Package ‘nuts’

July 13, 2024

Title Convert European Regional Data

Version 1.1.0

Description Motivated by changing administrative boundaries over time, the 'nuts' package can convert European regional data with NUTS codes between versions (2006, 2010, 2013, 2016 and 2021) and levels (NUTS 1, NUTS 2 and NUTS 3). The package uses spatial interpolation as in Lam (1983) <doi:10.1559/152304083783914958> based on granular (100m x 100m) area, population and land use data provided by the European Commission's Joint Research Center.

License MIT + file LICENSE

URL https://docs.ropensci.org/nuts/, https://github.com/ropensci/nuts

BugReports https://github.com/ropensci/nuts

Depends R (>= 3.5.0)

Imports cli, dplyr, glue, lifecycle, rlang

Suggests distill, eurostat, formatR, ggalluvial, ggfittext, ggplot2, ggpubr, ggrepel, gridExtra, kableExtra, knitr, raster, RColorBrewer, readr, rmarkdown, sf, stringr, terra, testthat, tidyr, withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Author Moritz Hennicke [aut, cre, cph]
  (<https://orcid.org/0000-0001-6811-1821>), Werner Krause [aut, cph] (<https://orcid.org/0000-0002-5069-7964>), Pueyo-Ros Josep [rev] (Josep reviewed the package for rOpenSci, see https://github.com/ropensci/software-review/issues/623#issuecomment-1951446662), Le Meur Nolwenn [rev] (Nolwenn reviewed the package for rOpenSci, see https://github.com/ropensci/software-review/issues/623#issuecomment-1961501137)
all_nuts_codes

Maintainer Moritz Hennicke <Aoritz@posteo.de>
Repository CRAN
Date/Publication 2024-07-13 10:50:02 UTC

Contents

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>all_nuts_codes</td>
<td>2</td>
</tr>
<tr>
<td>cross_walks</td>
<td>3</td>
</tr>
<tr>
<td>manure</td>
<td>3</td>
</tr>
<tr>
<td>nuts_aggregate</td>
<td>4</td>
</tr>
<tr>
<td>nuts_classify</td>
<td>5</td>
</tr>
<tr>
<td>nuts_convert_version</td>
<td>7</td>
</tr>
<tr>
<td>nuts_get_data</td>
<td>9</td>
</tr>
<tr>
<td>nuts_get_missing</td>
<td>10</td>
</tr>
<tr>
<td>nuts_get_version</td>
<td>11</td>
</tr>
<tr>
<td>nuts_test_multiple_versions</td>
<td>12</td>
</tr>
<tr>
<td>patents</td>
<td>13</td>
</tr>
</tbody>
</table>

Index 14

---

**all_nuts_codes** List of all NUTS codes and classifications

**Description**

The data frame stores all NUTS codes in hierarchical levels 1, 2 and 3 by NUTS classification versions 2006, 2010, 2013, 2016 and 2021.

**Usage**

```r
all_nuts_codes
```

**Format**

```r
all_nuts_codes:
A data frame with 8,896 rows and 2 columns:
code NUTS code
version NUTS versions
country Country name
```

**Source**

**Description**

The table contains population, area and surface flows between two NUTS regions and different NUTS code classifications. NUTS regions are at 1st, 2nd and 3rd level. NUTS versions are 2006, 2010, 2013, 2016 and 2021.

**Usage**

`cross_walks`

**Format**

cross_walks:
A data frame with 47,340 rows and 9 columns:
- **from_code**: Departing NUTS code
- **to_code**: Desired NUTS code
- **from_version**: Departing NUTS version
- **to_version**: Desired NUTS version
- **level**: NUTS division level
- **country**: Country name
- **areaKm**: Area size flow
- **pop18**: 2018 population flow
- **pop11**: 2011 population flow
- **artif_surf18**: 2018 artificial surfaces flow
- **artif_surf12**: 2012 artificial surfaces flow

**Source**


---

**Description**

The data frame contains the number of different manure storage facilities from the Farm Structure Survey in all (former) EU member states, such as Iceland, Norway, Switzerland and Montenegro at the NUTS 3 level. Please see the link indicated below for more information.
Usage

manure

Format

manure:
A data frame with 17,151 rows and 4 columns:
  indic_ag 9 indicators: All manure storage facilities, solid dung, liquid manure slurry, slurry: tank, slurry: lagoon; covered facilities with either dung, liquid manure, slurry
  geo  NUTS 1, 2, 3 or National level
  time  Years 2000, 2003 and 2010
  values  Number

Source


nuts_aggregate  Aggregate to higher order NUTS levels

Description

nuts_aggregate() transforms regional NUTS data between NUTS levels.

Usage

nuts_aggregate(
  data,
  to_level,
  variables,
  weight = NULL,
  missing_rm = FALSE,
  missing_weights_pct = FALSE,
  multiple_versions = c("error", "most_frequent")
)

Arguments

data  A nuts.classified object returned by nuts-classify().
to_level  Number corresponding to the desired NUTS level to be aggregated to: 1 or 2.
variables  Named character specifying variable names and variable type ('absolute' or 'relative'), e.g. c('var_name' = 'absolute').
weight  String with name of the weight used for conversion. Can be area size 'areaKm' (default), population in 2011 'pop11' or 2018 'pop18', or artificial surfaces in 2012 'artif_surf12' and 2018 'artif_surf18'.

