Package 'MetaMicrobiome'

January 30, 2019

Type Package

Title An R package for meta-analysis and visualization of Microbiome.

Version 1.0

Date 2019-01-04

Author Shuangbin Xu

Maintainer Shuangbin Xu <xshuangbin@163.com>

Description MetaMicrobiome was designed to performance the meta-analysis, visualization and the module building. MetaMicrobiome provides function for computing the count data for the measures of risk and a chi-squared test. the package also provides a function for creating forest plots for the results of the measures of risk. Moreover, the package also provides functions for the module building and testing based on the randomforest, and a function for the visualization for the result with the ROC curves.

Depends epiR,
metafor

Imports ggplot2,
dplyr,
tidyr,
caret,
pROC

License GPL (>= 2.0)

RoxygenNote 6.1.1

R topics documented:

`droptax` *Dropping Species with Few abundance and Few Occurrences*

Description

Drop species or features from the feature data frame that occur fewer than or equal to a threshold number of occurrences and fewer abundance than to a threshold abundance.

Usage

```
droptax(taxtab, rmode = FALSE, minocc = 0, minabu = 0)
```

Arguments

`taxtab`	dataframe; a dataframe of species (or features), default is (n_sample, n_feature).
`rmode`	boolean; whether transpose the taxtab, default is False.
`minocc`	numeric; the threshold number of occurrences to be dropped, if < 1.0, it will be the threshold ratios of occurrences, default is 0.
`minabu`	numeric: the threshold abundance, if fewer than the threshold will be dropped, default is 0.

Value

a list contained feature dataframe dropped, and the call, arguments.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
data <- read.csv(system.file("data", package="MetaMicrobiome", "Baxter_16_crc_genera_group.csv.gz"))
data$Group <- NULL
dim(data)
head(data)
newdat <- droptax(data, rmode=FALSE, minocc=0.2, minabu=0)
dim(newdat)
head(newdat)
```

get_mapply_predict_test *model predictions*

Description

predict the multi-test datasets with the results of the multi-models.

Usage

```
get_mapply_predict_test(study, dataset, models, classvariable, classtype)
```

Arguments

study	character, the names of the specified test dataset.
dataset	list, a list contained multi test datasets.
models	list, a list contained multi model object for predictions.
classvariable	character, the header name of the train of test dataframe for the classification.
classtype	character, the name of the positive group for classification.

Details

TODO

Author(s)

Shuangbin Xu

get_train_test_data *get train and test data*

Description

Build the train and test datasets with a multi-datasets.

Usage

```
get_train_test_data(test_study, datasets)
```

Arguments

test_study	character, the names of the list datasets.
datasets	list, a list contianed the multi-dataframes with features and target infromation.

Details

TODO

Value

a list contianed the test datasets and train datasets.

Author(s)

Shuangbin Xu

`getthresholds` *Creat the thresholds vector.*

Description

Creat the thresholds for the `high_low_vector`

Usage

```
getthresholds(dataset, var_of_interest, type = "median")
```

Arguments

`dataset`	dataframe, a dataframe contain interesting variable.
`var_of_interest`	character, a vector interesting variables
`type`	character, the method of choose thresholds, (median or mean), default is median

Details

TODO

Value

a vector thresholds of interesting variable.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
data <- read.csv(system.file("data",
         package="MetaMicrobiome",
      "Baxter_16_alpha_data.csv.gz"))
thresVetor <- getthresholds(dataset=data,
              var_of_interest=c("Shannon",
                                "Observe"),
              type="median")
```

`ggforest` *forest plot base ggplot2 with the result of RunPoolEffect and multiVarRRTab.*

Description

Polt a forest with the result of RunPoolEffect and multiVarRRTab base-on the `ggplot`.

Usage

```
ggforest(dataset, manualcolors, manualshapes, Logscale = TRUE, x, y,
  lower, upper, pointsize = 2, linesize = 0.4, errorbarhheight = 0.2,
  colorVar, shapeVar, facetx = NULL, facety = NULL, xlabs, ylabs,
  setTheme = TRUE)
```

