Package ‘ieugwasr’

June 18, 2025

Title Interface to the 'OpenGWAS' Database API

Version 1.0.4

Description Interface to the 'OpenGWAS' database API <https://api.opengwas.io/api/>. Includes a wrapper to make generic calls to the API, plus convenience functions for specific queries.

License MIT + file LICENSE

URL https://github.com/MRCIEU/ieugwasr, https://mrcieu.github.io/ieugwasr/

BugReports https://github.com/MRCIEU/ieugwasr/issues

Depends R (>= 4.0)

Imports dplyr, httr, jsonlite, magrittr, stats

Suggests knitr, utils, rmarkdown, testthat

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Gibran Hemani [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-0920-1055>),
Ben Elsworth [aut] (ORCID: <https://orcid.org/0000-0001-7328-4233>),
Tom Palmer [aut] (ORCID: <https://orcid.org/0000-0003-4655-4511>),
Rita Rasteiro [aut] (ORCID: <https://orcid.org/0000-0002-4217-3060>)

Maintainer Gibran Hemani <g.hemani@bristol.ac.uk>

Repository CRAN

Date/Publication 2025-06-18 11:10:02 UTC

Contents

`afl2_chrpos` *Look up allele frequencies and LD scores for 1000 genomes populations by chrpos*

Description

Look up allele frequencies and LD scores for 1000 genomes populations by chrpos

Usage

```
afl2_chrpos(chrpos, reference = "1000g", opengwas_jwt = get_opengwas_jwt())
```

Arguments

`chrpos`	list of `<chr>:<pos>` in build 37, e.g. `c("3:46414943", "3:122991235")`. Also allows ranges e.g `"7:105561135-105563135"`
`reference`	Default=`"1000g"`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Data frame containing ancestry specific LD scores and allele frequencies for each variant

---

`afl2_list` *Retrieve a allele frequency and LD scores for pre-defined lists of variants*

Description

Data frame includes 1000 genomes metadata including sample sizes, allele frequency and LD score, separated by 5 super populations (EUR = European, AFR = African, EAS = East Asian, AMR = Admixed American, SAS = South Asian)

Usage

```
afl2_list(variantlist = "reduced", opengwas_jwt = get_opengwas_jwt())
```

Arguments

`variantlist`	Choose pre-defined list. `"reduced"` = ~20k SNPs that are common in all super populations (default). `"hapmap3"` = ~1.3 million hm3 SNPs
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Data frame containing ancestry specific LD scores and allele frequencies for each variant

---

`afl2_rsid` *Look up allele frequencies and LD scores for 1000 genomes populations by rsid*

Description

Look up allele frequencies and LD scores for 1000 genomes populations by rsid

Usage

```
afl2_rsid(rsid, reference = "1000g", opengwas_jwt = get_opengwas_jwt())
```

Arguments

`rsid`	Vector of rsids
`reference`	Default=`"1000g"`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Data frame containing ancestry specific LD scores and allele frequencies for each variant

---

`api_query` *Wrapper for sending queries and payloads to API*

Description

There are a number of different GET and POST endpoints in the GWAS database API. This is a generic way to access them.

Usage

```
api_query(
  path,
  query = NULL,
  opengwas_jwt = get_opengwas_jwt(),
  method = "GET",
  silent = TRUE,
  encode = "json",
  timeout = 300,
  override_429 = FALSE
)
```

Arguments

`path`	Either a full query path (e.g. for get) or an endpoint (e.g. for post) queries
`query`	If post query, provide a list of arguments as the payload. `NULL` by default
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.
`method`	`"GET"` (default) or `"POST"`, `"DELETE"` etc
`silent`	`TRUE`/`FALSE` to be passed to httr call. `TRUE` by default
`encode`	Default = `"json"`, see `httr::POST` for options
`timeout`	Default = `300`, avoid increasing this, preferentially simplify the query first.
`override_429`	Default=`FALSE`. If allowance is exceeded then the query will error before submitting a request to avoid getting blocked. If you are sure you want to submit the request then set this to TRUE.

