## README.md (https://github.com/guiboucher/health/tree/v0.1.0.9000)

# health

This package offers functionality for a variety of health-promoting tasks and tools for visualizing results.

## Installation

You can install the published version of health from [Github](https://github.com/guiboucher/health) with:

remotes::install\_github("guiboucher/health", build\_vignettes = TRUE)

## Polypharmacy

These functions analyse prescription drugs deliveries to calculate several indicators of polypharmacy corresponding to the various definitions found in the literature.

It is essential to know the concepts used to calculate the various polypharmacy indicators to adequately use this package.

The core of the package is the data\_process function that creates the data.table of pharmacists drug deliveries by restructuring the drug delivery records (usually extracted from a pharmacy or a health insurance information system) into continuous periods of drug availability, applying user-defined arguments such as the grace periods between renewals or the longest treatment duration that an individual may accumulate through the successive renewals.

Then, each polypharmacy indicator can be computed using the corresponding function (ind\_simult, ind\_stdcumul, ind\_wcumul, ind\_stdcontinuous, ind\_ucontinuous) or using the overall function indicators and select all the desired indicator(s) to be calculated at once.

Prior to running data\_process the user may need to pre-process the table of original drug delivery records to break down combination drug into their individual components (drugs\_bkdn) and/or to overwrite the treatment duration of specified drugs with constant time periods (cst\_trt\_dur).

Data tables

Cohort table providing the unique identifiers of the study cohort and, optionally, columns relating to the individual’s characteristics such as demographics (e.g. age and sex). All optional descriptive columns are carried into the result table of individuals’ polypharmacy indicators.

|  |  |  |
| --- | --- | --- |
| **Column name** | **Value** | **Class** |
| id | Individual unique identifier. The individual may or may not have been provided drugs. | any |
| age | Optional descriptive column | any |
| sex | Optional descriptive column | any |
| ---- | Optional descriptive column | any |

Rx\_deliv table listing all prescription drug deliveries to be analyzed

|  |  |  |
| --- | --- | --- |
| **Column name** | **Value** | **Class** |
| id | Individual unique identifier (i.e. drug recipient) | same as Cohort$id |
| drug\_code | Drug unique identifier | any |
| drug\_deliv | Date the drug was delivered | as.date or  as.caracter (yyyy-mm-dd) or  integer (0 = January 1st, 1970) |
| drug\_duration | Duration of treatment delivered in days | integer |

Hosp\_stays table listing all hospital stays

|  |  |  |
| --- | --- | --- |
| **Column name** | **Value** | **Class** |
| id | Individual unique identifier | Same as Cohort$id |
| hosp\_admis | Date of admission | as.date or  as.caracter (yyyy-mm-dd) or  integer (0 = January 1st, 1970) |
| hosp\_discharge | Date of discharge | as.date or  as.caracter (yyyy-mm-dd) or  integer (0 = January 1st, 1970) |

Optional data tables

Combin\_drugs\_bkdn table of the single active molecules that make up combination drugs

|  |  |  |
| --- | --- | --- |
| **Column name** | **Value** | **Class** |
| drug\_code | Combination drug unique identifier | any |
| ind\_drug\_code | Single active molecule unique identifier | any |

Cst\_trt\_dur table used to overwrite the treatment duration of the listed drugs with a constant value

|  |  |  |
| --- | --- | --- |
| **Column name** | **Value** | **Class** |
| drug\_code | Drug unique identifier | any |
| cst\_duration | Constant value to replace the treatment duration that is originally provided in Rx\_deliveries$duration | integer |

**Package ‘health’**

September 22, 2020

**Title** Convenience Functions for Health Data

**Version** 0.1.0.9000

**Description** This package provides functionality for a variety of health- promoting tasks and tools for visualizing results.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.1

**Imports** data.table (>= 1.13.0), lubridate (>= 1.7.9)

**Suggests** microbenchmark, testthat

**Depends** R (>= 2.10)

R **topics documented:**

combine\_time\_periods [3](#_bookmark3)

cst\_tx\_dur [4](#_bookmark4)

data\_process [5](#_bookmark5)

drug\_bkdn [7](#_bookmark6)

Index [8](#_bookmark7)

age *Calculate ages*

## Description

Provides the age on on\_date for each date of birth (dob).

