Package 'RcometsAnalytics'

January 7, 2022

Type Package

Title Comets Analytics for consortium based metabolomic analyses

Version 2.0.4.0

Author Ella Temprosa, Ewy Mathe

Maintainer Bill Wheeler <wheelerb@imsweb.com>

Description This R package support all cohort-specific analyses of the COMETS consortium. Data are not saved in the system but output must be downloaded and submitted for meta-analyses. This package can be used in several ways: supports the CBIIT implementation of comets-analytics, local shiny based add-in called shinycomets, or console based analysis. See example vignette. The version of the package is noted as: level 1 and 2 reflect the comets-analytics version, 3rd level reflects the number of major revision and 4th level for bug fixes.

Depends R (>= 3.5.0)

Imports readxl, rio, dplyr, plyr, plotly, tidyr, heatmaply, stringr, data.table, caret, subselect, broom, psych, MASS, ppcor, survival

Suggests Hmisc, knitr, testthat, rmarkdown

VignetteBuilder knitr

License GPL-3

RoxygenNote 7.1.2

NeedsCompilation no

URL http://comets-analytics.org/

BugReports https://github.com/CBIIT/nci-webtools-comets-analytics/issues

R topics documented:

`RcometsAnalytics-package`

RcometsAnalytics R package

Description

This R package supports all cohort-specific analyses of the COMETS consortium https://www.comets-analytics.org/. Data are not saved in the system but output must be downloaded and submitted for meta-analyses.

Details

Functions for analysis:
`runCorr` (correlation analysis)
`runModel` (correlation, glm, lm, or coxph)
`runAllModels` (run models in batch mode from models sheet)
Functions for graphics:
`plotVar` (metabolite variance distribution plot)
`plotMinvalues` (distribution of missing values)
`showHeatmap` (heat map of metabolite correlations)
`showHClust` (interactive heat map with hierarchical clustering)
Functions for saving results to files:
`OutputCSVResults` (write to .csv file)
`OutputXLSResults` (write to excel file)
`OutputListToExcel` (write list of data frames to excel file with multiple sheets)

clogit.options *options list for clogit*

Description

A list of 2:

- `method` One of: "exact", "approximate", "efron", "breslow". The default is "exact".
- `weights` A variable name to specify weights. The default is NULL.

Examples

```
model.options <- list(method="efron", weights="weightVarInData")
```

correlation.options *options list for* `model="correlation"`

Description

A list of 1:

- `method` Correlation method to use. It must be one of "spearman", "pearson", "kendall". The default value is "spearman".

Examples

```
model.options <- list(method="pearson")
```

coxph.options *options list for coxph*

Description

A list of 6:

- `ties` One of: "efron", "breslow", "exact". The default is "efron".
- `robust` TRUE or FALSE for computing a robust covariance matrix). The default is FALSE.
- `weights` A variable name to specify weights. The default is NULL.
- `singular.ok` See coxph. The default is TRUE.
- `Surv.type` See the `type` option in Surv. The default is NULL.

Examples

```
model.options <- list(robust=TRUE, weights="weightVarInData")
```

Effects *Effects table*

Description

The Effects data frame contains the estimates for each exposure, and will contain multiple rows for categorical exposure variables. Depending on the model run and options specified, all the below names may not appear in the data frame.

- `corr` The correlation between `term` and the outcome
- `estimate` The regression coefficient of `term`
- `estimate.lower` The lower confidence limit for `term`
- `estimate.upper` The upper confidence limit for `term`
- `exp.estimate` The exponentiated regression coefficient of `term`
- `exp.estimate.lower` The exponentiated lower confidence limit for `term`
- `exp.estimate.upper` The exponentiated upper confidence limit for `term`
- `exp.std.error` The standard error of `exp.estimate` from the delta method
- `exposurespec` Exposure variable
- `model` Model label from `getModelData`
- `model_number` Model number used in `runAllModels`
- `outcomespec` Outcome variable
- `pvalue` The p-value of the test
- `run` Run number that can be used to link with the `ModelSummary` table
- `statistic` The test statistic for `term`
- `std.error` The standard error of `estimate`
- `stratavar` Stratum variable(s)
- `strata` Stratum level(s)
- `term` Variable in the model

Details

Missing values will appear if a model did not converge, produced an error, or not run because of too many missing values in the outcome.