Value

httr response object

`api_status` *OpenGWAS server status*

Description

OpenGWAS server status

Usage

```
api_status()
```

Value

List of values regarding status

`associations` *Query specific variants from specific GWAS*

Description

Every rsid is searched for against each requested GWAS id. To get a list of available GWAS ids, or to find their meta data, use `gwasinfo`. Can request LD proxies for instances when the requested rsid is not present in a particular GWAS dataset. This currently only uses an LD reference panel composed of Europeans in 1000 genomes version 3. It is also restricted to biallelic single nucleotide polymorphisms (no indels), with European MAF > 0.01.

Usage

```
associations(
  variants,
  id,
  proxies = 1,
  r2 = 0.8,
  align_alleles = 1,
  palindromes = 1,
  maf_threshold = 0.3,
  opengwas_jwt = get_opengwas_jwt()
)
```

Arguments

`variants`	Array of variants e.g. `c("rs234", "7:105561135-105563135")`
`id`	Array of GWAS studies to query. See `gwasinfo` for available studies
`proxies`	`0` or (default) `1` - indicating whether to look for proxies
`r2`	Minimum proxy LD rsq value. Default=`0.8`
`align_alleles`	Try to align tag alleles to target alleles (if `proxies = 1`). `1` = yes (default), `0` = no
`palindromes`	Allow palindromic SNPs (if `proxies = 1`). `1` = yes (default), `0` = no
`maf_threshold`	MAF threshold to try to infer palindromic SNPs. Default = `0.3`.
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe

`batches` *Get list of data batches in IEU OpenGWAS database*

Description

Get list of data batches in IEU OpenGWAS database

Usage

```
batches(opengwas_jwt = get_opengwas_jwt())
```

Arguments

`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

data frame

`batch_from_id` *Extract batch name from study ID*

Description

Extract batch name from study ID

Usage

```
batch_from_id(id)
```

Arguments

`id`	Array of study IDs

Value

Array of batch names

`check_access_token` *Check if authentication has been made*

Description

Deprecated. Use `get_opengwas_jwt()` instead. See `https://mrcieu.github.io/ieugwasr/articles/guide.html#authentication` for more information.

Usage

```
check_access_token()
```

Value

NULL or access_token depending on current authentication state

---

`check_reset` *Check if OpenGWAS allowance needs to be reset*

Description

This function checks if a recent query indicated that the OpenGWAS allowance has been used up. To prevent the IP being blocked, it will error if the new query is being submitted before the reset time. If the allowance has been used up, it displays a message indicating the time when the allowance will be reset. By default, the function will throw an error if the allowance has been used up, but this behavior can be overridden by setting `override_429` to `TRUE`.

Usage

```
check_reset(override_429 = FALSE)
```

Arguments

`override_429`	Logical value indicating whether to override the allowance reset check (default: `FALSE`)

`editcheck` *Check datasets that are in process of being uploaded*

Description

Check datasets that are in process of being uploaded

Usage

```
editcheck(id, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`id`	ID
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe

`fill_n` *Look up sample sizes when meta data is missing from associations*

Description

Look up sample sizes when meta data is missing from associations

Usage

```
fill_n(d, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`d`	Output from `associations`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Updated version of d

`get_opengwas_jwt` *Retrieve OpenGWAS JSON Web Token from .Renviron file*

Description

Retrieve OpenGWAS JSON Web Token from .Renviron file

Usage

```
get_opengwas_jwt()
```

Value

JWT string

`get_query_content` *Parse out json response from httr object*

Description

Parse out json response from httr object

Usage

```
get_query_content(response)
```

Arguments

`response`	Output from `api_query()`

Value

Parsed json output from query, often in form of data frame. If status code is not successful then return the actual response

`gwasinfo` *Get list of studies with available GWAS summary statistics through API*

Description

Get list of studies with available GWAS summary statistics through API

Usage

```
gwasinfo(id = NULL, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`id`	List of OpenGWAS IDs to retrieve. If `NULL` (default) retrieves all available datasets
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe of details for all available studies

`gwasinfo_files` *Get list of download URLs for each file associated with a dataset through API*

Description

`gwasinfo_files()` returns a list of download URLs for each file (.vcf.gz, .vcf.gz.tbi, _report.html) associated with a dataset. The URLs will expire in 2 hours. If a dataset is missing from the results, that means either the dataset doesn't exist or you don't have access to it. If a dataset is in the results but some/all links are missing, that means the files are unavailable.

Usage

