Package 'GNET2'

October 16, 2019

Type Package

Title Constructing gene regulatory networks from expression data through functional module inference

Version 1.0.0

Author Chen Chen, Jie Hou and Jianlin Cheng

Maintainer Chen Chen <ccm3x@mail.missouri.edu>

Description Cluster genes to functional groups with E-M process. Iteratively perform TF assigning and Gene assigning, until the assignment of genes did not change, or max number of iterations is reached.

License Apache License 2.0

Encoding UTF-8

LazyData true

LinkingTo Rcpp

Depends R (>= 3.5)

Imports ggplot2,xgboost,Rcpp,reshape2,grid,DiagrammeR,methods,stats,matrixStats,graphics,SummarizedExperiment

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

biocViews GeneExpression, Regression, Network, NetworkInference, Software

URL https://github.com/chrischen1/GNET2

BugReports https://github.com/chrischen1/GNET2/issues

git_url https://git.bioconductor.org/packages/GNET2

git_branch RELEASE_3_9

git_last_commit 756b7d1

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

`build_module` *Fit a regression tree.*

Description

Fit a regression tree based on Gaussian Likelihood score. Provided in case the best split is not applicable for R dnorm() function.

Usage

```
build_module(X, Y, max_depth, cor_cutoff, min_divide_size)
```

Arguments

`X`	A n by p matrix as input.
`Y`	A n by q matrix as response.
`max_depth`	Maximum depth of the tree.
`cor_cutoff`	Cutoff for within group Pearson correlation coefficient, if all data belong to a node have average correlation greater or equal to this, the node would not split anymore.
`min_divide_size`	Minimum number of data belong to a node allowed for further split of the node.

Value

A matrix for sample informatrion for each partition level. First column is feature index used by the node and second is the value used to split, the rest of the columns are the split of sample: 0 means less or equal, 1 means greater and -1 means the sample does not belong to this node.

Examples

```
build_module(X = matrix(rnorm(5*10),5,10), Y = matrix(rnorm(5*10),5,10),
                        max_depth=3,cor_cutoff=0.9,min_divide_size=3)
```

build_moduleR *Build regression tree.*

Description

Build regression tree based on Gaussian Likelihood score.

Usage

```
build_moduleR(X, Y, max_depth, cor_cutoff, min_divide_size)
```

Arguments

X	A n by p matrix as input.
Y	A n by q matrix as response.
max_depth	Maximum depth of the tree.
cor_cutoff	Cutoff for within group Pearson correlation coefficient, if all data belong to a node have average correlation greater or equal to this, the node would not split anymore.
min_divide_size	Minimum number of data belong to a node allowed for further split of the node.

Value

A matrix for sample informatrion for each tree level. First column is feature index used by the node and second is the value used to split, the rest of the columns are the split of sample: 0 means less or equal, 1 means greater and -1 means the sample does not belong to this node.

Examples

```
build_moduleR(X = matrix(rnorm(5*10),5,10), Y = matrix(rnorm(5*10),5,10),
                            max_depth=3,cor_cutoff=0.9,min_divide_size=3)
```

build_moduleR_heuristic

Build regression tree with splits are detemined by K-means heuristicly.

Description

Build regression tree based on Gaussian Likelihood score.

Usage

```
build_moduleR_heuristic(X, Y, max_depth, cor_cutoff, min_divide_size)
```

Arguments

X	A n by p matrix as input.
Y	A n by q matrix as response.
max_depth	Maximum depth of the tree.
cor_cutoff	Cutoff for within group Pearson correlation coefficient, if all data belong to a node have average correlation greater or equal to this, the node would not split anymore.
min_divide_size	Minimum number of data belong to a node allowed for further split of the node.

Value

A matrix for sample informatrion for each tree level. First column is feature index used by the node and second is the value used to split, the rest of the columns are the split of sample: 0 means less or equal, 1 means greater and -1 means the sample does not belong to this node.

Examples

```
build_moduleR_heuristic(X = matrix(rnorm(5*10),5,10), Y = matrix(rnorm(5*10),5,10),
                            max_depth=3,cor_cutoff=0.9,min_divide_size=3)
```

---

calc_likelihood_score *Calculate Gaussian Likelihood score.*

Description

Calculate Gaussian Likelihood score.

Usage

```
calc_likelihood_score(x, labels)
```

Arguments

x	A n by p matrix.
labels	A vector of length n, indicating the group of rows.

Value

The sum of log likelihood score of each group on each column.

Examples

```
calc_likelihood_score(x = matrix(rnorm(5*10),5,10), labels = c(rep(1,2),rep(2,3)))
```

get_correlation_list *Calculate correlation within each group.*

Description

Calculate Pearson correlation coefficient within each group.

Usage

```
get_correlation_list(x, labels)
```

Arguments

x	A n by p matrix.
labels	A vector of length n, indicating the group of rows.

Value

An array of Pearson correlation coefficient for each row, rows belong to the same group have same values.

