Package 'klassR'

August 21, 2026

Type Package

Title Classifications for Statistics Norway

Version 1.0.7

Description Functions to search, retrieve, apply and update classification standards and code lists 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 (>= 2.1.1), 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

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

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

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

Maintainer Øyvind I. Berntsen <Oyvind.Berntsen@ssb.no>

Repository CRAN

Date/Publication 2026-08-21 12:00:02 UTC

Contents

`api_endringer_1963` *Test data for changes in municipality codes from 1963 onwards*

Description

A data.frame containing information about changes in municipality codes

Usage

```
api_endringer_1963
```

Format

An object of class `data.frame` with 1172 rows and 7 columns.

api_endringer_2019 *Test data for changes in municipality codes from 2019 onwards*

Description

A data.frame containing information about changes in municipality codes

Usage

```
api_endringer_2019
```

Format

An object of class `data.frame` with 444 rows and 7 columns.

apply_klass *Match and convert a classification*

Description

Match and convert a classification

Usage

```
apply_klass(
  x,
  classification,
  date = NULL,
  variant = NULL,
  correspond = NULL,
  language = "nb",
  output_level = NULL,
  output = "name",
  format = TRUE
)

ApplyKlass(
  x,
  klass,
  date = NULL,
  variant = NULL,
  correspond = NULL,
  language = "nb",
  output_level = NULL,
  output = "name",
  format = TRUE
)
```

Arguments

x	Input vector of classification codes. Vector must match "code" column from a call to get_klass().
classification	Classification number
date	String for the required date of the classification. Format must be "yyyy-mm-dd". For an inverval, provide two dates as a vector. If blank, will default to today's date.
variant	The classification variant to fetch (if a variant is wanted).
correspond	ID number for target in correspondence table. For correspondence between two dates within the same classification, use correspond = TRUE.
language	Default "nb" for Norwegian (Bokmål). Also "nn" (Nynorsk) and "en" (English available for some classifications)
output_level	Desired output level
output	String describing output. May be "name" (default), "code" or "both".
format	Logical for whther to run formatting av input vector x (Default = TRUE), important to check if formatting is in one level.
klass	Deprecated; use classification instead.

Value

A vector or data frame is returned with names and/or code of the desired output level.

Examples

```
data(klassdata)
kommune_names <- apply_klass(
  x = klassdata$kommune,
  classification = 131,
  language = "en",
  format = FALSE
)
```

---

correspond_list *Correspondence list Print a list of correspondence tables for a given classification with source and target IDs*

Description

Correspondence list Print a list of correspondence tables for a given classification with source and target IDs

Usage

```
correspond_list(classification, date = NULL)

CorrespondList(klass, date = NULL)
```

Arguments

classification	Classification number
date	Date for classification (format = "YYYY-mm-dd"). Default is current date
klass	Deprecated; use classification instead.

Value

Data frame with list of corrsepondence tables, source ID and target ID.

Examples

```
correspond_list("7")
```

---

find_equivalent_codes *Find equivalent sets of codes in a Klass classification*

Description

Find which codes should be combined to reconstruct older or newer codes across multiple dates. Creates groups dynamically to fit the provided date range, with flexible labelling functionality.

Usage

```
find_equivalent_codes(
  classification,
  dates,
  labels = TRUE,
  graph = klass_graph(classification),
  date_format = "%Y"
)

find_equivalents(
  classification,
  dates,
  labels = TRUE,
  graph = klass_graph(classification),
  date_format = "%Y"
)
```

Arguments

classification	The Klass classification to be used
dates	The dates that equivalent sets of codes should be found for.

`labels` — `TRUE`, `FALSE` or a named list of functions.

This parameter controls whether or not to add group labels to groups of equivalent sets. If `TRUE`, labels are constructed using the codes valid at the latest provided date, and comma separated like in the example below. By default, labels will be placed in a column named `label`.

```
"1508 Ålesund, 1580 Haram"
```

If `FALSE`, no labels will be applied.

This parameter also accepts a named list of labelling functions. The resulting dataset will contain a label column for each of the supplied functions. The names of the label columns are specified using the names of the list of functions.

The functions provided in this parameter can accept any of the following parameters: `date`, `code`, `name`, `validFrom` and `validTo`, representing the corresponding values of each code in a group. The functions must also provide a `...` parameter, unless using all of the above. The functions can expect that the input variables have the same length of 1 or longer. The functions should return a character vector of length one or the same length as the input variables.

