Package 'tidycensus'

January 31, 2018

Type Package

Title Load US Census Boundary and Attribute Data as 'tidyverse' and 'sf'-Ready Data Frames

Version 0.4.1

Date 2018-01-31

URL https://github.com/walkerke/tidycensus

BugReports https://github.com/walkerke/tidycensus/issues

Description
An integrated R interface to the decennial US Census and American Community Survey APIs and the US Census Bureau's geographic boundary files. Allows R users to return Census and ACS data as
tidyverse-ready data frames, and optionally returns a list-column with feature geometry for many geographies.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.3.0)

Imports httr, sf, dplyr (>= 0.7.0), tigris, stringr, jsonlite, purrr, rvest, tidyr (>= 0.7.0), rappdirs, readr, xml2, units, utils

RoxygenNote 6.0.1

NeedsCompilation no

Author Kyle Walker [aut, cre], Kris Eberwein [ctb]

Maintainer Kyle Walker <kyle.walker@tcu.edu>

Repository CRAN

Date/Publication 2018-01-31 20:32:43 UTC

R topics documented:

census_api_key 2
fips_codes 3
get_acs 4
get_decennial 6
load_variables 7
moe_product 8
moe_prop 8
moe_ratio 9
moe_sum 10
tidycensus 10

Index **11**

`census_api_key` *Install a CENSUS API Key in Your* `.Renviron` *File for Repeated Use*

Description

This function will add your CENSUS API key to your `.Renviron` file so it can be called securely without being stored in your code. After you have installed your key, it can be called any time by typing `Sys.getenv("CENSUS_API_KEY")` and can be used in package functions by simply typing CENSUS_API_KEY If you do not have an `.Renviron` file, the function will create on for you. If you already have an `.Renviron` file, the function will append the key to your existing file, while making a backup of your original file for disaster recovery purposes.

Usage

```
census_api_key(key, overwrite = FALSE, install = FALSE)
```

Arguments

`key`	The API key provided to you from the Census formated in quotes. A key can be acquired at http://api.census.gov/data/key_signup.html
`overwrite`	If this is set to TRUE, it will overwrite an existing CENSUS_API_KEY that you already have in your `.Renviron` file.
`install`	if TRUE, will install the key in your `.Renviron` file for use in future sessions. Defaults to FALSE.

Examples

```
## Not run:
census_api_key("111111abc", install = TRUE)
# First time, reload your environment so you can use the key without restarting R.
readRenviron("~/.Renviron")
# You can check it with:
```

```
Sys.getenv("CENSUS_API_KEY")

## End(Not run)

## Not run:
# If you need to overwrite an existing key:
census_api_key("111111abc", overwrite = TRUE, install = TRUE)
# First time, reload your environment so you can use the key without restarting R.
readRenviron("~/.Renviron")
# You can check it with:
Sys.getenv("CENSUS_API_KEY")

## End(Not run)
```

`fips_codes` *Dataset with FIPS codes for US states and counties*

Description

Built-in dataset for smart state and county lookup. To access the data directly, issue the command `data(fips_codes)`.

- `county`: County name, title-case
- `county_code`: County code. (3-digit, 0-padded, character)
- `state`: Upper-case abbreviation of state
- `state_code`: State FIPS code (2-digit, 0-padded, character)
- `state_name`: Title-case name of state

Usage

```
data(fips_codes)
```

Format

An object of class `data.frame` with 3237 rows and 5 columns.

Details

Dataset with FIPS codes for US states and counties

Built-in dataset for use with the `lookup_code` function. To access the data directly, issue the command `data(fips_codes)`.

get_acs *Obtain data and feature geometry for the five-year American Community Survey*

Description

Obtain data and feature geometry for the five-year American Community Survey

Usage

```
get_acs(geography, variables = NULL, table = NULL, cache_table = FALSE,
  year = 2016, endyear = NULL, output = "tidy", state = NULL,
  county = NULL, geometry = FALSE, keep_geo_vars = FALSE,
  summary_var = NULL, key = NULL, moe_level = 90, survey = "acs5", ...)
```

