Package 'UCSCXenaShiny'

May 15, 2024

Title Interactive Analysis of UCSC Xena Data

Version 2.1.0

Maintainer Shixiang Wang <w_shixiang@163.com>

Description Provides functions and a Shiny application for downloading, analyzing and visualizing datasets from UCSC Xena (<http://xena.ucsc.edu/>), which is a collection of UCSC-hosted public databases such as TCGA, ICGC, TARGET, GTEx, CCLE, and others.

License GPL (>= 3)

URL https://github.com/openbiox/UCSCXenaShiny, https://openbiox.github.io/UCSCXenaShiny/

BugReports https://github.com/openbiox/UCSCXenaShiny/issues

Depends R (>= 3.5)

Imports digest, dplyr (>= 0.8.3), ezcox, forcats, ggplot2 (>= 3.2.0), ggpubr (>= 0.2), httr, magrittr (>= 1.5), ppcor, psych, purrr, rlang, shiny (>= 1.3.2), stats, stringr, tibble (>= 2.1.3), tidyr, UCSCXenaTools, utils

Suggests covr (>= 3.2.1), cowplot, DT (>= 0.5), furrr, future, ggrepel, ggstatsplot, knitr, pacman, plotly, plyr, RColorBrewer (>= 1.1.2), rmarkdown, Rtsne, scales, survival, survminer, testthat (>= 2.0.1), umap,

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

NeedsCompilation no

Author Shixiang Wang [aut, cre] (<https://orcid.org/0000-0001-9855-7357>), Shensuo Li [aut], Yi Xiong [aut] (<https://orcid.org/0000-0002-4370-9824>), Longfei Zhao [aut] (<https://orcid.org/0000-0002-6277-0137>), Kai Gu [aut] (<https://orcid.org/0000-0002-0177-0774>), Yin Li [aut], Fei Zhao [aut]

Repository CRAN

Date/Publication 2024-05-15 14:10:06 UTC

R topics documented:

.opt_pancan . . . 3
analyze_gene_drug_response_asso . . . 4
analyze_gene_drug_response_diff . . . 5
app_run . . . 6
available_hosts . . . 7
ccle_absolute . . . 7
ccle_info . . . 8
ccle_info_fine . . . 8
ezcor . . . 9
ezcor_batch . . . 10
ezcor_partial_cor . . . 11
get_ccle_cn_value . . . 12
keep_cat_cols . . . 15
load_data . . . 15
mol_quick_analysis . . . 17
pcawg_info . . . 17
pcawg_info_fine . . . 18
pcawg_purity . . . 18
query_general_value . . . 19
query_molecule_value . . . 20
query_pancan_value . . . 21
query_tcga_group . . . 23
query_toil_value_df . . . 24
tcga survival analysis . . . 25
TCGA.organ . . . 27
tcga_clinical . . . 27
tcga_clinical_fine . . . 28
tcga_genome_instability . . . 28
tcga_gtex . . . 29
tcga_purity . . . 29
tcga_subtypes . . . 30
tcga_surv . . . 30
tcga_tmb . . . 31
toil_info . . . 31
vis_ccle_gene_cor . . . 32
vis_ccle_tpm . . . 33
vis_dim_dist . . . 33
vis_gene_cor . . . 35
vis_gene_cor_cancer . . . 36
vis_gene_drug_response_asso . . . 37
vis_gene_drug_response_diff . . . 37
vis_gene_immune_cor . . . 38

vis_gene_msi_cor . . . 39
vis_gene_pw_cor . . . 40
vis_gene_stemness_cor . . . 41
vis_gene_TIL_cor . . . 42
vis_gene_tmb_cor . . . 43
vis_identifier_cor . . . 44
vis_identifier_dim_dist . . . 45
vis_identifier_grp_comparison . . . 46
vis_identifier_grp_surv . . . 48
vis_identifier_multi_cor . . . 50
vis_pancan_anatomy . . . 51
vis_pcawg_dist . . . 52
vis_pcawg_gene_cor . . . 53
vis_pcawg_unicox_tree . . . 54
vis_toil_Mut . . . 55
vis_toil_Mut_cancer . . . 56
vis_toil_TvsN . . . 57
vis_toil_TvsN_cancer . . . 59
vis_unicox_tree . . . 60

Index **61**

`.opt_pancan` *A default setting for pan-cancer studies*

Description

A default setting for pan-cancer studies

Usage

```
.opt_pancan
```

Format

An object of class `list` of length 16.

analyze_gene_drug_response_asso *Analyze Association between Gene (Signature) and Drug Response with CCLE Data*

Description

Analyze partial correlation of gene-drug association after controlling for tissue average expression.

Usage

```
analyze_gene_drug_response_asso(gene_list, combine = FALSE)
```

Arguments

gene_list	a gene symbol list.
combine	if TRUE, combine the expression of gene list as a gene signature.

Value

a data.frame

- If combine is TRUE, genes are combined as signature.
- mean.diff and median.diff indicate mean and median of normalized expression difference between High IC50 cells and Low IC50 cells. The cutoff between High and Low are median IC50.

Examples

```
## Not run:
analyze_gene_drug_response_asso("TP53")
analyze_gene_drug_response_asso(c("TP53", "KRAS"))
analyze_gene_drug_response_asso(c("TP53", "KRAS"), combine = TRUE)

# Visualization
vis_gene_drug_response_asso("TP53")

## End(Not run)
```

analyze_gene_drug_response_diff

Analyze Difference of Drug Response (IC50 Value (uM)) between Gene (Signature) High and Low Expression with CCLE Data

Description

Analyze Difference of Drug Response (IC50 Value (uM)) between Gene (Signature) High and Low Expression with CCLE Data

Usage

```
analyze_gene_drug_response_diff(
  gene_list,
  drug = "ALL",
  tissue = "ALL",
  combine = FALSE,
  cutpoint = c(50, 50)
)
```

Arguments

gene_list	a gene symbol list.
drug	a drug name. Check examples.
tissue	a tissue name. Check examples.
combine	if TRUE, combine the expression of gene list as a gene signature.
cutpoint	cut point (in percent) for High and Low group, default is c(50, 50).

Value

a data.frame.

Examples

```
tissue_list <- c(
 "prostate", "central_nervous_system", "urinary_tract", "haematopoietic_and_lymphoid_tissue",
  "kidney", "thyroid", "soft_tissue", "skin", "salivary_gland",
  "ovary", "lung", "bone", "endometrium", "pancreas", "breast",
  "large_intestine", "upper_aerodigestive_tract", "autonomic_ganglia",
  "stomach", "liver", "biliary_tract", "pleura", "oesophagus"
)

drug_list <- c(
  "AEW541", "Nilotinib", "17-AAG", "PHA-665752", "Lapatinib",
  "Nutlin-3", "AZD0530", "PF2341066", "L-685458", "ZD-6474", "Panobinostat",
  "Sorafenib", "Irinotecan", "Topotecan", "LBW242", "PD-0325901",
  "PD-0332991", "Paclitaxel", "AZD6244", "PLX4720", "RAF265", "TAE684",
```

```
  "TKI258", "Erlotinib"
)

target_list <- c(
  "IGF1R", "ABL", "HSP90", "c-MET", "EGFR", "MDM2", "GS", "HDAC",
  "RTK", "TOP1", "XIAP", "MEK", "CDK4", "TUBB1", "RAF", "ALK", "FGFR"
)
## Not run:
analyze_gene_drug_response_diff("TP53")
analyze_gene_drug_response_diff(c("TP53", "KRAS"), drug = "AEW541")
analyze_gene_drug_response_diff(c("TP53", "KRAS"),
  tissue = "kidney",
  combine = TRUE
)

# Visualization
vis_gene_drug_response_diff("TP53")

## End(Not run)
```

---

app_run *Run UCSC Xena Shiny App*

Description

Run UCSC Xena Shiny App

Usage

```
app_run(runMode = "client", port = getOption("shiny.port"))
```

Arguments

runMode	default is 'client' for personal user, set it to 'server' for running on server.
port	The TCP port that the application should listen on. If the `port` is not specified, and the `shiny.port` option is set (with `options(shiny.port = XX)`), then that port will be used. Otherwise, use a random port between 3000:8000, excluding ports that are blocked by Google Chrome for being considered unsafe: 3659, 4045, 5060, 5061, 6000, 6566, 6665:6669 and 6697. Up to twenty random ports will be tried.

Examples

```
## Not run:
app_run()

## End(Not run)
```

available_hosts *Show Available Hosts*

Description

Show Available Hosts

Usage

```
available_hosts()
```

Value

hosts

Examples

```
available_hosts()
```

ccle_absolute *ABSOLUTE Result of CCLE Database*

Description

ABSOLUTE Result of CCLE Database

Format

A `data.frame`

Source

see "data_source" attribute.

Examples

```
data("ccle_absolute")
```

ccle_info *Phenotype Info of CCLE Database*

Description

Phenotype Info of CCLE Database

Format

A `data.frame`

Source

UCSC Xena.