Errors_Warnings *Errors and Warnings table*

Description

Columns in the Errors_Warnings table.

- `type` WARNING or ERROR
- `object` The object that produced the warning or error. This is typically a variable or a particular stratum.
- `message` Message describing the warning or error
- `model` Model label from `getModelData`
- `model_number` Model number used in `runAllModels`

Details

The kinds of warnings and errors stored in this matrix are ones that apply to all models or all outcomes for an exposure variable. An error message for a particular exposure-outcome pair will be stored in the `message` column of the `ModelSummary` table.

filterCOMETSinput *Function that subsets input data based on "where variable"*

Description

Function that subsets input data based on "where variable"

Usage

```
filterCOMETSinput(readData, where = NULL)
```

Arguments

`readData`	list from readComets
`where`	users can specify which subjects to perform the analysis by specifying this parameter. 'where' expects a vector with a variable name, a comparison operator ("<", ">", "=","<=",">="), and a value. For example, "where = c("Gender=Female")".

Value

filtered list

getModelData *Prepares data for the models to be run as specified in the input. Can be run in interactive or batch mode. Each model is checked for validity (correlation between predictors, zero variance, etc.).*

Description

Prepares data for the models to be run as specified in the input. Can be run in interactive or batch mode. Each model is checked for validity (correlation between predictors, zero variance, etc.).

Usage

```
getModelData(
  readData,
  modelspec = "Batch",
  modlabel = "",
  outcomes = "All metabolites",
  exposures = "",
  adjvars = NULL,
  strvars = NULL,
  wgtvar = NULL,
  offvar = NULL,
  timevar = NULL,
  groupvar = NULL,
  where = NULL
)
```

Arguments

readData	List from readCOMETSinput
modelspec	How model is specified (Interactive or Batch). The default is Batch
modlabel	If batch, chosen model specified by batch mode (the MODEL column in the Models sheet). If interactive, then the model label.
outcomes	If Interactive, a vector of outcome variables (see details), the default is All metabolites)
exposures	If Interactive, a vector of exposure variables (see details)
adjvars	If Interactive, a vector adjustment covariates (see details)
strvars	If Interactive, stratification covariates (see details)
wgtvar	If Interactive, a variable of weights (see details)
offvar	If Interactive, an offset variable (see details)
timevar	If Interactive, time variable(s) for survival models (see details)
groupvar	If Interactive, a group variable for conditional logistic models (see details)
where	users can specify which subjects to perform the analysis on by specifying this parameter. 'where' expects a vector of strings with a variable name, a comparison operator (e.g. "<", ">", "<=", ">=", "!=", "="), and a value. For example, where = c("age>50","bmi > 22") uses all subjects with age > 50 AND bmi > 22. Note that when running in Batch mode, rules in the WHERE column of the Models sheet must be separated by a comma.

Details

All metabolite variables specified should be listed in the `metabolite_name` column of the `Metabolites` sheet of the Excel file. All non-metabolite variables should be listed in the `VARREFERENCE` column of the `VarMap` sheet. The `wgtvar`, `offvar`, and `timevar` are only used for specific models. See the `model` option in `options`.

Value

a list comprising:
1: subset data: gdta
2: exposure variables: ccovs
3: outcome variables: rcovs
4: adjustment variables: acovs
5: stratification variable: scovs
6: model specification: modspec
7: model label: modlab
8: whether all metabolites vs all metabolites is run: allvsall
9: weight variables: wgtcov
10: offset variables: offcov