Arguments

geography	The geography of your data.
variables	Character string or vector of character strings of variable IDs. tidycensus automatically returns the estimate and the margin of error associated with the variable.
table	The ACS table for which you would like to request all variables. Uses lookup tables to identify the variables; performs faster when variable table already exists through `load_variables(cache = TRUE)`.
cache_table	Whether or not to cache table names for faster future access. Defaults to FALSE; if TRUE, only needs to be called once per dataset. If variables dataset is already cached via the `load_variables` function, this can be bypassed.
year	The year, or endyear, of the ACS sample. 2010 through 2016 are available. Defaults to 2016.
endyear	Deprecated and will be removed in a future release.
output	One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns.
state	The state for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to NULL.
county	The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. Defaults to NULL.
geometry	if FALSE (the default), return a regular tibble of ACS data. if TRUE, uses the tigris package to return an sf tibble with simple feature geometry in the 'geometry' column. state, county, tract, block group, block, and ZCTA geometry are supported.
keep_geo_vars	if TRUE, keeps all the variables from the Census shapefile obtained by tigris. Defaults to FALSE.

summary_var	Character string of a "summary variable" from the ACS to be included in your output. Usually a variable (e.g. total population) that you'll want to use as a denominator or comparison.
key	Your Census API key. Obtain one at http://api.census.gov/data/key_signup.html
moe_level	The confidence level of the returned margin of error. One of 90 (the default), 95, or 99.
survey	The ACS contains one-year, three-year, and five-year surveys expressed as "acs1", "acs3", and "acs5". The default selection is "acs5."
...	Other keyword arguments

Value

A tibble or sf tibble of ACS data

Examples

```
## Not run:
library(tidycensus)
library(tidyverse)
library(viridis)
census_api_key("YOUR KEY GOES HERE")

tarr <- get_acs(geography = "tract", variables = "B19013_001",
                state = "TX", county = "Tarrant", geometry = TRUE)

ggplot(tarr, aes(fill = estimate, color = estimate)) +
  geom_sf() +
  coord_sf(crs = 26914) +
  scale_fill_viridis(option = "magma") +
  scale_color_viridis(options = "magma")


vt <- get_acs(geography = "county", variables = "B19013_001", state = "VT")

vt %>%
mutate(NAME = gsub(" County, Vermont", "", NAME)) %>%
 ggplot(aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_errorbarh(aes(xmin = estimate - moe, xmax = estimate + moe)) +
  geom_point(color = "red", size = 3) +
  labs(title = "Household income by county in Vermont",
       subtitle = "2012-2016 American Community Survey",
       y = "",
       x = "ACS estimate (bars represent margin of error)")


## End(Not run)
```

get_decennial *Obtain data and feature geometry for the decennial Census*

Description

Obtain data and feature geometry for the decennial Census

Usage

```
get_decennial(geography, variables = NULL, table = NULL,
  cache_table = FALSE, year = 2010, sumfile = "sf1", state = NULL,
  county = NULL, geometry = FALSE, output = "tidy",
  keep_geo_vars = FALSE, summary_var = NULL, key = NULL, ...)
```

Arguments

geography	The geography of your data.
variables	Character string or vector of character strings of variable IDs.
table	The Census table for which you would like to request all variables. Uses lookup tables to identify the variables; performs faster when variable table already exists through `load_variables(cache = TRUE)`.
cache_table	Whether or not to cache table names for faster future access. Defaults to FALSE; if TRUE, only needs to be called once per dataset. If variables dataset is already cached via the `load_variables` function, this can be bypassed.
year	The year for which you are requesting data. 1990, 2000, and 2010 are available.
sumfile	The Census summary file. Defaults to sf1; the function will look in sf3 if it cannot find a variable in sf1.
state	The state for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to NULL.
county	The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. Defaults to NULL.
geometry	if FALSE (the default), return a regular tibble of ACS data. if TRUE, uses the tigris package to return an sf tibble with simple feature geometry in the 'geometry' column. state, county, tract, and block group are supported for 1990 through 2010; block and ZCTA geometry are supported for 2000 and 2010.
output	One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns.
keep_geo_vars	if TRUE, keeps all the variables from the Census shapefile obtained by tigris. Defaults to FALSE.
summary_var	Character string of a "summary variable" from the decennial Census to be included in your output. Usually a variable (e.g. total population) that you'll want to use as a denominator or comparison.