Examples

```
data("ccle_info")
```

ccle_info_fine *Cleaned Phenotype Info of CCLE Database for grouping*

Description

Cleaned Phenotype Info of CCLE Database for grouping

Format

A `data.frame`

Source

UCSC Xena.

Examples

```
data("ccle_info_fine")
```

ezcor *Run Correlation between Two Variables and Support Group by a Variable*

Description

Run Correlation between Two Variables and Support Group by a Variable

Usage

```
ezcor(
  data = NULL,
  split = FALSE,
  split_var = NULL,
  var1 = NULL,
  var2 = NULL,
  cor_method = "pearson",
  adjust_method = "none",
  use = "complete",
  sig_label = TRUE,
  verbose = TRUE
)
```

Arguments

data	a data.frame containing variables
split	whether perform correlation grouped by a variable, default is 'FALSE'
split_var	a character, the group variable
var1	a character, the first variable in correlation
var2	a character, the second variable in correlation
cor_method	method="pearson" is the default value. The alternatives to be passed to cor are "spearman" and "kendall"
adjust_method	What adjustment for multiple tests should be used? ("holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none")
use	use="pairwise" will do pairwise deletion of cases. use="complete" will select just complete cases
sig_label	whether add symbal of significance. P < 0.001,***; P < 0.01,**; P < 0.05,*; P >=0.05,""
verbose	if TRUE, print extra info.

Value

a data.frame

Author(s)

Yi Xiong

`ezcor_batch` *Run correlation between two variables in a batch mode and support group by a variable*

Description

Run correlation between two variables in a batch mode and support group by a variable

Usage

```
ezcor_batch(
  data,
  var1,
  var2,
  split = FALSE,
  split_var = NULL,
  cor_method = "pearson",
  adjust_method = "none",
  use = "complete",
  sig_label = TRUE,
  parallel = FALSE,
  verbose = FALSE
)
```

Arguments

`data`	a `data.frame` containing variables
`var1`	a character, the first variable in correlation
`var2`	a character, the second variable in correlation
`split`	whether perform correlation grouped by a variable, default is 'FALSE'
`split_var`	a `character`, the group variable
`cor_method`	method="pearson" is the default value. The alternatives to be passed to cor are "spearman" and "kendall"
`adjust_method`	What adjustment for multiple tests should be used? ("holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none")
`use`	use="pairwise" will do pairwise deletion of cases. use="complete" will select just complete cases
`sig_label`	whether add symbal of significance. P < 0.001,***; P < 0.01,**; P < 0.05,*; P >=0.05,""
`parallel`	if `TRUE`, do parallel computation by **furrr** package.
`verbose`	if `TRUE`, print extra info.

Value

a data.frame

Author(s)

Yi Xiong, Shixiang Wang

---

ezcor_partial_cor *Run partial correlation*

Description

Run partial correlation

Usage

```
ezcor_partial_cor(
  data = NULL,
  split = FALSE,
  split_var = NULL,
  var1 = NULL,
  var2 = NULL,
  var3 = NULL,
  cor_method = "pearson",
  sig_label = TRUE,
  ...
)
```

Arguments

data	a data.frame containing variables
split	whether perform correlation grouped by a variable, default is 'FALSE'
split_var	a character, the group variable
var1	a character, the first variable in correlation
var2	a character, the second variable in correlation
var3	a character or character vector, the third variable in correlation
cor_method	method="pearson" is the default value. The alternatives to be passed to cor are "spearman" and "kendall"
sig_label	whether add symbal of significance. P < 0.001,"**'; P < 0.01,'**"*'; P < 0.05,'*"; P >=0.05,""
...	other arguments passed to methods

Value

a data.frame

Author(s)

Yi Xiong

See Also

`ppcor::pcor.test()` which this function wraps.

`get_ccle_cn_value` *Fetch Identifier Value from Pan-cancer Dataset*

Description

Identifier includes gene/probe etc.

Usage

```
get_ccle_cn_value(identifier)

get_ccle_gene_value(identifier, norm = c("rpkm", "nc"))

get_ccle_protein_value(identifier)

get_ccle_mutation_status(identifier)

get_pancan_value(
  identifier,
  subtype = NULL,
  dataset = NULL,
  host = available_hosts(),
  samples = NULL,
  ...
)

get_pancan_gene_value(identifier, norm = c("tpm", "fpkm", "nc"))

get_pancan_transcript_value(identifier, norm = c("tpm", "fpkm", "isopct"))

get_pancan_protein_value(identifier)

get_pancan_mutation_status(identifier)

get_pancan_cn_value(identifier, gistic2 = TRUE, use_thresholded_data = FALSE)

get_pancan_methylation_value(
  identifier,
  type = c("450K", "27K"),
  rule_out = NULL,
```

```
  aggr = c("NA", "mean", "Q0", "Q25", "Q50", "Q75", "Q100")
)

get_pancan_miRNA_value(identifier)

get_pcawg_gene_value(identifier)

get_pcawg_fusion_value(identifier)

get_pcawg_promoter_value(identifier, type = c("raw", "relative", "outlier"))

get_pcawg_miRNA_value(identifier, norm = c("TMM", "UQ"))

get_pcawg_APOBEC_mutagenesis_value(
  identifier = c("tCa_MutLoad_MinEstimate", "APOBECtCa_enrich", "A3A_or_A3B",
    "APOBEC_tCa_enrich_quartile", "APOBECrtCa_enrich", "APOBECytCa_enrich",
   "APOBECytCa_enrich-APOBECrtCa_enrich", "BH_Fisher_p-value_tCa", "ntca+tgan",
    "rtCa_to_G+rtCa_to_T", "rtca+tgay", "tCa_to_G+tCa_to_T",
  "ytCa_rtCa_BH_Fisher_p-value", "ytCa_rtCa_Fisher_p-value", "ytCa_to_G+ytCa_to_T",
    "ytca+tgar")
)
```

Arguments

identifier	a length-1 character representing a gene symbol, ensembl gene id, or probe id. Gene symbol is highly recommended.
norm	the normalization method.
subtype	a length-1 chracter representing a regular expression for matching DataSubtype column of UCSCXenaTools::XenaData.
dataset	a length-1 chracter representing a regular expression for matching XenaDatasets of UCSCXenaTools::XenaData.
host	a character vector representing host name(s), e.g. "toilHub".
samples	a character vector representing samples want to be returned.
...	other parameters.
gistic2	if TRUE (default), use GISTIC2 data.
use_thresholded_data	if TRUE, use GISTIC2-thresholded value.
type	methylation type, one of "450K" and "27K". for function get_pcawg_promoter_value, it can be one of "raw", "relative", "outlier".
rule_out	methylation sites to rule out before analyzing.
aggr	apporaches to aggregate the methylation data, default is 'NA', in such case, a mean value is obtained for gene-level methylation. Allowed value is one of c("NA", "mean", "Q0", "Q25", "Q50", "Q75", "Q100"). Here, Q50 is median.

Value

a named vector or `list`.

Functions

- `get_ccle_cn_value()`: Fetch copy number value from CCLE dataset
- `get_ccle_gene_value()`: Fetch gene expression value from CCLE dataset
- `get_ccle_protein_value()`: Fetch gene protein expression value from CCLE dataset
- `get_ccle_mutation_status()`: Fetch gene mutation info from CCLE dataset
- `get_pancan_value()`: Fetch identifier value from pan-cancer dataset
- `get_pancan_gene_value()`: Fetch gene expression value from pan-cancer dataset
- `get_pancan_transcript_value()`: Fetch gene transcript expression value from pan-cancer dataset
- `get_pancan_protein_value()`: Fetch protein expression value from pan-cancer dataset
- `get_pancan_mutation_status()`: Fetch mutation status value from pan-cancer dataset
- `get_pancan_cn_value()`: Fetch gene copy number value from pan-cancer dataset processed by GISTIC 2.0
- `get_pancan_methylation_value()`: Fetch gene expression value from CCLE dataset
- `get_pancan_miRNA_value()`: Fetch miRNA expression value from pan-cancer dataset
- `get_pcawg_gene_value()`: Fetch specimen-level gene expression value from PCAWG cohort
- `get_pcawg_fusion_value()`: Fetch specimen-level gene fusion value from PCAWG cohort
- `get_pcawg_promoter_value()`: Fetch specimen-level gene promoter activity value from PCAWG cohort
- `get_pcawg_miRNA_value()`: Fetch specimen-level miRNA value from PCAWG cohort
- `get_pcawg_APOBEC_mutagenesis_value()`: Fetch specimen-level gene fusion value from PCAWG cohort

Examples

```
## Not run:
# Fetch TP53 expression value from pan-cancer dataset
t1 <- get_pancan_value("TP53",
  dataset = "TcgaTargetGtex_rsem_isoform_tpm",
  host = "toilHub"
)
t2 <- get_pancan_gene_value("TP53")
t3 <- get_pancan_protein_value("AKT")
t4 <- get_pancan_mutation_status("TP53")
t5 <- get_pancan_cn_value("TP53")

## End(Not run)
```

keep_cat_cols *Keep Only Columns Used for Sample Selection*

Description

Keep Only Columns Used for Sample Selection

Usage

```
keep_cat_cols(x, keep_sam_cols = TRUE, return_idx = TRUE)
```

Arguments

x	a data.frame with many columns.
keep_sam_cols	if TRUE (default), keep columns with pattern 'sample', 'patient', etc.
return_idx	if TRUE (default), return index of 5 (at most) columns, it is useful in Shiny.