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,modlabel="1 Gender adjusted")
```

`glm.options` *options list for glm*

Description

A list of 6:

- `family` One of: "binomial", "gaussian", "Gamma", "inverse.gaussian", "poisson", "quasi", "quasibinomial", "quasipoisson". The default is "gaussian".
- `link` NULL or a string for the link function to use (see `family`). The default is to use the canonical link for `family`.
- `weights` A variable name to specify weights. The default is NULL.
- `offset` A variable name to specify an offset. The default is NULL.
- `control` See `glm`. The default is `glm.control`.
- `singular.ok` See `glm`. The default is TRUE.

Examples

```
model.options <- list(family="binomial", weights="weightVarInData")
```

lm.options *options list for lm*

Description

A list of 4:

- `weights` A variable name to specify weights. The default is NULL.
- `offset` A variable name to specify an offset. The default is NULL.
- `tol` See `lm`. The default is 1e-7.
- `singular.ok` See `lm`. The default is TRUE.

Examples

```
model.options <- list(weights="weightVarInData")
```

ModelSummary *ModelSummary table*

Description

The ModelSummary data frame contains one row of model summary results for each exposure/outcome combination. Depending on the model run and options specified, all the below names may not appear in the data frame.

- `adjspec` Original adjustment variables specified
- `adjvars` Adjustment variables included in the model
- `adjvars.removed` Adjustment variables removed from the model
- `adj_uid` Adjustment variable universal ids
- `adj.r.squared` Adjusted R-squared
- `aic` Akaike information criterion
- `bic` Bayesian information criterion
- `cohort` String passed into `runModel`
- `converged` TRUE or FALSE for model convergence
- `deviance` Deviance of the fitted model
- `df.null` NULL model degrees of freedom
- `df.residual` Residual degrees of freedom
- `exposure` Exposure variable
- `exposure_uid` Exposure universal id
- `exposurespec` Exposure variable
- `loglik` Log-likelihood of the fitted model
- `message` Error message produced from the modeling function
- `model` Model label from `getModelData`

- model_function Model function used in runModel
- model_number Model number used in runAllModels
- nobs Number of observations used
- null.deviance Deviance of the NULL model
- outcome Outcome variable
- outcomespec Outcome variable
- outcome_uid Outcome universal id
- run Run number that can be used to link with the Effects table
- runmode "Batch" or "Interactive"
- r.squared R-squared, the fraction of variance explained by the model
- sigma Square root of the estimated variance of the random error
- stratavar Stratum variable(s)
- strata Stratum level(s)
- term Variable in the model
- wald.pvalue P-value from the Wald test of the given variable. Note that this test may be a multi-df test if the variable is categorical.

Details

Missing values will appear if a model did not converge, produced an error, or not run because of too many missing values in the outcome.

`options` *options list*

Description

A list of 16:

- check.cor.cutoff Cutoff value to remove highly correlated columns in the design matrix. The default value is 0.97.
- check.cor.method Correlation method to remove highly correlated columns in the design matrix. It must be one of "spearman", "pearson", "kendall". The default value is "spearman".
- check.design TRUE or FALSE to check the design matrix for linearly dependent columns, highly correlated columns, or for being ill-conditioned whenever it is updated. The default is TRUE.
- check.illCond TRUE or FALSE to check for an ill-conditioned design matrix. The default is TRUE.
- check.nsubjects Minimum number of subjects. The default is 25.
- max.nstrata The maximum number of strata for a stratified analysis. The default is 10.
- model String for the model function. Currently, it must be one of "correlation", "lm", "glm", "coxph", or "clogit". The default is "correlation".
- model.options List of options specific for the model. See correlation.options, glm.options, lm.options, coxph.options and clogit.options for options specific to model="correlation","lm","glm","coxph","clogit" respectively. The default is NULL.

- output.ci_alpha Confidence interval level for estimated from glm models. This option must be a number >= 0 and < 1, where 0 is for not creating confidence intervals. The default value is 0.95.
- output.Effects A string to define the terms output in the returned Effects and ModelSummary data frames. Currently, it must be "exposure" or "all". If set to "all", then summary statistics for the exposure and adjustment variables will be output. Otherwise, only summary statistics for the exposure will be output. This option is ignored with model = "correlation". The default is "exposure".
- output.exp_parms TRUE, FALSE or NULL to exponentiate glm parameter estimates. Standard errors are obtained from the delta method. The default is NULL, so that estimates from glm models with family="binomial" will be exponentiated, and not otherwise.
- output.metab.cols Character vector of column names in the METABOLITES sheet to be output in the ModelSummary and Effects data frames. Metabolite ids are matched first using the outcomespec column and then using the exposurespec column. The default is "metabolite_name".
- output.ModelSummary A string to defines the columns output in the returned ModelSummary data frame. Currently, it must be "anova" or "all". This option is ignored with model = "correlation". The default is "anova".
- output.type "rda" or "xlsx" to define the type of output file(s) when runAllModels is called. See output.common.cols and output.merge.
  The default is "xlsx".
- output.common.cols 0 or 1 to only output the common column names from the model results when the model results are merged together (output.merge != none). The default is 1.
- output.merge One of the following strings: "all", "by_function", "by_modelspec" or "none". This option is used to merge model results together in order to reduce the number of output files. Setting to "all" will merge all model results together and output them to a single file. Setting to "by_function" will merge results from the same model function together and output to a file with the model function contained in the output file name. Similarly for "by_modelspec", where the MODELSPEC column in the MODELS sheet of the input Excel file is used to identify the models that will be merged together. Setting to "none" will not merge results. The default is "none".

Details

Before any analysis is performed, an initial design matrix is constructed using the above options as follows:
1. Adjustment variables with less than two distinct non-missing values, or with less than check.nsubjects non-missing values are removed.
2. The design matrix is created from the remaining adjustment variables and any linearly dependent columns are removed.
3. If check.cor.cutoff > 0, then highly correlated columns are removed by computing the correlation matrix cor_matrix = cor(design_matrix,method=check.cor.method).
4. If check.illCond is TRUE, then the function subselect::trim.matrix is applied to the correlation matrix to determine if additional columns will be removed.

If check.design is TRUE, then steps 2-4 are repeated each time an exposure variable is added to the design matrix or when rows of the design matrix are removed due to missing values in an outcome variable.

Examples