key	Your Census API key. Obtain one at http://api.census.gov/data/key_signup.html
...	Other keyword arguments

Value

a tibble or sf tibble of decennial Census data

Examples

```
## Not run:
# Plot of race/ethnicity by county in Illinois for 2010
library(tidycensus)
library(tidyverse)
library(viridis)
census_api_key("YOUR KEY GOES HERE")
vars10 <- c("P0050003", "P0050004", "P0050006", "P0040003")

il <- get_decennial(geography = "county", variables = vars10, year = 2010,
                    summary_var = "P0010001", state = "IL", geometry = TRUE) %>%
  mutate(pct = 100 * (value / summary_value))

ggplot(il, aes(fill = pct, color = pct)) +
  geom_sf() +
  facet_wrap(~variable)



## End(Not run)
```

---

load_variables *Load variables from a decennial Census or American Community Survey dataset to search in R*

Description

Load variables from a decennial Census or American Community Survey dataset to search in R

Usage

```
load_variables(year, dataset, cache = FALSE)
```

Arguments

year	The year for which you are requesting variables. Either the year of the decennial Census, or the endyear for a 5-year ACS sample.
dataset	One of "sf1", "sf3", "acs1", "acs3", "acs5", "acs1/profile", "acs3/profile, "acs5/profile", "acs1/subject", "acs3/subject", or "acs5/subject".
cache	Whether you would like to cache the dataset for future access, or load the dataset from an existing cache. Defaults to FALSE.

Value

A tibble of variables from the requested dataset.

Examples

```
## Not run:
v15 <- load_variables(2015, "acs5", cache = TRUE)
View(v15)

## End(Not run)
```

`moe_product` *Calculate the margin of error for a derived product*

Description

Calculate the margin of error for a derived product

Usage

```
moe_product(est1, est2, moe1, moe2)
```

Arguments

`est1`	The first factor in the multiplication equation (an estimate)
`est2`	The second factor in the multiplication equation (an estimate)
`moe1`	The margin of error of the first factor
`moe2`	The margin of error of the second factor

Value

A margin of error for a derived product

`moe_prop` *Calculate the margin of error for a derived proportion*

Description

Calculate the margin of error for a derived proportion

Usage

```
moe_prop(num, denom, moe_num, moe_denom)
```

Arguments

num	The numerator involved in the proportion calculation (an estimate)
denom	The denominator involved in the proportion calculation (an estimate)
moe_num	The margin of error of the numerator
moe_denom	The margin of error of the denominator

Value

A margin of error for a derived proportion

---

moe_ratio *Calculate the margin of error for a derived ratio*

Description

Calculate the margin of error for a derived ratio

Usage

```
moe_ratio(num, denom, moe_num, moe_denom)
```

Arguments

num	The numerator involved in the ratio calculation (an estimate)
denom	The denominator involved in the ratio calculation (an estimate)
moe_num	The margin of error of the numerator
moe_denom	The margin of error of the denominator

Value

A margin of error for a derived ratio

moe_sum *Calculate the margin of error for a derived sum*

Description

Generates a margin of error for a derived sum. The function requires a vector of margins of error involved in a sum calculation, and optionally a vector of estimates associated with the margins of error. If the associated estimates are not specified, the user risks inflating the derived margin of error in the event of multiple zero estimates. It is recommended to inspect your data for multiple zero estimates before using this function and setting the inputs accordingly.

Usage

```
moe_sum(moe, estimate = NULL)
```

Arguments

moe	A vector of margins of error involved in the sum calculation
estimate	A vector of estimates, the same length as moe, associated with the margins of error

Value

A margin of error for a derived sum

See Also

https://www2.census.gov/programs-surveys/acs/tech_docs/accuracy/MultiyearACSAccuracyofData2015.pdf

tidycensus *Return tidy data frames from the US Census Bureau API*

Description

This packages uses US Census Bureau data but is neither endorsed nor supported by the US Census Bureau.

Author(s)

Kyle Walker

Index

∗Topic **datasets**
    fips_codes, 3

census_api_key, 2

fips_codes, 3

get_acs, 4
get_decennial, 6

load_variables, 7

moe_product, 8
moe_prop, 8
moe_ratio, 9
moe_sum, 10

tidycensus, 10
tidycensus-package *(*tidycensus*)*, 10