Package 'CrossICC'

October 17, 2020

Type Package

Title An Interactive Consensus Clustering Framework for Multi-platform Data Analysis

Version 1.2.0

Description CrossICC utilizes an iterative strategy to derive the optimal gene set and cluster number from consensus similarity matrix generated by consensus clustering and it is able to deal with multiple cross platform datasets so that requires no between-dataset normalizations. This package also provides abundant functions for visualization and identifying subtypes of cancer. Specially, many cancer-related analysis methods are embedded to facilitate the clinical translation of the identified cancer subtypes.

License GPL-3 | file LICENSE

Encoding UTF-8

Suggests rmarkdown, testthat, knitr, shiny, shinydashboard, shinyWidgets, shinycssloaders, DT, ggthemes, ggplot2, pheatmap, RColorBrewer, tibble, ggalluvial

RoxygenNote 6.1.1

Imports data.table, methods, MergeMaid, ConsensusClusterPlus, limma, cluster, dplyr, Biobase, grDevices, stats, graphics, utils

Depends R (>= 3.5), MASS

biocViews Software, GeneExpression, DifferentialExpression, GUI, GeneSetEnrichment, Classification, Clustering, FeatureExtraction, Survival, Microarray, RNASeq, BatchEffect, Normalization, Preprocessing, Visualization

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

git_url https://git.bioconductor.org/packages/CrossICC

git_branch RELEASE_3_11

git_last_commit 97414a7

git_last_commit_date 2020-04-27

Date/Publication 2020-10-16

Author Yu Sun [aut, cre] (<https://orcid.org/0000-0003-4269-7187>), Qi Zhao [aut] (<https://orcid.org/0000-0002-8683-6145>)

Maintainer Yu Sun <suny226@mail2.sysu.edu.cn>

R topics documented:

`Cal.ARI` *Title Adjust Rank Index*

Description

Title Adjust Rank Index

Usage

```
Cal.ARI(df, col1, col2)
```

Arguments

`df`	input data frame
`col1`	name of interest variable 1 column in df
`col2`	name of interest variable 2 column in df

Value

adjust ARI value

`centroidOfcentroid` *Return centroid of centroid from each platform*

Description

Return centroid of centroid from each platform

Usage

```
centroidOfcentroid(centroid.list, cluster)
```

Arguments

`centroid.list`	a list stored the centroid
`cluster`	a named vector with gene names as name and the cluster number as vector value

Value

a list contains a vecter that store the predict clusters and a normalized expression matrix

---

`CrossICC` *CrossICC: Automatically Aggregating and Summarizing Bioinformatics Results for Interactive Report.*

Description

CrossICC: Automatically Aggregating and Summarizing Bioinformatics Results for Interactive Report.

The Main Function of the package

Usage

```
CrossICC(..., study.names, filter.cutoff = 0.5, fdr.cutoff = 0.001,
  output.dir = "~", max.K = 10, max.iter = 20, rep.runs = 1000,
  n.platform = 2, pItem = 0.8, pFeature = 1, clusterAlg = "hc",
  distance = "euclidean", sil.filter = "soft",
  heatmap.order = "up.based", com.mode = "overlap", cc.seed = NULL,
  cluster.cutoff = 0.05, ebayes.cutoff = 0.1, ebayes.mode = "up",
  cross = "cluster", supercluster.method = "hclust",
  skip.merge.dup = TRUE, skip.mm = FALSE, skip.mfs = FALSE,
  use.shiny = FALSE, overwrite = FALSE)
```

Arguments

`...`	all datasets (matrices is better)
`study.names`	a vector containing all study names
`filter.cutoff`	low variability (median absolute deviation (MAD)) cutoff threshold, default is 0.5.
`fdr.cutoff`	cutoff value during fdr filtering.
`output.dir`	the results' output directory.
`max.K`	the maximum cluster number of ConsensusClusterPlus. Default is 10, but was set as number of samples when there're less than 10 samples.
`max.iter`	the maximum number of iterations.
`rep.runs`	number of subsamples during clustering.
`n.platform`	to filter the signature with it's meta-cluster group in platforms. That is, if the parameter is set to 2 (default), the signature (like hgnc symbol ESR1) in a certain meta-cluser (like K1) must exists more than 2 times among data of all platforms; otherwise, it will not be reported.
`pItem`	proportion of items to sample during clustering.

