Package ‘censusxy’

December 6, 2022

Title Access the U.S. Census Bureau's Geocoding API System

Version 1.1.1

Description Provides access to the U.S. Census Bureau's API for matching American street addresses with their longitude and latitude. This includes both single address matching as well as batch functionality for multiple addresses. Census geographies can be appended to addresses if desired, and reverse geocoding of point locations to census geographies is also supported.

Depends R (>= 3.4)

License GPL-3

URL https://chris-prener.github.io/censusxy/

BugReports https://github.com/chris-prener/censusxy/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Imports doParallel, foreach, httr, parallel, utils

Suggests covr, knitr, rmarkdown, sf, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Christopher Prener [aut, cre] (<https://orcid.org/0000-0002-4310-9888>), Branson Fox [aut] (<https://orcid.org/0000-0002-4361-2811>), Christopher Kenny [ctb] (<https://orcid.org/0000-0002-9386-6860>)

Maintainer Christopher Prener <chris.prener@gmail.com>

Repository CRAN

Date/Publication 2022-12-06 13:00:08 UTC

R topics documented:

cxy_benchmarks .................................................. 2
cxy_geocode ..................................................... 2
cxy_geocode

Description

Batch Geocode Parsed Addresses

Provide access to the US Census Bureau batch endpoints for locations and geographies. The function implements iteration and optional parallelization in order to geocode datasets larger than the API limit of 1,000 and more efficiently than sending 10,000 per request. It also supports multiple outputs, including (optionally, if sf is installed,) sf class objects.

cxy_benchmarks

Get Current Valid Benchmarks

Description

Get Current Valid Benchmarks

Usage

cxy_benchmarks()

Value

A data.frame containing valid Census Benchmarks

Examples

```r
cxy_benchmarks()
```
Usage

cxy_geocode(
  .data,
  id = NULL,
  street,
  city = NULL,
  state = NULL,
  zip = NULL,
  return = "locations",
  benchmark = "Public_AR_Current",
  vintage = NULL,
  timeout = 30,
  parallel = 1,
  class = "dataframe",
  output = "simple"
)

Arguments

.data  data.frame containing columns with structured address data
id     Optional String - Name of column containing unique ID
street String - Name of column containing street address
city   Optional String - Name of column containing city
state  Optional String - Name of column containing state
zip    Optional String - Name of column containing zip code
return One of 'locations' or 'geographies' denoting returned information from the API. If you would like Census geography data, you must specify a valid vintage for your benchmark.
benchmark Optional Census benchmark to geocode against. To obtain current valid benchmarks, use the cxy_benchmarks() function.
vintage Optional Census vintage to geocode against. You may use the cxy_vintages() function to obtain valid vintages.
timeout Numeric, in minutes, how long until request times out
parallel Integer, number of cores greater than one if parallel requests are desired. All operating systems now use a SOCK cluster, and the dependencies are not longer suggested packages. Instead, they are installed by default. Note that this value may not represent more cores than the system reports are available. If it is larger, the maximum number of available cores will be used.
class  One of 'dataframe' or 'sf' denoting the output class. 'sf' will only return matched addresses.
output One of 'simple' or 'full' denoting the returned columns. Simple returns just coordinates.
Details

Parallel requests are supported across platforms. If supported (POSIX platforms) the process is forked, otherwise a SOCK cluster is used (Windows). You may not specify more cores than the system reports are available.

Value

A data.frame or sf object containing geocoded results

Examples

```r
# load data
x <- stl_homicides[1:10,]

# geocode
cxy_geocode(x, street = 'street_address', city = 'city', state = 'state', zip = 'postal_code',
             return = 'locations', class = 'dataframe', output = 'simple')
```

---

cxy_geography | Geocode Single Coordinate Pair

Description

Provides access to the GeoLookup API of the US Census Bureau. Returns census geographies for a single geographic point.

Usage

```r
cxy_geography(
  lon, 
  lat,  
  benchmark = "Public_AR_Current",  
  vintage = "Current_Current"
)
```

Arguments

- `lon` | Numeric or String Containing Longitude (x) of Point
- `lat` | Numeric or String Containing Latitude (y) of Point
- `benchmark` | Optional ID or Name of Census Benchmark. See Details.
- `vintage` | Optional ID or Name of Census Vintage. See Details.
Details

This function can be used to locate geographic information given a geographic point. It does not provide an address like a reverse-geocoder.