Value

a data.frame or a list.

load_data *Load Dataset Provided by This Package*

Description

Load data from builtin or Zenodo. Option xena.zenodoDir can be used to set default path for storing extra data from Zenodo, e.g., options(xena.zenodoDir = "/home/xxx/dataset").

Usage

```
load_data(name)
```

Arguments

name	a dataset name. Could be one of **Builtin datasets**: • ccle_absolute: CCLE ABSOLUTE result. • ccle_info: CCLE information. • ccle_info_fine: cleaned CCLE information for TPC analysis. • pcawg_info: PCAWG information. • pcawg_info_fine: cleaned PCAWG information for TPC analysis. • pcawg_purity: PCAWG tumor purity, ploidy and WGD data. • tcga_clinical: TCGA clinical data.

- tcga_clinical_fine: cleaned TCGA information for TPC analysis.
- tcga_genome_instability: TCGA genome instability data.
- tcga_gtex: TCGA and GTEX sample info.
- tcga_purity: TCGA tumor purity data.
- tcga_subtypes: TCGA subtypes data.
- tcga_surv: TCGA survival data.
- TCGA.organ: TCGA organ data.
- toil_info: Toil hub information.

Remote datasets stored in Zenodo:

- pcawg_promoter_id: PCAWG promoter identifiers.
- transcript_identifier: Common transcript identifiers.
- ccle_expr_and_drug_response: CCLE expression and drug response data.
- ccle_drug_response_extend: CCLE drug response extended data.
- pancan_MSI: Pan-cancer MSI data.
- tcga_chr_alteration: TCGA chromosome alteration data.
- tcga_MSI: TCGA MSI data.
- tcga_pan_immune_signature: TCGA pan-cancer immune signature.
- tcga_stemness: TCGA tumor stemness data.
- tcga_TIL: TCGA TIL data.
- tcga_PW: ssGSEA scores of HALLMARK, KEGG, IOBR terms for TCGA samples.
- tcga_PW_meta: metadata annotation for HALLMARK, KEGG, IOBR terms.
- tcga_tmb: TCGA TMB data.
- tcga_armcalls: TCGA arm alteration calls and Aneuploidy data.
- tcga_dna_repair: TCGA DNA repair data.
- pancancer_conserved_immune_subtype: Pan-cancer conserved immune subtypes.
- pcawg_TIL: PCAWG TIL data.
- pcawg_PW: ssGSEA scores of HALLMARK, KEGG, IOBR terms for PCAWG samples.
- ...

Value

a dataset, typically a data.frame.

Examples

```
data1 <- load_data("tcga_surv")
data1

data2 <- load_data("tcga_armcalls")
data2
```

mol_quick_analysis *Quick molecule analysis and report generation*

Description

Quick molecule analysis and report generation

Usage

```
mol_quick_analysis(molecule, data_type, out_dir = ".", out_report = FALSE)
```

Arguments

molecule	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	data type. Can be one of "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
out_dir	path to save analysis result and report, default is '.'
out_report	logical value wheather to generate html report

Value

a list.

pcawg_info *Phenotype Info of PCAWG Database*

Description

Phenotype Info of PCAWG Database

Format

A data.frame

Source

UCSC Xena.

Examples

```
data("pcawg_info")
```

pcawg_info_fine *Cleaned Phenotype Info of PCAWG Database for grouping*

Description

Cleaned Phenotype Info of PCAWG Database for grouping

Format

A `data.frame`

Source

UCSC Xena.

Examples

```
data("pcawg_info_fine")
```

pcawg_purity *Purity Data of PCAWG*

Description

Purity Data of PCAWG

Format

A `data.frame`

Source

UCSC Xena.

Examples

```
data("pcawg_purity")
```

query_general_value *download data for shiny general analysis*

Description

download data for shiny general analysis

Usage

```
query_general_value(
  L1,
  L2,
  L3,
  database = c("toil", "pcawg", "ccle"),
  tpc_value_nonomics = NULL,
  opt_pancan = NULL,
  custom_metadata = NULL
)
```

Arguments

L1	level 1 main datatype
L2	level 2 sub datatype
L3	level 3 identifier
database	one of c("toil","pcawg","ccle")
tpc_value_nonomics	non-omics matrix data of one database
opt_pancan	molecular datasets parameters
custom_metadata	user customized metadata

Examples

```
## Not run:
general_value_id = UCSCXenaShiny:::query_general_id()
tcga_value_option = general_value_id[["value"]][[1]]
tcga_index_value = tcga_value_option[["Tumor index"]]
tcga_immune_value = tcga_value_option[["Immune Infiltration"]]
tcga_pathway_value = tcga_value_option[["Pathway activity"]]
tcga_phenotype_value = tcga_value_option[["Phenotype data"]]
clinical_phe = tcga_phenotype_value[["Clinical Phenotype"]]
x_data = UCSCXenaShiny:::query_general_value(
           "Molecular profile", "mRNA Expression", "TP53", "toil",
           tcga_index_value, tcga_immune_value, tcga_pathway_value,
           clinical_phe)

y_data = UCSCXenaShiny:::query_general_value(
```

```
            "Immune Infiltration", "CIBERSORT", "Monocyte", "toil",
            tcga_index_value, tcga_immune_value, tcga_pathway_value,
            clinical_phe)

## End(Not run)
```

`query_molecule_value` *Get Molecule or Signature Data Values from Dense (Genomic) Matrix Dataset of UCSC Xena Data Hubs*

Description

Get Molecule or Signature Data Values from Dense (Genomic) Matrix Dataset of UCSC Xena Data Hubs

Usage

```
query_molecule_value(dataset, molecule, host = NULL)
```

Arguments

`dataset`	a UCSC Xena dataset in dense matrix format (rows are features (e.g., gene, cell line) and columns are samples).
`molecule`	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature (`"TP53 + 2 * KRAS - 1.3 * PTEN"`). **NOTE**, when a signature is specified, a space must exist in the input.
`host`	a UCSC Xena host, default is `NULL`, auto-detect from the dataset.

Value

a named vector.

Examples

```
# What does dense matrix mean?
table(UCSCXenaTools::XenaData$Type)
# It is a the UCSC Xena dataset with "Type" equals to "genomicMatrix"
## Not run:
dataset <- "ccle/CCLE_copynumber_byGene_2013-12-03"
x <- query_molecule_value(dataset, "TP53")
head(x)

signature <- "TP53 + 2*KRAS - 1.3*PTEN" # a space must exist in the string
y <- query_molecule_value(dataset, signature)
head(y)

## End(Not run)
```

query_pancan_value *Query Single Identifier or Signature Value from Pan-cancer Database*

Description

Query Single Identifier or Signature Value from Pan-cancer Database

Usage

```
query_pancan_value(
  molecule,
 data_type = c("mRNA", "transcript", "protein", "mutation", "cnv", "methylation",
    "miRNA", "fusion", "promoter", "APOBEC"),
  database = c("toil", "ccle", "pcawg"),
  reset_id = NULL,
  opt_pancan = .opt_pancan
)
```

Arguments

molecule	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	data type. Can be one of "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
database	database, either 'toil' for TCGA TARGET GTEx, or 'ccle' for CCLE.
reset_id	if not NULL, set the specified variable at parent frame to "Signature".
opt_pancan	other extra parameters passing to the underlying functions.

Details

query_pancan_value() provide convenient interface to download multi-omics data from 3 databases by specifying one or several canonical datasets. It is derived from query_pancan_value() and support query for genomic signature. To query comprehensive datasets that UCSCXena supports, users can check UCSCXenaTools::XenaData and use get_pancan_value() directly.

Option opt_pancan is a nested list and allow to adjust the downloading details. For now, only cnv(toil),methylation(toil),miRNA(toil),miRNA(pcawg),promoter(pcawg) support optional parameters. The default set is .opt_pancan and we check meanings of sublist(parameters) through the following relationship.

Value

a list.

