Package 'klassR'

November 15, 2024

Type Package

Title Classifications and Codelists for Statistics Norway

Version 1.0.0

Description Functions to search, retrieve, apply and update classifications and codelists using Statistics Norway's API <https://www.ssb.no/klass> from the system 'KLASS'. Retrieves classifications by date with options to choose language, hierarchical level and formatting.

Depends R (>= 3.5.0)

Imports tm, httr, jsonlite, igraph, methods

URL https://statisticsnorway.github.io/ssb-klassr/

BugReports https://github.com/statisticsnorway/ssb-klassr/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Suggests rmarkdown, knitr, testthat (>= 3.0.0), kableExtra, magrittr, dplyr

Config/testthat/edition 3

NeedsCompilation no

Author Susie Jentoft [aut, cre],
Diana-Cristina Iancu [aut],
Lisa Li [aut],
Øyvind I. Berntsen [aut],
Statistics Norway [cph]

Maintainer Susie Jentoft <susie.jentoft@ssb.no>

Repository CRAN

Date/Publication 2024-11-15 10:10:06 UTC

Contents

`count_neighbors` *Count the neighbors of a node.*

Description

Count the neighbors of a node.

Usage

```
count_neighbors(graph, node, mode)
```

Arguments

`graph`	A graph generated by `klass_graph`.
`node`	A node as returned by `klass_node` or `V`.
`mode`	Whether to query outgoing ('out'), incoming ('in') edges, or both types ('all'). This is ignored for undirected graphs.

Value

A numeric vector of length one giving the number of neighbors.

`find_dates` *For a given Klass code, produce a table of dates describing the valid-from and valid-to dates of all versions of the code*

Description

For a given Klass code, produce a table of dates describing the valid-from and valid-to dates of all versions of the code

Usage

```
find_dates(code, api_alle, api_endringer)
```

Arguments

`code`	A Klass code
`api_alle`	A table of all codes in the classification. See example.
`api_endringer`	A table of all changes in the classification See example.

Value

A `data.frame` with number of rows equal to the number of variants of the combination of code and name, determined by the changes the code has been involved in. The `data.frame` has two columns:

- "validFrom"
- "validTo"

`find_name` *Find the name of a code valid at a specific date.*

Description

Find the name of a code valid at a specific date.

Usage

```
find_name(code, validFrom, validTo, api_alle)
```

Arguments

`code`	A Klass code
`validFrom`	The date the code is valid from, in `YYYY-MM-DD` format.
`validTo`	The date the code is valid to, in `YYYY-MM-DD` format.
`api_alle`	A table of all codes in the classification. See example.

Value

The name of the code.

`find_variant_from` *Find the variant of a code corresponding to a change *from* a specific code.*

Description

Find the variant of a code corresponding to a change *from* a specific code.

Usage

```
find_variant_from(x, changeOccurred, variants)
```

Arguments

`x`	The code that is being changed
`changeOccurred`	The date the change occurred
`variants`	The variants lookup-table.

Value

The variant corresponding to the code `x` at date `changeOccurred`.

See Also

[find_variant_to]

`find_variant_to` *Find the variant of a code corresponding to a change *to* a specific code.*

Description

Find the variant of a code corresponding to a change *to* a specific code.

Usage

```
find_variant_to(x, changeOccurred, variants)
```

Arguments

`x`	The code that is being changed
`changeOccurred`	The date the change occurred
`variants`	The variants lookup-table.

Value

The variant corresponding to the code `x` at date `changeOccurred`.

See Also

[find_variant_from()]

`get_klass_changes` *Get all changes that have occurred in a Klass classification*

Description

Get all changes that have occurred in a Klass classification

Usage

```
get_klass_changes(classification)
```

Arguments

`classification`	The ID of the desired classification.

Value

A `data.frame` containing the code changes.

`is_combined` *Given a graph and a node, determine if the node is a result of combinations of multiple codes.*

Description

Given a graph and a node, determine if the node is a result of combinations of multiple codes.

Usage

```
is_combined(graph, node, compare_node = NULL)
```

Arguments

`graph`	A graph generated by `klass_graph`.
`node`	A node as returned by `klass_node` or `V`.
`compare_node`	Optional. A node to compare `node` with when determining whether `node` is combined. See details.

Details

The function will attempt to reconcile nodes that have split and then later merged again when evaluating a node's combinedness.

If `compare_node == NULL`, a node is considered to be combined if more than one node that does not itself have a parent (i.e. codes at the start of a sequence of changes) contribute to `node`.

If `compare_node != NULL`, a node is considered to be combined if any node that is not an ancestor of `compare_node` contributes to `node`, i.e. all paths from `node` to the parents of `node` pass through `compare_node`.