```
gwasinfo_files(id, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`id`	List of OpenGWAS IDs to retrieve.
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a JWT. Provide the JWT string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe of details for requested studies

`infer_ancestry` *Infer ancestry of GWAS dataset by matching against 1000 genomes allele frequencies*

Description

Uses ~20k SNPs selected for common frequency across 5 major super populations

Usage

```
infer_ancestry(d, snpinfo = NULL, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`d`	Data frame containing at least `rsid` and `eaf` columns. e.g. output from associations
`snpinfo`	Output from `afl2_list`, `afl2_rsid` or `afl2_chrpos`. If `NULL` then `afl2_list()` is used by default
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

data frame ordered by most likely ancestry, each row represents a super population and cor column represents the correlation between the GWAS dataset and the 1000 genomes super population allele frequencies

---

`ld_clump` *Perform LD clumping on SNP data*

Description

Uses PLINK clumping method, where SNPs in LD within a particular window will be pruned. The SNP with the lowest p-value is retained.

Usage

```
ld_clump(
  dat = NULL,
  clump_kb = 10000,
  clump_r2 = 0.001,
  clump_p = 0.99,
  pop = "EUR",
  opengwas_jwt = get_opengwas_jwt(),
  bfile = NULL,
  plink_bin = NULL
)
```

Arguments

`dat`	Dataframe. Must have a variant name column (`rsid`) and pval column called `pval`. If `id` is present then clumping will be done per unique id.
`clump_kb`	Clumping kb window. Default is very strict, `10000`
`clump_r2`	Clumping r2 threshold. Default is very strict, `0.001`
`clump_p`	Clumping sig level for index variants. Default = `1` (i.e. no threshold)
`pop`	Super-population to use as reference panel. Default = `"EUR"`. Options are `"EUR"`, `"SAS"`, `"EAS"`, `"AFR"`, `"AMR"`. `'legacy'` also available - which is a previously used version of the EUR panel with a slightly different set of markers
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.
`bfile`	If this is provided then will use the API. Default = `NULL`
`plink_bin`	If `NULL` and `bfile` is not `NULL` then will detect packaged plink binary for specific OS. Otherwise specify path to plink binary. Default = `NULL`

Details

This function interacts with the OpenGWAS API, which houses LD reference panels for the 5 super-populations in the 1000 genomes reference panel. It includes only bi-allelic SNPs with MAF > 0.01, so it's quite possible that a variant you want to include in the clumping process will be absent. If it is absent, it will be automatically excluded from the results.

You can check if your variants are present in the LD reference panel using `ld_reflookup()`.

This function does put load on the OpenGWAS servers, which makes life more difficult for other users. We have implemented a method and made available the LD reference panels to perform clumping locally, see `ld_clump()` and related vignettes for details.

Value

Data frame

`ld_clump_api` *Perform clumping on the chosen variants using through API*

Description

Perform clumping on the chosen variants using through API

Usage

```
ld_clump_api(
  dat,
  clump_kb = 10000,
  clump_r2 = 0.1,
  clump_p,
  pop = "EUR",
  opengwas_jwt = get_opengwas_jwt()
)
```

Arguments

`dat`	Dataframe. Must have a variant name column (`variant`) and pval column called `pval`. If `id` is present then clumping will be done per unique id.
`clump_kb`	Clumping kb window. Default is very strict, `10000`
`clump_r2`	Clumping r2 threshold. Default is very strict, `0.001`
`clump_p`	Clumping sig level for index variants. Default = `1` (i.e. no threshold)
`pop`	Super-population to use as reference panel. Default = `"EUR"`. Options are `"EUR"`, `"SAS"`, `"EAS"`, `"AFR"`, `"AMR"`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Data frame of only independent variants

`ld_clump_local` *Wrapper for clump function using local plink binary and ld reference dataset*

Description

Wrapper for clump function using local plink binary and ld reference dataset

Usage