**nuts_classify**

**Description**

`nuts_classify()` can identify the NUTS version year and level from a variable containing NUTS codes.

**missing_rm**

Boolean that is FALSE by default. TRUE removes regional flows that depart from missing NUTS codes.

**missing_weights_pct**

Boolean that is FALSE by default. TRUE computes the percentage of missing weights due to missing departing NUTS regions for each variable.

**multiple_versions**

By default equal to 'error', when providing multiple NUTS versions within groups. If set to 'most_frequent' data is converted using the best-matching NUTS version.

**Details**

Console messages can be controlled with `rlang::local_options(nuts.verbose = "quiet")` to silence messages and `nuts.verbose = "verbose"` to switch messages back on.

**Value**

A tibble containing NUTS codes, aggregated variable values, and possibly grouping variables.

**Examples**

```r
library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

# Data varies at the NUTS level x indicator x year x country x NUTS code level
head(manure)

# Aggregate from NUTS 3 to 2 by indicator x year
manure %>%
  filter(nchar(geo) == 5) %>%
  nuts_classify(nuts_code = "geo",
                group_vars = c('indic_ag','time')) %>%
# Group vars are automatically passed on
nuts_aggregate(to_level = 2,
               variables = c('values'='absolute'),
               weight = 'pop18')
```

---

**nuts_classify**

Classify version of NUTS codes
Usage

nuts_classify(
  data,
  nuts_code,
  group_vars = NULL,
  ties = c("most_recent", "oldest")
)

Arguments

data A data frame or tibble that contains a variable with NUTS 1, 2 or 3 codes and possibly other variables. NUTS codes must be of the same level and need to be unique, unless additional grouping variables are specified. No duplicate NUTS codes within groups allowed.
nuts_code Variable name containing NUTS codes
group_vars Variable name(s) for classification within groups. nuts_classify() always computes overlap within country. Hence, country variables should not be specified. NULL by default.
ties Picks 'most_recent' or 'oldest' version when overlap is identical across multiple NUTS versions. 'most_recent' by default.

Details

Console messages can be controlled with rlang::local_options(nuts.verbose = "quiet") to silence messages and nuts.verbose = "verbose" to switch messages back on.

Value

A list of three tibbles. The first tibble contains the original data with the classified NUTS version, level, and country. The second tibble lists the group-specific overlap with each NUTS version. The third tibble shows missing NUTS codes for each group.

The output can be passed to nuts_convert_version() to convert data across NUTS versions and nuts_aggregate() to aggregate across NUTS levels.

Examples

library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

# Data varies at the NUTS level x indicator x year x country x NUTS code level
head(manure)

# Classify version of NUTS 2 codes in Germany
manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('"DE', geo)) %>%

# Load EUROSTAT data of manure storage deposits
data(manure)

# Data varies at the NUTS level x indicator x year x country x NUTS code level
head(manure)

# Classify version of NUTS 2 codes in Germany
manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('"DE', geo)) %>%

# Load EUROSTAT data of manure storage deposits
data(manure)

# Data varies at the NUTS level x indicator x year x country x NUTS code level
head(manure)

# Classify version of NUTS 2 codes in Germany
manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('"DE', geo)) %>%
nuts_convert_version

Description

nuts_convert_version() transforms regional NUTS data between NUTS versions.

Usage

nuts_convert_version(
  data, 
  to_version, 
  variables, 
  weight = NULL, 
  missing_rm = FALSE, 
  missing_weights_pct = FALSE, 
  multiple_versions = c("error", "most_frequent")
)

Arguments

data A nuts.classified object returned by nuts_classify().
to_version String with desired NUTS version the function should convert to. Possible versions: '2006', '2010', '2013', '2016' or '2021'.
variables Named character specifying variable names and variable type ('absolute' or 'relative') e.g. c('var_name' = 'absolute')
weight String with name of the weight used for conversion. Can be area size 'areaKm' (default), population in 2011 'pop11' or 2018 'pop18', or artificial surfaces in 2012 'artif_surf12' and 2018 'artif_surf18'.
missing_rm Boolean that is FALSE by default. TRUE removes regional flows that depart from missing NUTS codes.
missing_weights_pct

Boolean that is FALSE by default. TRUE computes the percentage of missing weights due to missing departing NUTS regions for each variable.

multiple_versions

By default equal to 'error', when providing multiple NUTS versions within groups. If set to 'most_frequent' data is converted using the best-matching NUTS version.

Details

Console messages can be controlled with\texttt{rlang::local_options(nuts.verbose = "quiet")} to silence messages and\texttt{nuts.verbose = "verbose"} to switch messages back on.

Value

A tibble containing NUTS codes, converted variable values, and possibly grouping variables.

