Package ‘sociome’

August 4, 2020

Type Package

Title Operationalizing Social Determinants of Health Data for Researchers

Version 1.4.2

Maintainer Nik Krieger <nk@case.edu>

Description Accesses raw data via API and calculates social determinants of health measures for user-specified locations in the US, returning them in tidyverse- and sf-compatible data frames.

License MIT + file LICENSE

BugReports https://github.com/NikKrieger/sociome/issues

Depends R (>= 3.3.0)

Imports censusapi (>= 0.6.0), dplyr (>= 1.0.1), magrittr (>= 1.5), mice (>= 3.10.0.1), psych (>= 2.0.7), purrr (>= 0.3.4), rlang (>= 0.4.7), stringr (>= 1.4.0), tidycensus (>= 0.9.9.5), tidyr (>= 1.1.0)

Suggests ggplot2 (>= 3.3.2), sf (>= 0.9-5), testthat (>= 2.3.2), tibble (>= 3.0.3), tigris (>= 1.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Nik Krieger [aut, cre],
Jarrod Dalton [aut],
Cindy Wang [aut],
Adam Perzynski [aut],
National Institutes of Health/National Institute on Aging [fnd] (The development of this software package was supported by a research grant from the National Institutes of Health/National Institute on Aging, (Principal Investigators: Jarrod E. Dalton, PhD and Adam T. Perzynski, PhD; Grant Number: 5R01AG055480-02). All of its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NIH.)

Repository CRAN

Date/Publication 2020-08-03 22:50:06 UTC

R topics documented:

`acs_vars` *ACS variable names for ADI calculation*

Description

A dataset of the ACS variable names used to calculate the Area Deprivation Index (ADI).

Usage

```
acs_vars
```

Format

A `tibble` with 138 rows and 8 variables:

variable ACS variable name

description Brief description of the data the variable contains

set1 Logical, indicating the variables to be used when calculating ADI using the 1- or 3-year estimates from 2011 and later or when using the 5-year estimates from 2012 or later

set2 Logical, indicating the variables to be used when calculating ADI at the block group level using the 2015 or 2016 estimates

set3 Logical, indicating the variables to be used when calculating ADI using the 2011 5-year estimates

set4 Logical, indicating the variables to be used when calculating ADI using the 2010 1- or 3-year estimates

set5 Logical, indicating the variables to be used when calculating ADI using the 2010 5-year estimates

set6 Logical, indicating the variables to be used when calculating ADI using the 2008 or 2009 1-year estimates

set7 Logical, indicating the variables to be used when calculating ACS estimates not previously mentioned, including the 2009 5-year estimates

dec2010 Logical, indicating the variables to use in conjunction with the few actual 2010 decennial census variables when running `get_adi(year = 2010,dataset = "decennial")`

Note that not all year/estimate combinations are currently supported by the census API and/or `tidycensus`, and some may never be supported.

See Also

`decennial_vars`

`calculate_adi` *Calculate ADI from census data.*

Description

Calculate the area deprivation index using decennial US census or American Community Survey (ACS) variables.

Usage

```
calculate_adi(data_raw, keep_indicators = FALSE, seed = NULL)
```

Arguments

`data_raw`	A data frame, `tibble`, or `sf` ultimately obtained via `tidycensus::get_acs()` or `tidycensus::get_decennial()`, having the data necessary to compute the indicators of the ADI. The columns of his data frame must be named according to the elements of the `variable` column in `sociome::acs_vars` and/or `sociome::decennial_vars`. The easiest way to obtain data like this is to run `sociome::get_adi(raw_data_only = TRUE)`.
`keep_indicators`	Logical indicating whether or not to keep the component indicators of the ADI as well as the original census variables used to calculate them. Defaults to `FALSE`. See `acs_vars` and `decennial_vars` for basic descriptions of the raw census variables.
`seed`	Passed to `set.seed()` when imputation is needed.

Details

The function `get_adi()` calls this function by default as its final step, but some users may want to calculate ADI values for different combinations of areas in a given data set. `get_adi(raw_data_only = TRUE)` returns the raw census data used to calculate ADI. Users may select subsets of such a data set and pipe them into `calculate_adi()`.

This function discerns what kind of census data that `data` contains (ACS, or one of the decennial censuses) by checking for the existence of key variables unique to each kind of data set.