Arguments

`dataset`	dataframe; a dataframe of result of RunPoolEffect and multiVarRRTab
`manualcolors`	character; the ponit colors.
`manualshapes`	character; the point shape.
`Logscale`	logical; log2 for x-axis (default) or not?
`x`	character; the header name in the dataframe for map to axes of point.
`y`	character; the header name in the dataframe for map to ayes of point.
`lower`	character; the header name in the dataframe for map to lower of the errorbar.
`upper`	character; the header name in the dataframe for map to upper of the errorbar.
`pointsize`	numeric; the point size, default is 3.0 .
`linesize`	numeric; the errorbar line size, default is 0.4 .
`errorbarhheight`	numeric; the errorbar height size, default is 0.2 .
`colorVar`	character; the header name in the dataframe for map to the color of point.
`shapeVar`	character; the header name in the dataframe for map to the shape of point.
`facetx`	character; the header name in the dataframe for map to the facet of row, default is NULL.
`facety`	character; the header name in the dataframe for map to the facet of col, default is NULL.
`xlabs`	character; label for the x-axis.
`ylabs`	character; label for the y-axis.
`setTheme`	logical; whether set the default theme.

Details

TODO

Value

Returns a ggplot object.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
data <- system.file("data", package="MetaMicrobiome", "ggforestDemo.rda")
load(data)
head(ggforestDemoData)
ggforestDemoData$study <- factor(ggforestDemoData$study,
                                 levels=rev(unique(ggforestDemoData$study)))
print(levels(ggforestDemoData$study))
pointcolors <- rev(c("#E41A1C",
                     "#4DAF4A",
    "#984EA3",
    "#FF7F00",
    "#FFFF33",
    "#A65628",
    "#F781BF",
    "#999999"))
pointshape <- c(18, 20)
data1 <- ggforestDemoData[ggforestDemoData$measure=="Shannon",]
head(data1)
data2 <- ggforestDemoData
head(data2)
p1 <- ggforest(dataset=data1,
      manualcolors=pointcolors,
      manualshapes=pointshape,
      x="est",
      y="study",
      Logscale=TRUE,
      lower="lower",
      upper="upper",
      colorVar="study",
      shapeVar="unite",
      pointsize=3,
      linesize=0.4,
      errorbarhheight=0.05,
      xlabs="Odds Ratio",
      ylabs="",
      setTheme=TRUE)

p2 <- ggforest(dataset=data2,
         manualcolors=pointcolors,
         manualshapes=pointshape,
         x="est",
         y="study",
         Logscale=TRUE,
         lower="lower",
         upper="upper",
         colorVar="study",
         shapeVar="unite",
         pointsize=3,
         linesize=0.4,
         errorbarhheight=0.05,
```

```
            xlabs="Odds Ratio",
            ylabs="",
            facetx="measure",
            facety="group",
            setTheme=TRUE)
```

---

high_low_vector *creat a vector with high/low versus the threshold.*

Description

Creat a vector with hith/low versus the threshold, suitable for analysis with [metafor]{rma}.

Usage

```
high_low_vector(var_of_interest, dataset, threshold)
```

Arguments

var_of_interest	character, the interesting variable names.
dataset	dataframe. a dataframe contain the inteeresting variable.
threshold	numeric, the threshold.

Details

TODO

Value

a vector with high/low versus the threshold.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
testfile <- system.file("data", package="MetaMicrobiome", "Baxter_16_alpha_data.csv.gz")
data <- read.csv(testfile, header=TRUE, check.names=FALSE)
thresVetor <- getthresholds(dataset=data,
c("Shannon", "Observe", "J"),
type="median")
highlowVector <- high_low_vector(dataset=data,
threshold=thresVetor,
var_of_interest="Shannon")
head(highlowVector)
```

`make_RF_model` *model training*

Description

build a model with training datasets.

Usage

```
make_RF_model(train_data, study, numtree = 500, number_try,
  numbercv = 10, classvariable)
```

Arguments

`train_data`	list or dataframe, the dataframe contianed the features or a list contianed multi-dataframe with features.
`study`	character,if train_data is a list, we can use 'study' extract the dataframe, default is NULL.
`numtree`	numeric, the number of trees for randomforest, see `[randomForest]` details.
`number_try`	numeric, the number of variables randomly sampled as candidates at each split, see `[randomForest]` details, default is 'round(sqrt(ncol(train_data)))'.
`numbercv`	numeric, the number of cross-validation.
`classvariable`	character, the header name of the train dataframe for the classification.

Details

TODO

Value

a model object, see `[randomForest]` details.

Author(s)

Shuangbin Xu

`MultiHighLow` *creat list of vectors with high/low versus the threshold.*

Description

Creat list of vetors with high/low versus the threshold, suitable for analysis with `[metafor]{rma}`

Usage

```
MultiHighLow(var_of_interest, dataset, threshold)
```

Arguments

var_of_interest	vector, the vector of interesting variables.
dataset	dataframe, the dataframe contained the interesting variables.
threshold	vector, the threshold values vector.