"toil" database

1. mRNA–`get_pancan_gene_value()`
2. transcript–`get_pancan_transcript_value()`
3. protein–`get_pancan_protein_value()`
4. mutation–`get_pancan_mutation_status()`
5. cnv–`get_pancan_cn_value()`
6. methylation–`get_pancan_methylation_value()`
7. miRNA–`get_pancan_miRNA_value()`

"ccle" database

1. mRNA–`get_ccle_gene_value()`
2. protein–`get_ccle_protein_value()`
3. mutation–`get_ccle_mutation_status()`
4. cnv–`get_ccle_cn_value()`

"pcawg" database

1. mRNA–`get_pcawg_gene_value()`
2. miRNA–`get_pcawg_miRNA_value()`
3. promoter–`get_pcawg_promoter_value()`
4. fusion–`get_pcawg_fusion_value()`
5. APOBEC–`get_pcawg_APOBEC_mutagenesis_value()`

Examples

```
## Not run:
query_pancan_value("KRAS")
query_pancan_value("KRAS", database = "ccle")
query_pancan_value("KRAS", database = "pcawg")


query_pancan_value("ENSG00000000419",
  database = "pcawg",
  data_type = "fusion"
) # gene symbol also work

.opt_pancan

opt_pancan = list(toil_cnv = list(use_thresholded_data = FALSE))
query_pancan_value("PTEN",data_type = "cnv", database = "toil", opt_pancan = opt_pancan)


opt_pancan = list(toil_methylation = list(type = "450K",rule_out = "cg21115430", aggr = "Q25"))
query_pancan_value("PTEN",data_type = "methylation", database = "toil", opt_pancan = opt_pancan)


## End(Not run)
```

query_tcga_group *Group TPC samples by build-in or custom phenotype and support filtering or merging operations*

Description

Group TPC samples by build-in or custom phenotype and support filtering or merging operations

Usage

```
query_tcga_group(
  database = c("toil", "pcawg", "ccle"),
  cancer = NULL,
  custom = NULL,
  group = "Gender",
  filter_by = NULL,
  filter_id = NULL,
  merge_by = NULL,
  merge_quantile = FALSE,
  return_all = FALSE
)
```

Arguments

database	one of c("toil","pcawg","ccle")
cancer	select cancer cohort(s)
custom	upload custom phenotype data
group	target group names
filter_by	filter samples by one or multiple criterion
filter_id	directly filter samples by provided sample ids
merge_by	merge the target group for main categories
merge_quantile	whether to merge numerical variable by percentiles
return_all	return the all phenotype data

Value

a list object with grouping samples and statistics

Examples

```
## Not run:
query_tcga_group(group = "Age")

query_tcga_group(cancer="BRCA",
              group = "Stage_ajcc"
```

```
                )

query_tcga_group(cancer="BRCA",
                group = "Stage_ajcc",
                filter_by = list(
                  c("Code",c("TP"),"+"),
                  c("Stage_ajcc",c(NA),"-"))
                )

query_tcga_group(cancer="BRCA",
                  group = "Stage_ajcc",
                  filter_by = list(
                    c("Age",c(0.5),"%>"))
)

query_tcga_group(cancer="BRCA",
                  group = "Stage_ajcc",
                  filter_by = list(
                    c("Age",c(60),">"))
)

query_tcga_group(cancer="BRCA",
                group = "Stage_ajcc",
                merge_by = list(
                  "Early"=c("Stage I"),
                  "Late" = c("Stage II","Stage III","Stage IV"))
                )

query_tcga_group(cancer="BRCA",
                  group = "Age",
                  merge_by = list(
                    "Young"= c(20, 60),
                    "Old"= c(60, NA)
                  )
)

query_tcga_group(cancer="BRCA",
                  group = "Age",
                  merge_quantile = TRUE,
                  merge_by = list(
                    "Young"= c(0, 0.5),
                    "Old"= c(0.5, 1)
                  )
)

## End(Not run)
```

---

query_toil_value_df *Obtain ToilHub Info for Single Molecule*

Description

Obtain ToilHub Info for Single Molecule

Obtain ToilHub Info for Single Gene

Usage

```
query_toil_value_df(identifier = "TP53")

query_toil_value_df(identifier = "TP53")
```

Arguments

identifier	a length-1 character representing a gene symbol, ensembl gene id, or probe id. Gene symbol is highly recommended.

Value

a tibble

a tibble

Examples

```
## Not run:
t <- query_toil_value_df()
t

## End(Not run)
## Not run:
t <- query_toil_value_df()
t

## End(Not run)
```

---

tcga survival analysis

TCGA Survival Analysis

Description

- Firstly, get merged data of one molecular profile value and associated clinical data from TCGA Pan-Cancer dataset.
- Secondly, filter data as your wish.
- Finally, show K-M plot.

Usage

```
tcga_surv_get(
  item,
  TCGA_cohort = "LUAD",
 profile = c("mRNA", "miRNA", "methylation", "transcript", "protein", "mutation", "cnv"),
 TCGA_cli_data = dplyr::full_join(load_data("tcga_clinical"), load_data("tcga_surv"), by
    = "sample"),
  opt_pancan = .opt_pancan
)

tcga_surv_plot(
  data,
  time = "time",
  status = "status",
  cutoff_mode = c("Auto", "Custom"),
  cutpoint = c(50, 50),
  cnv_type = c("Duplicated", "Normal", "Deleted"),
 profile = c("mRNA", "miRNA", "methylation", "transcript", "protein", "mutation", "cnv"),
  palette = "aaas",
  ...
)
```

Arguments

item	a molecular identifier, can be gene symbol (common cases), protein symbol, etc.
TCGA_cohort	a TCGA cohort, e.g. "LUAD" (default), "LUSC", "ACC".
profile	a molecular profile. Option can be one of "mRNA" (default), "miRNA", "methylation", "transcript", "protein", "mutation", "cnv".
TCGA_cli_data	a data.frame containing TCGA clinical data. Default use pre-compiled TCGA clinical data in this package.
opt_pancan	specify one dataset for some molercular profiles
data	a subset of result from tcga_surv_get().
time	the column name for "time".
status	the column name for "status".
cutoff_mode	mode for grouping samples, can be "Auto" (default) or "Custom".
cutpoint	cut point (in percent) for "Custom" mode, default is c(50, 50).
cnv_type	only used when profile is "cnv", can select from c("Duplicated", "Normal", "Deleted").
palette	color palette, can be "hue", "grey", "RdBu", "Blues", "npg", "aaas", etc. More see ?survminer::ggsurvplot.
...	other parameters passing to survminer::ggsurvplot

Value

a data.frame or a plot.

Examples

```
## Not run:
# 1. get data
data <- tcga_surv_get("TP53")
# 2. filter data (optional)

# 3. show K-M plot
tcga_surv_plot(data, time = "DSS.time", status = "DSS")

## End(Not run)
```

`TCGA.organ` *TCGA: Organ Data*

Description

TCGA: Organ Data

Format

A data.frame

Examples

```
data("TCGA.organ")
```

`tcga_clinical` *Toil Hub: TCGA Clinical Data*

Description

See `tcga_surv` for TCGA survival data.

Format

A data.frame

Source

Generate from data-raw

Examples

```
data("tcga_clinical")
```

tcga_clinical_fine *Toil Hub: Cleaned TCGA Clinical Data for grouping*

Description

See tcga_surv for TCGA survival data.

Format

A data.frame

Source

Generate from data-raw

Examples

```
data("tcga_clinical_fine")
```

tcga_genome_instability *TCGA: Genome Instability Data*

Description

TCGA: Genome Instability Data

Format

A data.frame

Source

https://gdc.cancer.gov/about-data/publications/PanCanStemness-2018

Examples

```
data("tcga_genome_instability")
```

tcga_gtex *Toil Hub: Merged TCGA GTEx Selected Phenotype*

Description

Toil Hub: Merged TCGA GTEx Selected Phenotype

Format

A data.frame

Examples

```
data("tcga_gtex")
```

tcga_purity *TCGA: Purity Data*

Description

TCGA: Purity Data

Format

A data.frame

Source

https://www.nature.com/articles/ncomms9971#Sec14

Examples

```
data("tcga_purity")
```

tcga_subtypes *TCGA Subtype Data*

Description

TCGA Subtype Data

Format

A data.frame

Source

UCSC Xena.

