

Package ‘OncoScore’

October 16, 2018

Version 1.8.0

Date 2018-03-09

Title A tool to identify potentially oncogenic genes

Maintainer Daniele Ramazzotti <daniele.ramazzotti@yahoo.com>

Depends R (>= 3.4),

Imports biomaRt, grDevices, graphics, utils,

Suggests BiocGenerics, BiocStyle, knitr, testthat,

Description OncoScore is a tool to measure the association of genes to cancer based on citation frequencies in biomedical literature. The score is evaluated from PubMed literature by dynamically updatable web queries.

Encoding UTF-8

LazyData TRUE

License file LICENSE

URL <https://github.com/danro9685/OncoScore>

BugReports <https://github.com/danro9685/OncoScore>

biocViews BiomedicalInformatics

RoxygenNote 6.0.1

VignetteBuilder knitr

git_url <https://git.bioconductor.org/packages/OncoScore>

git_branch RELEASE_3_7

git_last_commit a625a79

git_last_commit_date 2018-04-30

Date/Publication 2018-10-15

Author Luca De Sano [aut],
Carlo Gambacorti Passerini [ctb],
Rocco Piazza [ctb],
Daniele Ramazzotti [aut, cre],
Roberta Spinelli [ctb]

R topics documented:

combine.query.results	2
combine.single.matrix	3
compute.frequencies.scores	3
compute.oncoscore	4
compute.oncoscore.from.region	4
compute.oncoscore.timeseries	5
estimate.oncogenes	6
genes	6
get.genes.from.biomart	7
get.list.from.xml	7
get.pubmed.driver.analysis	8
perform.query	8
perform.query.from.region	9
perform.query.timeseries	9
plot.oncoscore	10
plot.oncoscore.timeseries	11
query	12
query.timepoints	12
timepoints	13
try.scan	13
Index	14

combine.query.results *combine.query.results*

Description

Merge a set of genes in a unique one in order to account for possible aliases

Usage

```
combine.query.results(query, genes, new.name)
```

Arguments

query	The result of perform.query, perform.query.timeseries or perform.query.from.region
genes	A list of genes to be merged
new.name	A string containing the new name to be used for the new genes

Value

The frequencies of the genes in the cancer related documents and in all the documents retrieved on PubMed

Examples

```
data(query)
combine.query.results(query, c('IDH1', 'IDH2'), 'new_gene')
```

combine.single.matrix *combine.single.matrix*

Description

Perform merge procedure on a matrix

Usage

```
combine.single.matrix(query, genes, new.name)
```

Arguments

query	The result of perform.query, perform.query.timeseries or perform.query.from.region
genes	A list of genes to be merged
new.name	A string containing the new name to be used for the new genes

Value

a merged matrix

compute.frequencies.scores
compute.frequencies.scores

Description

compute the logarithmic scores based on the frequencies of the genes

Usage

```
compute.frequencies.scores(data, filter.threshold = 1,  
  analysis.mode = "Log2")
```

Arguments

data	input data as result of the function perform.query
filter.threshold	threshold to filter for a minimum number of citations for the genes
analysis.mode	logarithmic scores to be computed, i.e., log10, log2, natural log or log5

Value

the computed scores

```
compute.oncoscore      compute.oncoscore
```

Description

compute the OncoScore for a list of genes

Usage

```
compute.oncoscore(data, filter.threshold = 0, analysis.mode = "Log2",
  cutoff.threshold = 21.09, file = NULL, filter.invalid = TRUE)
```

Arguments

```
data          input data as result of the function perform.query
filter.threshold
              threshold to filter for a minimum number of citations for the genes
analysis.mode  logarithmic scores to be computed, i.e., log10, log2, natural log or log5
cutoff.threshold
              threshold to be used to asses the oncogenes
file          should I save the results to text files?
filter.invalid auto-remove genes with invalid count
```

Value

the computed OncoScores and the clusters for the genes

Examples

```
data(query)
compute.oncoscore(query)
```

```
compute.oncoscore.from.region
      compute.oncoscore.from.region
```

Description

Perform OncoScore analysis on a given chromosomal region

Usage

```
compute.oncoscore.from.region(chromosome, start = NA, end = NA,
  gene.num.limit = 100, filter.threshold = NA, analysis.mode = "Log2",
  cutoff.threshold = 21.09, file = NULL)
```

Arguments

chromosome	chromosome to be retrieved
start	initial position to be used
end	final position to be used
gene.num.limit	A limit to the genes to be considered in the analysis; this is done to limit the number of queries to PubMed
filter.threshold	threshold to filter for a minimum number of citations for the genes
analysis.mode	logarithmic scores to be computed, i.e., log10, log2, natural log or log5
cutoff.threshold	threshold to be used to asses the oncogenes
file	should I save the results to text files?