```
# Logistic regression with all default options
```

```
model.op <- list(family="binomial")
op <- list(model="glm", model.options=model.op)
# Compute Pearson correlations requiring at least 100 subjects
model.op <- list(method="pearson")
op <- list(model="correlation", check.nsubjects=100, model.options=model.op)
```

`OutputCSVResults` *Create output CSV file*

Description

Create output CSV file

Usage

```
OutputCSVResults(filename, dataf, cohort = "")
```

Arguments

`filename`	name of CSV file and can include path
`dataf`	correlation output (from function runCorr())
`cohort`	cohort name

Value

the filename of the CSV file with results named with cohort

Examples

```
## Not run:
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,exposures="age",modlabel="1 Gender adjusted",
outcomes=c("lactose","lactate"))
corrmatrix <-runCorr(modeldata,exmetabdata,"DPP")
# Get correlation results
OutputCSVResults(filename="corr",dataf=corrmatrix,cohort="DPP")
# Get harmonization results
OutputCSVResults(filename="harmonization",dataf=exmetabdata$metab,cohort="DPP")

## End(Not run)
```

OutputListToExcel *Create an excel xlsx file from a list of data frames*

Description

Create an excel xlsx file from a list of data frames

Usage

```
OutputListToExcel(filename, obj)
```

Arguments

filename	Name of file and can include path. It must have a ".xlsx" extension.
obj	List of data frames or matrices

OutputXLSResults *Create output XLSX file*

Description

Create output XLSX file

Usage

```
OutputXLSResults(filename, datal, cohort = "")
```

Arguments

filename	name of file and can include path
datal	data list to output (each item on list outputs to a worksheet)
cohort	cohort name

Value

the filename of the XLSX file with results named with cohort

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,exposures="age",modlabel="1 Gender adjusted",
outcomes=c("lactose","lactate"))
# Get descriptive data
descdata <-runDescrip(exmetabdata)
OutputXLSResults(filename="corr",datal=descdata,cohort="DPP")
```