Examples

```
data("tcga_subtypes")
```

tcga_surv *Toil Hub: TCGA Survival Data*

Description

Toil Hub: TCGA Survival Data

Format

A data.frame

Source

Generate from data-raw

Examples

```
data("tcga_surv")
```

tcga_tmb *TCGA: TMB (Tumor Mutation Burden) Data*

Description

TCGA: TMB (Tumor Mutation Burden) Data

Format

A data.frame

Source

https://gdc.cancer.gov/about-data/publications/panimmune

Examples

```
data("tcga_tmb")
```

toil_info *Toil Hub: TCGA TARGET GTEX Selected Phenotype*

Description

Toil Hub: TCGA TARGET GTEX Selected Phenotype

Format

A data.frame

Source

Generate from data-raw

Examples

```
data("toil_info")
```

vis_ccle_gene_cor *Visualize CCLE Gene Expression Correlation*

Description

Visualize CCLE Gene Expression Correlation

Usage

```
vis_ccle_gene_cor(
  Gene1 = "CSF1R",
  Gene2 = "JAK3",
  data_type1 = "mRNA",
  data_type2 = "mRNA",
  cor_method = "spearman",
  use_log_x = FALSE,
  use_log_y = FALSE,
  use_regline = TRUE,
  SitePrimary = "prostate",
  use_all = FALSE,
  alpha = 0.5,
  color = "#000000",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene1	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Gene2	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type1	choose gene profile type for the first gene, including "mRNA","transcript","methylation","miRNA","prote
data_type2	choose gene profile type for the second gene, including "mRNA","transcript","methylation","miRNA","pr
cor_method	correlation method
use_log_x	if TRUE, log X values.
use_log_y	if TRUE, log Y values.
use_regline	if TRUE, add regression line.
SitePrimary	select cell line origin tissue.
use_all	use all sample, default FALSE.
alpha	dot alpha.
color	dot color.
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

`vis_ccle_tpm` *Visualize CCLE Gene Expression*

Description

Visualize CCLE Gene Expression

Usage

```
vis_ccle_tpm(
  Gene = "TP53",
  data_type = "mRNA",
  use_log = FALSE,
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	support genomic profile for CCLE, currently "mRNA", "protein","cnv" are supported
use_log	if TRUE, log values.
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

`vis_dim_dist` *Visualize the distribution difference of samples after dimension reduction analysis*

Description

Visualize the distribution difference of samples after dimension reduction analysis

Usage

```
vis_dim_dist(
  ids = c("TP53", "KRAS", "PTEN", "MDM2", "CDKN1A"),
  data_type = "mRNA",
  group_info = NULL,
  DR_method = c("PCA", "UMAP", "tSNE"),
  palette = "Set1",
  add_margin = NULL,
  opt_pancan = .opt_pancan
)
```

Arguments

ids	molecular identifiers (>=3)
data_type	molecular types, refer to query_pancan_value() function
group_info	two-column grouping information with names 'Sample','Group'
DR_method	the dimension reduction method
palette	the color setting of RColorBrewer
add_margin	the marginal plot (NULL, "density", "boxplot")
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object or rawdata list

Examples

```
## Not run:
group_info = tcga_clinical_fine %>%
  dplyr::filter(Cancer=="BRCA") %>%
  dplyr::select(Sample, Code) %>%
  dplyr::rename(Group=Code)

vis_dim_dist(
  ids = c("TP53", "KRAS", "PTEN", "MDM2", "CDKN1A"),
  group_info = group_info
)


## End(Not run)
```

vis_gene_cor *Visualize Gene-Gene Correlation in TCGA*

Description

Visualize Gene-Gene Correlation in TCGA

Usage

```
vis_gene_cor(
  Gene1 = "CSF1R",
  Gene2 = "JAK3",
  data_type1 = "mRNA",
  data_type2 = "mRNA",
  use_regline = TRUE,
  purity_adj = TRUE,
  alpha = 0.5,
  color = "#000000",
  filter_tumor = TRUE,
  opt_pancan = .opt_pancan
)
```

Arguments

Gene1	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Gene2	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type1	choose gene profile type for the first gene, including "mRNA","transcript","methylation","miRNA","prote
data_type2	choose gene profile type for the second gene, including "mRNA","transcript","methylation","miRNA","pr
use_regline	if TRUE, add regression line.
purity_adj	whether performing partial correlation adjusted by purity
alpha	dot alpha.
color	dot color.
filter_tumor	whether use tumor sample only, default TRUE
opt_pancan	specify one dataset for some molercular profiles

vis_gene_cor_cancer *Visualize Gene-Gene Correlation in a TCGA Cancer Type*

Description

Visualize Gene-Gene Correlation in a TCGA Cancer Type

Usage

```
vis_gene_cor_cancer(
  Gene1 = "CSF1R",
  Gene2 = "JAK3",
  data_type1 = "mRNA",
  data_type2 = "mRNA",
  purity_adj = TRUE,
  cancer_choose = "GBM",
  use_regline = TRUE,
  cor_method = "spearman",
  use_all = FALSE,
  alpha = 0.5,
  color = "#000000",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene1	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Gene2	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type1	choose gene profile type for the first gene, including "mRNA","transcript","methylation","miRNA","prote
data_type2	choose gene profile type for the second gene, including "mRNA","transcript","methylation","miRNA","pr
purity_adj	whether performing partial correlation adjusted by purity
cancer_choose	TCGA cohort name, e.g. "ACC".
use_regline	if TRUE, add regression line.
cor_method	correlation method.
use_all	use all sample, default FALSE.
alpha	dot alpha.
color	dot color.
opt_pancan	specify one dataset for some molercular profiles

vis_gene_drug_response_asso

Visualize Gene and Drug-Target Association with CCLE Data

Description

See analyze_gene_drug_response_asso for examples.

Usage

```
vis_gene_drug_response_asso(
  Gene = "TP53",
  x_axis_type = c("mean.diff", "median.diff"),
  output_form = c("plotly", "ggplot2")
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
x_axis_type	set the value type for X axis.
output_form	plotly or ggplot2.

Value

plotly or ggplot2 object.

vis_gene_drug_response_diff

Visualize Gene and Drug Response Difference with CCLE Data

Description

See analyze_gene_drug_response_diff for examples.

Usage

```
vis_gene_drug_response_diff(
  Gene = "TP53",
  tissue = "lung",
  Show.P.label = TRUE,
  Method = "wilcox.test",
  values = c("#DF2020", "#DDDF21"),
  alpha = 0.5
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
tissue	select cell line origin tissue.
Show.P.label	TRUE or FALSE present p value with number or label *, **, *** and ****
Method	default method is wilcox.test
values	the color to fill tumor or normal
alpha	set alpha for dots.

Value

a ggplot object.

---

vis_gene_immune_cor *Heatmap for Correlation between Gene and Immune Signatures*

Description

Heatmap for Correlation between Gene and Immune Signatures

Usage

```
vis_gene_immune_cor(
  Gene = "TP53",
  cor_method = "spearman",
  data_type = "mRNA",
  Immune_sig_type = "Cibersort",
  Plot = "TRUE",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
cor_method	correlation method
data_type	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
Immune_sig_type	quantification method, default is "Cibersort"
Plot	output the plot directly, default 'TRUE'
opt_pancan	specify one dataset for some molercular profiles

Examples

```
## Not run:
p <- vis_gene_immune_cor(Gene = "TP53")

## End(Not run)
```

`vis_gene_msi_cor` *Visualize Correlation between Gene and MSI (Microsatellite instability)*

Description

Visualize Correlation between Gene and MSI (Microsatellite instability)

Usage

```
vis_gene_msi_cor(
  Gene = "TP53",
  cor_method = "spearman",
  data_type = "mRNA",
  Plot = "TRUE",
  opt_pancan = .opt_pancan
)
```

Arguments

`Gene`	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature (`"TP53 + 2 * KRAS - 1.3 * PTEN"`).
`cor_method`	correlation method
`data_type`	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
`Plot`	output the plot directly, default 'TRUE'
`opt_pancan`	specify one dataset for some molercular profiles

Examples

```
## Not run:
p <- vis_gene_msi_cor(Gene = "TP53")

## End(Not run)
```

vis_gene_pw_cor *Visualize Correlation between Gene and Pathway signature Score*

Description

Visualize Correlation between Gene and Pathway signature Score

Usage

```
vis_gene_pw_cor(
  Gene = "TP53",
  data_type = "mRNA",
  pw_name = "HALLMARK_ADIPOGENESIS",
  cancer_choose = "GBM",
  use_regline = TRUE,
  cor_method = "spearman",
  use_all = FALSE,
  alpha = 0.5,
  color = "#000000",
  filter_tumor = TRUE,
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
pw_name	the queried Pathway name, see the supported pathway from 'load("toil_sig_score")'default is NULL
cancer_choose	select cancer cohort(s)
use_regline	if TRUE, add regression line.
cor_method	select correlation coefficient (pearson/spearman)
use_all	use all sample, default FALSE.
alpha	dot alpha.
color	dot color.
filter_tumor	whether use tumor sample only, default TRUE
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object or dataframe

Examples

```
## Not run:
vis_gene_pw_cor(Gene = "TP53", data_type = "mRNA",
                pw_name = "HALLMARK_ADIPOGENESIS",
                cancer_choose = "BRCA")

## End(Not run)
```

`vis_gene_stemness_cor` *Visualize Correlation between Gene and Tumor Stemness*

Description

Visualize Correlation between Gene and Tumor Stemness

Usage

```
vis_gene_stemness_cor(
  Gene = "TP53",
  cor_method = "spearman",
  data_type = "mRNA",
  Plot = "TRUE",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
cor_method	correlation method
data_type	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
Plot	output the plot directly, default 'TRUE'
opt_pancan	specify one dataset for some molercular profiles

