

Package ‘drugprepr’

July 23, 2026

Title Prepare Electronic Prescription Record Data to Estimate Drug Exposure

Version 0.0.7

Date 2026-07-22

Maintainer David Selby <David.Selby@dfki.de>

URL <https://github.com/belayb/drugprepr>

BugReports <https://github.com/belayb/drugprepr/issues>

Description Prepare prescription data (such as from the Clinical Practice Research Datalink) into an analysis-ready format, with start and stop dates for each patient's prescriptions. Based on Pye et al (2018) <[doi:10.1002/pds.4440](https://doi.org/10.1002/pds.4440)>.

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

LazyData true

Imports dplyr, rlang, tidyr, sqldf, stringr, purrr, DescTools, doseminer

Depends R (>= 2.10)

Suggests knitr, rmarkdown, testthat, kableExtra

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Belay Birlie Yimer [aut] (ORCID:
<<https://orcid.org/0000-0001-8621-6539>>),
David Selby [aut, cre, cph] (ORCID:
<<https://orcid.org/0000-0001-8026-5663>>),
Meghna Jani [aut],
Goran Nenadic [aut],
Mark Lunt [aut],
William G. Dixon [aut]

Repository CRAN

Date/Publication 2026-07-23 10:50:28 UTC

Contents

clean_duration	2
close_small_gaps	3
compute_ndd	4
cprd	5
decision_1	6
decision_10	7
decision_2	7
decision_3	8
decision_4	9
decision_5	10
decision_6	11
decision_7	12
decision_8	12
decision_9	13
drug_prep	14
example_therapy	15
get_mode	15
impute	16
impute_duration	17
impute_ndd	18
impute_qty	18
isolate_overlaps	19
make_decisions	20
min_max_dat	22
outside_range	22
shift_interval	23
Index	24

clean_duration	<i>Clean implausibly-long prescription durations</i>
----------------	--

Description

Given a prescription length limit, truncate any prescriptions that appear to be longer than this, or mark them as missing.

Usage

```
clean_duration(data, max_months = Inf, method = c("truncate", "remove"))
```

Arguments

data	A data frame containing a column called duration
max_months	The maximum plausible prescription length in months
method	Either 'truncate' or 'remove'. See details

Details

The method 'truncate' causes any duration longer than max_months to be replaced with the value of max_months (albeit converted to days). The method 'remove' causes such durations to be replaced with NA. There is no explicit 'ignore' method, but if you want to 'do nothing', simply set max_months to an arbitrarily high number. By default, the maximum is infinite, so nothing should happen. (Of course, you could also just *not* run the function...)

Value

A data frame of the same structure as the input, possibly with some elements of the duration column changed

Note

Currently the variable name is hard-coded as 'duration', but in principle this could be parametrised for datasets where the column has a different name.

Examples

```
long_presc <- data.frame(duration = c(100, 300, 400, 800))
clean_duration(long_presc, 6)
clean_duration(long_presc, 12, 'remove')
```

close_small_gaps	<i>Close small gaps between successive prescriptions</i>
------------------	--

Description

Given a series of prescriptions in data, if one prescription (for the same patient and drug) starts $\leq$ min_gap days after the previous one finishes, we extend the length of the previous prescription to cover the gap.

Usage

```
close_small_gaps(data, min_gap = 0L)
```

Arguments

data	A data frame containing columns procode, patid, start_date and stop_date
min_gap	Size of largest gaps to close. Default is zero, i.e. do nothing

Value

The input data frame data, possibly with some of the stop_dates changed.

Examples

```
gappy_data <- data.frame(
  patid = 1,
  prodcode = 'a',
  start_date = Sys.Date() + (0:6) * 7,
  stop_date = Sys.Date() + (0:6) * 7 + 4
)
close_small_gaps(gappy_data)
close_small_gaps(gappy_data, 7)
```

compute_ndd	<i>Compute numerical daily dose from free-text prescribing instructions</i>
-------------	---

Description

The function calls the R package **doseminer** to extract dose information from free-text prescribing instructions, then computes the average numerical daily dose according to a given decision rule.