Details

TODO

Value

list of vector with high/low versus the threshold.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
testfile <- system.file("data", package="MetaMicrobiome", "Baxter_16_alpha_data.csv.gz")
data <- read.csv(testfile, header=TRUE, check.names=FALSE)
thresVetor <- getthresholds(dataset=data,
c("Shannon", "Observe", "J"),
type="median")
multiVariableHL <- MultiHighLow(var_of_interest=c("Shannon",
     "Observe",
     "J"),
dataset=data,
threshold=thresVetor)
head(multiVariableHL)
```

---

multiRunRR *Summary measures base on epiR for multi variables*

Description

Computes summary measures of risk and a chi-squared test for difference in the observed proportions from count data presented in a 2 by 2 table with high low vector. With multiple strata the function returns crude and Mantel-Haenszel adjusted measures of association and chi-squared tests of homogeneity for multi variables (based on [epiR]{epi.2by2}.

Usage

```
multiRunRR(multiHighLow = NULL, metadavector = NULL,
  prefix = "Disease", grouptype = "Case", method = "cohort.count",
  conf.level = 0.95, score = "OR.strata.score", ...)
```

Arguments

multiHighLow	list, mulit variable high-lower-vector.
metadavector	dataframe, metada dataframe contained the group information.
prefix	character, the header names of the metada data frame, default is 'Group'.
grouptype	character, the positive group names, default is "Case".
method	characer, a character string indicating the study design on which the tabular data has been based. Options are cohort.count, cohort.time, case.control, or cross.sectional, default is cohort.count. See epi.2by2 for details.
conf.level	numeric, magnitude of the returned confidence intervals. Must be a single number between 0 and 1. See epi.2by2 for details.
score	character, Wald and score confidence intervals for the effect value for each strata, default is OR.strata.score, See epi.2by2 for details .
...	Additional arguments passed to epi.2by2.

Details

TODO

Value

the summary measures of risk and a chi-squared test

Author(s)

ShuangbinXu

---

multiVarRRTab *collate results for summary measure of multi-variables*

Description

Collate results for summary measure of multi-variables.

Usage

```
multiVarRRTab(multiRunRRTab, var_of_interest)
```

Arguments

multiRunRRTab	list, the results of the
var_of_interest	vector, the interesting variables.

Details

TODO

Author(s)

Shuangbin Xu

predict_test *Model Predictions*

Description

predict the test datasets with the results of the models.

Usage

```
predict_test(model, teststudy, dataset, classvariable, classtype,
  Trainstudy)
```

Arguments

model	object, a model object for predict.
teststudy	character, the names of the dataset, if the dataset is a list.
dataset	list or dataframe, the test dataset.
classvariable	character, the header name of the train or test dataframe for the classification.
classtype	character, the name of the positive group for classification.
Trainstudy	chraracter, the name of the origin data of the model.

Details

TODO

Value

a list contianed the predict prob and roc results with sensitivity and specificity.

Author(s)

Shuangbin Xu

predict_train *models results*

Description

Predict the training datasets for a model

Usage

```
predict_train(model, study, minus = FALSE, classtype)
```

Arguments

model	object, a model object.
study	character, a names of training datasets.
minus	logical, whether minus the part of the ‘study‘, default is FALSE.
classtype	character, the name of the positive group for classificaion.

Details

TODO

Value

a dataframe for the roc curve plot.

Author(s)

Shuangbin Xu

`ROCplot` *plot the roc curve*

Description

Plot the ROC curve base on the `[ggplot2]`

Usage

```
ROCplot(rocplotdata, x, y, xlab, ylab, roccolors, legendkeyheight = 0.05,
  legendposition = c(0.67, 0.17), ...)
```

Arguments

`rocplotdata`	dataframe, a dataframe of result contained the sensitivity and specificity.
`x`	character, the header name in dataframe.
`y`	character, the header name in dataframe.
`xlab`	character, the label for the x-axis.
`ylab`	character, the label for the y-axis.
`roccolors`	vector, the colors for the roc curve.
`legendkeyheight`	the height of the legend, default is ‘unit(0.05, "mm")‘.
`legendposition`	vector, the position of legend, default is ‘c(0.67, 0.17)‘.
`...`	Additional arguments passed to `aes`

Details

TO DO

Value

Returns a ggplot object.

Author(s)

Shuangbin Xu

run_rr *Summary measures base on epiR*

Description

Computes summary measures of risk and a chi-squared test for difference in the observed proportions from count data presented in a 2 by 2 table with high low vector. With multiple strata the function returns crude and Mantel-Haenszel adjusted measures of association and chi-squared tests of homogeneity(based on [epiR]{epi.2by2}.

Usage

