Package 'cancensus'

November 20, 2018

Type Package

Title Canadian Census Data and Geography from the 'CensusMapper' API

Version 0.1.8

Description Integrated, convenient, and uniform access to Canadian Census data and geography retrieved using the 'CensusMapper' API. This package produces analysis-ready tidy data frames and spatial data in multiple formats, as well as convenience functions for working with Census variables, variable hierarchies, and region selection. API keys are freely available with free registration at <https://censusmapper.ca/api>. Census data and boundary geometries are reproduced and distributed on an ``as is'' basis with the permission of Statistics Canada (Statistics Canada 2006; 2011; 2016).

License MIT + file LICENSE

Encoding UTF-8

LazyData yes

ByteCompile yes

NeedsCompilation no

Imports digest (>= 0.1), dplyr (>= 0.7), httr (>= 1.0.0), jsonlite (>= 1.0), rlang

RoxygenNote 6.1.0

Suggests knitr, ggplot2, leaflet, rmarkdown, readr, rgdal, rgeos, scales, sf, sp, tidyr

VignetteBuilder knitr

URL https://github.com/mountainMath/cancensus, https://mountainmath.github.io/cancensus/, https://censusmapper.ca/api

BugReports https://github.com/mountainMath/cancensus/issues

Author Jens von Bergmann [aut] (API creator and maintainer), Dmitry Shkolnik [aut, cre] (Package maintainer, responsible for correspondence), Aaron Jacobs [aut]

Maintainer Dmitry Shkolnik <shkolnikd@gmail.com>

Repository CRAN

Date/Publication 2018-11-20 09:40:04 UTC

R topics documented:

`as_census_region_list` *Convert a (suitably filtered) data frame from* `list_census_regions` *to a list suitable for passing to* `get_census`.

Description

Convert a (suitably filtered) data frame from `list_census_regions` to a list suitable for passing to `get_census`.

Usage

```
as_census_region_list(tbl)
```

Arguments

`tbl`	A data frame, suitably filtered, as returned by `list_census_regions`.

Examples

```
## Not run:
library(dplyr, warn.conflicts = FALSE)

# Query the CensusMapper API for the total occupied dwellings
# of 20 random Census Subdivisions, in Census 2016.
regions <- list_census_regions("CA16") %>%
  filter(level == "CSD") %>%
  sample_n(20) %>%
  as_census_region_list()
```

```
occupied <- get_census("CA16", regions = regions,
                           vectors = c("v_CA16_408"),
                           level = "Regions")

## End(Not run)
```

`census_vectors` *Return Census variable names and labels as a tidy data frame (Deprecated)*

Description

Return Census variable names and labels as a tidy data frame (Deprecated)

Usage

```
census_vectors(x)
```

Arguments

`x`	A data frame, `sp` or `sf` object returned from `get_census` or similar.

Value

A data frame with a column `variable` containing the truncated variable name, and a column `label` describing it.

Examples

```
## Not run:
# Query census data with truncated labels:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf", labels="short")

# Get details for truncated vectors:
census_vectors(census_data)

## End(Not run)
```

`child_census_vectors` *List all child variables from vector hierarchical based on a (sub-)list of census variables returned by* `list_census_vectors` *or* `search_census_vectors`.

Description

List all child variables from vector hierarchical based on a (sub-)list of census variables returned by `list_census_vectors` or `search_census_vectors`.

Usage

```
child_census_vectors(vector_list, leaves_only = FALSE)
```

Arguments

`vector_list`	The list of vectors to be used
`leaves_only`	Boolean flag to indicate if only leaf vectors should be returned, i.e. vectors that don't have children

Examples

```
library(dplyr, warn.conflicts = FALSE)

list_census_vectors("CA16") %>%
  filter(vector == "v_CA16_4092") %>%
  child_census_vectors(TRUE)
```

`get_census` *Access to Canadian census data through the CensusMapper API*

Description

This function allows convenient access to Canadian census data and boundary files through the CensusMapper API. An API key is required to retrieve data.

