Package 'incadata'

July 13, 2018

Type Package

Title Recognize and Handle Data in Formats Used by Swedish Cancer Centers

Version 0.6.4

Description Handle data in formats used by cancer centers in Sweden, both from 'INCA' (the current register platform, (see <http://rcc.incanet.se> for more information) and
by the older register platform 'Rockan' (used in the Western and Northern part of the country).
All variables are coerced to suitable classes based on their format.
Dates (from various formats such as with missing month or day, with or without century prefix or with just a week number) are all recognized as dates and coerced to the ISO 8601 standard (Y-m-d).
Boolean variables (internally stored either as 0/1 or ``True''/``False''/blanks when exported) are coerced to logical.
Variable names ending in '_Beskrivning' and '_Varde' will be character, and 'PERSNR' will be coerced (if possible) to a valid personal identification number 'pin' (by the 'sweidnumbr' package).
The package also allow the user to interactively choose if a variable should be coerced into a potential format even though not all of its values might conform to the recognized pattern.
It also contain a caching mechanism in order to temporarily store data sets with its newly decided formats in order to not rerun the identification process each time.
And finally, it also include a mechanism to aid the documentation process connected to projects build on data from 'INCA'.

License GPL-2

Depends R (>= 2.10)

Imports backports, decoder, dplyr, rccmisc, rvest, sweidnumbr, xml2

Suggests testthat, knitr, rmarkdown, R.rsp

VignetteBuilder R.rsp

URL https://www.bitbucket.org/cancercentrum/incadata

BugReports https://www.bitbucket.org/cancercentrum/incadata/issues

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Erik Bulow [aut, cre] (<https://orcid.org/0000-0002-9973-456X>)

Maintainer Erik Bulow <erik.bulow@rccvast.se>

Repository CRAN

Date/Publication 2018-07-13 09:30:03 UTC

R topics documented:

`as.Dates` *Converting potential date to Date vector*

Description

The function recognizes dates in formats used by INCA and Rockan.

Usage

```
as.Dates(x)
```

Arguments

x	atomic vector

Details

Regular expressions are used to match any of the following date formats:

- `Y-m-d`: The ISO 8601 standard such as "2017-02-16" as used by INCA.
- `Ymd`: such as "20160216" as used by the Rockan registers
- Any of the above with missing day such as "2017-02-00" or "20170200" as used if the exact date is unknown.
- Any of the above with missing month such as "2017-00-00" or "20170000" as sometimes used if the exact date is unknown.
- Dates between 1950 and 1980 can have missing century prefix, such as "67-01-01", "670101", "670100", "670000" etc as earlier used for some dates in the Rockan registers.
- Dates from the 20th century can also have month and day changed to week number such as "6723" or "196723" as sometimes used for death dates in the cancer register (originating from the population register).
- The special INCA variable `SKAPAD_DATUM` is also recognized as data but is originally a date and time object (`POSIXct`

All dates are coerced to `Y-m-d` (ISO 8601):

- a missing day is set to 15
- a missing month is set to July
- a week number is translated to the "median day" of that week
- `SKAPAD_DATUM` has its time stamp dropped

An alternative would be to use random assignments of dates within specified periods. This would have some benefits but does not conform to behavior used elsewhere by INCA.

Value

vector of class "Date"

Possible date range

All potential dates are accepted as such. RCC data should however only contain historic data. Dates from the future does therefore raise warnings. The same is true for dates before 1830. The Swedish cancer register was initiated in 1958. The earliest possible dates found in the register should therefore originate from birth date of really old people diagnosed with cancer during that year.

See Also

as.Date

Examples

```
as.Dates(c(1212121212, "20000101", "2014-10-15", 5806))

## Not run:
# Note that the as.Date (as oppose to as.Dates)
# does not handle missing dates as empty strings
as.Date(c("", "2017-02-16")) # Error
as.Dates(c("", "2017-02-16")) # NA "2017-02-16"

## End(Not run)
```

---

as.incadata *Identify data formats used by INCA and Rockan*

Description

Coerce data of any form to its relevant type as identified either by column/vector names or by variable content and convert all variable names to lower case.

