Package: MMINP (via r-universe)

November 6, 2024

Title Microbe-Metabolite Interactions-Based Metabolic Profiles Predictor

Version 0.1.1

Description Implements a computational framework to predict microbial community-based metabolic profiles with 'O2PLS' model. It provides procedures of model training and prediction. Paired microbiome and metabolome data are needed for modeling, and the trained model can be applied to predict metabolites of analogous environments using new microbial feature abundances.

Depends R (>= 4.1.0)

Imports magrittr (>= 2.0.1), OmicsPLS (>= 2.0.2), utils, stats, forecast

Suggests rmarkdown, knitr, prettydoc, testthat (>= 3.0.0)

License GPL (>= 3.0)

URL https://github.com/YuLab-SMU/MMINP

BugReports https://github.com/YuLab-SMU/MMINP/issues

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Config/testthat/edition 3

VignetteBuilder knitr

Config/pak/sysreqs libssl-dev

Repository https://yulab-smu.r-universe.dev

RemoteUrl https://github.com/yulab-smu/mminp

RemoteRef HEAD

RemoteSha 77359aeb9af3419a9c93a186f1717d1326628a62

Contents

`calOrthVIP` *Calculation of the orthogonal variable influence on projection*

Description

Calculation of the orthogonal variable influence on projection

Usage

```
calOrthVIP(SSDAO, SSD, loading)
```

Arguments

`SSDAO`	a value of sum of squares (SSDao in step2) for each deflated matrix
`SSD`	the sum of square values
`loading`	the normalized loading matrices

calPredVIP *Calculation of the predictive variable influence on projection*

Description

Calculation of the predictive variable influence on projection

Usage

```
calPredVIP(SSXAP, SSYAP, SSD, loading)
```

Arguments

SSXAP	the sum of squares values of deflated X matrix for the predictive VIPO2PLS
SSYAP	the sum of squares values of deflated Y matrix for the predictive VIPO2PLS
SSD	the sum of square values
loading	the normalized loading matrices

checkInputdata *Check if input data satisfies input conditions*

Description

This function throws an error if x is not a numeric matrix or a data frame with all numeric-alike variables, or if any elements of x is NA.

Usage

```
checkInputdata(x)
```

Arguments

x	A matrix or data frame.

Value

No return value

`compareFeatures` *Compare features' abundance obtained by prediction and measurement.*

Description

Compare features' abundance obtained by prediction and measurement.

Usage

```
compareFeatures(
  predicted,
  measured,
  method = "spearman",
  adjmethod = "fdr",
  rsignif = 0.3,
  psignif = 0.05
)
```

Arguments

`predicted`	A matrix or data frame. The feature table obtained by prediction.
`measured`	A matrix or data frame. The feature table obtained by measurement. The abundances are expected to be normalized (i.e. proportion) or be preprocessed by `MMINP.preprocess`.
`method`	A character string indicating which correlation coefficient is to be used for the `cor.test`. One of "pearson", "kendall", or "spearman", can be abbreviated.
`adjmethod`	A character string indicating correction method (`p.adjust`). One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none", can be abbreviated.
`rsignif`	A numeric ranging from 0 to 1, the minimum correlation coefficient of features which considered as well-predicted features.
`psignif`	A numeric ranging from 0 to 1, the maximum adjusted p value of features which considered as well-predicted features.

Value

A list containing a table of correlation results and a vector of well-predicted features.

`filterFeatures` *Filter features of input table according to prevalence and/or abundance*

Description

Filter features of input table according to prevalence and/or abundance.

Usage

```
filterFeatures(x, prev = NA, abund = NA)
```

Arguments

x	A matrix or data frame.
prev	A numeric ranging from 0 to 1, the minimum prevalence of features to be retained. If set to NA, means no need to filter prevalence.
abund	A numeric greater than 0, the minimum abundance (mean) of features to be retained. If set to NA, means no need to filter abundance.

Value

A filtered feature table will be returned.

