Package ‘TCGAretriever’

January 23, 2024

Type Package
Title Retrieve Genomic and Clinical Data from CBioPortal Including TCGA Data
Version 1.9.1
Date 2024-01-22
Author Damiano Fantini
Maintainer Damiano Fantini <damiano.fantini@gmail.com>
Depends R (>= 3.5.0)
Imports httr, reshape2, jsonlite
Suggests stats, utils, knitr, rmarkdown, ggplot2, dplyr
VignetteBuilder knitr
Description The Cancer Genome Atlas (TCGA) is a program aimed at improving our understanding of Cancer Biology. Several TCGA Datasets are available online. ‘TCGAretriever’ helps accessing and downloading TCGA data hosted on 'cBioPortal' via its Web Interface (see <https://www.cbioportal.org/> for more information).
License GPL-3
LazyData true
URL https://www.data-pulse.com/dev_site/TCGAretriever/
Encoding UTF-8
RoxygenNote 7.2.3
NeedsCompilation no
Repository CRAN
Date/Publication 2024-01-23 21:52:48 UTC

R topics documented:

  blcaOutputExamples .................................................. 2
  expand_cases ............................................................ 3
  fetch_all_tcgadata ................................................... 4
blcaOutputExamples

get_cancer_studies ........................................ 5
get_cancer_types ........................................... 6
get_case_lists .............................................. 7
get_clinical_data .......................................... 8
get_genetic_profiles ....................................... 9
get_gene_identifiers ...................................... 10
get_molecular_data ......................................... 11
get_mutation_data ......................................... 12
make_groups .................................................. 13

Index 15

blcaOutputExamples TCGAretriever Examples

Description

A list of objects including examples of the output returned by different 'TCGAretriever' functions. The objects were obtained from the "blca_tcga" study (bladder cancer).

Usage

data(blcaOutputExamples)

Format

A list including 7 elements.

exmpl_1 data.frame (dimensions: 10 by 13). Sample output of the 'get_cancer_studies()' function.
exmpl_2 data.frame (dimensions: 10 by 5). Sample output of the 'get_cancer_types()' function.
exmpl_3 data.frame (dimensions: 9 by 5). Sample output of the 'get_case_lists()' function.
exmpl_4 list including 9 elements. Sample output of the 'expand_cases()' function.
exmpl_5 data.frame (dimensions: 10 by 94). Sample output of the 'get_clinical_data()' function.
exmpl_6 data.frame (dimensions: 9 by 8). Sample output of the 'get_genetic_profiles()' function.
exmpl_7 data.frame (dimensions: 10 by 3). Sample output of the 'get_gene_identifiers()' function.
exmpl_8 data.frame (dimensions: 2 by 10). Sample output of the 'get_molecular_data()' function.
exmpl_9 data.frame (dimensions: 6 by 27). Sample output of the 'get_mutation_data()' function.

Details

The object was built using the following lines of code. blcaOutputExamples <- list(exmpl_1 = head(get_cancer_studies(), 10), exmpl_2 = head(get_cancer_types(), 10), exmpl_3 = head(get_case_lists("blca_tcga"), 10), exmpl_4 = expand_cases("blca_tcga"), exmpl_5 = head(get_clinical_data("blca_tcga"), 10), exmpl_6 = head(get_genetic_profiles("blca_tcga"), 10), exmpl_7 = head(get_gene_identifiers(), 10), exmpl_8 = get_molecular_data(case_list_id = "blca_tcga_3way_complete", gprofile_id = "blca_tcga_rna_seq_v2_mrna", glist = c("TP53", "E2F1"))[1:10], exmpl_9 = head(get_mutation_data(case_list_id = "blca_tcga_sequenced", gprofile_id = "blca_tcga_mutations", glist = c("TP53", "PTEN"))))
expand_cases

Examples

data(blcaOutputExamples)
blcaOutputExamples$exmpl_1

expand_cases  Samples Included in a List of Samples.