Usage

```
compute_ndd(data, dose_fn = mean, freq_fn = mean, interval_fn = mean)
```

Arguments

data	a data frame containing free-text prescribing instructions in a column called text.
dose_fn	function to summarise range of numbers by a single value
freq_fn	function to summarise range of frequencies by a single value
interval_fn	function to summarise range of intervals by a single value

Details

The general formula for computing numerical daily dose (ndd) is given by

$$\text{ndd} = \text{DF} \times \text{DN} / \text{DI},$$

where

DF is dose frequency, the number of dose 'events' per day

DN is dose number, or number of units of drug taken during each dose 'event'

DI is dose interval, or the number of days between 'dose days', where an interval of 1 means every day

Prescriptions can have a variable dose frequency or dose number, such as '2-4 tablets up to 3 times per day'. In this case, the user can choose to reduce these ranges to single values by taking the minimum, maximum or average of these endpoints.

Value

A data frame mapping the raw text to structured dosage information.

Examples

```
compute_ndd(cprd, min, min, mean)
```

cprd

Example data from the Clinical Practice Research Datalink (CPRD).

Description

A dataset containing prescription information for two individuals. The dataset is a hypothetical dataset resembling the real CPRD data.

Usage

```
cprd
```

Format

A data frame with 18 rows and 9 variables:

patid unique identifier given to a patient in CPRD GOLD

pracid unique identifier given to a practice in CPRD GOLD

start_date Beginning of the prescription period

prodcode CPRD unique code for the treatment selected by the GP

dosageid Identifier that allows dosage information on the event to be retrieved from a Common Dosages lookup table

text Prescription instruction for the prescribed product, as entered by the GP

qty Total quantity entered by the GP for the prescribed product

numdays Number of treatment days prescribed for a specific therapy event

dose_duration an estimated prescription duration, as entered by CPRD ...

Source

<https://www.cprd.com/sites/default/files/2022-02/CPRD%20GOLD%20Full%20Data%20Specification%20v2.5.pdf>

See Also

[min_max_dat](#)

decision_1

Decision 1: impute implausible total quantities

Description

A light wrapper around [impute_qty](#).

Usage

```
decision_1(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" do nothing; leave implausible values as-is "b" set implausible values to missing "c1" set to mean for individual's prescriptions for that drug "c2" set to mean for practice's prescriptions for that drug "c3" set to mean for populations's prescriptions for that drug "d1" set to median for individual's prescriptions for that drug "d2" set to median for practice's prescriptions for that drug "d3" set to median for population's prescriptions for that drug "e1" set to mode for individual's prescriptions for that drug "e2" set to mode for practice's prescriptions for that drug "e3" set to mode for population's prescriptions for that drug "f1" use value of individual's next prescription "f2" use value of practice's next prescription "f3" use value of population's next prescription "g1" use value of individual's previous prescription "g2" use value of practice's previous prescription "g3" use value of population's previous prescription

Note

Decisions f and g are not yet implemented.

See Also

Other decision functions: [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_10

*Decision 10: close small gaps between successive prescriptions***Description**

Where one prescription (for the same drug and patient) starts only a short time after the previous finishes, this function can close the gap, as if the prescription was continuous over the entire period.

Usage

```
decision_10(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" do nothing "b_15" change stop date of first prescription to start date of next if gap is ≤ 15 days "b_30" change stop date of first prescription to start date of next if gap is ≤ 30 days "b_60" change stop date of first prescription to start date of next if gap is ≤ 60 days

Details

The underlying function is called `close_small_gaps`

See Also

Other decision functions: [decision_1\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_2

*Decision 2: impute missing total quantities***Description**

A light wrapper around [impute_qty](#).