Examples

```
data(train_metag)
d <- filterFeatures(train_metag, prev = 0.8)
dim(train_metag)
dim(d)
```

`get_Components` *Estimate components for O2-PLS method*

Description

get components number using Cross-validate procedure of O2-PLS

Usage

```
get_Components(
  metag,
  metab,
  compmethod = NULL,
  n = 1:10,
  nx = 0:5,
  ny = 0:5,
  seed = 1234,
  nr_folds = 3,
  nr_cores = 1
)
```

Arguments

metag	Training data of sequence features' relative abundances. Must have the exact same rows (subjects/samples) as metab.
metab	Training data of metabolite relative abundances. Must have the exact same rows (subjects/samples) as metag.
compmethod	A character string indicating which Cross-validate procedure of O2PLS is to be used for estimating components, must be one of "NULL", "cvo2m" or "cvo2m.adj". If set to "NULL", depends on the features number.
n	Integer. Number of joint PLS components. Must be positive. More details in crossval_o2m and crossval_o2m_adjR2.
nx	Integer. Number of orthogonal components in metag. Negative values are interpreted as 0. More details in crossval_o2m and crossval_o2m_adjR2.
ny	Integer. Number of orthogonal components in metab. Negative values are interpreted as 0. More details in crossval_o2m and crossval_o2m_adjR2.
seed	a random seed to make the analysis reproducible, default is 1234.
nr_folds	Positive integer. Number of folds to consider. Note: kcv=N gives leave-one-out CV. Note that CV with less than two folds does not make sense. More details in crossval_o2m and crossval_o2m_adjR2.
nr_cores	Positive integer. Number of cores to use for CV. You might want to use detectCores(). Defaults to 1. More details in crossval_o2m and crossval_o2m_adjR2.

Value

A data frame of components number

`get_cvo2mComponent` *get components number from Cross-validate procedure of O2PLS*

Description

get components number from Cross-validate procedure of O2PLS

Usage

```
get_cvo2mComponent(x)
```

Arguments

`x`	List of class "cvo2m", produced by `crossval_o2m`.

Value

A data frame of components number

`MMINP.predict` *Predict metabolites from new microbiome samples using MMINP model.*

Description

This function aims to predict potentially metabolites in new microbial community using trained MMINP model. If genes in model are not appear in newdata, then this procedure will fill them up with 0. Note that this function does not center or scale the new microbiome matrixs, you would better do preprocessing on newdata in advance.

Usage

```
MMINP.predict(model, newdata, minGeneSize = 0.5)
```

Arguments

`model`	List of class `"mminp"` or `"o2m"`, produced by `MMINP.train` or `o2m`.
`newdata`	New matrix of microbial genes, each column represents a gene.
`minGeneSize`	A numeric between 0-1, minimal size of genes in model contained in newdata.

Details

The model must be class 'mminp' or 'o2m'. The column of newdata must be microbial genes.

Value

Predicted Data

Examples

```
data(MMINP_trained_model)
data(test_metag)
test_metag_preprocessed <- MMINP.preprocess(test_metag, normalized = FALSE)
pred_metab <- MMINP.predict(model = MMINP_trained_model$model,
newdata = test_metag_preprocessed)
```

MMINP.preprocess *Data Preprocessing function for MMINP*

Description

Before doing MMINP analysis, abundances of both microbial features and metabolites should be preprocessed. Both measurements are expected to be transformed to relative abundance (i.e. proportion) and be log-transformed. To meet the need of O2-PLS method, data must be scaled.

Usage

```
MMINP.preprocess(
  data,
  normalized = TRUE,
  prev = NA,
  abund = NA,
  transformed = "none",
  scaled = TRUE
)
```

Arguments

data	A numeric matrix or data frame containing measurements of metabolites or microbial features.
normalized	Logical, whether to transform measurements into relative abundance or not.
prev	A numeric ranging from 0 to 1, the minimum prevalence of features to be retained. If set to NA, means no need to filter prevalence.
abund	A numeric greater than 0, the minimum abundance (mean) of features to be retained. If set to NA, means no need to filter abundance.
transformed	character, select a transformation method: "boxcox", "log", or "none".
scaled	Logical, whether scale the columns of data or not.

Details

The rows of data must be samples and columns of data must be metabolites or microbial features. The filtering process (prev and abund) is before log/boxcox transformation and scale transformation.

Value

A preprocessed numeric matrix for analysis of MMINP.

