Package 'genophenoR'

April 3, 2017

Type Package

Title Integrative analysis of phenotype and genotype information from electronic health record data

Version 0.99.0

Description Package to integrate, analyze and visualize phenotype and genotype information from electronic health record data.

Depends R (>= 3.2.3), stringr

License MIT + file LICENSE

LazyData true

Suggests parallel

Imports stringr, igraph, ggplot2, plotrix, plyr, reshape, parallel, gridExtra, scales, plotly, RCurl, d3heatmap, gtools, shiny, httr

NeedsCompilation no

Author Alba Gutierrez-Sacristan [aut],

Maintainer Alba Gutierrez-Sacristan <alba.gutierrez@upf.edu>

biocViews Software, BiomedicalInformatics, Genetics, DataRepresentation

RoxygenNote 6.0.0

R topics documented:

demographicSummary — *Describes the demographic characteristics (sex, age) of the population under study*

Description

Given an object of class genopheno, and the characters used to specify the gender, a graphic containing 3 plots, one of the age distribution, another one of the age distribution and the third one with the relation between age and gender distribution, is obtained.

Usage

```
demographicSummary(input, maleCode, femaleCode, verbose = FALSE,
  warnings = TRUE)
```

Arguments

input	Object of genopheno class.
maleCode	Characters(s) used to determine the male condition of a patient. Depending on the database it can be determined, for example, as Male, . MALE, M, with digits as 0 or 1.
femaleCode	Characters(s) used to determine the female condition of a patient. Depending on the database it can be determined, for example, as Female, . FEMALE, F, with digits as 0 or 1.
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.
warnings	By default TRUE. Change it to FALSE to don't see the warnings.

Value

A multiple graph containing a barplot with age distribution, a boxplot representing age distribution by gender and a pie chart representing gender distribution.

Examples

```
load(system.file("extdata", "genophenoExData.RData", package="genophenoR"))
demographicSummary( input = genophenoExData,
                    maleCode   = "male",
                    femaleCode = "female"
          )
```

extract — *Obtain the raw query from a* genophenoComor *object.*

Description

Obtain the raw query from a genophenoComor object.

Usage

```
## S4 method for signature 'genopheno'
extract(object)
```

Arguments

object	of class genophenoComor or genopheno object

Value

A data.frame containing the raw result from the initial data (genopheno)or comorbidity analysis (genophenoComor)

Examples

```
## Not run:
#Being x an genophenoComor
qr <- extract(x)

## End(Not run)
```

---

genoPhenoComorbidity *Comorbidity Analysis* genophenoComor

Description

Given an object of type genopheno, a comorbidity analysis is perform, for the subset of population under specific conditions of age, gender and gene status. It generates a genophenoComor object.

Usage

```
genoPhenoComorbidity(input, pth, ageRange = c(0, 100), aggregate = TRUE,
  gender = "ALL", mutation = c("", ""), nfactor = 10, score, fdr,
  oddsRatio, rr, phi, cores = 1, verbose = FALSE)
```

Arguments

input	A genopheno object, obtained with the queryPheno function.
pth	Determines the path where the required file with phenotype data is located. This file is generated applying the phenotypeSummary function.
ageRange	Determines what is the age range of interest for performing the comorbidity analysis. By default it is set from 0 to 100 years old.
aggregate	By default TRUE. Change it to FALSE if you want to analyze the comorbidity taking into all the values of each phenotype.
gender	Determine what is the gender of interest for performing the comorbidity analysis. By default ALL. Change it to the gender of interest for your comorbidity analysis.
mutation	Determine what is the mutation value of interest for performing the comorbidity analysis. By default c("ALL", "ALL"). Change it to the value of interest for your comorbidity analysis. For example, c("CHD8", "yes")

nfactor	By default 10. Change it into other number if you consider there is any categorical variable with more than nfactor values.
score	The comorbidity score is a measure based on the observed comorbidities and the expected ones, based on the occurrence of each disease.
fdr	A Fisher exact test for each pair of diseases is performed to assess the null hypothesis of independence between the two diseases. The Benjamini-Hochberg false discovery rate method (FDR) is applied to correct for multiple testing.
oddsRatio	The odds ratio represents the increased chance that someone suffering disease X will have the comorbid disorder Y.
rr	The relative risk refers to the fraction between the number of patients diagnosed with both diseases and random expectation based on disease prevalence.
phi	The Pearsons correlation for binary variables (Phi) measures the robustness of the comorbidity association.
cores	By default 1. To run parallel computations on machines with multiple cores or CPUs, the cores argument can be changed.
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.