Examples

```
library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

# Data varies at the NUTS level x indicator x year x country x NUTS code level
head(manure)

# Convert NUTS 2 codes in Germany from 2006 to 2021 version
manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('^DE', geo)) %>%
  filter(time == 2003) %>%
  select(-indic_ag, -time) %>%
  nuts_classify(nuts_code = "geo") %>%
  nuts_convert_version(to_version = '2021',
                       weight = 'pop18',
                       variables = c('values' = 'absolute'))

# Convert NUTS 3 codes by country x year, classifying version first
manure %>%
  filter(nchar(geo) == 5) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  select(-indic_ag) %>%
  nuts_classify(nuts_code = 'geo', group_vars = c('time')) %>%
  nuts_convert_version(to_version = '2021',
                       weight = 'pop18',
                       variables = c('values' = 'absolute'))
```
Description

`nuts_get_data()` returns the classified data after running `nuts_classify()`.

Usage

`nuts_get_data(data)`

Arguments

data A `nuts.classified` object returned by `nuts_classify()`.

Details

Console messages can be controlled with `rlang::local_options(nuts.verbose = "quiet")` to silence messages and `nuts.verbose = "verbose"` to switch messages back on.

Value

A tibble containing the original data with the classified NUTS version, level, and country.

Examples

```r
library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

data(classified <- manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('DE', geo)) %>%
  filter(time == 2003) %>%
  select(-indic_ag, -time) %>%
# Data varies at the NUTS code level
  nuts_classify(nuts_code = 'geo'))

nuts_get_data(classified)
```
nuts_get_missing

Return missing NUTS codes in classified NUTS data

Description

nuts_get_missing() returns the classified data after running nuts_classify().

Usage

nuts_get_missing(data)

Arguments

data A nuts.classified object returned by nuts_classify().

Details

Console messages can be controlled with rlang::local_options(nuts.verbose = "quiet") to silence messages and nuts.verbose = "verbose" to switch messages back on.

Value

A tibble listing missing NUTS codes for each group.

Examples

library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

classified <- manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('DE', geo)) %>%
  filter(time == 2003) %>%
  select(-indic_ag, -time) %>%
  # Data varies at the NUTS code level
  nuts_classify(nuts_code = 'geo')

nuts_get_missing(classified)
nuts_get_version

Return version overlap of classified NUTS data

Description

nuts_get_version() returns the classified data after running nuts_classify().

Usage

nuts_get_version(data)

Arguments

data A nuts.classified object returned by nuts_classify().

Details

Console messages can be controlled with rlang::local_options(nuts.verbose = "quiet") to silence messages and nuts.verbose = "verbose" to switch messages back on.

Value

A tibble that lists the group-specific overlap with each NUTS version.

Examples

library(dplyr)

# Load EUROSTAT data of manure storage deposits
data(manure)

# Classify version of NUTS 2 codes in Germany
classified <- manure %>%
  filter(nchar(geo) == 4) %>%
  filter(indic_ag == 'I07A_EQ_Y') %>%
  filter(grepl('DE', geo)) %>%
  filter(time == 2003) %>%
  select(-indic_ag, -time) %>%
# Data varies at the NUTS code level
nuts_classify(nuts_code = 'geo')

nuts_get_version(classified)
nuts_test_multiple_versions

Helper function to test for multiple versions

Description

nuts_test_multiple_versions is called from either nuts_convert_version or nuts_aggregate to selects the most frequent version within groups or throw an error.

Usage

nuts_test_multiple_versions(group_vars, multiple_versions, data_versions, data)

Arguments

- **group_vars**: Variable name(s) for classification within groups. Always computes overlap within country. NULL by default.
- **multiple_versions**: By default equal to 'error', when providing multiple NUTS versions within groups.
- **data_versions**: Data versions
- **data**: A nuts.classified object returned by nuts_classify().

Value

A tibble containing NUTS codes, the potential number of rows dropped and a message with the results of the test.

Examples

```r
library(dplyr)
df <- manure %>%
  filter(nchar(geo) == 5) %>%
  select(geo, indic_ag, values) %>%
  distinct(geo, .keep_all = TRUE) %>%
  nuts_classify(nuts_code = "geo",
                group_vars = "indic_ag",
                data = .)

nuts_test_multiple_versions(group_vars = "indic_ag",
                            multiple_versions = "most_frequent",
                            data_versions = df$versions_data,
                            data = df$data)
```
Description

The data frame contains information on patent applications to the European Patent Office by year and NUTS 3 regions.

Usage

patents

Format

patents:
A data frame with 104,106 rows and 4 columns:

unit  4 indicators: Number, Nominal GDP in billion euro, Per million habitants, Per million of population in the labor force
geo   NUTS 1, 2, 3 or National level
values Values

Source

Index

∗ datasets
  all_nuts_codes, 2
  cross_walks, 3
  manure, 3
  patents, 13

all_nuts_codes, 2

cross_walks, 3

manure, 3

nuts_aggregate, 4
nuts_aggregate(), 6
nuts_classify, 5
nuts_classify(), 4, 7, 9–12
nuts_convert_version, 7
nuts_convert_version(), 6
nuts_get_data, 9
nuts_get_missing, 10
nuts_get_version, 11
nuts_test_multiple_versions, 12

patents, 13