`plotMinvalues` *Plot the distribution of the number of missing values for each metabolite*

Description

Plot the distribution of the number of missing values for each metabolite

Usage

```
plotMinvalues(
  cometsdata,
  title = "Distribution of the Number/Missing Values",
  xlabel = "Number of minimum/missing values",
  ylabel = "Frequency",
  xylabelsize = 12,
  titlesize = 16
)
```

Arguments

`cometsdata`	output of readCOMETSinput function
`title`	main title for the plot (default is "Distribution of the Number/Missing Values")
`xlabel`	x-axis label (default is "Number of minimum/missing values")
`ylabel`	y-axis label (default is "Frequency")
`xylabelsize`	size of x and y labels (default=8)
`titlesize`	size of title (default, 20)

Value

a distribution plot

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
plotMinvalues(exmetabdata)
```

plotVar *Plot the variance distribution of transformed metabolite abundances*

Description

Plot the variance distribution of transformed metabolite abundances

Usage

```
plotVar(
  cometsdata,
  title = "Distribution of Variance",
  titlesize = 16,
  xlabel = "Variance of transformed metabolite abundances",
  ylabel = "Frequency",
  xylabelsize = 12
)
```

Arguments

cometsdata	output of readCOMETSinput function
title	main title for the plot (default is "Distribution of Variance")
titlesize	size of title (default, 20)
xlabel	x-axis label (default is "Variance of transformed metabolite abundances")
ylabel	y-axis label (default is "Frequency")
xylabelsize	size of x and y labels (default=8)

Value

a distribution plot

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
plotVar(exmetabdata)
```

readCOMETSinput *Read in Excel file that contains metabolite data, covariate data, models, and model options.*

Description

Read in Excel file that contains metabolite data, covariate data, models, and model options.

Usage

```
readCOMETSinput(file)
```

Arguments

file	path of Excel file to be read in. This file must contain sheets with names **Metabolites**, **SubjectMetabolites**, **SubjectData**, **VarMap**, and optionally **Models**, **ModelOptions** (see details).

Details

Additional information regarding each sheet in the input Excel file is given below.

Metabolites
A table with the columns METABID, METABOLITE_NAME, and possibly other columns of information about the metabolites. The METABID column is used for harmonizing the metabolite names across different cohorts when meta-analyses are performed.

SubjectMetabolites
A table with the subject ids in the first column and metabolites as the other columns.

SubjectData
A table with the subject ids in the first column and covariates as the other columns.

VarMap
A table with at least the required columns VARREFERENCE, COHORTVARIABLE, VARTYPE, and VARDEFINITION. The COHORTVARIABLE column must contain names that match the column names in the **SubjectData** table. These names will be renamed to their corresponding name in the VARREFERENCE column. The VARTYPE column should have values `continuous` or `categorical` for each row.

Models
A table where each row represents a model to be run, and with columns MODEL, OUTCOMES, EXPOSURE, ADJUSTMENT, STRATIFICATION, WHERE, and optionally MODELSPEC, TIME, GROUP. All variable names in this table must match variable names in the VARREFERENCE column of the **VarMap** sheet. The MODEL column is a label for the model. The OUTCOMES and EXPOSURE columns define the outcome and exposure variables for the model. Use `All metabolites` to specify that all metabolite variables are to be included as outcomes or exposures, otherwise use a space separated list of variable names. The ADJUSTMENT column contains a space separated list of covariate adjustment variables; use an empty cell for no covariate adjustment. The STRATIFICATION column is used for stratified analyses, with a space separated list of stratification variables. If more than one stratification variable is specified, then the strata are defined by all unique combinations of the stratification variables that appear in the data. The WHERE column is used to define a subset of subjects to include in the analysis, and has the form `variable operator value`, where `operator` can be one of the following `>,<,>=,<= !=,=`. An example WHERE condition is `age > 50`, which will include all subjects older than 50 in the analysis. Multiple WHERE conditions must be separated by a comma and are logically connected with the `&` operator. For example, the WHERE condition `age > 50 ,bmi >= 22` will include the subjects older than 50 AND with bmi >= 22. Values in the MODELSPEC column must match with the MODELSPEC column in the **ModelOptions** sheet. The TIME column is only required when survival models are run. This column can contain a single time variable or two time variables separated by a space. The GROUP column is only required when conditional logistic regression models are run. This column can contain only a single variable that defines the groups (sets, strata). This sheet is not required when running in interactive mode, but is required when running in batch mode.