Description

Each study includes one or more "case lists". Each case list is a collection of samples that were analyzed using one or more platforms/assays. It is possible to obtain a list of all sample identifiers for each case list of interest.

Usage

expand_cases(csid, dryrun = FALSE)

Arguments

csid   String corresponding to a TCGA Cancer Study identifier.
dryrun Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

Value

list containing as many elements as TCGA case lists available for a given TCGA Study. Each element is named after a case list identifier. Also, each element is a character vector including all sample identifiers (case ids) corresponding to the corresponding case list identifier.

Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/TCGAretriever/

Examples

# Set `dryrun = FALSE` (default option) in production!
x <- expand_cases("blca_tcg", dryrun = TRUE)
lapply(x, utils:::head)
fetch_all_tcgadata

Fetch All Molecular Data for a Cancer Profile of Interest.

Description

Recursively query cbioportal to retrieve data corresponding to all available genes. Data are returned as a `data.frame` that can be easily manipulated for downstream analyses.

Usage

```r
fetch_all_tcgadata(case_list_id, gprofile_id, mutations = FALSE)
```

Arguments

- `case_list_id` string corresponding to the identifier of the TCGA Case List of interest
- `gprofile_id` string corresponding to the identifier of the TCGA Profile of interest
- `mutations` logical. If TRUE, extended mutation data are fetched instead of the standard TCGA data

Value

A `data.frame` is returned, including the desired TCGA data. Typically, rows are genes and columns are cases. If "extended mutation" data are retrieved (mutations = TRUE), rows correspond to individual mutations while columns are populated with mutation features

Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/TCGAretriever/

Examples

```r
# The examples below require an active Internet connection.
# Note: execution may take several minutes.

## Not run:

# Download all brca_pub mutation data (complete samples)
all_brca_MUT <- fetch_all_tcgadata(case_list_id = "brca_tcga_pub_complete",
gprofile_id = "brca_tcga_pub_mutations",
mutations = TRUE)

# Download all brca_pub RNA expression data (complete samples)
all_brca_RNA <- fetch_all_tcgadata(case_list_id = "brca_tcga_pub_complete",
gprofile_id = "brca_tcga_pub_mrna",
mutations = FALSE)
```
## Description

Retrieve information about the studies or datasets available at cbioportal.org. Information include a ‘studyId’, description, references, and more.

## Usage

```r
get_cancer_studies(dryrun = FALSE)
```

## Arguments

- `dryrun` Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

## Value

Data Frame including cancer study information.

## Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

## References

[https://www.data-pulse.com/dev_site/TCGAretriever/](https://www.data-pulse.com/dev_site/TCGAretriever/)

## Examples

```r
# Set `dryrun = FALSE` (default option) in production!
all_studies <- get_cancer_studies(dryrun = TRUE)
utils::head(all_studies)
```
get_cancer_types  Retrieve Cancer Types.

Description
Retrieve information about cancer types and corresponding abbreviations from cbioportal.org. Information include identifiers, names, and parental cancer type.

Usage
get_cancer_types(dryrun = FALSE)

Arguments
dryrun Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

Value
A data.frame including cancer type information.

Author(s)
Damiano Fantini, <damiano.fantini@gmail.com>

References
https://www.data-pulse.com/dev_site/TCGAretriever/

Examples
# Set `dryrun = FALSE` (default option) in production!
all_canc <- get_cancer_types(dryrun = TRUE)
utils::head(all_canc)
**get_case_lists**

**Retrieve Case List Available for a Specific Cancer Study.**

**Description**

Each study includes one or more "case lists". Each case list is a collection of samples that were analyzed using one or more platforms/assays. It is possible to obtain a list of case list identifiers from cbioportal.org for a cancer study of interest. Identifier, name, description and category are returned for each entry.