The following list, when supplied to this parameter, creates two label columns: one containing the codes and names (`label1`), and another with only the codes `label2`. In this example, `label1` creates the same labels as the default labelling used when `labels == TRUE`.

```
list(
  label1 = function(code, name, date, ...) {
    label_codes <- date == max(date)
    paste(code[label_codes], name[label_codes], collapse = ", ")
  },
  label2 = function(code, date, ...) {
    label_codes <- date == max(date)
    paste(code[label_codes], collapse = ", ")
  }
)
```

`graph` — Optional. Generating the graph using `klass_graph` manually beforehand and providing it in this parameter can save time if running `find_equivalent_codes` multiple times in sequence.

`date_format` — Optional. Passed directly to format, this is used to specify the output format for the `date` column. The default keeps just the year (`"YYYY"`). To get the full date in `"YYYY-MM-DD"` format, use `"%Y-%m-%d"`. See strptime for complete functionality.

Details

This function provides a solution to the problem of split or combined codes in Klass classifications. When using `update_klass` to ask "what is this code in this version of the classification in this other version of the classification?", the answer is sometimes that the code has been split into two or more codes (or combined from two or more codes, if trying to back-date a code), and therefore that the code cannot be updated.

The solution provided by `find_equivalent_codes` is answering the question: "in these versions of the classification, which codes were equivalent to this code in this other version of the classification?".

Consider the following example of two codes combining into one. Here, `"a"` and `"b"` are valid at t1, and are combined into `"c"` at t2.

```
t1      t2
a ---|
b ---+-> c
```

`update_klass` would inform us that `"a"` can be updated to `"c"` at t2, unless we specified `combine = FALSE`, in which case the result would be `NA`. `find_equivalent_codes()` would inform us that the equivalent of the codes `"a"` and `"b"` in t1 at t2 is `"c"`.

We can also consider a code splitting into two. In this example, `"a"` is valid at t1, and splits into `"b"` and `"c"` at t2.

```
t1     t2
a
|-----> b
|-----> c
```

`update_klass` is unable to provide an updated code due to the split, and would return `NA`. `find_equivalent_codes` would inform us that the equivalent codes of `"a"` at t1 is `"b"` and `"c"` at t2.

`find_equivalent_codes` can handle more than two dates. In the following example, `"a"` splits into `"b"` and `"c"` at t2, and `"b"` and `"c"` combine into `"d"` at t3. `find_equivalent_codes` can inform us that `"a"` is equivalent to `"b"` and `"c"` at t2, and `"d"` at t3.

```
t1     t2     t3
a
|-----> b |
|-----> c +-> d
```

`find_equivalent_codes` will only search in the time range we specify. As a consequence, generating sets of equivalent codes over longer time spans will generally create larger sets than using shorter time spans.

To illustrate this behavior, we can add a new code `"e"` to the previous example, and have `"d"` and `"e"` combine into `"f"` at t4.

```
t1     t2     t3     t4
a
|-----> b |
|-----> c |-> d |
e---------------+--> f
```

Finding the equivalents of `"a"` in t1 at t2 and t3 returns the same sets as before:

- t1: "a"
- t2: "b" and "c"
- t3: "d"

However, if we also wanted to know the equivalent set for t4, the result would be:

- t1: "a" and "e"
- t2: "b", "c" and "e"
- t3: "d" and "e"
- t4: "f"

find_equivalents is a legacy alias for find_equivalent_codes.

Value

A data.frame with columns:

- date containing the input dates
- code containing the set of equivalent codes in each date
- name containing the names of each code
- validFrom and validTo values for each code returned
- By default, labels, giving a unique group label for each group of equivalent sets.

---

find_equivalent_nodes *Find the equivalent sets of a node at various dates*

Description

Find the equivalent sets of a node at various dates

Usage

```
find_equivalent_nodes(node, dates, graph)
```

Arguments

node	The node that we're finding the equivalent sets of
dates	The dates that we want to find the equivalent sets in
graph	The graph that the nodes come from

Value

A named list of length(dates) containing the equivalent nodes for each date. The names of the returned list are the dates provided in date, coerced to character.

get_family *Identify corresponding family from a classification number*

Description

Identify corresponding family from a classification number

Usage

```
get_family(classification)

GetFamily(klass)
```

Arguments

`classification`	Classification number
`klass`	Deprecated; use `classification` instead.