Usage

```
as.incadata(x, ...)

is.incadata(x)

## S3 method for class 'data.frame'
as.incadata(x, decode = TRUE, id = TRUE, ask = TRUE,
  ...)

## Default S3 method:
as.incadata(x, n_i = NULL, ...)
```

Arguments

x	data
...	arguments passed to exceed_threshold (of most use is probably "threshold" and "force", see the "interactive use" section below)
decode	Should decode be applied to variables with identified variable names? (TRUE by default).
id	Should an id-column be added (see id)?
ask	ask for input if unsure how to coerce variables (see the "interactive use" section below)
n_i	used internally between methods (should not be set by the user)

Details

Vectors are coerced to identified formats in the following order:

- vectors recognized as Boolean by `is.incalogical` are coerced to logical (this is a strict format than can not be contaminated with any unwanted values, section "interactive use" below does therefore not apply to these values)
- vectors with an already specified class attribute (except the common "factor" class) remains as members of that class
- columns or vectors names 'persnr' or 'pnr' will be coerced to the 'pin' class by `as.pin`
- columns or vectors with names ending in '_Beskrivning', '_Varde', '_Gruppnamn' or '_id' are always treated as character (not factors; see section "factors" below)
- column or vectors named "PAT_ID", "KON_VALUE" and "LAN_VALUE" are also always treated as character. These could also be thought of as numerics but are treated as character internally by INCA. To stay with that format ensures the assumption of a stable format.
- If all values of a vector are NA, it is coerced from logical to character. This might be a faulty assumption but it is in fact more likely that an empty vector is a character variable (since most INCA variables are of type character) than that it is a Boolean vector (that has its own format in INCA).
- Dates in formats recognized by `as.Dates` are coerced to such.
- Integers (even if stored as characters or factors) without leading zeros (except when the zero is the only digit) are coerced to integers
- Numerics (even if stored as characters or factors) containing either a Swedish decimal comma or an English decimal point are coerced to numeric (with possible commas changed to points).
- all other formats are coerced to character. This includes integers with leading zeroes (since these might be unit codes where a leading zero might bear meaning).

Value

`as.incadata.data.frame` object of class `incadata` based on the "tibble"-class used within the "tidyverse" with all variables possibly coerced as described above.

`as.incadata.default` input vector coerced to relevant class

`is.incadata` `TRUE` for objects of class `incadata`, otherwise `FALSE`

factors

Note that the `incadata` format does not include factors. Factors can be really useful for some applications but our philosophy is that they should be explicitly stated as such when needed. It is otherwise common that factor levels are created just by the responses present in a certain data set. These might or might not contain a complete list of possible alternatives from a INCA variable with a fixed value set.

interactive use

Some vectors can be undoubtedly recognized according to specifications above. It is however possible that a vector of an intended format might have been "contaminated" with data of some other

form. This might happen for example when a numeric variable is technically a character in INCA. For example a hospital unit code like `c(111, 123, "?")` might suddenly occur (if someone use a question mark as placeholder for an unknown code). Ordinary coercing rules of R would treat this vector as a character (see `c`), although it might be more correct to treat it as a numeric with "?" set to `NA`.

The `as.incadata` function relies on `exceed_threshold` to ignore such contaminated values if they represent only a (preferably small) proportion of the values.

By default, if contaminated values exist but only to a proportion of less than 10 percent, the function will stop and ask the user for input on how to handle this variable. If the proportion exceeds 10 percent, ordinary coercing principles will apply.

The 10 percent limit can be modified by argument `threshold` and it is possible to force vectors with contaminated values to the otherwise potential format (without the need of individual confirmation) by setting argument `force = TRUE` (passed to `exceed_threshold`).

`documents` *Download and possibly open INCA documentation*

Description

Download and possibly open INCA documentation

Usage

```
documents(..., dir = ".", max_open = 3)
```

Arguments

`...`	Arguments passed on to `find_documents` **pattern** part of document name to look for **x** name of register to look for
`dir`	directory where to save files
`max_open`	maximum number of files to open automatically (only on Mac OS X). Set to 0 to avoid any opening of files.

Value

Nothing. The function is called for its side effects.

Examples

```
## Not run:
documents("lunga", "uppfoljning")

## End(Not run)
```

ex_data *Synthetic example data from INCA*

Description

A data set resembling the typical form of INCA data. Variable names are real but all data has been carefully anonymized!

Usage

```
ex_data
```

Format

A data frame (not an object of class `incadata` with 497 rows and 433 variables

Details

All data is random! There is no logical relation between any variables, not even between `x_Beskrivning` and `x_Varde`!