**Usage**

```r
get_case_lists(csid, dryrun = FALSE)
```

**Arguments**

- `csid` String corresponding to the Identifier of the Study of Interest
- `dryrun` Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when `dryrun` is TRUE.

**Value**

Data Frame including Case List information.

**Author(s)**

Damiano Fantini, <damiano.fantini@gmail.com>

**References**

[https://www.data-pulse.com/dev_site/TCGAretriever/](https://www.data-pulse.com/dev_site/TCGAretriever/)

**Examples**

```r
# Set `dryrun = FALSE` (default option) in production!
blca_case_lists <- get_case_lists("blca_tcga", dryrun = TRUE)
bbla_case_lists
```
**get_clinical_data**

Retrieve Clinical Information for a Cancer Study.

**Description**

Retrieve Clinical Information about the samples included in a cancer study of interest. For each sample/case, information about the corresponding cancer patient are returned. These may include sex, age, therapeutic regimen, tumor stage, survival status, as well as other information.

**Usage**

```r
get_clinical_data(csid, case_list_id = NULL, dryrun = FALSE)
```

**Arguments**

- **csid**: String corresponding to a TCGA Cancer Study identifier.
- **case_list_id**: String corresponding to the case_list identifier of interest. This can be NULL.
- **dryrun**: Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when `dryrun` is TRUE.

**Value**

data.frame including clinical information of a list of samples/cases of interest.

**Author(s)**

Damiano Fantini, <damiano.fantini@gmail.com>

**References**

[https://www.data-pulse.com/dev_site/TCGAretriever/](https://www.data-pulse.com/dev_site/TCGAretriever/)

**Examples**

```r
# Set `dryrun = FALSE` (default option) in production!
clinic_data <- get_clinical_data("blca_tcga", dryrun = TRUE)
utils::head(clinic_data[, 1:7])
```
get_genetic_profiles  

Retrieve Genetic Profiles for a TCGA Study of Interest

Description

Retrieve Information about all genetic profiles associated with a cancer study of interest. Each cancer study includes one or more types of molecular analyses. The corresponding assays or platforms are referred to as genetic profiles. A genetic profile identifier is required to download molecular data.

Usage

get_genetic_profiles(csid = NULL, dryrun = FALSE)

Arguments

csid  String corresponding to the cancer study id of interest
dryrun Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

Value

data.frame including information about genetic profiles.

Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/TCGAretriever/

Examples

# Set `dryrun = FALSE` (default option) in production!
get_genetic_profiles("blca_tcga", dryrun = TRUE)
**get_gene_identifiers**  
*Retrieve All Gene Identifiers*

**Description**

Obtain all valid gene identifiers, including ENTREZ gene identifiers and HUGO gene symbols. Genes are classified according to the gene type (*e.g.*, ‘protein-coding’, ‘pseudogene’, ‘miRNA’, ...). Note that miRNA and phosphoprotein genes are associated with a negative entrezGeneId.

**Usage**

```r
get_gene_identifiers(dryrun = FALSE)
```

**Arguments**

- `dryrun` Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

**Value**

Data Frame including gene identifiers.

**Author(s)**

Damiano Fantini, <damiano.fantini@gmail.com>

**References**

[https://www.data-pulse.com/dev_site/TCGAretriever/](https://www.data-pulse.com/dev_site/TCGAretriever/)

**Examples**

```r
# Set 'dryrun = FALSE' (default option) in production!
x <- get_gene_identifiers(dryrun = TRUE)
```
get_molecular_data

Retrieve Molecular Data corresponding to a Genetic Profile of Interest.

Description

Retrieve Data corresponding to a Genetic Profile of interest from a cancer study of interest. This function is the workhorse of the TCGAretriever package and can be used to fetch data concerning several genes at once. For retrieving mutation data, please use the ‘get_mutation_data()’ function. For large queries (more than 500 genes), please use the ‘fetch_all_tcgadata()’ function.