Value

the computed scores

Examples

```

chromosome = 15
start = 200000
end = 300000
compute.oncoscore.from.region(chromosome, start, end)

```

```

compute.oncoscore.timeseries
      compute.oncoscore.timeseries

```

Description

perform the OncoScore time series analysis for a list of genes and data times

Usage

```

compute.oncoscore.timeseries(data, filter.threshold = 0,
  analysis.mode = "Log2", cutoff.threshold = 21.09, file = NULL)

```

Arguments

data	input data as result of the function perform.query.timeseries
filter.threshold	threshold to filter for a minimum number of citations for the genes
analysis.mode	logarithmic scores to be computed, i.e., log10, log2, natural log or log5
cutoff.threshold	threshold to be used to asses the oncogenes
file	should I save the results to text files?

Value

the performed OncoScores time series analysis

Examples

```
data(query.timepoints)
compute.oncoscore.timeseries(query.timepoints)
```

```
estimate.oncogenes      estimate.oncogenes
```

Description

estimate the oncoscore for the genes

Usage

```
estimate.oncogenes(data, cutoff.threshold = 21.09)
```

Arguments

`data` input data as result of the function `compute.frequencies.scores`
`cutoff.threshold` threshold to be used to asses the oncogenes

Value

the computed scores and oncogenes

```
genes                   A list of genes
```

Description

This dataset contains a list of genes to be used in the analysis as an example

Usage

```
genes
```

Format

```
txt
```

Value

list of 5 elements

Source

example data

`get.genes.from.biomart`
get.genes.from.biomart

Description

Get a gene list from biomart

Usage

```
get.genes.from.biomart(chromosome, start = NA, end = NA)
```

Arguments

chromosome	chromosome to be retrieved
start	initial position to be used
end	final position to be used

Value

A list of genes

Examples

```
chromosome = 15  
start = 200000  
end = 300000  
ch15 = get.genes.from.biomart(chromosome, start, end)
```

`get.list.from.xml` *get.list.from.xml*

Description

process the result of the query

Usage

```
get.list.from.xml(webget)
```

Arguments

webget	The result from the query to PubMed
--------	-------------------------------------

Value

Processed result obtained from the query to PubMed

```
get.pubmed.driver.analysis
      get.pubmed.driver.analysis
```

Description

query PubMed for a list of genes

Usage

```
get.pubmed.driver.analysis(keywords, gene)
```

Arguments

keywords	The set of keywords to be used for the query to PubMed
gene	The name of a gene to be used for the query to PubMed

Value

The frequency for the current gene retrieved with the query on the provided set of keywords

```
perform.query      perform.query
```

Description

perform the query to PubMed

Usage

```
perform.query(list.of.genes, gene.num.limit = 100, custom.search = NA)
```

Arguments

list.of.genes	The list of genes to be used in the queries to PubMed
gene.num.limit	A limit to the genes to be considered in the analysis; this is done to limit the number of queries to PubMed
custom.search	A custom set of keywords to be used when querying PubMed

Value

The frequencies of the genes in the cancer related documents and in all the documents retrieved on PubMed

Examples

```
data(genes)
perform.query(genes[1:2])
```

`perform.query.from.region`
perform.query.from.region

Description

Perform the query to PubMed on a given chromosomal region

Usage

```
perform.query.from.region(chromosome, start = NA, end = NA,  
  gene.num.limit = 100)
```

Arguments

<code>chromosome</code>	chromosome to be retrieved
<code>start</code>	initial position to be used
<code>end</code>	final position to be used
<code>gene.num.limit</code>	A limit to the genes to be considered in the analysis; this is done to limit the number of queries to PubMed

Value

The frequencies of the genes in the cancer related documents and in all the documents retrieved on PubMed

Examples

```
chromosome = 15  
start = 200000  
end = 300000  
perform.query.from.region(chromosome, start, end)
```