To obtain current valid benchmarks, use the `cxy_benchmarks()` function.

To use this function, you must specify a valid vintage for your benchmark. You may use the `cxy_vintages()` function to obtain valid Vintages for a given benchmark. See vignette('censusxy') for a full walkthrough.

Value

A data.frame containing matched address or NULL if not matches.

Examples

cxy_geography(lon = -90.23324, lat = 38.63593)

cxy_oneline(address, return = "locations", benchmark = "Public_AR_Current", vintage = NULL)

---

cxy_oneline Geocode Single One Line Address

Description

Provides access to the oneline single address geocoding API from the US Census Bureau. This can be used with an address that is not parsed.

Usage

cxy_oneline(
  address,
  return = "locations",
  benchmark = "Public_AR_Current",
  vintage = NULL
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>String containing a single line address</td>
</tr>
<tr>
<td>return</td>
<td>One of 'locations' or 'geographies' See Details.</td>
</tr>
<tr>
<td>benchmark</td>
<td>Optional ID or Name of Census Benchmark. See Details.</td>
</tr>
<tr>
<td>vintage</td>
<td>Optional ID or Name of Census Vintage. See Details.</td>
</tr>
</tbody>
</table>

Details

To obtain current valid benchmarks, use the `cxy_benchmarks()` function.

If you want to append census geographies, you must specify a valid vintage for your benchmark. You may use the `cxy_vintages()` function to obtain valid Vintages. See vignette('censusxy') for a full walkthrough.
Value

A data.frame containing matched address or NULL if not matches

Examples

cxy_oneline(address = "20 N Grand Blvd, St Louis, MO 63108", return = "locations")

---

cxy_single  Geocode Single Parsed Address

Description

Provides access to the structured single address geocoding API from the US Census Bureau.

Usage

cxy_single(
  street,
  city = NULL,
  state = NULL,
  zip = NULL,
  return = "locations",
  benchmark = "Public_AR_Current",
  vintage = NULL
)

Arguments

street  String containing street address
city   Optional String containing city
state  Optional String containing state
zip    Optional String or Integer containing 5-digit Zip Code
return One of 'locations' or 'geographies' See Details.
benchmark Optional ID or Name of Census Benchmark. See Details.
vintage Optional ID or Name of Census Vintage. See Details.

Details

To obtain current valid benchmarks, use the cxy_benchmarks() function.
If you want to append census geographies, you must specify a valid vintage for your benchmark.
You may use the cxy_vintages() function to obtain valid Vintages. See vignette('censusxy') for a full walkthrough.
**cxy_vintages**

**Value**

A data.frame containing matched address or NULL if not matches

**Examples**

```r
  cxy_single(street = "20 N Grand Blvd", city = "St Louis", state = "MO", zip = "63108", return = "locations")
```

---

**cxy_vintages**  
*Get Current Valid Vintages*

**Description**

Get Current Valid Vintages

**Usage**

```r
  cxy_vintages(benchmark)
```

**Arguments**

- `benchmark` Name or ID of Census Benchmark

**Value**

A data.frame containing valid Census Vintages for a given benchmark

**Examples**

```r
  cxy_vintages("Public_AR_Current")
```

---

**stl_homicides**  
*Homicides in the City of St. Louis, 2008 - 2018*

**Description**

An example data set containing the addresses for homicides reported by the Saint Louis Metropolitan Police Department

**Usage**

```r
  data(stl_homicides)
```
Format
A tibble with 1822 rows and 6 variables:

- **street_address** number, street and street suffix where homicide occurred
- **year** year homicide occurred
- **date** date homicide occurred
- **state** state abbreviation of location, in these data, all "MO"
- **postal_code** zipcode/postal code of location, in these data all NA
- **city** city of location, in these data all "St. Louis"

Source
St. Louis Metropolitan Police Department

Examples
- `str(stl_homicides)`
- `head(stl_homicides)`
Examples

str(stl_homicides_small)
head(stl_homicides_small)
Index

* datasets
  stl_homicides, 7
  stl_homicides_small, 8

  cxy_benchmarks, 2
  cxy_geocode, 2
  cxy_geography, 4
  cxy_oneline, 5
  cxy_single, 6
  cxy_vintages, 7

  stl_homicides, 7
  stl_homicides_small, 8