Value

`TRUE` if the node is a combination of two or more nodes, otherwise `FALSE`.

`is_split` *Given a graph and a node, determine if the node is a split code.*

Description

Given a graph and a node, determine if the node is a split code.

Usage

```
is_split(graph, node)
```

Arguments

`graph`	A graph generated by `klass_graph`.
`node`	A node as returned by `klass_node` or `V`.

Details

The function will attempt to reconcile nodes that have split and then later merged again. A node is considered to be split if there is more than one node that does not itself have children (i.e. nodes at the end of a sequence of changes) that can be reached from `node`

Value

`TRUE` if the node is split, otherwise `FALSE`.

klassdata *Testdata for klassR package*

Description

A dataset containing variables for testing of Statistics Norways classification API with the klassR package. Some observations are missing or incorrect for testing and demonstrations.

Usage

```
klassdata
```

Format

A data frame containing 100 rows and 7 variables:

ID Identification number

sex 1/2 variable for sex

education 4-digit number for education standard ISCED97 (level and subject area) NUS (klass = 66) 2015.01.01

kommune 4-digit code for Norwegian municipality (klass = 131). Based on 2015.01.01

kommune2 Numeric variable for Norwegian municipality with dropped leading zero's for testing (klass = 131). Based on 2015.01.01

nace5 5-digit code for industry (NACE). Based on 01.01.2015 standard industry codes (klass = 7)

occupation 4-digit occupation codes using standard for STYRK-08 (klass = 7) 2015.01.01

klass_131_1964_graph *Test Graph data for municipalities in 1964*

Description

A nested list of graph data for using in testing

Usage

```
klass_131_1964_graph
```

Format

An object of class `igraph` of length 1936.

klass_131_2020_graph *Test Graph data for municipalities in 2020*

Description

A nested list of graph data for using in testing

Usage

```
klass_131_2020_graph
```

Format

An object of class `igraph` of length 1936.

klass_131_graph *Test Graph data for municipalities in 2024*

Description

A nested list of graph data for using in testing

Usage

```
klass_131_graph
```

Format

An object of class `igraph` of length 1936.

klass_graph *Build a directed graph of code changes based on a Klass classification*

Description

Build a directed graph of code changes based on a Klass classification

Usage

```
klass_graph(classification, date = NULL)
```

Arguments

classification	The ID of the desired classification.
date	The date which the edges of the graph should be directed towards. Defaults to the current year plus one, which ensures the graph is directed to the most recent codes.

Value

An igraph object with the vertexes representing codes, and edges representing changes between codes. The direction of the edges represent changes towards the date specified in date.

Examples

```
library(klassR)

# Build a graph directed towards the most recent codes
## Not run:
klass_131 <- klass_graph(131)

## End(Not run)

# Build a graph directed towards valid codes in 2020.
## Not run:
klass_131_2020 <- klass_graph(131, "2020-01-01")

## End(Not run)
```

---

klass_node *Given a Klass graph, find the node corresponding to a code and (optionally) a date.*

Description

Given a Klass graph, find the node corresponding to a code and (optionally) a date.

Usage

```
klass_node(graph, x, date = NA)
```

Arguments

graph	A graph generated by klass_graph.
x	The code to search for.
date	Optional. The specific date the supplied code is valid in.

Value

The node in the graph corresponding to the supplied code. If date is not provided, the node with the most recent code is returned. If date is provided, the code with date between `validFrom` and `validTo` is returned.

Examples

```
# Build a graph directed towards the most recent codes.
library(klassR)
## Not run:
klass_131 <- klass_graph(131)

## End(Not run)

# Find the most recent node in the graph representing the code "0101" (Halden,
# valid to 2020.)
## Not run:
halden_node <- klass_node(klass_131, "0101")

## End(Not run)
```

`update_code` *Update a code found in a directed graph based on a Klass-classification*

Description

Update a code found in a directed graph based on a Klass-classification

Usage

```
update_code(
  graph,
  code,
  date = NA,
  output = "code",
  combine = TRUE,
  report = FALSE
)
```

Arguments

`graph`	A graph generated by `klass_graph`.
`code`	A Klass code
`date`	Optional. The specific date the supplied code is valid in.
`output`	Either a character vector, containing one or more of the items in the list below, or `TRUE` to include all columns.

"code" The Klass code.

"name" The Klass name.

"validFrom" The date that the code is valid from.

"validTo" The date that the code is valid to.

"split" Logical: Does the code split into two or more codes?

"combined" Logical: Does two or more codes become this code?

"nextCode" If split == FALSE, gives the code this code changed into. NA otherwise.

combine TRUE or FALSE. See the return section.

report TRUE or FALSE. See the return section.