## Usage

age(dob, on\_date = lubridate::today())

1

2 *age\_grps*

## Arguments

dob Vector of date(s) of birth.

on\_date Vector of date(s) on which the age is calculated

## Details

dob and on\_date can be 1) as.Date("yyyy-mm-dd"), 2) as.character("yyyy-mm-dd") or 3)

as.integer() where 0 is January 1*st*, 1970.

on\_date can be a vector of length 1 or same length as dob. If only one value is provide for on\_date, all ages are calculated on the same date.

## Value

Vector of same length as dob and on\_date.

## Examples

age(dob = c("1990-06-01", "1965-11-14"),

oon\_date = c("2010-06-01", "1985-11-14"))

age(dob = as.Date(c("1990-06-01", "1965-11-14")),

on\_date = as.Date("2010-01-01"))

age(dob = as.integer(as.Date(c("1990-06-01", "1965-11-14"))),

on\_date = as.integer(as.Date(c("2010-06-01", "1985-11-14")))) ## Not run:

# Error

age(dob = c("1990-06-01", "1965-11-14"),

on\_date = c("2010-06-01", "1985-11-14", "2006-01-17"))

## End(Not run)

# Different class of variables

vec\_chr <- c("1990-06-01", "2020-01-01")

str(vec\_chr)

age(dob = vec\_chr[1], on\_date = vec\_chr[2]) vec\_date <- as.Date(vec\_chr)

str(vec\_date)

age(dob = vec\_date[1], on\_date = vec\_date[2]) vec\_int <- as.integer(vec\_date)

str(vec\_int)

age(dob = vec\_int[1], on\_date = vec\_int[2])

age\_grps *Generate age-groups*

## Description

Determines the age-group according to user-defined categories.

## Usage

age\_grps(age, grp\_breaks, max\_age = TRUE, as\_factor = TRUE)

*combine\_time\_periods* 3

## Arguments

age Vector age (integer or numeric).

grp\_breaks Vector of the lower bounds of the increasing age-group categories (integer or numeric). A category includes all values equal or greater to the lower bound and lower than the next category lower bound (except for the last value when max\_age = FALSE where the last values is the upper bound (not included) of the last age-group.

max\_age logical, TRUE (default): last grp\_breaks value is the last age-group category **lower bound**. FALSE: last grp\_breaks value is the last age\_group category **upper bound** (not included in the category). If grp\_breaks = x, y, z and max\_age=TRUE then the last age-group is defined by [z, max(age)+1[. Conversely, if max\_age=FALSE, the last age-group is defined by [y, z[

as\_factor logical, TRUE (default): the values are encoded as a factor. FALSE: values are converted into character and loose their built-in ordering property.

## Details

grp\_breaks can neither contain the same value more than once nor NAs.

## Value

Vector of length(age).

## Examples

age\_grps(age = 0:19, grp\_breaks = c(0, 5, 10, 15), max\_age = TRUE, as\_factor = TRUE)

age\_grps(age = 0:19, grp\_breaks = c(0, 5, 10, 15), max\_age = FALSE, as\_factor = FALSE)

aggreg\_time\_periods *Time periods aggregation*

## Description

Aggregates time periods that are overlapping and that are no more than ndays apart.

## Usage

aggreg\_time\_periods( dt,

start, end,

by = NULL,

nday = 2, final\_as\_date = TRUE, verif\_cols = TRUE

)

4 *polyRx\_cst\_tx\_dur*

## Arguments

dt Input data name.

start, end Columns of dt indicating the start and end date of individual time periods ('yyyy-mm-dd').

by Columns of dt by which the time periods are combined. NULL by default.

nday integer that specify the maximum number of days allowed between the end of a time period (end[i]) and the beginning of the next one (start[i+1]) to aggregate the two time periods from start[i] to end[i+1]. Usually 1 or 2. Time periods are merged if start[i+1]

* end[i] < nday.

final\_as\_date logical, indicates if start and end columns should be returned as\_date (TRUE). Else, dates are returned as character (FALSE). TRUE by default.

verif\_cols logical,indicates if the program should verify start and end column format (safe).

TRUE by default.

## Value

data.table of the aggregated time periods at least nday apart for each distinct value of the by columns. The data.table includes the *by*, *start* and *end* columns.