ModelOptions
A table where each row specifies an option and has columns MODELSPEC, FUNCTION, OPTION, and VALUE. For an example sheet and additional information about this sheet, see the Excel file `/extdata/cometsInput.xlsx`. This sheet is optional, but is required when the **Models** sheet contains the column MODELSPEC.

Value

a list comprising of data and information needed for getModelData.

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
```

`runAllModels` *This function allows users to run all models that are provided in the "Models" sheet of the input Excel file.*

Description

This function allows users to run all models that are provided in the "Models" sheet of the input Excel file.

Usage

```
runAllModels(readData, cohortLabel = "", writeTofile = T)
```

Arguments

`readData`	list from `readCOMETSinput`
`cohortLabel`	cohort label (e.g. DPP, NCI, Shanghai)
`writeTofile`	T/F (whether or not to write results for each model into separate xlsx files). Files are written to current directory. Default is TRUE.

Value

A list of return objects from `runModel` or `runCorr`. The `ith` element in this list is the output from the `ith` model run.

Examples

```
## Not run:
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
allmodeloutput <- runAllModels(exmetabdata)

## End(Not run)
```

runCorr *Calculate correlations for input model.*

Description

Calculate correlations for input model.

Usage

```
runCorr(modeldata, metabdata, cohort = "")
```

Arguments

modeldata	list from function getModelData
metabdata	metabolite data list from readCOMETSinput
cohort	cohort label (e.g DPP, NCI, Shanghai)

Details

This function is a special case of runModel with the option op$model = "correlation", however for backwards compatibility, it returns a data frame as in the original version of the **COMETS** R package.

Value

data frame with each row representing the correlation for each combination of outcomes and exposures represented as specified in the model (*spec), label (*lab), and universal id (*_uid) with additional columns for n, pvalue, method of model specification (Interactive or Batch), universal id for outcomes (outcome_uid) and exposures (exposure_uid) name of the cohort, adjustment (adjvars) and stratification (stratavar,strata) variables. Attribute of dataframe includes ptime for processing time of model run.

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,exposures="age",modlabel="1 Gender adjusted",
outcomes=c("lactose","lactate"), modelspec="Interactive")
corrmatrix <- runCorr(modeldata,exmetabdata, "DPP")
```

runDescrip *This function provides a description of the input data (for categorical data, the number of samples of each type; for continous data, the median and other statistics for each variable)*

Description

This function provides a description of the input data (for categorical data, the number of samples of each type; for continous data, the median and other statistics for each variable)

Usage

```
runDescrip(readData)
```

Arguments

readData	list from readComets

Value

a list with 2 data frames, continuous and categorical summaries. Type of variable is defined in varmap

Examples

```
## Not run:
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
allmodeloutput <- runAllModels(exmetabdata)
# Get descriptive data
descdata <-runDescrip(exmetabdata)
OutputXLSResults(filename="corr",datal=descdata,cohort="DPP")

## End(Not run)
```

runModel *Main function for running the models*

Description

Main function for running the models

Usage

```
runModel(modeldata, metabdata, cohortLabel = "", op = NULL)
```

Arguments

modeldata	list from function getModelData
metabdata	metabolite data list from readCOMETSinput
cohortLabel	cohort label (e.g DPP, NCI, Shanghai)
op	list of options when running in Interactive mode (see options).

Details

This function will check for stratification variables, and if present run within each stratum. The design matrix is checked for validity (see options). When running in Batch mode, the options are obtained from the Options sheet in the excel file.