Value

An object of class genophenoComor

Examples

```
load(system.file("extdata", "genophenoExData.RData", package="genophenoR"))
ex1 <- genoPhenoComorbidity(
              input         = genophenoExData,
              pth           = system.file("extdata", package="genophenoR"),
              aggregate     = TRUE,
              ageRange      = c(0,16),
              gender        = "MALE",
              mutation      = c("CHD8", "ALL")
              )
```

---

genoPhenoHeatmap *Plot the comorbidity analysis results in a heatmap.*

Description

Given an object of class genophenoComor obtained from a comorbidity analysis, a heatmap is obtained.

Usage

```
genoPhenoHeatmap(input, selectValue = "score", cutOff = 0.05, npairs = 0,
  interactive = FALSE, lowColor = "#cde6ff", highColor = "red",
  verbose = FALSE)
```

Arguments

input	A genophenoComor object, obtained by applying the comorbidityAnalysis function
selectValue	By default "patientsPhenoAB" variable will be selected. Change it to any of the other possible variables ('score',('fdr','odds ratio', 'phi', 'rr', 'PercentagePhenoAB').
cutOff	By default '0.05'. The value of the argument can be changed to any other numeric variable, according to the range of the selected value.
npairs	by default '0'. The value of the argument can be changed to any other numeric variable to show in the network only those comorbidities suffered by at least npairs of patients.
interactive	Determines if the output heatmap is interactive or not. By default the interactive argument is set up as FALSE. The value of the argument can be changed to TRUE, as a result an interactive heatmap will be obtained.
lowColor	Determines the heatmap color for the lowest value. By default it is set to "#cde6ff".
highColor	Determines the heatmap color for the highest value. By default it is set to "red".
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.

Value

A heatmap

Examples

```
load(system.file("extdata", "genophenoComor.RData", package="genophenoR"))
htmp <- genoPhenoHeatmap( input = genophenoComor,
              selectValue  = "patientsPhenoAB",
              cutOff       = 1,
              npairs       = 1
              )
htmp
```

---

genoPhenoNetwork *Plot the comorbidity analysis results in a network*

Description

Given an object of class genophenoComor obtained from a comorbidity analysis, a network is obtained.

Usage

```
genoPhenoNetwork(input, layout = "layout.circle",
  selectValue = "patientsPhenoAB", title = "Phenotype comorbidity network",
  cutOff = 0, npairs = 0, prop = 1, interactive = FALSE,
  comorColor = "lightblue", verbose = FALSE)
```

Arguments

input	A genophenoComor object, obtained by applying the comorbidityAnalysis function
layout	By default 'layout.fruchterman.reingold'. It can be set to any other of the possible igraph layouts.
selectValue	By default "AB" variable will be selected. Change it to any of the other possible variables ('score',('fdr','odds ratio', 'phi', 'rr').
title	Determines the title of the network figure. By default 'Comorbidity network'.
cutOff	By default '0.05'. The value of the argument can be changed to any other numeric variable, according to the range of the selected value.
npairs	by default '0'. The value of the argument can be changed to any other numeric variable to show in the network only those comorbidities suffered by at least npairs of patients.
prop	Determines the node size proportionality. By default it is set to 1. The value of the argument can be changed to any other numeric variable.
interactive	Determines if the output network is interactive or not. By default the interactive argument is set up as FALSE. The value of the argument can be changed to TRUE, as a result an interactive network will be obtained.
comorColor	Determines the node color for the comorbid disorders. By default it is set to "lightblue".
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.
databasePth	Determines the path where the intermediate RData objects have been created. It is the same path where the three required input files (patientData, diagnosisData, admissionData) are located.