Areas listed as having zero households are excluded from ADI calculation. Their resulting ADIs will be `NA`.

If calling this function directly (i.e., not via `get_adi()`) on a data set that contains median household income (B19013_001) and does not contain median family income (B19113_001), median household income will be used in place of median family income, with a `warning()`. See the "Missingness and imputation" section of `get_adi()`.

Value

A `tibble` with the same number of rows as `data`. Columns include `GEOID`, `NAME`, and `ADI`. Further columns containing the indicators and raw values will also be present if `keep_indicators = TRUE`.

See Also

For more information, see `get_adi()`, especially **ADI factor loadings** and **Missingness and imputation**.

Examples

```
## Not run:
# Wrapped in \dontrun{} because these examples require a Census API key.

raw_census <- get_adi("state", raw_data_only = TRUE)

calculate_adi(raw_census)

calculate_adi(raw_census, keep_indicators = TRUE)

## End(Not run)
```

`census_api_key` *Census API Key installer*

Description

See `tidycensus::census_api_key()`.

`decennial_vars` *Decennial census variable names for ADI calculation*

Description

A dataset of the decennial census variable names used to calculate the Area Deprivation Index (ADI).

Usage

```
decennial_vars
```

Format

A `tibble` with 137 rows and 4 variables:

variable Decennial census variable name

sumfile The summary tape file of the decennial census variable

year The year of the decennial census variable

description Brief description of the data the variable contains

See Also

`acs_vars`

`get_adi` *Get area deprivation index (ADI)*

Description

Returns the ADIs of user-specified areas.

Usage

```
get_adi(
  geography,
  state = NULL,
  county = NULL,
  geoid = NULL,
  zcta = NULL,
  year = 2017,
  dataset = c("acs5", "acs3", "acs1", "decennial"),
  geometry = FALSE,
  shift_geo = FALSE,
  keep_indicators = FALSE,
  raw_data_only = FALSE,
  cache_tables = TRUE,
  key = NULL,
  seed = NULL,
  ...
)
```

Arguments

`geography`	A character string denoting the level of census geography whose ADIs you'd like to obtain. Must be one of `c("state","county","tract","block group",or "zcta")`. Required.
`state`	A character string specifying states whose ADI data is desired. Defaults to `NULL`. Can contain full state names, two-letter state abbreviations, or a two-digit FIPS code/GEOID (must be a vector of strings, so use quotation marks and leading zeros if necessary). Must be left as `NULL` blank if using the `geoid` or `zcta` parameter.
`county`	A vector of character strings specifying the counties whose ADI data you're requesting. Defaults to `NULL`. If not `NULL`, the `state` parameter must have a length of 1. County names and three-digit FIPS codes are accepted (must contain strings, so use quotation marks and leading zeros if necessary). Must be blank if using the `geoid` parameter.
`geoid`	A character vector of GEOIDs (use quotation marks and leading zeros). Defaults to `NULL`. Must be blank if `state`, `county`, or `zcta` is used. Can contain different levels of geography (see details).
`zcta`	A character vector of ZCTAs or the leading digit(s) of ZCTAs (use quotation marks and leading zeros). Defaults to `NULL`. Must be blank if `state`, `county`, or `geoid` is used. Strings under 5 digits long will yield all ZCTAs that begin with those digits. Requires that `geography = "zcta"`. If `geography = "zcta"` and `zcta = NULL`, all ZCTAs in the US will be used.
`year`	Single integer specifying the year of US Census data to use. Defaults to 2017.
`dataset`	The data set used to calculate ADIs. Must be one of `c("acs5","acs3","acs1","decennial")`, denoting the 5-, 3-, and 1-year ACS along with the decennial census. Defaults to `"acs5"`. When `dataset = "decennial"`, `year` must be in `c(1990,2000,2010)`. The 2010 decennial census did not include the long-form questionnaire used in the 1990 and 2000 censuses, so this function uses the 5-year estimates from the 2010 ACS to supply the data not included in the 2010 decennial census. In fact, the only 2010 decennial variables used are H003002, H014002, P020002, and P020008. Important: data are not always available depending on the level of geography and data set chosen. See `https://www.census.gov/programs-surveys/acs/guidance/estimates.html`.
`geometry`	Logical value indicating whether or not shapefile data should be included in the result, making the result an `sf` tibble instead of a plain `tibble`. Defaults to `FALSE`. The shapefile data that is returned is somewhat customizable: see the `shift_geo` and `...` arguments.
`shift_geo`	Logical value. See the `shift_geo` argument of `tidycensus::get_acs()` or `tidycensus::get_decennial()` for details. See `...` below for other ways to customize the shapefile data returned.