Examples

```
## Not run:
p <- vis_gene_stemness_cor(Gene = "TP53")
p

## End(Not run)
## To generate a radar plot, uncomment the following code
# pdata <- p$data %>%
#   dplyr::mutate(cor = round(cor, digits = 3), p.value = round(p.value, digits = 3))
#
# df <- pdata %>%
```

```
#   select(cor, cancer) %>%
#   pivot_wider(names_from = cancer, values_from = cor)
#
# ggradar::ggradar(
#   df[1, ],
#   font.radar = "sans",
#   values.radar = c("-1", "0", "1"),
#   grid.min = -1, grid.mid = 0, grid.max = 1,
#   # Background and grid lines
#   background.circle.colour = "white",
#   gridline.mid.colour = "grey",
#   # Polygons
#   group.line.width = 1,
#   group.point.size = 3,
#   group.colours = "#00AFBB") +
#   theme(plot.title = element_text(hjust = .5))
```

`vis_gene_TIL_cor` *Heatmap for Correlation between Gene and Tumor Immune Infiltration (TIL)*

Description

Heatmap for Correlation between Gene and Tumor Immune Infiltration (TIL)

Usage

```
vis_gene_TIL_cor(
  Gene = "TP53",
  cor_method = "spearman",
  data_type = "mRNA",
 sig = c("B cell_TIMER", "T cell CD4+_TIMER", "T cell CD8+_TIMER", "Neutrophil_TIMER",
    "Macrophage_TIMER", "Myeloid dendritic cell_TIMER"),
  Plot = "TRUE",
  opt_pancan = .opt_pancan
)
```

Arguments

`Gene`	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature (`"TP53 + 2 * KRAS - 1.3 * PTEN"`).
`cor_method`	correlation method
`data_type`	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
`sig`	Immune Signature, default: result from TIMER
`Plot`	output the plot directly, default 'TRUE'
`opt_pancan`	specify one dataset for some molercular profiles

Examples

```
## Not run:
p <- vis_gene_TIL_cor(Gene = "TP53")

## End(Not run)
```

`vis_gene_tmb_cor` *Visualize Correlation between Gene and TMB (Tumor Mutation Burden)*

Description

Visualize Correlation between Gene and TMB (Tumor Mutation Burden)

Usage

```
vis_gene_tmb_cor(
  Gene = "TP53",
  cor_method = "spearman",
  data_type = "mRNA",
  Plot = "TRUE",
  opt_pancan = .opt_pancan
)
```

Arguments

`Gene`	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature (`"TP53 + 2 * KRAS - 1.3 * PTEN"`).
`cor_method`	correlation method
`data_type`	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
`Plot`	output the plot directly, default 'TRUE'
`opt_pancan`	specify one dataset for some molercular profiles

Examples

```
## Not run:
p <- vis_gene_tmb_cor(Gene = "TP53")

## End(Not run)
```

`vis_identifier_cor` *Visualize Identifier-Identifier Correlation*

Description

NOTE: the dataset must be dense matrix in UCSC Xena data hubs.

Usage

```
vis_identifier_cor(
  dataset1,
  id1,
  dataset2,
  id2,
  samples = NULL,
  use_ggstats = FALSE,
  use_simple_axis_label = TRUE,
  line_color = "blue",
  alpha = 0.5,
  ...
)
```

Arguments

`dataset1`	the dataset to obtain `id1`.
`id1`	the first molecule identifier.
`dataset2`	the dataset to obtain `id2`.
`id2`	the second molecule identifier.
`samples`	default is `NULL`, can be common sample names for two datasets.
`use_ggstats`	if `TRUE`, use ggstatsplot package for plotting.
`use_simple_axis_label`	if `TRUE` (default), use simple axis labels. Otherwise, data subtype will be labeled.
`line_color`	set the color for regression line.
`alpha`	set the alpha for dots.
`...`	other parameters passing to ggscatter.

Value

a (gg)plot object.

Examples

```
## Not run:
dataset <- "TcgaTargetGtex_rsem_isoform_tpm"
id1 <- "TP53"
id2 <- "KRAS"
vis_identifier_cor(dataset, id1, dataset, id2)

samples <- c(
  "TCGA-D5-5538-01", "TCGA-VM-A8C8-01",
  "TCGA-ZN-A9VQ-01", "TCGA-EE-A17X-06",
  "TCGA-05-4420-01"
)
vis_identifier_cor(dataset, id1, dataset, id2, samples)

dataset1 <- "TCGA-BLCA.htseq_counts.tsv"
dataset2 <- "TCGA-BLCA.gistic.tsv"
id1 <- "TP53"
id2 <- "KRAS"
vis_identifier_cor(dataset1, id1, dataset2, id2)

## End(Not run)
```

---

vis_identifier_dim_dist

Visualize the distribution difference of samples after Molecule Identifier dimension reduction analysis

---

Description

NOTE: the dataset must be dense matrix in UCSC Xena data hubs.

Usage

```
vis_identifier_dim_dist(
  dataset = NULL,
  ids = NULL,
  grp_df,
  samples = NULL,
  return.data = FALSE,
  DR_method = c("PCA", "UMAP", "tSNE"),
  add_margin = NULL,
  palette = "Set1"
)
```

Arguments

dataset	the dataset to obtain identifiers.
ids	the molecule identifiers.

grp_df	When dataset and id are all not NULL, it should be a data.frame with 2 columns. • The first column refers to sample ID. • The second column refers to groups indicated in axis X.
samples	default is NULL, can be common sample names for two datasets.
return.data	whether to reture the raw meta/matrix data (list) instead of plot
DR_method	the dimension reduction method
add_margin	the marginal plot (NULL, "density", "boxplot")
palette	the color setting of RColorBrewer

Value

a ggplot object.

Examples

```
library(UCSCXenaTools)
expr_dataset <- "TCGA.LUAD.sampleMap/HiSeqV2_percentile"
ids = c("TP53", "KRAS", "PTEN", "MDM2", "CDKN1A")

cli_dataset <- "TCGA.LUAD.sampleMap/LUAD_clinicalMatrix"
cli_df <- XenaGenerate(
  subset = XenaDatasets == cli_dataset
) %>%
  XenaQuery() %>%
  XenaDownload() %>%
  XenaPrepare()
grp_df = cli_df[, c("sampleID", "gender")]
vis_identifier_dim_dist(expr_dataset, ids, grp_df, DR_method="PCA")
```

---

vis_identifier_grp_comparison

Visualize Comparison of an Molecule Identifier between Groups

---

Description

NOTE: the dataset must be dense matrix in UCSC Xena data hubs.

Usage

```
vis_identifier_grp_comparison(
  dataset = NULL,
  id = NULL,
  grp_df,
```

```
  samples = NULL,
  fun_type = c("betweenstats", "withinstats"),
  type = c("parametric", "nonparametric", "robust", "bayes"),
  pairwise.comparisons = TRUE,
 p.adjust.method = c("holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr",
    "none"),
  ggtheme = cowplot::theme_cowplot(),
  ...
)
```

Arguments

dataset	the dataset to obtain identifiers.
id	the molecule identifier.
grp_df	When dataset and id are all not NULL, it should be a data.frame with 2 or 3 columns. • The first column refers to sample ID. • The second column refers to groups indicated in axis X. • The third column is optional, which indicates facet variable. When any of dataset and id is NULL, it should be a data.frame with 3 or 4 columns. • The first column refers to sample ID. • The second column refers to values indicated in axis Y. • The third column refers to groups indicated in axis X. • The fourth column is optional, which indicates facet variable.
samples	default is NULL, can be common sample names for two datasets.
fun_type	select the function to compare groups.
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter.
pairwise.comparisons	whether pairwise comparison
p.adjust.method	Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
ggtheme	A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.

... other parameters passing to ggstatsplot::ggbetweenstats or ggstatsplot::ggwithinstats.

Value

a (gg)plot object.

Examples

```
## Not run:
library(UCSCXenaTools)
expr_dataset <- "TCGA.LUAD.sampleMap/HiSeqV2_percentile"
cli_dataset <- "TCGA.LUAD.sampleMap/LUAD_clinicalMatrix"
id <- "TP53"
cli_df <- XenaGenerate(
  subset = XenaDatasets == "TCGA.LUAD.sampleMap/LUAD_clinicalMatrix"
) %>%
  XenaQuery() %>%
  XenaDownload() %>%
  XenaPrepare()

# group data.frame with 2 columns
vis_identifier_grp_comparison(expr_dataset, id, cli_df[, c("sampleID", "gender")])
# group data.frame with 3 columns
vis_identifier_grp_comparison(
  expr_dataset, id,
  cli_df[, c("sampleID", "pathologic_M", "gender")] %>%
    dplyr::filter(pathologic_M %in% c("M0", "MX"))
)

# When not use the value of `identifier` from `dataset`
vis_identifier_grp_comparison(grp_df = cli_df[, c(1, 2, 71)])
vis_identifier_grp_comparison(grp_df = cli_df[, c(1, 2, 71, 111)])

## End(Not run)
```

---

vis_identifier_grp_surv

Visualize Identifier Group Survival Difference

Description

NOTE: the dataset must be dense matrix in UCSC Xena data hubs.