## Examples

DT <- data.frame(

id = 1, age = 55, sex = "f",

start = c("2020-01-01", "2020-01-10", "2020-01-20", "2020-01-26"),

end\_date = c("2020-01-10", "2020-01-19", "2020-01-24", "2020-01-31")

)

dt1 <- unite\_periods(

dt = DT, start = "start", end = "end\_date", by = NULL, nday = 2, verif\_cols = TRUE

)

dt2 <- unite\_periods(

dt = DT, start = "start", end = "end\_date", by = c("id", "sex"), nday = 2, verif\_cols = TRUE

)

polyRx\_cst\_tx\_dur *Constant treatment duration drugs*

## Description

Overwrites the treatment durations with user-specified constant durations for each drug\_code in- cluded in table cst\_tx\_dur.

## Usage

polyRx\_cst\_tx\_dur(Rx\_deliveries, cst\_tx\_dur)

*polyRx\_data\_process* 5

## Arguments

Rx\_deliveries Table listing all prescription drug deliveries to be analyzed. Data with **at least** 2 columns:

* + drug\_code : Drug unique identifier.
  + duration : Duration of treatment delivered in days.

cst\_tx\_dur Table used to verwrite the treatment duration of the listed drugs with a constant value. Data with 2 columns:

* + drug\_code : Drug unique identifier.
  + cst\_duration: Constant treatment duration.

## Value

data.table containing constant duration treatment. duration as integer.

## Examples

dt1 <- data.frame(id = c(1, 2, 3),

drug\_code = c("A", "B", "C"), duration = 15)

dt2 <- polyRx\_cst\_tx\_dur( Rx\_deliveries = dt1,

cst\_tx\_dur = data.frame(drug\_code = c("A", "C", "E"),

cst\_duration = c(40, 60, 20))

)

data\_process *Creates the data.table required for the calculation of polypharmacy indicators*

## Description

Reads a table of successive drug delivery records (usually extracted from a pharmacy or a health insurance information system) and creates the dataset required for the calculation of the polypharmacy indicators by applying various user-defined arguments, incorporating hospital stays into the treatment periods and merging quasi continuous and overlapping treatment periods of identical drugs.

## Usage

polyRx\_data\_process( Rx\_deliv,  
Cohort = NULL,  
Hosp\_stays = NULL,   
id,  
drug\_code  
drug\_deliv,  
drug\_duration,  
hosp\_admis,  
hosp\_discharge,  
study\_start = NULL, study\_end = NULL,  
max\_hosp\_gap = 2, grace\_factor = 0.5,

grace\_fix = 0, max\_reserve = NULL, final\_as\_date = TRUE, verif\_cols = TRUE

)

## Arguments

Rx\_deliv Table listing all prescription drug delivered including a run-in period of 7 months prior to study\_start.

Cohort Table providing the unique identifiers of the study cohort. Only the ids listed in both the Cohort and the Rx\_deliv tables will be returned. if Cohort=NULL, all ids of the Rx-deliv table will be returned.

Hosp\_stays Table listing all hospital stays.

id Column name of Rx\_deliv, Cohort, and Hosp\_stays containing individuals’ unique identifiers (any format)

drug\_code Column name of Rx\_deliv that contains the drugs’ unique identifiers (any format)

drug\_deliv Column name of Rx\_deliv that contains the dates of the drug deliveries

drug\_duration Column name of Rx\_deliv that contains the delivered treatment duration

hosp\_admis Column name of Hosp\_stays that contains the date of admission in hospital

hosp\_discharge Column name of Hosp\_stays that contains the date of discharge from hospital

study\_start, study\_end

"yyyy-mm-dd". Defines the first and last day of the study period for which the polypharmacy indicator(s) need be calculated All treatment periods prior to study start and past study\_end are not transcribed into the result table. Could be

number values where 1970-01-01 = 0. Exemple: as.integer(as.Date("2020-01-01"))

= 18262.

max\_hosp\_gap integer. Maximum number of days allowed between the end of a hospital stay (hosp\_discharge[i]) and the beginning of the next one (hosp\_admis[i+1]) to aggregate the two as one single hospital stay, from hosp\_admis [i] to hosp\_discharge [i+1]. Usually equal to 1 or 2.

grace\_fctr, grace\_cst

numbers *≥* 0. Two types of grace periods can be applied. One is proportional to the treatment duration of the previous delivery (grace\_fctr) and the other is a constant number of days. (grace\_cst). See Details.

max\_reserve Integer 0 or NULL. Longest treatment duration, in days, that can be stored from successive overlapping deliveries. When max\_reserve=NULL no limit is applied. When max\_reserve=0 no accumulation of extra treatment duration is accounted for.