`pFeature`	proportion of features to sample during clustering.
`clusterAlg`	cluster algorithm. Could be 'hc' heirarchical (hclust), 'pam' for paritioning around medoids, 'km' for k-means upon data matrix, 'kmdist' for k-means upon distance matrices (former km option), or a function that returns a clustering.
`distance`	Could be 'pearson': (1 - Pearson correlation), 'spearman' (1 - Spearman correlation), 'euclidean', 'canberra', 'minkowski" or custom distance function.
`sil.filter`	silhouetee width filtering mode. Could be "soft" or "hard". If "hard", all negtive silhouetee width value will be set to 0. Default is "soft" (to do nothing).
`heatmap.order`	gene order for heatmaps. Default is "up.based", with which genes will be arranged as up-regulated order in meta-clusters across all matrices. Or can be set to "concordant" for all in same order.
`com.mode`	mode for choose common features when pre-processing data. Could be "overlap" (use intersection, default) or "merge" (keep all features).
`cc.seed`	sets random seed for reproducible results.
`cluster.cutoff`	cutoff value during determining cluster numbers.
`ebayes.cutoff`	p-value cutoff when select differentially expressed probes.
`ebayes.mode`	'up' or 'both'. Choose only up-regulated genes or all differentially expressed genes when determining MDEGs. default is 'up'
`cross`	object type when determining meta-cluster. Could be "cluster" for clusters by ConsencusClusterPlus, "sample" for samples or "none" (only used for single dataset).
`supercluster.method`	method for super-clustering. Default is 'hclust', can also be 'kmeans'.
`skip.merge.dup`	skip merge multiple probes for one gene (duplicates) or not. Default is TRUE (it is highly recommended that user has their data pre-processed well).
`skip.mm`	skip MergeMaid processing or not. Default is FALSE (not skip).
`skip.mfs`	by default, the datasets will be normalized at the start, and the genes or features that have no or few contributions to the final clusters will be filtered out. To skip this process, you can set this parameter to TRUE. Only try when you're sure that you're working with pre-processed datasets.
`use.shiny`	if TRUE, a shiny app will appear after running this main function. Note: You must keep output.dir with default value '~' for using shiny app.
`overwrite`	if user allow overwrite result file? Default is FALSE.

Value

A nested list with iteration time as its name and list containing consensus cluster, gene signature and balanced cluster as its value.

See Also

`ConsensusClusterPlus`

Examples

```
data("demo.platforms")
CrossICC.obj <- CrossICC(demo.platforms, skip.mfs = TRUE, max.iter = 1, overwrite = TRUE, output.dir = tempdir(
```

`CrossICCInput` *Read file into CrossICC input*

Description

Read file into CrossICC input

Usage

```
CrossICCInput(files)
```

Arguments

`files`	a list for filenames, usually a returned value of list.files() function

Value

list contains matrices from each platform parsing from file provided .

Examples

```
files <- list.files(path=".", pattern = '.csv')
CrossICC.input <- CrossICCInput(files)
```

`demo.platforms` *list containing several different (eSet) matrix.*

Description

each matrix comes from GSE file with feature names.

Usage

```
demo.platforms
```

Format

Note: this is format of single matrix. A matrix with rows of features (genes) and column of samples:

sample1 GSM*

sample2 GSM* ...