Value

A network

Examples

```
load(system.file("extdata", "genophenoComor.RData", package="genophenoR"))
genoPhenoNetwork <- network(
              input            = genophenoComor,
              selectValue      = "score",
              cutOff           = 1.5,
              npairs           = 2,
              title            = "Example comorbidity network"
              )
```

---

genoPhenoPatientsSelection

Patients selection genophenoPatientsSelection

Description

Given an object of type genopheno and two phenotypes of interest, the patients identifiers of those patients having both phenotypes are selected.

Usage

```
genoPhenoPatientsSelection(input, pth, ageRange = c(0, 100), phenotypeA,
  phenotypeB, aggregate = TRUE, gender = "ALL", mutation = c("ALL",
  "ALL"), nfactor = 10, cores = 1, verbose = FALSE, warnings = TRUE)
```

Arguments

input	A genopheno object, obtained with the queryPheno function.
pth	Determines the path where the required input file with the yes/no phenotype data is located.
ageRange	Determines what is the age range of interest for performing the comorbidity analysis. By default it is set from 0 to 100 years old.
phenotypeA	One of the phenotypes of interest and the value in which user is interested. For example, c("FacialExpression", "yes)
phenotypeB	The second phenotype of interest.and the value in which user is interested. For example, c("HandMovement", "yes)
aggregate	By default TRUE. Change it to FALSE if you want to analyze the comorbidity taking into all the values of each phenotype.
gender	Determine what is the gender of interest for performing the comorbidity analysis. By default ALL. Change it to the gender of interest for your comorbidity analysis.
mutation	Determine what is the mutation value of interest for performing the comorbidity analysis. By default c("ALL", "ALL"). Change it to the value of interest for your comorbidity analysis. For example, c("CHD8", "yes")
nfactor	By default 10. Change it into other number if you consider there is any categorical variable with more than nfactor values.
cores	By default 1. To run parallel computations on machines with multiple cores or CPUs, the cores argument can be changed.
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.
warnings	By default TRUE. Change it to FALSE to don't see the warnings.

Value

An object of class cgpAnalysis

Examples

```
load(system.file("extdata", "genopheno.RData", package="genophenoR"))
ex1 <- genoPhenoPatientsSelection(
             input         = queryExample,
             pth           = system.file("extdata", package="genophenoR"),
             phenotypeA    = c("HandMovement", "yes"),
             phenotypeB    = c("FacialExpression", "yes"),
             aggregate     = TRUE,
             ageRange      = c(0,16),
             gender        = "MALE",
             mutation      = c("CHD8", "ALL")
             )
```

genoPhenoPrevalence *Generates a table with the prevalence of each phenotype*

Description

Given an object of class genophenoComor obtained from a comorbidity analysis, a prevalence table is obtained.

Usage

```
genoPhenoPrevalence(input, verbose = FALSE, warnings = FALSE)
```

Arguments

input	A genophenoComor object, obtained by applying the comorbidityAnalysis function
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.
warnings	By default TRUE. Change it to FALSE to don't see the warnings.

Value

A table

Examples

```
load(system.file("extdata", "genophenoComor.RData", package="genophenoR"))
prevalence <- genoPhenoPrevalence(
              input            = genophenoComor
              )
```

genophenoZscore *Transform continuous in categorical variables and generates a new* genopheno *object.*

Description

Given an object of class genopheno, it transforms continuous into categorical variable applying Z-score. As a result a new genopheno object is generated. Note that is the number of individuals is lower than 5000 a Saphiro test is done to test the normal distribution, otherwhise a Kolmogorov-Smirnov test is performed.

Usage

```
genophenoZscore(input, cutOff = c(-2, 2), nfactor = 10, verbose = FALSE)
```

Arguments

input	Object of genopheno class.
cutOff	Z-score cut-off to categorize the continuous variable. By default it is set to -2 and 2.
nfactor	By default 10. Change it into other number if you consider there is any categorical variable with more than nfactor values.
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.

Value

A genopheno class object with the continuous variable transformed into a categorical variable, if possible.

Examples

```
load(system.file("extdata", "genophenoExData.RData", package="genophenoR"))
genophenoZscore( input      = genophenoExData,
                 verbose    = FALSE
           )
```

---

irctquery *Query analysis to the API*

Description

Given an url, a key and a JSON object, it generates a data.frame object with the output of the query.