Examples

```
# Inspect the data
dplyr::glimpse(ex_data)

# Coerce to incadata
as.incadata(ex_data)
```

find_documents *List all documents for a register*

Description

List all documents for a register

Usage

```
find_documents(x, pattern = NULL)
```

Arguments

`x`	name of register to look for
`pattern`	part of document name to look for

Value

data frame with names of documents and corresponding URL:s for specified register

Examples

```
## Not run:
find_documents("all")
find_documents("peniscancer", "uppfoljning")

## End(Not run)
```

`find_register` *Find register by name*

Description

Find register by name

Usage

```
find_register(x = NULL)
```

Arguments

x	name of register to look for

Value

Name of existing register according to <www.incanet.se>

Examples

```
find_register("all") # "akut lymfatiskt leukemi all"
## Not run:
find_register("kronisk") # More than one possible alternative

## End(Not run)
```

id *Add id variables to data frame*

Description

Construct id variable for patient data.

Usage

```
id(x, id = c("persnr", "pnr", "pat_id", "pn", "id"), ignore.case = TRUE)
```

Arguments

x	data frame
id	names of a possible id variable found in x
ignore.case	should name matching be done regardless of character case?

Value

Character variable with either the first name from id found in x or rownames(x) if no named column found.

is.incalogical *Coerce to logical if value is logical according to INCA*

Description

Boolean vectors in INCA are stored internally as 0/1 and are changed to "True"/blank when exported. These functions identify such a variable as Boolean and can coerce it to such.

Usage

```
is.incalogical(x)

incalogical2logical(x)
```

Arguments

x	vector (potentially logical)

Details

It is common that check boxes are blanks by default but that this should be interpreted as TRUE. There are however some uncommon cases were the boxes are marked with "False" for FALSE. We can therefore not be certain of the meaning of a blank value. These will therefore be treated as NA.

Value

`is.incalogical` returns `TRUE` if the vector is logical according to INCA:s internal rules, `FALSE` otherwise. `incalogical2logical` returns a logical vector if `x` can be coerced to such.

Examples

```
is.incalogical(c("", "", "True", ""))  # TRUE
is.incalogical(c("", "False", "", "")) # TRUE
is.incalogical(c("", "FALSE", "", "")) # FALSE
is.incalogical(logical(2)) # will be recognised as well
```

---

lt *Lead time from one date to another*

Description

Lead time from one date to another

Usage

```
lt(from, to, neg = FALSE)
```

Arguments

`from, to`	start and stop dates (in formats that can be recognized as RCC dates).
`neg`	except negative lead times (set to `NA` if `neg = FALSE`)?

Value

Numeric vector

Examples

```
lt("2017-02-10", "2017-02-16") # 6
lt("2017-02-16", "2017-02-10") # negative lead times ignored by default
lt("2017-02-16", "2017-02-10", TRUE) # -6
```

---

next_method *Function to create methods for generics*

Description

Function to create methods for generics

Usage

```
next_method()
```

use_incadata *Use incadata from file or dataframe df*

Description

Read in a file (locally) or use global object named `df` (on INCA) and coerce to `incadata`-object.

Usage

```
use_incadata(file, cache = TRUE, sep = ";", dec = ",", ...)
```

Arguments

`file`	file name as character (ignored if called from INCA)
`cache`	use cache to speed up the loading (see section: "Cache" below)
`sep, dec`	arguments passed to `read.csv2`
`...`	arguments passed to `as.incadata`.

Value

object returned by `as.incadata`

Cache

To process all data through `as.incadata` can be time consuming for large data sets. It is therefore advised to use caching (argument `cache = TRUE`) to avoid unnecessary processing of already formatted data. If `cache = TRUE`, the function will read and process the data only the first time (or if the original data is later changed). A processed and cached version of the data is saved with suffix ".rds". The cached version is always compared to the original file by its MD5 sum and is always updated if needed.

Examples

```
## Not run:
# Create a csv file with example data in a temporary directory
fl <- tempfile("ex_data", fileext = ".csv2")
write.csv2(incadata::ex_data, fl)

# First time the file is read from csv2
use_incadata(fl)
dir(tempdir) # a cache file is saved along the original csv2-file
use_incadata(fl) # Next time file loaded from cache

## End(Not run)
```

Index