```
ld_clump_local(dat, clump_kb, clump_r2, clump_p, bfile, plink_bin)
```

Arguments

`dat`	Dataframe. Must have a variant name column (`variant`) and pval column called `pval`. If `id` is present then clumping will be done per unique id.
`clump_kb`	Clumping kb window. Default is very strict, `10000`
`clump_r2`	Clumping r2 threshold. Default is very strict, `0.001`
`clump_p`	Clumping sig level for index variants. Default = `1` (i.e. no threshold)
`bfile`	If this is provided then will use the API. Default = `NULL`
`plink_bin`	Specify path to plink binary. Default = `NULL`. See `https://github.com/MRCIEU/genetics.binaRies` for convenient access to plink binaries

Value

data frame of clumped variants

`ld_matrix` *Get LD matrix for list of SNPs*

Description

This function takes a list of SNPs and searches for them in a specified super-population in the 1000 Genomes phase 3 reference panel. It then creates an LD matrix of r values (signed, and not squared). All LD values are with respect to the major alleles in the 1000G dataset. You can specify whether the allele names are displayed.

Usage

```
ld_matrix(
  variants,
  with_alleles = TRUE,
  pop = "EUR",
  opengwas_jwt = get_opengwas_jwt(),
  bfile = NULL,
  plink_bin = NULL
)
```

Arguments

`variants`	List of variants (rsids)
`with_alleles`	Whether to append the allele names to the SNP names. Default: `TRUE`

`pop`	Super-population to use as reference panel. Default = `"EUR"`. Options are `"EUR"`, `"SAS"`, `"EAS"`, `"AFR"`, `"AMR"`. `'legacy'` also available - which is a previously used version of the EUR panel with a slightly different set of markers
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.
`bfile`	If this is provided then will use the API. Default = `NULL`
`plink_bin`	If `NULL` and bfile is not `NULL` then will detect packaged plink binary for specific OS. Otherwise specify path to plink binary. Default = `NULL`

Details

The data used for generating the LD matrix includes only bi-allelic SNPs with MAF > 0.01, so it's quite possible that a variant you want to include will be absent. If it is absent, it will be automatically excluded from the results.

You can check if your variants are present in the LD reference panel using `ld_reflookup()`

This function does put load on the OpenGWAS servers, which makes life more difficult for other users, and has been limited to analyse only up to 500 variants at a time. We have implemented a method and made available the LD reference panels to perform the operation locally, see `ld_matrix()` and related vignettes for details.

Value

Matrix of LD r values

`ld_matrix_local` *Get LD matrix using local plink binary and reference dataset*

Description

Get LD matrix using local plink binary and reference dataset

Usage

```
ld_matrix_local(variants, bfile, plink_bin, with_alleles = TRUE)
```

Arguments

`variants`	List of variants (rsids)
`bfile`	Path to bed/bim/fam ld reference panel
`plink_bin`	Specify path to plink binary. Default = `NULL`. See `https://github.com/MRCIEU/genetics.binaRies` for convenient access to plink binaries
`with_alleles`	Whether to append the allele names to the SNP names. Default: `TRUE`

Value

data frame

`ld_reflookup` *Check which rsids are present in a remote LD reference panel*

Description

Provide a list of rsids that you may want to perform LD operations on to check if they are present in the LD reference panel. If they are not then some functions e.g. `ld_clump` will exclude them from the analysis, so you may want to consider how to handle those variants in your data.

Usage

```
ld_reflookup(rsid, pop = "EUR", opengwas_jwt = get_opengwas_jwt())
```

Arguments

`rsid`	Array of rsids to check
`pop`	Super-population to use as reference panel. Default = `"EUR"`. Options are `"EUR"`, `"SAS"`, `"EAS"`, `"AFR"`, `"AMR"`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Array of rsids that are present in the LD reference panel

`legacy_ids` *Convert current IDs to legacy IDs*

Description

Convert current IDs to legacy IDs

Usage

```
legacy_ids(x)
```

Arguments

`x`	Vector of ids

Value

Vector of back compatible ids

`logging_info` *Details of how access token logs are used*

Description

Details of how access token logs are used

Usage

```
logging_info()
```

Value

No return value, called for side effects

`phewas` *Perform fast phewas of a specific variants against all available GWAS datasets*

Description

This is faster than doing it manually through `associations`

Usage

```
phewas(variants, pval = 1e-05, batch = c(), opengwas_jwt = get_opengwas_jwt())
```

Arguments

`variants`	Array of variants e.g. `c("rs234", "7:105561135-105563135")`
`pval`	p-value threshold. Default = `0.00001`
`batch`	Vector of batch IDs to search across. If `c()` (default) then returns all batches
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe

`print.ApiStatus` *Print API status*

Description

Print API status

Usage