Usage

get_molecular_data(
  case_list_id,
  gprofile_id,
  glist = c("TP53", "E2F1"),
  dryrun = FALSE
)

Arguments

case_list_id  String corresponding to the Identifier of a list of cases.
gprofile_id   String corresponding to the Identifier of a genetic Profile of interest.
glist         Vector including one or more gene identifiers (ENTREZID or OFFICIAL_SYMBOL). ENTREZID gene identifiers should be passed as numeric.
dryrun        Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

Value

data.frame including the molecular data of interest. Rows are genes, columns are samples.

Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/TCGAretriever/
Examples

```r
# Set 'dryrun = FALSE' (default option) in production!
x <- get_molecular_data(case_list_id = 'blca_tcga_3way_complete',
gprofile_id = 'blca_tcga_rna_seq_v2_mrna',
glist = c("TP53", "E2F1"), dryrun = TRUE)
x[, 1:10]
```

---

**get_mutation_data**  
Retrieves Mutation Data corresponding to a Genetic Profile of Interest.

**Description**

Retrieve DNA Sequence Variations (Mutations) identified by exome sequencing projects. This function is the workhorse of the TCGAretriever package for mutation data and can be used to fetch data concerning several genes at once. For retrieving non-mutation data, please use the `get_molecular_data()` function. For large queries (more than 500 genes), please use the `fetch_all_tcgadata()` function.

**Usage**

```r
get_mutation_data(
  case_list_id,  
gprofile_id,  
glist = c("TP53", "E2F1"),  
dryrun = FALSE
)
```

**Arguments**

- **case_list_id**: String corresponding to the Identifier of a list of cases.
- **gprofile_id**: String corresponding to the Identifier of a genetic Profile of interest.
- **glist**: Vector including one or more gene identifiers (ENTREZID or OFFICIAL_SYMOL). ENTREZID gene identifiers should be passed as numeric.
- **dryrun**: Logical. If TRUE, all other arguments (if any) are ignored and a representative example is returned as output. No Internet connection is required for executing the operation when ‘dryrun’ is TRUE.

**Value**

data Frame inluding one row per mutation

**Author(s)**

Damiano Fantini, <damiano.fantini@gmail.com>
make_groups

Split Numeric Vectors in Groups.

Description

Assign each element of a numeric vector to a group. Grouping is based on ranks: numeric values are sorted and then split in 2 or more groups. Values may be sorted in an increasing or decreasing fashion. The vector is returned in the original order. Labels may be assigned to each group.

Usage

make_groups(num_vector, groups, group_labels = NULL, desc = FALSE)

Arguments

num_vector numeric vector. It includes the values to be assigned to the different groups

groups integer. The number of groups that will be generated

group_labels character vector. Labels for each group. Note that the length of group_labels has to be equal to the number of groups

desc logical. If TRUE, the sorting is applied in a decreasing fashion

Value

data.frame including the vector provided as argument in the original order ("value") and the grouping vector ("rank"). If labels are provided as an argument, group labels are also included in the data.frame ("labels").

Author(s)

Damiano Fantini, <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/TCGAretriever/
Examples

```r
exprs_geneX <- c(19.1, 18.4, 22.4, 15.5, 20.2, 17.4, 9.4, 12.4, 31.2, 33.2, 18.4, 22.1)
groups_num <- 3
groups_labels <- c("high", "med", "low")
make_groups(exprs_geneX, groups_num, groups_labels, desc = TRUE)
```
Index

* datasets
  blcaOutputExamples, 2
blcaOutputExamples, 2
expand_cases, 3
fetch_all_tcgadata, 4
get_cancer_studies, 5
get_cancer_types, 6
get_case_lists, 7
get_clinical_data, 8
gene_identifiers, 10
get_genetic_profiles, 9
gene_identifiers, 10
get_molecular_data, 11
get_mutation_data, 12
make_groups, 13