbasedUFE)
set.seed(0)
matrix1 <- matrix(runif(1000),20) #row features, column samples
matrix2 <- matrix(runif(2000),40) #row features, column samples
Z <- prepareTensorfromMatrix(t(matrix1),t(matrix2))
Z <- prepareTensorRect(sample=as.character(seq_len(50)),
  feature=list(as.character(seq_len(20)),as.character(seq_len(40))),
  sampleData=list(rep(seq_len(2),each=25)),value=Z)
HOSVD <- computeHosvd(Z)
cond <- list(attr(Z,"sampleData")[[1]],NULL,NULL)
index_all <- selectFeatureTransRect(HOSVD,cond,de=c(0.1,0.1),
  input_all=2,p0=1e-10)
```
Description

Class definitions

Slots

- sample character.
- feature list.
- value array.
- featureRange GRanges.
- sampleData list.

transSVD

Convert SVD to that for the case where samples are shared between two matrices

Description

Convert SVD to that for the case where samples are shared between two matrices

Usage

transSVD(SVD)

Arguments

- SVD input SVD object generated from computeSVD function

Value

computed SVD objects

Examples

```r
class1 <- matrix(runif(200),20)
class2 <- matrix(runif(400),20)
SVD <- computeSVD(class1,class2)
SVD <- transSVD(SVD)
```
Index

* internal
  TDbasedUFEdv-package, 2

computeSVD, 3
prepareCondDrugandDisease, 3
prepareCondTCGA, 4
prepareexpDrugandDisease, 5
prepareTensorfromList, 6
prepareTensorfromMatrix, 6
prepareTensorRect, 7
selectFeatureProj, 8
selectFeatureRect, 9
selectFeatureTransRect, 10

TDbasedUFEdv (TDbasedUFEdv-package), 2
TDbasedUFEdv-package, 2
TensorRect-class, 11
transSVD, 11