Value

Family number

Examples

```
get_family(classification = 7)
```

get_klass *Fetch Statistics Norway classification data using API*

Description

Fetch Statistics Norway classification data using API

Usage

```
get_klass(
  classification,
  date = NULL,
  correspond = NULL,
  correspondID = NULL,
  variant = NULL,
  output_level = NULL,
  language = "nb",
  output_style = "normal",
  notes = FALSE,
  quiet = TRUE
```

```
)

GetKlass(
  klass,
  date = NULL,
  correspond = NULL,
  correspondID = NULL,
  variant = NULL,
  output_level = NULL,
  language = "nb",
  output_style = "normal",
  notes = FALSE,
  quiet = TRUE
)
```

Arguments

`classification`	Number/string of the classification ID/number. (use klass_list() to find this)
`date`	String for the required date of the classification. Format must be "yyyy-mm-dd". For an inverval, provide two dates as a vector. If blank, will default to today's date.
`correspond`	Number/string of the target classification for correspondence table (if a correspondence table is requested).
`correspondID`	ID number of the correspondence table to retrieve. Use as an alternative to correspond.
`variant`	The classification variant to fetch (if a variant is wanted).
`output_level`	Number/string specifying the requested hierarchy level (optional).
`language`	Two letter string for the requested language output. Default is Bokmål ("nb"). Nynorsk ("nn") and English ("en") also available for some classification.)
`output_style`	String variable for the output type. Default is "normal". Specify "wide" for a wide formatted table output.
`notes`	Logical for if notes should be returned as a column. Default FALSE
`quiet`	Logical for whether to suppress the printing of the API address. Default TRUE.
`klass`	Deprecated; use `classification` instead.

Value

The function returns a data frame of the specified classification/correspondence table. Output variables include: code, parentCode, level, and name for standard lists. For correspondence tables variables include: sourceCode, sourceName, targetCode and targetName. For date correspondence tables variables include: oldCode, oldName, newCode and newName. For "wide" output, code and name with level suffixes is specified. For date ranges, validFromInRequestedRange and validToInRequestedRange give the dates for the classification. Variable ChangeOccured gives the effective date for classification change in classification change tables.

Examples

```
# Get classification for occupation classifications
head(get_klass(classification = "7"))
# Get classification for occupation classifications in English
head(get_klass(classification = "7", language = "en"))
```

`get_name` *Get the name of a classification version*

Description

Get the name of a classification version

Usage

```
get_name(version)

GetName(version)
```

Arguments

`version`	Version number

Value

string or vector of strings with name of version

Examples

```
get_name("33")
```

`get_version` *Get version number of a class given a date*

Description

Get version number of a class given a date

Usage

```
get_version(classification = NULL, date = NULL, family = NULL, klassNr = FALSE)

GetVersion(klass = NULL, date = NULL, family = NULL, klassNr = FALSE)
```

Arguments

`classification`	Classification number
`date`	Date for version to be valid
`family`	Family ID number if a list of version number for all classes is desired
`klassNr`	True/False for whether to output classification numbers. Default = FALSE
`klass`	Deprecated; use `classification` instead.

Value

Number, vector or data frame with version numbers and calssification numbers if specified.

Examples

```
get_version(7)
```

`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 2000.

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 2000.

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 2000.

`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

```
## Not run:
# 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")

## End(Not run)
```

`list_family` *Classification family list Print a list of all families and the number of classifications in each*

Description

Classification family list Print a list of all families and the number of classifications in each

Usage

```
list_family(family = NULL, codelists = FALSE, language = "nb")

ListFamily(family = NULL, codelists = FALSE, language = "nb")
```

Arguments

family	Input family ID number to get a list of classifications in that family
codelists	True/False for whether to include codelists. Default = FALSE
language	Two letter string for the requested language output. Default is Bokmål ("nb"). Nynorsk ("nn") and English ("en").

Value

dataset containing a list of families

Examples

```
list_family(family = 1)
```

---

list_klass *Classification list Get a full list of all classifications and codelists*

Description

Classification list Get a full list of all classifications and codelists

Usage

```
list_klass(codelists = FALSE, language = "nb")

ListKlass(codelists = FALSE, language = "nb")
```

Arguments

codelists	True/False for whether to include codelists. Default = FALSE
language	Two letter string for the requested language output. Default is Bokmål ("nb"). Nynorsk ("nn") and English ("en").

Value

A data frame containing a full list of classifications. The data frame includes the classification name, number, family and type.

Examples

```
head(list_klass(codelists = TRUE))
```

search_klass *Search Klass*

Description

Search Klass

Usage

```
search_klass(query, codelists = FALSE, size = 20)

SearchKlass(query, codelists = FALSE, size = 20)
```

Arguments

query	String with key word to search for
codelists	True/False for whether to include codelists. Default = FALSE
size	The number of results to show. Default = 20.

Value

Data frame of possible classifications that match the query

Examples

```
search_klass("occupation")
```

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

```
## Not run:
# 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)

## End(Not run)
```

Index