Usage

```
decision_2(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" Leave as missing (implicitly drop this prescription) "b1" set to mean for individual's prescriptions for that drug "b2" set to mean for practice's prescriptions for that drug "b3" set to mean for populations's prescriptions for that drug "c1" set to median for individual's prescriptions for that drug "c2" set to median for practice's prescriptions for that drug "c3" set to median for population's prescriptions for that drug "d1" set to mode for individual's prescriptions for that drug "d2" set to mode for practice's prescriptions for that drug "d3" set to mode for population's prescriptions for that drug "e1" use value of individual's next prescription "e2" use value of practice's next prescription "e3" use value of population's next prescription "f1" use value of individual's previous prescription "f2" use value of practice's previous prescription "f3" use value of population's previous prescription

Note

Decisions e and f are not yet implemented.

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_3

Decision 3: impute implausible daily doses

Description

A light wrapper around [impute_ndd](#).

Usage

```
decision_3(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" do nothing; leave implausible values as-is "b" set implausible values to missing "c1" set to mean for individual's prescriptions for that drug "c2" set to mean for practice's prescriptions for that drug "c3" set to mean for populations's prescriptions for that drug "d1" set to median for individual's prescriptions for that drug "d2" set to median for practice's prescriptions for that drug "d3" set to median for population's prescriptions for that drug "e1" set to mode for individual's prescriptions for that drug "e2" set to mode for practice's prescriptions for that drug "e3" set to mode for population's prescriptions for that drug "f1" use value of individual's next prescription "f2" use value of practice's next prescription "f3" use value of population's next prescription "g1" use value of individual's previous prescription "g2" use value of practice's previous prescription "g3" use value of population's previous prescription

Note

Decisions f and g are not yet implemented.

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

 decision_4

Decision 4: impute missing daily doses

Description

A light wrapper around [impute_ndd](#).

Usage

```
decision_4(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" Leave as missing (implicitly drop this prescription) "b1" set to mean for individual's prescriptions for that drug "b2" set to mean for practice's prescriptions for that drug "b3" set to mean for populations's prescriptions for that drug "c1" set to median for individual's prescriptions for that drug "c2" set to median for practice's prescriptions for that drug "c3" set to median for population's prescriptions for that drug "d1" set to mode for individual's prescriptions for that drug "d2" set to mode for practice's prescriptions for that drug "d3" set to mode for population's prescriptions for that drug "e1" use value of individual's next prescription "e2" use value of practice's next prescription "e3" use value of population's next prescription "f1" use value of individual's previous prescription "f2" use value of practice's previous prescription "f3" use value of population's previous prescription

Note

Decisions e and f are not yet implemented.

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

 decision_5

Decision 5: impute implausible prescription durations

Description

A light wrapper around [clean_duration](#).

Usage

```
decision_5(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" leave duration as-is "b_6" set to missing if > 6 months "b_12" set to missing if > 12 months "b_24" set to missing if > 24 months "c_6" set to 6 months if > 6 months "c_12" set to 12 months if > 12 months "c_24" set to 24 months if > 24 months

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_6

*Decision 6: choose method of calculating prescription duration***Description**

This is just shorthand for defining a column equal to one of the specified formulae. If the column(s) corresponding to decision are missing, an error will be thrown. If you have already calculated or obtained the column duration from elsewhere, this step is not necessary.

Usage

```
decision_6(data, decision = "c")
```

Arguments

data	a data frame
decision	one of the following strings: "a" numdays "b" dose_duration "c" qty / ndd

Note

This step actually takes place *before* [decision_5](#).

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_7

Decision 7: impute missing prescription durations

Description

A light wrapper around [impute_duration](#).