Value

A list of objects with names ModelSummary, Effects, and Errors_Warnings.
Important: check the adjvars.removed **column in the** ModelSummary **data frame to see if any adjustment variables were dropped from the model, and use caution interpreting results when variables are removed. The** Errors_Warnings **object may also contain additional variables removed.** Attribute of this list includes ptime for processing time of model run.

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,exposures="age",modlabel="1 Gender adjusted",
outcomes=c("lactose","lactate"))
obj <- runModel(modeldata,exmetabdata, cohortLabel="DPP")
```

---

showCorr *Function that returns top N lines of the* runCorr *output*

Description

Function that returns top N lines of the runCorr output

Usage

```
showCorr(corr, nlines = 50)
```

Arguments

corr	returned object from runCorr
nlines	number of lines to return (default 50)

Value

first 50 lines of output

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,modlabel="1 Gender adjusted")
corrmatrix <- runCorr(modeldata,exmetabdata,"DPP")
showCorr(corrmatrix)
```

`showHClust` *Show interactive heatmap using heatmaply_cor with hierarchical clustering*

Description

This function outputs a heatmap with hierarchical clustering. It thus requires you to have at least 2 outcome and 2 exposure variables in your models.

Usage

```
showHClust(
  ccorrList,
  strata = NULL,
  clust = TRUE,
  colscale = "RdYlBu",
  showticklabels = TRUE
)
```

Arguments

`ccorrList`	correlation object (output of `runCorr`)
`strata`	Only valid if ccorrList is from a stratified analysis. If NULL, then results from the first stratum will be used in the plot.
`clust`	TRUE or FALSE to show hierarchical clustering. The default is TRUE.
`colscale`	colorscale, can be custom or named ("Hots","Greens","Blues","Greys","Purples") see `RColorBrewer_colors`
`showticklabels`	TRUE or FALSE to show axis labels. The default is TRUE.

Value

a heatmap with outcomes as rows and exposures in columns.

References

For colorscale reference: `RColorBrewer_colors`

Examples

```
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata, modelspec="Interactive",
exposures=c("age","bmi_grp"))
corrmatrix <-runCorr(modeldata,exmetabdata,"DPP")
showHClust(corrmatrix)
```

---

showHeatmap *Show interactive heatmap using plot_ly*

Description

Show interactive heatmap using plot_ly

Usage

```
showHeatmap(
  ccorrList,
  strata = NULL,
  rowsortby = "estimate",
  plothgt = 700,
  plotwid = 800,
  colscale = "RdYlBu"
)
```

Arguments

ccorrList	correlation object (output of runCorr)
strata	Only valid if ccorrList is from a stratified analysis. If NULL, then results from the first stratum will be used in the plot.
rowsortby	How row labels are sorted
plothgt	Plot height default 700
plotwid	Plot width default 800
colscale	colorscale, can be custom or named ("Hots","Greens","Blues","Greys","Purples") see https://plot.ly/ipython-notebooks/color-scales/

Value

a heatmap with outcomes as rows and exposures in columns.

References

For colorscale reference: https://plot.ly/ipython-notebooks/color-scales/

Examples

```
## Not run:
dir <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata <- getModelData(exmetabdata,modlabel="1 Gender adjusted")
corrmatrix <-runCorr(modeldata,exmetabdata,"DPP")
showHeatmap(corrmatrix)

## End(Not run)
```

---

showModel *Function that displays the first N rows of each data frame in the the* runModel *output.*

Description

Function that displays the first N rows of each data frame in the the runModel output.

Usage

```
showModel(obj, nlines = 10)
```

Arguments

obj	returned object from runModel
nlines	number of lines to display (default 10)

Examples

```
dir         <- system.file("extdata", package="RcometsAnalytics", mustWork=TRUE)
csvfile     <- file.path(dir, "cometsInputAge.xlsx")
exmetabdata <- readCOMETSinput(csvfile)
modeldata   <- getModelData(exmetabdata,modlabel="1 Gender adjusted")
result      <- runModel(modeldata,exmetabdata, cohortLabel="DPP")
showModel(result)
```

Index