Usage

```
irctquery(url, apiKey, query, outputPath = getwd(), verbose = FALSE)
```

Arguments

url	The url.
apiKey	The key to access to the data.
query	A text file containing the JSON query body.
outputPath	Path where the output file will be saved.By default it will be saved in your working directory
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.

Value

An object of class data.frame with the query output.

Examples

```
query <- irctquery(
              url         = "https://nhanes.hms.harvard.edu/",
              apiKey      = "a77l42anvcgdtkfcvbl7hnp2v9",
              query  = system.file("extdata", "jsonQueryNhanes", package="genophenoR"))
              )
```

`phenotypeSummary` *Describes the phenotypic characteristics for the whole study population then according to the status regarding one selected gene*

Description

Given an object of class genopheno, a file with the different values for each phenotype, and the prevalence of each one in general population and according to the gene status selected is generated. A figure containing a barplot for each phenotype is displayed. Each barplot shows the population percentage suffering each type of the phenotypes according to the values it takes, and distinguishing between those having or not a mutation. Furthermore, a data.frame with the numerical values is obtained.

Usage

```
phenotypeSummary(input, mutation = FALSE, nfactor = 10, showTable = TRUE,
  showFigures = FALSE, path = getwd(), verbose = FALSE)
```

Arguments

`input`	Object of genopheno class.
`mutation`	Determines the mutation of interest for wchich you want to analyze the phenotype values. By default FALSE.
`nfactor`	By default 10. Change it into other number if you consider there is any categorical variable with more than `nfactor` values.
`showTable`	By default TRUE. Change it into FALSE in order to not visualize the table with the ressults.
`showFigures`	By default FALSE. Change it into TRUE in order to visualize the table with the ressults.
`path`	By default the working directory. Define the path where you want the file to be saved
`verbose`	By default `FALSE`. Change it to `TRUE` to get a on-time log from the function.

Value

A file .

Examples

```
load(system.file("extdata", "genophenoExData.RData", package="genophenoR"))
phenotypeSummary( input       = genophenoExData,
                  showTable   = TRUE,
                  showFigures = TRUE,
                 verbose      = FALSE
          )
```

`prevalenceNetwork` *Plot the phenotype prevalence related to a mutation in a network*

Description

Given mutation and a `genophenoComor` object table a network is obtained.

Usage

```
prevalenceNetwork(input, layout = "layout.circle",
  title = "Phenotype prevalence network", interactive = FALSE,
  verbose = FALSE)
```

Arguments

`input`	A `genophenoComor` object, obtained by applying the `comorbidityAnalysis` function
`layout`	By default `'layout.fruchterman.reingold'`. It can be set to any other of the possible igraph layouts.
`title`	Determines the title of the network figure. By default `'Comorbidity network'`.
`interactive`	Determines if the output network is interactive or not. By default the `interactive` argument is set up as `FALSE`. The value of the argument can be changed to `TRUE`, as a result an interactive network will be obtained.
`verbose`	By default `FALSE`. Change it to `TRUE` to get a on-time log from the function.
`warnings`	By default `TRUE`. Change it to `FALSE` to don't see the warnings.

Value

A network

Examples

```
load(system.file("extdata", "genophenoComor.RData", package="genophenoR"))
ntwk <- prevalenceNetwork(
             input            = genophenoComor
             )
```

queryGenoPheno *Query your data and generates a* genopheno

Description

Given a tabulated file, checks if it contains the data in the correct format and generates a phenotype object.

Usage

```
queryGenoPheno(inputDataFile, verbose = FALSE, warnings = TRUE)
```

Arguments

inputDataFile	Determines the file with the complete path where the required input file is located. This input file must contain three columns: "patient_id" with the patient identifier, "Gender" and "Age". Mutation columns must start with "M." while phenotype ones must start with a "P."
verbose	By default FALSE. Change it to TRUE to get a on-time log from the function.
warnings	By default TRUE. Change it to FALSE to don't see the warnings.

Value

An object of class phenotype

Examples

```
queryExample <- queryGenoPheno( inputDataFile = paste0(system.file("extdata", package="genophenoR"),
                                                       "/queryOutput.txt"),
                                verbose     = TRUE)
queryExample
```

Index