Usage

```
vis_identifier_grp_surv(
  dataset = NULL,
  id = NULL,
```

```
  surv_df,
  samples = NULL,
  cutoff_mode = c("Auto", "Custom", "None"),
  cutpoint = c(50, 50),
  palette = "aaas",
  ...
)
```

Arguments

dataset	the dataset to obtain identifiers.
id	the molecule identifier.
surv_df	a data.frame. The "time" should be in unit of "days". • If there are 3 columns, the names should be "sample", "time", "status". • If there are 4 columns, the names should be "sample", "value", "time", "status".
samples	default is NULL, can be common sample names for two datasets.
cutoff_mode	mode for grouping samples, can be "Auto" (default) or "Custom" or "None" (for groups have been prepared).
cutpoint	cut point (in percent) for "Custom" mode, default is c(50, 50).
palette	color palette, can be "hue", "grey", "RdBu", "Blues", "npg", "aaas", etc. More see ?survminer::ggsurvplot.
...	other parameters passing to survminer::ggsurvplot

Value

a (gg)plot object.

Examples

```
## Not run:
library(UCSCXenaTools)
expr_dataset <- "TCGA.LUAD.sampleMap/HiSeqV2_percentile"
cli_dataset <- "TCGA.LUAD.sampleMap/LUAD_clinicalMatrix"
id <- "KRAS"
cli_df <- XenaGenerate(
  subset = XenaDatasets == "TCGA.LUAD.sampleMap/LUAD_clinicalMatrix"
) %>%
  XenaQuery() %>%
  XenaDownload() %>%
  XenaPrepare()

# Use individual survival data
surv_df1 <- cli_df[, c("sampleID", "ABSOLUTE_Ploidy", "days_to_death", "vital_status")]
surv_df1$vital_status <- ifelse(surv_df1$vital_status == "DECEASED", 1, 0)
vis_identifier_grp_surv(surv_df = surv_df1)

# Use both dataset argument and vis_identifier_grp_surv(surv_df = surv_df1)
```

```
surv_df2 <- surv_df1[, c(1, 3, 4)]
vis_identifier_grp_surv(expr_dataset, id, surv_df = surv_df2)
vis_identifier_grp_surv(expr_dataset, id,
  surv_df = surv_df2,
  cutoff_mode = "Custom", cutpoint = c(25, 75)
)

## End(Not run)
```

`vis_identifier_multi_cor` *Visualize Correlation for Multiple Identifiers*

Description

NOTE: the dataset must be dense matrix in UCSC Xena data hubs.

Usage

```
vis_identifier_multi_cor(
  dataset,
  ids,
  samples = NULL,
  matrix.type = c("full", "upper", "lower"),
  type = c("parametric", "nonparametric", "robust", "bayes"),
  partial = FALSE,
  sig.level = 0.05,
 p.adjust.method = c("holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr",
    "none"),
  color_low = "#E69F00",
  color_high = "#009E73",
  ...
)
```

Arguments

`dataset`	the dataset to obtain identifiers.
`ids`	the molecule identifiers.
`samples`	default is `NULL`, can be common sample names for two datasets.
`matrix.type`	Character, `"upper"` (default), `"lower"`, or `"full"`, display full matrix, lower triangular or upper triangular matrix.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"`

	You can specify just the initial letter.
partial	Can be TRUE for partial correlations. For Bayesian partial correlations, "full" instead of pseudo-Bayesian partial correlations (i.e., Bayesian correlation based on frequentist partialization) are returned.
sig.level	Significance level (Default: 0.05). If the *p*-value in *p*-value matrix is bigger than sig.level, then the corresponding correlation coefficient is regarded as insignificant and flagged as such in the plot.
p.adjust.method	Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
color_low	the color code for lower value mapping.
color_high	the color code for higher value mapping.
...	other parameters passing to ggstatsplot::ggcorrmat.

Value

a (gg)plot object.

Examples

```
## Not run:
dataset <- "TcgaTargetGtex_rsem_isoform_tpm"
ids <- c("TP53", "KRAS", "PTEN")
vis_identifier_multi_cor(dataset, ids)

## End(Not run)
```

---

vis_pancan_anatomy *Visualize Single Gene Expression in Anatomy Location*

Description

Visualize Single Gene Expression in Anatomy Location

Usage

```
vis_pancan_anatomy(
  Gene = "TP53",
  Gender = c("Female", "Male"),
  data_type = "mRNA",
  option = "D",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Gender	a string, "Female" (default) or "Male".
data_type	choose gene profile type, including "mRNA","transcript","methylation","miRNA","protein","cnv"
option	A character string indicating the color map option to use. Eight options are available: • "magma" (or "A") • "inferno" (or "B") • "plasma" (or "C") • "viridis" (or "D") • "cividis" (or "E") • "rocket" (or "F") • "mako" (or "G") • "turbo" (or "H")
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

---

vis_pcawg_dist *Visualize molecular profile in PCAWG*

Description

Visualize molecular profile in PCAWG

Usage

```
vis_pcawg_dist(
  Gene = "TP53",
  Mode = c("Boxplot", "Violinplot"),
  data_type = "mRNA",
  Show.P.value = TRUE,
  Show.P.label = TRUE,
  Method = c("wilcox.test", "t.test"),
  values = c("#DF2020", "#DDDF21"),
  draw_quantiles = c(0.25, 0.5, 0.75),
  trim = TRUE,
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Mode	"Boxplot" or "Violinplot" to represent data
data_type	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
Show.P.value	TRUE or FALSE whether to count P value
Show.P.label	TRUE or FALSE present p value with number or label *, **, *** and ****
Method	default method is wilcox.test
values	the color to fill tumor or normal
draw_quantiles	draw quantiles for violinplot
trim	whether trim the violin
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

Examples

```
## Not run:
p <- vis_pcawg_dist(Gene = "TP53")

## End(Not run)
```

---

vis_pcawg_gene_cor *Visualize Gene-Gene Correlation in TCGA*

Description

Visualize Gene-Gene Correlation in TCGA

Usage

```
vis_pcawg_gene_cor(
  Gene1 = "CSF1R",
  Gene2 = "JAK3",
  data_type1 = "mRNA",
  data_type2 = "mRNA",
  cor_method = "spearman",
  purity_adj = TRUE,
  use_log_x = FALSE,
  use_log_y = FALSE,
  use_regline = TRUE,
```

```
  dcc_project_code_choose = "BLCA-US",
  use_all = FALSE,
  filter_tumor = TRUE,
  alpha = 0.5,
  color = "#000000",
  opt_pancan = .opt_pancan
)
```

Arguments

Gene1	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Gene2	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type1	choose gene profile type for the first gene, including "mRNA","transcript","methylation","miRNA","prote
data_type2	choose gene profile type for the second gene, including "mRNA","transcript","methylation","miRNA","pr
cor_method	correlation method
purity_adj	whether performing partial correlation adjusted by purity
use_log_x	if TRUE, log X values.
use_log_y	if TRUE, log Y values.
use_regline	if TRUE, add regression line.
dcc_project_code_choose	select project code.
use_all	use all sample, default FALSE.
filter_tumor	whether use tumor sample only, default TRUE
alpha	dot alpha.
color	dot color.
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

---

vis_pcawg_unicox_tree *Visualize Single Gene Univariable Cox Result in PCAWG*

Description

Visualize Single Gene Univariable Cox Result in PCAWG

Usage

```
vis_pcawg_unicox_tree(
  Gene = "TP53",
  measure = "OS",
  data_type = "mRNA",
  threshold = 0.5,
  values = c("grey", "#E31A1C", "#377DB8"),
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
measure	a survival measure, e.g. "OS".
data_type	choose gene profile type, including "mRNA","transcript","methylation","miRNA","protein","cnv"
threshold	a expression cutoff, 0.5 for median.
values	the color to fill tumor or normal
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

Examples

```
## Not run:
p <- vis_pcawg_unicox_tree(Gene = "TP53")

## End(Not run)
```

---

vis_toil_Mut *Visualize molecular profile difference between mutation and wild status of queried gene*

Description

Visualize molecular profile difference between mutation and wild status of queried gene

Usage

```
vis_toil_Mut(
  mut_Gene = "TP53",
  Gene = NULL,
  data_type = NULL,
  Mode = c("Boxplot", "Violinplot"),
```

```
  Show.P.value = TRUE,
  Show.P.label = TRUE,
  Method = c("wilcox.test", "t.test"),
  values = c("#DF2020", "#DDDF21"),
  draw_quantiles = c(0.25, 0.5, 0.75),
  trim = TRUE,
  opt_pancan = .opt_pancan
)
```

Arguments

mut_Gene	the queried gene to determine grouping based on mutation and wild status
Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	choose gene profile type, including "mRNA", "transcript", "methylation", "miRNA".
Mode	choose one visualize mode to represent data
Show.P.value	TRUE or FALSE whether to count P value
Show.P.label	TRUE or FALSE present p value with number or label *, **, *** and ****
Method	default method is wilcox.test
values	the color to fill mutation or wild status
draw_quantiles	draw quantiles for violinplot
trim	whether to trim the violin
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object or a tibble data.frame

Examples