Usage

```
get_census(dataset, regions, level = NA, vectors = c(),
  geo_format = NA, labels = "detailed", use_cache = TRUE,
  quiet = FALSE, api_key = getOption("cancensus.api_key"))

get_census_geometry(dataset, level, regions, geo_format = "sf", ...)
```

Arguments

dataset	A CensusMapper dataset identifier.
regions	A named list of census regions to retrieve. Names must be valid census aggregation levels.
level	The census aggregation level to retrieve, defaults to "Regions". One of "Regions", "PR", "CMA", "CD", "CSD", "CT" or "DA".
vectors	An R vector containing the CensusMapper variable names of the census variables to download. If no vectors are specified only geographic data will get downloaded.
geo_format	By default is set to NA and appends no geographic information. To include geographic information with census data, specify one of either "sf" to return an sf object (requires the sf package) or "sp" to return a SpatialPolygonsDataFrame object (requires the rgdal package).
labels	Set to "detailed" by default, but truncated Census variable names can be selected by setting labels = "short". Use label_vectors(...) to return variable label information in detail.
use_cache	If set to TRUE (the default) data will be read from the local cache if available.
quiet	When TRUE, suppress messages and warnings.
api_key	An API key for the CensusMapper API. Defaults to options() and then the CM_API_KEY environment variable.
...	Further arguments passed to get_census.

Details

get_census_geometry is a convenience function that retrieves only Census geography boundaries.

For help selecting regions and vectors, see list_census_regions and list_census_vectors, or check out the interactive selection tool at https://censusmapper.ca/api.

Source

Census data and boundary geographies are reproduced and distributed on an "as is" basis with the permission of Statistics Canada (Statistics Canada 2006; 2011; 2016).

Examples

```
# Query the API for data on dwellings in Vancouver, at the census subdivision
# level:
## Not run:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD')

# Query the API for data on dwellings in Vancouver, at the census subdivision
# level, and return the associated geography files in \code{sf} format:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
```

```
                          level='CSD', geo_format = "sf")

# Make the same query, but return geography in \code{sp} format:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf")

# Make the same query, but this time drop descriptive vector names:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf", labels="short")

# Get details for truncated vectors:
label_vectors(census_data)

## End(Not run)
## Not run:
# Query the API for census subdivision boundary geometry within Vancouver.
vc_csds <- get_census_geometry(dataset='CA16', regions=list(CMA="59933"),
                               level='CSD', geo_format = "sf")

## End(Not run)
```

`label_vectors` *Return Census variable names and labels as a tidy data frame*

Description

Return Census variable names and labels as a tidy data frame

Usage

```
label_vectors(x)
```

Arguments

`x`	A data frame, `sp` or `sf` object returned from `get_census` or similar.

Value

A data frame with a column `variable` containing the truncated variable name, and a column `label` describing it.

Examples

```
## Not run:
# Query census data with truncated labels:
label_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf", labels="short")
```

```
# Get details for truncated vectors:
label_vectors(label_data)

## End(Not run)
```

`list_census_datasets` *Query the CensusMapper API for available datasets.*

Description

Query the CensusMapper API for available datasets.

Usage

```
list_census_datasets(use_cache = FALSE, quiet = FALSE)
```

Arguments

`use_cache`	If set to TRUE, data will be read from a local cache, if available. If set to FALSE (the default), query the API for the data, and refresh the local cache with the result.
`quiet`	When TRUE, suppress messages and warnings.

Value

Returns a data frame with a column `dataset` containing the code for the dataset, and a column `description` describing it.

Examples

```
# List available datasets in CensusMapper
list_census_datasets()
```

`list_census_regions` *Query the CensusMapper API for available regions in a given dataset.*

Description

Query the CensusMapper API for available regions in a given dataset.

Usage

```
list_census_regions(dataset, use_cache = FALSE, quiet = FALSE)
```

Arguments

dataset	The dataset to query for available regions, e.g. "CA16".
use_cache	If set to TRUE, data will be read from a local cache, if available. If set to FALSE (the default), query the API for the data, and refresh the local cache with the result.
quiet	When TRUE, suppress messages and warnings.