*≥*

final\_as\_date Return start and end columns as\_date (TRUE). Else, dates are returned as character (FALSE). TRUE by default.

verif\_cols Verify start and end column format (safe).

TRUE by default.

## Details

**Variables**:

* id: Rx\_deliveries$id, Cohort$id and Hosp\_stays$id must be of the same *class* (integer,

numeric, character...).

* drug\_duration: integer.
* drug\_deliv, hosp\_admis, hosp\_discharge: can be 1) as.Date("yyyy-mm-dd"), 2) as.character("yyyy-mm-dd") or 3) as.integer() where 0 is January 1*st*, 1970.

## Hospital stays

Drug availability is considered to continue during the hospital stay as it is on the day prior admission. The patient is assumed to resume the consumption of the drugs delivered by community pharmacists (as recorded in Rx\_deliv) prior admission the day after hosp\_discharge.

## Run-in period

A run-in period is necessary to account for the medications that are available to the individuals on the day of study\_start.

## Grace period:

The grace period is used to determine if two successive deliveries can be considered as a continuous treatment even if there is a gap of a few days for which no treatment is apparently available. Two successive deliveries of an identical drug are considered part of a single continuous treatment if the next delivery doesn’t occur more than grace\_cst + (grace\_fctr ×drug\_duration) days after the end of the previous treatment duration. The availability of extra drugs accumulated over the successive deliveries is accounted for prior to evaluating the duration of the gap between deliveries.

## Performance:

For better performance, Rx\_deliveries$drug\_deliv is converted to *integer*.

*polyRx\_drug\_bkdn* 7

## Value

data.table with four (4) variables:

* id: Individual unique identifier (i.e. drug recipient).
* drug\_code: Drug unique identifier.
* drug\_start: First date of continued drug use.
* drug\_end: Last date of continued drug use.

polyRx\_drug\_bkdn *Translates compound drugs into single active molecules*

## Description

Reads the Rx\_deliveries table and breaks down the delivery of each combination drug into several single active molecule deliveries according to the correspondence table combi\_drug\_brkdwn.

## Usage

polyRx\_drug\_bkdn(Rx\_deliveries, combi\_drug\_bkdn)

## Arguments

Rx\_deliveries Table listing all prescription drug deliveries to be analyzed. Data with **at least** one (1) column:

* + drug\_code: Drug unique identifier.

combi\_drug\_bkdn

Table of the single active molecules that make up combination drugs. Data with two (2) columns:

* + drug\_code: Drug unique identifier.
  + ind\_drug\_code: Single active molecule unique identifier.

Details

Rx\_deliveries$drug\_code, combi\_drug\_bkdn$drug\_code and combi\_drug\_bkdn$ind\_drug\_code must have the same *class* (integer, numeric, character...).

## Value

data.table containing single active molecule deliveries.

## Examples

polyRx\_drugs\_bkdn(

Rx\_deliveries = data.frame(drug\_code = c("A", "B", "C")), combi\_drugs\_bkdn = data.frame(drug\_code = c("A", "A", "C", "C", "C"),

drug\_code\_bkdn = c("Ai", "Aj", "Cx", "Cy", "Cz"))

)

polyRx\_drugs\_bkdn(

Rx\_deliveries = data.frame(id = 1:3,

drug\_code = c("A", "B", "C")), combi\_drugs\_bkdn = data.frame(drug\_code = c("A", "A", "C", "C", "C"),

drug\_code\_bkdn = c("Ai", "Aj", "Cx", "Cy", "Cz"))

)

# Index

age, [1](#_bookmark0)

age\_grps, [2](#_bookmark2)

combine\_time\_periods, [3](#_bookmark3)

polyRx\_cst\_tx\_dur, [4](#_bookmark4)

polyRx\_data\_process, [5](#_bookmark5)

polyRx\_drug\_bkdn, [7](#_bookmark6)

polyRx\_struct, [*5*](#_bookmark5)

8