keep_indicators	Logical value indicating whether or not the resulting tibble or sf tibble will contain the socioeconomic measures used to calculate the ADI values. Defaults to FALSE. See acs_vars and decennial_vars for basic descriptions of the raw census variables.
raw_data_only	Logical, indicating whether or not to skip calculation of the ADI and only return the census variables. Defaults to FALSE.
cache_tables	The plural version of the cache_table argument in tidycensus::get_acs() or tidycensus::get_decennial(). (get_adi() calls the necessary tidycensus function many times in order to return ADIs, so many tables are cached if TRUE). Defaults to TRUE.
key	Your Census API key as a character string. Obtain one at http://api.census.gov/data/key_signup.html. Defaults to NULL. Not necessary if you have already loaded your key with census_api_key().
seed	Passed to set.seed() in calculate_adi() when imputation is needed.
...	Additional arguments to be passed onto tidycensus::get_acs() or tidycensus::get_decennial(). Currently, none of these functions' formal arguments can be meaningfully customized (doing so will either throw an error or have no effect). However, when setting geometry = TRUE, the tidycensus functions do pass meaningful arguments onto the appropriate tigris function (namely, one of states(), counties(), tracts(), block_groups(), or zctas(), according to the the value of geography). This enables the user to somewhat customize the shapefile data obtained.

Details

Returns a tibble or sf tibble of the area deprivation indices (ADIs) of user-specified locations in the United States, utilizing US Census data. Locations that are listed as having zero households are excluded from ADI calculation: their ADI values will be NA.

Value

If geometry = FALSE, (the default) a tibble. If geometry = TRUE is specified, an sf tibble.

Reference area

The concept of "reference area" is important to understand when using this function. The algorithm that produced the original ADIs employs factor analysis. As a result, the ADI is a relative measure; the ADI of a particular location is dynamic, varying depending on which other locations were supplied to the algorithm. In other words, **ADI will vary depending on the reference area you specify.**

For example, the ADI of Orange County, California is *x* when calculated alongside all other counties in California, but it is *y* when calculated alongside all counties in the US. The get_adi() function enables the user to define a **reference area** by feeding a vector of GEOIDs to its geoid parameter (or alternatively for convenience, states and/or counties to state and county). The function then gathers data from those specified locations and performs calculations using their data alone.

Areas listed as having zero households are excluded from the reference area, and their ADI values will be NA.

The geoid parameter

Elements of geoid can represent different levels of geography, but they all must be either 2 digits (for states), 5 digits (for counties), 11 digits (for tracts), or 12 digits (for block groups). It must contain character strings, so use quotation marks as well as leading zeros where applicable.

ADI factor loadings

The returned tibble or sf tibble is of class adi, and it contains an attribute called loadings, which contains a tibble of the PCA loadings of each factor. This is accessible through attr(name_of_tibble,"loadings").

Missingness and imputation

While this function allows flexibility in specifying reference areas (see the **Reference area** section above), data from the US Census are masked for sparsely populated places, resulting in many missing values.

Imputation is attempted via mice::mice(m = 1,maxit = 50,method = "pmm",seed = 500). If imputation is unsuccessful, an error is thrown, but the dataset of indicators on which imputation was unsuccessful is available via rlang::last_error()$adi_indicators and the raw census data are available via rlang::last_error()$adi_raw_data. The former excludes areas with zero households, but the latter includes them.

One of the ADI indicators is median family income, but methodological issues with the 2015 and 2016 ACS have rendered this variable unavailable at the block group level for those years. When requested, this function will use median household income in its place, with a warning(). See https://www.census.gov/programs-surveys/acs/technical-documentation/user-notes/2016-01.html.

API-related error handling

Depending on user input, this function may call its underlying functions (tidycensus::get_acs() or tidycensus::get_decennial()) many times in order to accommodate their behavior. When these calls are broken up by state or by state and county, a message is printed indicating the state or state and county whose data is being pulled. These calls are wrapped in purrr::insistently(rate = purrr::rate_delay(),quiet = FALSE), meaning that they are attempted over and over until success, and tidycensus error messages are printed as they occur.