Value

If report == TRUE and length(output) > 1 | TRUE, the result will be a data.frame with number of rows equal to the number of codes in the sequence of changes between the input code and output code. The columns in the data.frame are specified with output.

If report == TRUE and length(output) == 1, the result will be a character vector with length equal to the number of codes in the sequence of changes between the input code and output code. The contents of the character vector is specified with output.

If report == FALSE and length(output) > 1 | TRUE the result will be a data.frame with one row representing the last code in the change sequence and columns specified by output. If a code has been split, the result will be NA. If combine == FALSE and a code is the result of a combination of codes, the result will be NA.

If report == FALSE and length(output) == 1, the result will be a character vector of length one, containing information about the updated code specified by output. If a code has been split, the result will be NA. If combine == FALSE and a code is the result of a combination of codes, the result will be NA.

See Also

See [update_klass] for updating multiple codes in one function call.

---

update_klass *Update multiple Klass codes to a desired date.*

Description

Update multiple Klass codes to a desired date.

Usage

```
update_klass(
  codes,
  dates = NA,
  classification = NULL,
  date = NULL,
  graph = klass_graph(classification, date),
  output = "code",
  report = FALSE,
  combine = TRUE
)
```

Arguments

codes	Codes to be updated.
dates	Optional. Can be used to specify what date each of the codes was valid in. Supply a character vector of either length 1 to specify the same valid date for all codes, or of the same length as codes to specify valid dates for each code. The character vector(s) should have a format coercible by as.Date, e.g. YYYY-MM-DD. The function will return an error if a code was not valid at the specified date.
classification	The ID of the desired classification.
date	Optional. Can be used to specify the date the codes should be updated to, e.g. if you have codes that are valid in year T, but want to change the codes to the corresponding version in year T-1. If unspecified (the default), the function will update codes to the most recent version.
graph	Optional. A graph object generated by klass_graph. If you're making multiple calls to [update_klass], you can save some time by generating the graph beforehand and reusing it for each call to [update_klass] with this parameter. If providing the graph directly, you do not need to provide the classification and date parameters.
output	Either a character vector, containing one or more of the items in the list below, or TRUE to include all columns. "code" The Klass code. "name" The Klass name. "validFrom" The date that the code is valid from. "validTo" The date that the code is valid to. "split" Logical: Does the code split into two or more codes? "combined" Logical: Does two or more codes become this code? "nextCode" If split == FALSE, gives the code this code changed into. NA otherwise.
report	TRUE or FALSE. See the return section.
combine	TRUE or FALSE. See the return section.

Value

If `output = "code"`, a vector of length `length(codes)` containing either a code if the update is successful or `NA` if the code has been split. If `combine = FALSE`, a code being combined with another code will also return `NA`.

If `output == TRUE`, a list of length `length(codes)` containing `data.frames` detailing the codes visited through the node search. The tables have the following columns.

—

If `report == TRUE` and `length(output) > 1 | TRUE`, the result will be a list of `data.frames` with number of rows equal to the number of codes in the sequence of changes between the input codes and output codes. The columns in the `data.frames` are specified with `output`.

If `report == TRUE` and `length(output) == 1`, the result will be a list of character vectors with length equal to the number of codes in the sequence of changes between the input code and output code. The contents of the character vectors is specified with `output`.

If `report == FALSE` and `length(output) > 1 | TRUE` the result will be a list of `data.frames` with one row representing the last code in the change sequence and columns specified by `output`. If a code has been split, the result will be `NA`. If `combine == FALSE` and a code is the result of a combination of codes, the result will be `NA`.

If `report == FALSE` and `length(output) == 1`, the result will be a character vector containing information about the updated codes specified by `output`. If a code has been split, the result will be `NA`. If `combine == FALSE` and a code is the result of a combination of codes, the result will be `NA`.

Examples

```
library(klassR)
codes <- get_klass(131, date = "2020-01-01")[["code"]]

## Not run:
updated_codes <- update_klass(codes,
  dates = "2020-01-01",
  classification = 131
)

## End(Not run)
```

---

update_klass_node *Given a node and a graph, find the node at the end of a sequence of changes.*

Description

Given a node and a graph, find the node at the end of a sequence of changes.

Usage

```
update_klass_node(graph, node)
```

Arguments

graph	A graph generated by klass_graph.
node	A node as returned by klass_node or V.

Value

A sequence of vertices, starting with node and ending with the last visited node.

Examples

```
# Build a graph directed towards the most recent codes.
library(klassR)
klass_131 <- klass_graph(131)

# Find the most recent node in the graph representing the code "0101" (Halden,
# valid to 2020.)
halden_node <- klass_node(klass_131, "0101")

# Find the most recent code corresponding to 0101 Halden
halden_node_updated <- update_klass_node(klass_131, halden_node)
```

Index