```
## S3 method for class 'ApiStatus'
print(x, ...)
```

Arguments

`x`	Output from `api_status`
`...`	Unused, for extensibility

Value

Print out of API status

`print.GwasInfo` *Print GWAS information*

Description

Print GWAS information

Usage

```
## S3 method for class 'GwasInfo'
print(x, ...)
```

Arguments

`x`	Output from `gwasinfo`
`...`	Unused, for extensibility

Value

Print out of GWAS information

`select_api` *Toggle API address between development and release*

Description

Toggle API address between development and release

Usage

```
select_api(where = "public", silent = FALSE)
```

Arguments

`where`	Which API to use. Choice between `"public"`, `"private"`, `"dev1"`, `"dev2"`. Default = `"public"`.
`silent`	Silent? Default = `FALSE`

Value

No return value, called for side effects

`set_reset` *Set the reset time for OpenGWAS allowance*

Description

This function sets the reset time for the OpenGWAS allowance based on the retry-after header returned by the API response. It also displays a warning message indicating the time at which the allowance will reset.

Usage

```
set_reset(r)
```

Arguments

`r`	The API response object

Value

None

`tophits` *Obtain top hits from a GWAS dataset*

Description

By default performs clumping on the server side.

Usage

```
tophits(
  id,
  pval = 5e-08,
  clump = 1,
  r2 = 0.001,
  kb = 10000,
  pop = "EUR",
  force_server = FALSE,
  opengwas_jwt = get_opengwas_jwt()
)
```

Arguments

`id`	Array of GWAS studies to query. See `gwasinfo` for available studies
`pval`	use this p-value threshold. Default = `5e-8`
`clump`	whether to clump (`1`) or not (`0`). Default = 1
`r2`	use this clumping r2 threshold. Default is very strict, `0.001`
`kb`	use this clumping kb window. Default is very strict, `10000`
`pop`	Super-population to use as reference panel. Default = `"EUR"`. Options are `"EUR"`, `"SAS"`, `"EAS"`, `"AFR"`, `"AMR"`
`force_server`	Logical. By default will return preclumped hits. p-value threshold 5e-8, with r2 threshold 0.001 and kb threshold 10000, using only SNPs with MAF > 0.01 in the European samples in 1000 genomes. If force_server = `TRUE` then will recompute using server side LD reference panel.
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Dataframe

`user` *Get user details*

Description

Get user details

Usage

```
user(opengwas_jwt = get_opengwas_jwt())
```

Arguments

`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

user information

`variants_chrpos` *Obtain information about chr pos and surrounding region*

Description

For a list of chromosome and positions, finds all variants within a given radius

Usage

```
variants_chrpos(chrpos, radius = 0, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`chrpos`	list of <chr>:<pos> in build 37, e.g. `c("3:46414943", "3:122991235")`. Also allows ranges e.g. `"7:105561135-105563135"`
`radius`	Radius around each chrpos, default = `0`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

Data frame

`variants_gene` *Obtain variants around a gene*

Description

Provide a gene identified, either Ensembl or Entrez

Usage

```
variants_gene(gene, radius = 0, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`gene`	Vector of genes, either Ensembl or Entrez, e.g. `c("ENSG00000123374", "ENSG00000160791")` or `1017`
`radius`	Radius around the gene region to include. Default = `0`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

data frame with the following columns

`variants_rsid` *Obtain information about rsid*

Description

Obtain information about rsid

Usage

```
variants_rsid(rsid, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`rsid`	Vector of rsids
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

data frame

`variants_to_rsid` *Convert mixed array of rsid and chrpos to list of rsid*

Description

Convert mixed array of rsid and chrpos to list of rsid

Usage

```
variants_to_rsid(variants, opengwas_jwt = get_opengwas_jwt())
```

Arguments

`variants`	Array of variants e.g. `c("rs234", "7:105561135-105563135")`
`opengwas_jwt`	Used to authenticate protected endpoints. Login to `https://api.opengwas.io` to obtain a jwt. Provide the jwt string here, or store in .Renviron under the keyname OPENGWAS_JWT.

Value

list of rsids

Index