Value

Returns a data frame with the following columns:

region The region identifier.

name The name of that region.

level The census aggregation level of that region.

pop The population of that region.

municipal_status Additional identifiers to distinguish the municipal status of census subdivisions.

CMA_UID The identifier for the Census Metropolitan Area the region is in (if any).

CD_UID The identifier for the Census District the region is in (if any).

PR_UID The identifier for the Province the region is in (if unique).

Examples

```
list_census_regions('CA16')
```

---

list_census_vectors *Query the CensusMapper API for available vectors for a given dataset.*

Description

Query the CensusMapper API for available vectors for a given dataset.

Usage

```
list_census_vectors(dataset, use_cache = FALSE, quiet = TRUE)
```

Arguments

dataset	The dataset to query for available vectors, e.g. "CA16".
use_cache	If set to TRUE, data will be read from a local cache, if available. If set to FALSE (the default), query the API for the data, and refresh the local cache with the result.
quiet	When FALSE, shows messages and warnings. Set to TRUE by default.

Value

Returns a data frame detailing the available Census vectors (i.e. variables) for a given Census dataset. This data frame has columns `vector` containing the short code for the variable, `type` describing whether it's a female, male, or total aggregate, `label` indicating the name of the variable, `units` indicating whether the value represents a numeric integer, percentage, dollar figure, or ratio, `parent_vector` to show hierarchical relationship, `aggregation` indicating whether the value is additive or a transformation, and a column `details` with a detailed description of the variable generated by traversing all labels within its hierarchical structure.

Examples

```
# List all vectors for a given Census dataset in CensusMapper
list_census_vectors('CA16')
```

`parent_census_vectors` *List all parent variables from vector hierarchical based on a (sub-)list of census variables returned by* `list_census_vectors` *or* `search_census_vectors`.

Description

List all parent variables from vector hierarchical based on a (sub-)list of census variables returned by `list_census_vectors` or `search_census_vectors`.

Usage

```
parent_census_vectors(vector_list)
```

Arguments

`vector_list`	The list of vectors to be used

Examples

```
library(dplyr, warn.conflicts = FALSE)

list_census_vectors("CA16") %>%
  filter(vector == "v_CA16_4092") %>%
  parent_census_vectors()
```

search_census_regions *Query the CensusMapper API for regions with names matching a searchterm.*

Description

Query the CensusMapper API for regions with names matching a searchterm.

Usage

```
search_census_regions(searchterm, dataset, level = NA, ...)
```

Arguments

searchterm	The term to search for e.g. "Victoria". Search terms are case insensitive. If unable to find a given search term, this function will suggest the correct spelling to use when possible.
dataset	The dataset to query for available regions, e.g. "CA16".
level	One of NA, 'C', 'PR', 'CMA', 'CD', or 'CSD'. If specified, only return variables of specified 'level'.
...	Further arguments passed on to list_census_regions.

Examples

```
search_census_regions('Victoria', 'CA16')

## Not run:
# This will return a warning that no match was found, but will suggest similar named regions.
search_census_regions('Victorea', 'CA16')

# This will limit region results to only include CMA level regions
search_census_regions('Victoria', 'CA16', level = "CMA")

## End(Not run)
```

search_census_vectors *Query the CensusMapper API for vectors with descriptions matching a searchterm.*

Description

Query the CensusMapper API for vectors with descriptions matching a searchterm.

Usage

```
search_census_vectors(searchterm, dataset, type = NA, ...)
```

Arguments

searchterm	The term to search for e.g. "Ojibway". Search terms are case insensitive. If unable to find a given search term, this function will suggest the correct spelling to use when possible.
dataset	The dataset to query for available vectors, e.g. "CA16".
type	One of NA, 'Total', 'Male' or 'Female'. If specified, only return variables of specified 'type'.
...	Further arguments passed on to list_census_vectors.

Examples

```
search_census_vectors('Ojibway', 'CA16')
## Not run:
# This will return a warning that no match was found, but will suggest similar terms.
search_census_vectors('Ojibwe', 'CA16', 'Total')

## End(Not run)
```

Index