Examples

```
data(train_metag)
d <- MMINP.preprocess(train_metag)
d <- MMINP.preprocess(train_metag, prev = 0.3, abund = 0.001)
d[1:5, 1:5]
```

MMINP.train *Train MMINP model using paired microbial features and metabolites data*

Description

This function contains three steps. Step1, Build an O2-PLS model and use it to predict metabolites profile; Step2, Compare predicted and measured metabolites abundances, then filter those metabolites which predicted poorly (i.e. metabolites of which correlation coefficient less than rsignif or adjusted pvalue greater than psignif.); Step3, (iteration) Re-build O2-PLS model until all reserved metabolites are well-fitted.

Usage

```
MMINP.train(
  metag,
  metab,
  n = 1:3,
  nx = 0:3,
  ny = 0:3,
  seed = 1234,
  compmethod = NULL,
  nr_folds = 3,
  nr_cores = 1,
  rsignif = 0.4,
  psignif = 0.05,
  recomponent = FALSE
)
```

Arguments

metag	Training data of sequence features' relative abundances. Must have the exact same rows (subjects/samples) as metab.
metab	Training data of metabolite relative abundances. Must have the exact same rows (subjects/samples) as metag.
n	Integer. Number of joint PLS components. Must be positive. More details in crossval_o2m and crossval_o2m_adjR2.
nx	Integer. Number of orthogonal components in metag. Negative values are interpreted as 0. More details in crossval_o2m and crossval_o2m_adjR2.
ny	Integer. Number of orthogonal components in metab. Negative values are interpreted as 0. More details in crossval_o2m and crossval_o2m_adjR2.
seed	a random seed to make the analysis reproducible, default is 1234.
compmethod	A character string indicating which Cross-validate procedure of O2PLS is to be used for estimating components, must be one of "NULL", "cvo2m" or "cvo2m.adj". If set to "NULL", depends on the features number.
nr_folds	Positive integer. Number of folds to consider. Note: kcv=N gives leave-one-out CV. Note that CV with less than two folds does not make sense. More details in crossval_o2m and crossval_o2m_adjR2.
nr_cores	Positive integer. Number of cores to use for CV. You might want to use detectCores(). Defaults to 1. More details in crossval_o2m and crossval_o2m_adjR2.
rsignif	A numeric ranging from 0 to 1, the minimum correlation coefficient of features which considered as well-predicted features.
psignif	A numeric ranging from 0 to 1, the maximum adjusted p value of features which considered as well-predicted features.
recomponent	Logical, whether re-estimate components or not during each iteration.

Value

A list containing

model	O2PLS model
trainres	Final correlation results between predicted and measured metabolites of training samples
WFM	Well-fitted metabolites
components	Components number. If recomponent = TRUE, the components number is the result of last estimation.
re_estimate	Re-estimate information, i.e. whether re-estimate components or not during each iteration
trainnumb	Iteration number

Examples

```
data(test_metab)
data(test_metag)
a <- MMINP.preprocess(test_metag[, 1:20], normalized = FALSE)
b <- MMINP.preprocess(test_metab[, 1:20], normalized = FALSE)
mminp_model <- MMINP.train(metag = a,
                           metab = b,
                           n = 3:5, nx = 0:3, ny = 0:3,
                           nr_folds = 2, nr_cores = 1)
length(mminp_model$trainres$wellPredicted)
```

MMINP_trained_model *(Data) A MMINP model*

Description

This model was built using (MMINP.train) with preprocessed values in dataset $train_metag$ and $train_metab$.

Format

A list containing an 'o2m' model, results of correlation analysis between metabolites of training data and its predicted values, components number, re-estimate information and iteration number of modeling.

Examples

```
data(MMINP_trained_model)
```

O2PLSvip *Evaluate the importance of variables in O2PLS models*

Description

O2PLS-VIP, an approach for variable influence on projection (VIP) in O2PLS models, is a model-based method for judging the importance of variables. For both X and Y data blocks, it generates VIP profiles for (i) the predictive part of the model, (ii) the orthogonal part, and (iii) the total model.

Usage

```
O2PLSvip(x, y, model)
```

Arguments

x	Training data of sequence features' relative abundances. Must have the exact same rows (subjects/samples) as y.
y	Training data of metabolite relative abundances. Must have the exact same rows (subjects/samples) as x.
model	List of class "mminp" or "o2m", produced by MMINP.train or o2m. x and y must be the corresponding training data.