Examples

```
get_correlation_list(x = matrix(rnorm(5*10),5,10), labels = c(rep(1,2),rep(2,3)))
```

gnet *Run GNET2*

Description

Build regulation modules by iteratively perform regulator assigning and Gene assigning, until the assignment of genes did not change, or max number of iterations reached.

Usage

```
gnet(input, reg_names, init_method = "boosting", init_group_num = 4,
  max_depth = 3, cor_cutoff = 0.9, min_divide_size = 3,
  min_group_size = 2, max_iter = 5, heuristic = FALSE)
```

Arguments

input	A SummarizedExperiment object, or a p by n matrix of expression data of p genes and n samples, for example log2 RPKM from RNA-Seq.
reg_names	A list of potential upstream regulators names, for example a list of known transcription factors.
init_method	Cluster initialization, can be "boosting" or "kmeans", default is using "boosting".
init_group_num	Initial number of function clusters used by the algorithm.
max_depth	max_depth Maximum depth of the tree.

cor_cutoff	Cutoff for within group Pearson correlation coefficient, if all data belong to a node have average correlation greater or equal to this, the node would not split anymore.
min_divide_size	Minimum number of data belong to a node allowed for further split of the node.
min_group_size	Minimum number of genes allowed in a group.
max_iter	Maxumum number of iterations allowed if not converged.
heuristic	If the splites of the regression tree is determined by k-means heuristicly.

Value

A list of expression data of genes, expression data of regulators, within group score, table of tree structure and final assigned group of each gene.

Examples

```
set.seed(1)
init_group_num = 8
init_method = 'boosting'
exp_data <- matrix(rnorm(50*10),50,10)
reg_names <- paste0('TF',1:5)
rownames(exp_data) <- c(reg_names,paste0('gene',1:(nrow(exp_data)-length(reg_names))))
colnames(exp_data) <- paste0('condition_',1:ncol(exp_data))
se <- SummarizedExperiment::SummarizedExperiment(assays=list(counts=exp_data))
gnet_result <- gnet(se,reg_names,init_method,init_group_num)
```

---

kneepointDetection *Knee point detection.*

Description

Detect the knee point of the array.

Usage

```
kneepointDetection(vect)
```

Arguments

vect	A list of sorted numbers.

Value

The index of the data point which is the knee.

Examples

```
kneepointDetection(sort(c(runif(10,1,3),c(runif(10,5,10))),TRUE))
```

plot_gene_group *Plot a module*

Description

Plot the regulators module and heatmap of the expression inferred downstream genes for each sample. It can be interpreted as two parts: the bars at the top shows how samples are splited by the regression tree and the heatmap at the bottom shows how downstream genes are regulated by each subgroup determined by the regulators.

Usage

```
plot_gene_group(gnet_result, group_idx)
```

Arguments

gnet_result	Results returned by gnet().
group_idx	Index of the module.

Value

None

Examples

```
set.seed(1)
init_group_num = 5
init_method = 'boosting'
exp_data <- matrix(rnorm(50*10),50,10)
reg_names <- paste0('TF',1:5)
rownames(exp_data) <- c(reg_names,paste0('gene',1:(nrow(exp_data)-length(reg_names))))
colnames(exp_data) <- paste0('condition_',1:ncol(exp_data))
se <- SummarizedExperiment::SummarizedExperiment(assays=list(counts=exp_data))
gnet_result <- gnet(se,reg_names,init_method,init_group_num)
plot_gene_group(gnet_result,group_idx=0)
```

plot_group_correlation *Plot the correlation of each group*

Description

Plot the correlation of each group and auto detected knee point. It can be used to determined which clustered are kept for further analysis.

Usage

```
plot_group_correlation(gnet_result)
```

Arguments

gnet_result	Results returned by gnet().

Value

A list of indices of the data point with correlation higher than the knee point.

Examples

```
set.seed(1)
gnet_result <- list('group_score'=c(runif(10,1,3),c(runif(10,5,3))))
group_keep <- plot_group_correlation(gnet_result)
```

---

plot_tree *Plot the regression tree.*

Description

Plot the regression tree given the index of a module.

Usage

```
plot_tree(gnet_result, group_idx)
```

Arguments

gnet_result	Results returned by gnet().
group_idx	Index of the module.

Value

None

Examples

```
set.seed(1)
init_group_num = 5
init_method = 'boosting'
exp_data <- matrix(rnorm(50*10),50,10)
reg_names <- paste0('TF',1:5)
rownames(exp_data) <- c(reg_names,paste0('gene',1:(nrow(exp_data)-length(reg_names))))
colnames(exp_data) <- paste0('condition_',1:ncol(exp_data))
se <- SummarizedExperiment::SummarizedExperiment(assays=list(counts=exp_data))
gnet_result <- gnet(se,reg_names,init_method,init_group_num)
plot_tree(gnet_result,group_idx=0)
```

Index