```
run_rr(var_high_low, metadavector, prefix = "Group",
  grouptype = "Case", method = "cohort.count", conf.level = 0.95,
  score = "OR.strata.score", ...)
```

Arguments

var_high_low	list or vector, the list of vectors with high/low versus the threshold
metadavector	dataframe, metada dataframe contained the group information.
prefix	character, the header names of the metada data frame, default is 'Group'.
grouptype	character, the positive group names, default is "Case".
method	characer, a character string indicating the study design on which the tabular data has been based. Options are cohort.count, cohort.time, case.control, or cross.sectional, default is cohort.count. See epi.2by2 for details.
conf.level	numeric, magnitude of the returned confidence intervals. Must be a single number between 0 and 1. See epi.2by2 for details.
score	character, Wald and score confidence intervals for the effect value for each strata, default is OR.strata.score, See epi.2by2 for details .
...	Additional arguments passed to epi.2by2.

Details

TODO

Value

the summary measures of risk and a chi-squared test

Author(s)

Shuangbin Xu

RunPoolEffect *Calcutate effect sizes via linear (Mixed-Effects) models*

Description

The function can be used to calculate various effect sizes. See [metafor]{rma} and [metafor]{escalc} details.

Usage

```
RunPoolEffect(var_of_interest, dataset, measure = "OR",
  methodtype = "REML")
```

Arguments

var_of_interest	vector, the interesting variables.
dataset	dataframe, the results of multiRunRR.
measure	character, a character string indicating which effect size or outcome measure should be calculated. See [metafor]{rma} and [metafor]{escalc} details.
methodtype	character, the specifying whether a fixed- or a random/mixed-effects model should be fitted, See [metafor]{rma} details.

Details

TODO

Value

a results dataframe of the pooled data with the random-effect model ot fixed-effect model. See the metafor{rma} details.

Author(s)

Shuangbin Xu

Examples

```
library("MetaMicrobiome")
study <- c("Baxter_16",
"Deng_18",
"Flemer_17",
       "Flemer_18",
       "Hale_17",
"Mori_18",
       "Zeller_15")
data <- lapply(study,
        function(x){read.csv(system.file("data",
                                         package="MetaMicrobiome",
                                         paste(x, "_alpha_data.csv.gz", sep=""))))})
names(data) <- study
```

```
threholds <- mapply(getthresholds, data,
                    MoreArgs=list(var_of_interest=c("Observe", "Shannon", "J"),
                    type="median"), SIMPLIFY=FALSE)
multiHighLowVector <- mapply(MultiHighLow,
                         data,
                         threholds,
                         MoreArgs=list(var_of_interest=c("Observe",
"Shannon",
"J")),SIMPLIFY=FALSE)

multiRRresult <- mapply(multiRunRR,
                    multiHighLowVector,
                    data,
                    MoreArgs=list(prefix="Group", grouptype="CRC"),
                    SIMPLIFY=FALSE)

multistudyRRresult <- mapply(multiVarRRTab,
                           multiRRresult,
                           MoreArgs=list(var_of_interest=c("Observe",
"Shannon", "J")),
                           SIMPLIFY=FALSE)
multistudyRRresult2 <- dplyr::bind_rows(lapply(study,
                                               function(x)
dplyr::mutate(multistudyRRresult[[x]], study=x)))

multistudyRRresult2
pooledREML <- dplyr::bind_rows(mapply(RunPoolEffect,
c("Observe", "Shannon", "J"),
                                      MoreArgs=list(dataset=multistudyRRresult2,
                                      methodtype="REML"),
                                      SIMPLIFY=FALSE))
head(pooledREML)
```

`tidy_data` *collate results for summary measure*

Description

Collate results for summary measure

Usage

```
tidy_data(multiRunRRTab, var_of_int)
```

Arguments

`multiRunRRTab`	list, the results of
`var_of_int`	vector, interesting variables

Details

TODO

Value

a results of dataframe for summary measure.

Author(s)

ShuangbinXu

Index