Details

It generates 6 VIPO2PLS profiles in total:

1. Two VIP profiles for the predictive components, which uncover the X- and Y-variables that are more important for the model interpretation in relation to the variation correlated to the Y- and X- data matrices respectively;
2. Two VIP profiles for the orthogonal components for both the X-block and the Y-block severally, profiles that uncover the X- and Y- variables that are more relevant in relation to the variation uncorrelated to the Y- and X- data matrices respectively;
3. Two VIP profiles for the total model (i.e. including the contributions of both predictive and orthogonal components) for both the X- and the Y- blocks severally, these VIP profiles point at the X- and Y- variables that are more significant for the whole model.

Value

A list containing

xvip	For the X-block, the VIP profiles for the predictive part of the model, the orthogonal part, the total model.
yvip	For the Y-block, the VIP profiles for the predictive part of the model, the orthogonal part, the total model.

References

Galindo-Prieto B, Trygg J, Geladi P. A new approach for variable influence on projection (VIP) in O2PLS models. Chemometrics and Intelligent Laboratory Systems 2017; 160: 110–124.

Examples

```
#' data(test_metab)
data(test_metag)
a <- MMINP.preprocess(test_metag[, 1:20], normalized = FALSE)
b <- MMINP.preprocess(test_metab[, 1:20], normalized = FALSE)
mminp_model <- MMINP.train(metag = a,
                           metab = b,
                           n = 3:5, nx = 0:3, ny = 0:3,
                           nr_folds = 2, nr_cores = 1)
length(mminp_model$trainres$wellPredicted)
vipres <- O2PLSvip(a, b, mminp_model)
head(vipres$xvip)
head(vipres$yvip)
```

print.mminp *Print function for MMINP.train*

Description

This function is the print method for `MMINP.train`.

Usage

```
## S3 method for class 'mminp'
print(x, ...)
```

Arguments

`x`	A model (an object of class "mminp")
`...`	additional parameters

Value

Brief information about the object.

ssd *estimation of the sum of squares of deviations*

Description

estimation of the sum of squares of deviations

Usage

```
ssd(x)
```

Arguments

`x`	matrix

Value

the sum of squares of deviations

test_metab *(Data) Normalized metabolite abundances for MMINP prediction*

Description

This datasets were built from NLIBD dataset (Franzosa et al., 2019) by converting original HMDB IDs into KEGG compound IDs and removing unassigned and repeated features.

Format

A data frame of metabolite relative abundances (i.e. proportion), with 65 subjects in rows and 130 KEGG compound IDs in columns.

References

Franzosa EA et al. (2019). Gut microbiome structure and metabolic activity in inflammatory bowel disease. Nature Microbiology 4(2):293-305.

Examples

```
data(test_metab)
```

test_metag *(Data) Normalized gene family abundances for MMINP prediction*

Description

This datasets were built from NLIBD dataset (Franzosa et al., 2019) by converting original UniRef90 IDs into KEGG Orthology (KO) IDs and removing unassigned and repeated features.

Format

A data frame of gene family relative abundances (i.e. proportion), with 65 subjects in rows and 629 KEGG Orthology (KO) IDs in columns.

References

Franzosa EA et al. (2019). Gut microbiome structure and metabolic activity in inflammatory bowel disease. Nature Microbiology 4(2):293-305.

Examples

```
data(test_metag)
```

train_metab *(Data) Normalized metabolite abundances for MMINP training*

Description

This datasets were built from PRISM dataset (Franzosa et al., 2019) by converting original HMDB IDs into KEGG compound IDs and removing unassigned and repeated features.

Format

A data frame of metabolite relative abundances (i.e. proportion), with 155 subjects in rows and 135 KEGG compound IDs in columns.

References

Franzosa EA et al. (2019). Gut microbiome structure and metabolic activity in inflammatory bowel disease. Nature Microbiology 4(2):293-305.

Examples

```
data(train_metab)
```

train_metag *(Data) Normalized gene family abundances for MMINP training*

Description

This datasets were built from PRISM dataset (Franzosa et al., 2019) by converting original UniRef90 IDs into KEGG Orthology (KO) IDs and removing unassigned and repeated features.

Format

A data frame of gene family relative abundances (i.e. proportion), with 155 subjects in rows and 733 KEGG Orthology (KO) IDs in columns.

References

Franzosa EA et al. (2019). Gut microbiome structure and metabolic activity in inflammatory bowel disease. Nature Microbiology 4(2):293-305.

Examples

```
data(train_metag)
```

Index