`perform.query.timeseries`
perform.query.timeseries

Description

perform the query to PubMed for the time series analysis

Usage

```
perform.query.timeseries(list.of.genes, list.of.datatimes,  
  gene.num.limit = 100, timepoints.limit = 10, custom.search = NA)
```

Arguments

`list.of.genes` The list of genes to be used in the queries to PubMed
`list.of.datatimes`
 The list of time points to be used in the queries to PubMed
`gene.num.limit` A limit to the genes to be considered in the analysis; this is done to limit the
 number of queries to PubMed
`timepoints.limit`
 A limit to the time points to be considered in the analysis; this is done to limit
 the number of queries to PubMed
`custom.search` A custom set of keywords to be used when quering PubMed

Value

The frequencies of the genes in the cancer related documents and in all the documents retrieved on PubMed at the specified time points

Examples

```

data(genes)
data(timepoints)
perform.query.timeseries(genes[1:2], timepoints[1:2])
  
```

plot.oncoscore *plot.oncoscore*

Description

plot the OncoScore for a list of genes

Usage

```

## S3 method for class 'oncoscore'
plot(x, gene.number = 5, main = "OncoScore",
      xlab = "score", ylab = "genes", file = NA, ...)
  
```

Arguments

`x` input data as result of the function compute.OncoScore
`gene.number` number of genes to print
`main` the title
`xlab` description of x asix (default score)
`ylab` description of y asix (default genes)
`file` where to save the plot
`...` additional parameter to pass to the barplot function

Value

A plot

Examples

```
data(query)
result = compute.oncoscore(query)
plot.oncoscore(result)
```

```
plot.oncoscore.timeseries
      plot.oncoscore.timeseries
```

Description

plot the OncoScore for a list of genes

Usage

```
## S3 method for class 'oncoscore.timeseries'
plot(x, gene.number = 5, incremental = FALSE,
     relative = FALSE, main = "OncoScore", xlab = "timepoints",
     ylab = "score", legend.pos = "top", file = NA, ...)
```

Arguments

x	input data as result of the function compute.OncoScore
gene.number	number of genes to print
incremental	display the OncoScore increment
relative	display the incrementa as relative value
main	the title
xlab	description of x asix (default score)
ylab	description of y asix (default genes)
legend.pos	Position of the legend
file	where to save the plot
...	additional parameter to pass to the lines function

Value

A plot

Examples

```
data(query.timepoints)
result = compute.oncoscore.timeseries(query.timepoints)
plot.oncoscore.timeseries(result)
```

query	<i>The result of perform.web.query on genes</i>
-------	---

Description

This dataset contains the result of perform.web.query on genes

Usage

query

Format

rdata

Value

matrix 5 x 2

Source

example data

query.timepoints	<i>The result of perform.time.series.query on genes and timepoints</i>
------------------	--

Description

This dataset contains the result of perform.time.series.query on genes and timepoints

Usage

query.timepoints

Format

rdata

Value

list of 5 matrix 5 x 2

Source

example data

timepoints	<i>A list of timepoints</i>
------------	-----------------------------

Description

This dataset contains a list of time points to be used in the analysis as an example

Usage

timepoints

Format

txt

Value

list of 5 elements

Source

example data

try.scan	<i>try.scan</i>
----------	-----------------

Description

try to query the given URL

Usage

try.scan(getURL)

Arguments

getURL The given URL

Value

Result obtained from PubMed

Index

*Topic **datasets**

- genes, [6](#)
- query, [12](#)
- query.timepoints, [12](#)
- timepoints, [13](#)

- combine.query.results, [2](#)
- combine.single.matrix, [3](#)
- compute.frequencies.scores, [3](#)
- compute.oncoscore, [4](#)
- compute.oncoscore.from.region, [4](#)
- compute.oncoscore.timeseries, [5](#)

- estimate.oncogenes, [6](#)

- genes, [6](#)
- get.genes.from.biomart, [7](#)
- get.list.from.xml, [7](#)
- get.pubmed.driver.analysis, [8](#)

- perform.query, [8](#)
- perform.query.from.region, [9](#)
- perform.query.timeseries, [9](#)
- plot.oncoscore, [10](#)
- plot.oncoscore.timeseries, [11](#)

- query, [12](#)
- query.timepoints, [12](#)

- timepoints, [13](#)
- try.scan, [13](#)