```
## Not run:
p <- vis_toil_Mut(mut_Gene = "TP53")
p <- vis_toil_Mut(mut_Gene = "TP53", Gene = "TNF")
p <- vis_toil_Mut(mut_Gene = "TP53", Gene = "hsa-let-7d-3p", data_type = "miRNA")

## End(Not run)
```

---

vis_toil_Mut_cancer *Visualize molecular profile difference between mutation and wild status of queried gene in Single Cancer Type*

Description

Visualize molecular profile difference between mutation and wild status of queried gene in Single Cancer Type

Usage

```
vis_toil_Mut_cancer(
  mut_Gene = "TP53",
  Gene = NULL,
  data_type = NULL,
  Mode = c("Dotplot", "Violinplot"),
  Show.P.value = TRUE,
  Show.P.label = TRUE,
  Method = c("wilcox.test", "t.test"),
  values = c("#DF2020", "#DDDF21"),
  draw_quantiles = c(0.25, 0.5, 0.75),
  trim = TRUE,
  Cancer = "ACC",
  opt_pancan = .opt_pancan
)
```

Arguments

mut_Gene	the queried gene to determine grouping based on mutation and wild status
Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
data_type	choose gene profile type, including "mRNA", "transcript", "methylation", "miRNA".
Mode	choose one visualize mode to represent data
Show.P.value	TRUE or FALSE whether to count P value
Show.P.label	TRUE or FALSE present p value with number or label *, **, *** and ****
Method	default method is wilcox.test
values	the color to fill mutation or wild status
draw_quantiles	draw quantiles for violinplot
trim	whether to trim the violin
Cancer	select cancer cohort(s).
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object or a tibble data.frame.

---

vis_toil_TvsN *Visualize Pan-cancer TPM (tumor (TCGA) vs Normal (TCGA & GTEx))*

Description

Visualize Pan-cancer TPM (tumor (TCGA) vs Normal (TCGA & GTEx))

Usage

```
vis_toil_TvsN(
  Gene = "TP53",
  Mode = c("Boxplot", "Violinplot"),
  data_type = "mRNA",
  Show.P.value = TRUE,
  Show.P.label = TRUE,
  Method = c("wilcox.test", "t.test"),
  values = c("#DF2020", "#DDDF21"),
  TCGA.only = FALSE,
  draw_quantiles = c(0.25, 0.5, 0.75),
  trim = TRUE,
  include.Tumor.only = FALSE,
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
Mode	"Boxplot" or "Violinplot" to represent data
data_type	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
Show.P.value	TRUE or FALSE whether to count P value
Show.P.label	TRUE or FALSE present p value with number or label *, **, *** and ****
Method	default method is wilcox.test
values	the color to fill tumor or normal
TCGA.only	include samples only from TCGA dataset
draw_quantiles	draw quantiles for violinplot
trim	whether trim the violin
include.Tumor.only	if TRUE, include "UVM" and "MESO" these two types with matched normals samples.
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

Examples

```
## Not run:
p <- vis_toil_TvsN(Gene = "TP53", Mode = "Violinplot", Show.P.value = FALSE, Show.P.label = FALSE)
p <- vis_toil_TvsN(Gene = "TP53", Mode = "Boxplot", Show.P.value = FALSE, Show.P.label = FALSE)

## End(Not run)
```

`vis_toil_TvsN_cancer` *Visualize Gene TPM in Single Cancer Type (Tumor (TCGA) vs Normal (TCGA & GTEx))*

Description

Visualize Gene TPM in Single Cancer Type (Tumor (TCGA) vs Normal (TCGA & GTEx))

Usage

```
vis_toil_TvsN_cancer(
  Gene = "TP53",
  Mode = c("Violinplot", "Dotplot"),
  data_type = "mRNA",
  Show.P.value = FALSE,
  Show.P.label = FALSE,
  Method = "wilcox.test",
  values = c("#DF2020", "#DDDF21"),
  TCGA.only = FALSE,
  Cancer = "ACC",
  opt_pancan = .opt_pancan
)
```

Arguments

`Gene`	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature (`"TP53 + 2 * KRAS - 1.3 * PTEN"`).
`Mode`	"Boxplot" or "Violinplot" to represent data
`data_type`	choose gene profile type, including "mRNA", "transcript", "protein", "mutation", "cnv", "methylation", "miRNA".
`Show.P.value`	`TRUE` or `FALSE` whether to count P value
`Show.P.label`	`TRUE` or `FALSE` present p value with number or label *, **, *** and ****
`Method`	default method is wilcox.test
`values`	the color to fill tumor or normal
`TCGA.only`	include samples only from TCGA dataset
`Cancer`	select cancer cohort(s).
`opt_pancan`	specify one dataset for some molercular profiles

Value

a `ggplot` object.

vis_unicox_tree *Visualize Single Gene Univariable Cox Result from Toil Data Hub*

Description

Visualize Single Gene Univariable Cox Result from Toil Data Hub

Usage

```
vis_unicox_tree(
  Gene = "TP53",
  measure = "OS",
  data_type = "mRNA",
  threshold = 0.5,
  values = c("grey", "#E31A1C", "#377DB8"),
  opt_pancan = .opt_pancan
)
```

Arguments

Gene	a molecular identifier (e.g., "TP53") or a formula specifying genomic signature ("TP53 + 2 * KRAS - 1.3 * PTEN").
measure	a survival measure, e.g. "OS".
data_type	choose gene profile type, including "mRNA","transcript","methylation","miRNA","protein","cnv"
threshold	a expression cutoff, 0.5 for median.
values	the color to fill tumor or normal
opt_pancan	specify one dataset for some molercular profiles

Value

a ggplot object

Examples

```
## Not run:
p <- vis_unicox_tree(Gene = "TP53")

## End(Not run)
```

Index

∗ **datasets**
    .opt_pancan, 3
.opt_pancan, 3

analyze_gene_drug_response_asso, 4, *37*
analyze_gene_drug_response_diff, 5, *37*
app_run, 6
available_hosts, 7

ccle_absolute, 7
ccle_info, 8
ccle_info_fine, 8

data.frame, *27–31*

ezcor, 9
ezcor_batch, 10
ezcor_partial_cor, 11

get_ccle_cn_value, 12
get_ccle_gene_value
        *(*get_ccle_cn_value*)*, 12
get_ccle_mutation_status
        *(*get_ccle_cn_value*)*, 12
get_ccle_protein_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_cn_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_gene_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_methylation_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_miRNA_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_mutation_status
        *(*get_ccle_cn_value*)*, 12
get_pancan_protein_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_transcript_value
        *(*get_ccle_cn_value*)*, 12
get_pancan_value *(*get_ccle_cn_value*)*, 12
get_pcawg_APOBEC_mutagenesis_value
        *(*get_ccle_cn_value*)*, 12
get_pcawg_fusion_value
        *(*get_ccle_cn_value*)*, 12
get_pcawg_gene_value
        *(*get_ccle_cn_value*)*, 12
get_pcawg_miRNA_value
        *(*get_ccle_cn_value*)*, 12
get_pcawg_promoter_value
        *(*get_ccle_cn_value*)*, 12
ggstatsplot::ggbetweenstats, *48*
ggstatsplot::ggcorrmat, *51*
ggstatsplot::ggwithinstats, *48*

keep_cat_cols, 15

load_data, 15

mol_quick_analysis, 17

pcawg_info, 17
pcawg_info_fine, 18
pcawg_purity, 18
ppcor::pcor.test(), *12*

query_general_value, 19
query_molecule_value, 20
query_pancan_value, 21
query_tcga_group, 23
query_toil_value_df, 24

tcga survival analysis, 25
TCGA.organ, 27
tcga_clinical, 27
tcga_clinical_fine, 28
tcga_genome_instability, 28
tcga_gtex, 29
tcga_purity, 29
tcga_subtypes, 30
tcga_surv, 30

tcga_surv_get *(tcga survival analysis)*, 25
tcga_surv_plot *(tcga survival analysis)*, 25
tcga_tmb, 31
toil_info, 31

UCSCXenaTools::XenaData, *13*

vis_ccle_gene_cor, 32
vis_ccle_tpm, 33
vis_dim_dist, 33
vis_gene_cor, 35
vis_gene_cor_cancer, 36
vis_gene_drug_response_asso, 37
vis_gene_drug_response_diff, 37
vis_gene_immune_cor, 38
vis_gene_msi_cor, 39
vis_gene_pw_cor, 40
vis_gene_stemness_cor, 41
vis_gene_TIL_cor, 42
vis_gene_tmb_cor, 43
vis_identifier_cor, 44
vis_identifier_dim_dist, 45
vis_identifier_grp_comparison, 46
vis_identifier_grp_surv, 48
vis_identifier_multi_cor, 50
vis_pancan_anatomy, 51
vis_pcawg_dist, 52
vis_pcawg_gene_cor, 53
vis_pcawg_unicox_tree, 54
vis_toil_Mut, 55
vis_toil_Mut_cancer, 56
vis_toil_TvsN, 57
vis_toil_TvsN_cancer, 59
vis_unicox_tree, 60