Warnings and disclaimers

Please note that this function calls data from US Census servers, so execution may take a long time depending on the user's internet connection and the amount of data requested.

For advanced users, if changing the dataset argument, be sure to know the advantages and limitations of the 1-year and 3-year ACS estimates. See https://www.census.gov/programs-surveys/acs/guidance/estimates.html. for details.

Examples

```
## Not run:
# Wrapped in \dontrun{} because all these examples take >5 seconds
# and require a Census API key.
# ADI of all census tracts in Cuyahoga County, Ohio
get_adi(geography = "tract", state = "OH", county = "Cuyahoga")

# ADI of all counties in Connecticut, using the 2014 ACS1 survey.
# Returns a warning because there are only 8 counties.
# A minimum of 30 locations is recommended.
get_adi(geography = "county", state = "CT", year = 2014, dataset = "acs1")

# Areas with zero households will have an ADI of NA:
queens <-
  get_adi(
    "tract",
    state = "NY",
    county = "Queens",
    keep_indicators = TRUE,
    geometry = TRUE
  )
queens %>%
  dplyr::as_tibble() %>%
  dplyr::select(GEOID, NAME, ADI, households = B11005_001) %>%
  dplyr::filter(is.na(ADI) | households == 0) %>%
  print(n = Inf)

# geoid argument allows for highly customized reference populations.
# ADI of all census tracts in the GEOIDs stored in "delmarva" below:
# Notice the mixing of state- ("10") and county-level GEOIDs (the others).
delmarva_geoids <- c("10", "51001", "51131", "24015", "24029", "24035",
                     "24011", "24041", "24019", "24045", "24039", "24047")
delmarva <-
  get_adi(
    geography = "tract",
    geoid = delmarva_geoids,
    dataset = "decennial",
    year = 2000,
    geometry = TRUE
  )

# Demonstration of geom_sf() integration:
require(ggplot2)

# The na.value argument changes the fill of NA ADI areas.
delmarva %>% ggplot() + geom_sf(aes(fill = ADI), lwd = 0)

# Setting direction = -1 makes the less deprived areas the lighter ones
# The argument na.value changes the color of zero-household areas
queens %>%
  ggplot() +
  geom_sf(aes(fill = ADI), lwd = 0) +
  scale_fill_viridis_c(na.value = "red", direction = -1)

# Obtain factor loadings:
attr(queens, "loadings")
```

```
## End(Not run)
```

`get_geoids` *Obtain GEOIDs of places*

Description

Returns a `tibble` of GEOIDs, names, and decennial census population of user-specified locations.

Usage

```
get_geoids(
  geography,
  state = NULL,
  county = NULL,
  geoid = NULL,
  year = 2010,
  geometry = FALSE,
  cache_tables = TRUE,
  key = NULL,
  ...
)
```

Arguments

`geography`	A character string denoting the level of census geography whose GEOIDs you'd like to obtain. Must be one of `c("state","county","tract","block group","block")`. Note that block-level data cannot be obtained from 1990 and 2000 decennial census data due to limitations in `tidycensus::get_decennial()`. Whereas block-level 2010 decennial census data are available, block-level ADIs cannot be calculated due to the removal of the long-form questionnaire from the 2010 decennial census.
`state, county, geoid, geometry, cache_tables, key`	See the descriptions of the arguments in `get_adi()`.
`year`	Single integer specifying the year of US Census data to use. Defaults to 2010. Based on this year, data from the most recent decennial census will be returned (specifically, `year <-floor(year / 10) * 10` is run).
`...`	Additional arguments to be passed to `tidycensus::get_decennial()`. Not recommended; use at your own risk.

Details

This allows users to quickly obtain all GEOIDs in a specified location at a specific level of geography without having to manually look them up somewhere else.

This facilitates calls to `get_adi()` that involve somewhat complicated reference areas.

Examples

```
## Not run:
# Wrapped in \dontrun{} because it requires a Census API key.

# Get all tract GEOIDs for Manhattan
tracts <- get_geoids(geography = "tract", state = "New York", county = "New York")
tracts

# Get all block GEOIDs for the fifth tract on that list
get_geoids(geography = "block", geoid = tracts$GEOID[5])

## End(Not run)
```

Index