`get_jarrad_index_df_fromDF` *Title get jaccard index of list factors*

Description

Title get jaccard index of list factors

Usage

```
get_jarrad_index_df_fromDF(df1, df2, universe = NULL)
```

Arguments

`df1`	an annotated data frame with cluster at the seccond column
`df2`	an annotated data frame with cluster at the seccond column
`universe`	(Optional) total number of all strings that vec1 and vec2 comes from

Value

a data frame of Jaccard index or a list contains two dataframe (jaccard index and Fisher's test P value list)

`get_jarrad_index_df_fromlist` *Title get jaccard index of list factors*

Description

Title get jaccard index of list factors

Usage

```
get_jarrad_index_df_fromlist(list1, list2, universe = NULL)
```

Arguments

`list1`	a list that contains a lot vectors
`list2`	a list that contains a lot vectors
`universe`	(Optional) total number of all strings that vec1 and vec2 comes from

Value

a data frame of Jaccard index or a list contains two dataframe (jaccard index and Fisher's test P value list) get_overlap_test_by_fisher *Title Get fihser test p value for overlaps*

Description

Title Get fihser test p value for overlaps

Usage

```
get_overlap_test_by_fisher(vec1, vec2, universe)
```

Arguments

vec1	a string vector
vec2	a string vector
universe	total number of all strings that vec1 and vec2 comes from

Value

a P value

predictor *To calculate the correlation between the predictor centroid and the validation centroid.*

Description

To calculate the correlation between the predictor centroid and the validation centroid.

Usage

```
predictor(pre.dat, model)
```

Arguments

pre.dat	a eSet object or eSet-like matrix with features in rows and samples in columns
model	a list containing CrossICC result

Value

a list contains a vecter that store the predict clusters and a normalized expression matrix

Examples

```
data("demo.platforms")
CrossICC.object <- CrossICC(demo.platforms, skip.mfs = TRUE, max.iter = 1, overwrite = TRUE, output.dir = tempd
predicted <- predictor(demo.platforms[[1]], CrossICC.object)
```

rand.index *Title Adjust Rank Index*

Description

Title Adjust Rank Index

Usage

```
rand.index(df, col1, col2)
```

Arguments

`df`	input data frame
`col1`	name of interest variable 1 column in df
`col2`	name of interest variable 2 column in df

Value

adjust ARI value

ssGSEA *To get GSEA-like ranked matrix from CrossICC result.*

Description

To get GSEA-like ranked matrix from CrossICC result.

Usage

```
ssGSEA(x, gene.signature, geneset2gene, cluster)
```

Arguments

`x`	a eSet object or eSet-like matrix.
`gene.signature`	gene signatures calculated by CrossICC.
`geneset2gene`	a matrix contains geneset (cluster name) mapping to gene.
`cluster`	CrossICC returned clusters. Note: Must mapping to x!

Value

a matrix with samples' eigenvalue in different super clusters.

Examples

```
data("demo.platforms")
CrossICC.object <- CrossICC(demo.platforms, skip.mfs = TRUE, max.iter = 1, overwrite = TRUE, output.dir = tempd
Mcluster <- paste("K", CrossICC.object$clusters$clusters[[1]], sep = "")
CrossICC.ssGSEA <- ssGSEA(x = demo.platforms[[1]], gene.signature = CrossICC.object$gene.signature,
geneset2gene = CrossICC.object$unioned.genesets, cluster = Mcluster)
```

`summaryCrossICC` *Summary the CrossICC-returned list to produce human-readable output*

Description

Summary the CrossICC-returned list to produce human-readable output

Usage

```
summaryCrossICC(result)
```

Arguments

`result`	list-type CrossICC's return value.

Value

list contains: a matrix of genesets mapping to genes; a named atomic vetor of samples mapping to super clusters.

Examples

```
data("demo.platforms")
CrossICC.object <- CrossICC(demo.platforms, skip.mfs = TRUE, max.iter = 1, overwrite = TRUE, output.dir = tempd
CrossICC.summary <- summaryCrossICC(CrossICC.object)
```

Index