Usage

```
decision_7(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings: "a" Leave missing durations as-is (implicitly drop the prescription) "b" Use mean prescription duration for that drug, for that individual "c" Use mean prescription duration for that drug, for the population "d" Use individual mean duration; if not available use population mean

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

decision_8

Decision 8: disambiguate prescriptions with the same start date

Description

A light wrapper around [impute_duration](#), followed by removing duplicate rows with the same combination of prodcod, patid and start_date.

Usage

```
decision_8(data, decision = "a")
```

Arguments

data	a data frame
decision	one of the following strings "a" do nothing "b" replace with a prescription of duration equal to the mean "c" choose the shortest prescription "d" choose longest prescription "e" replace with a prescription of duration equal to the sum

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_9\(\)](#), [drug_prep\(\)](#)

`decision_9`*Decision 9: handle overlapping prescription periods*

Description

In situations where one prescription starts before another (for the same patient and drug) finishes, this function will either implicitly sum the doses (i.e. do nothing) or it will divide the intervals into non-overlapping subsets, shifting these sub-intervals forward in time until there is no overlap.

Usage

```
decision_9(data, decision = "a")
```

Arguments

<code>data</code>	a data frame
<code>decision</code>	one of the following strings: "a" allow overlapping prescriptions (implicitly sum doses) "b" move later prescription to next available time that this product is not prescribed

Details

The underlying algorithm for shifting overlapping intervals is implemented by the internal function `shift_interval`.

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [drug_prep\(\)](#)

`drug_prep`*Run drug preparation algorithm*

Description

Run drug preparation algorithm

Usage

```
drug_prep(data, plausible_values, decisions = rep("a", 10))
```

Arguments

<code>data</code>	data frame containing prescription data
<code>plausible_values</code>	data frame containing variables <code>prodcode</code> , <code>min_qty</code> , <code>max_qty</code> , <code>min_ndd</code> , <code>max_ndd</code> describing plausible ranges for values for each drug
<code>decisions</code>	character vector of length 10

Value

A data frame including estimated `stop_date` for each prescription

See Also

Other decision functions: [decision_1\(\)](#), [decision_10\(\)](#), [decision_2\(\)](#), [decision_3\(\)](#), [decision_4\(\)](#), [decision_5\(\)](#), [decision_6\(\)](#), [decision_7\(\)](#), [decision_8\(\)](#), [decision_9\(\)](#)

Examples

```
plausible_values <- data.frame(
  prodcode = c('a', 'b', 'c'),
  min_qty = 0,
  max_qty = c(50, 100, 200),
  min_ndd = 0,
  max_ndd = c(10, 20, 30)
)
drug_prep(example_therapy,
  plausible_values,
  decisions = c('a', 'a', 'a', 'a', 'a',
                'c', 'a', 'a', 'a', 'a'))
```

example_therapy	<i>Example electronic prescription dataset</i>
-----------------	--

Description

Based on a hypothetical 'therapy' file from the Clinical Practical Research Datalink (CPRD), a UK database of primary care records.

Usage

```
example_therapy
```

Note

This dataset is now generated deterministically, so it will not vary between sessions.

get_mode	<i>Get the mode (most common value) of a vector</i>
----------	---

Description

Get the mode (most common value) of a vector

Usage

```
get_mode(v, na.rm = TRUE)
```

Arguments

v	a vector
na.rm	Logical. If TRUE (the default), find mode of non-NA values

impute

*Impute missing or implausible values***Description**

This is a workhorse function used by `impute_ndd`, `impute_qty` and others.

Usage

```
impute(
  data,
  variable,
  method = c("ignore", "mean", "median", "mode", "replace", "min", "max", "sum"),
  where = is.na,
  group,
  ...,
  replace_with = NA_real_
)
```

Arguments

<code>data</code>	A data frame containing columns <code>procode</code> , <code>pracid</code> , <code>patid</code>
<code>variable</code>	Unquoted name of the column in dataset to be imputed
<code>method</code>	Method for imputing the values. See details.
<code>where</code>	Logical vector, or function applied to <code>variable</code> returning such a vector, indicating which elements to impute. Defaults to <code>is.na</code>
<code>group</code>	Level of structure for imputation. Defaults to whole study population.
<code>...</code>	Extra arguments, currently ignored
<code>replace_with</code>	if the method 'replace' is selected, which value should be inserted? • <code>ignore</code>. Do nothing, leaving input unchanged. • <code>mean</code>. Replace values with the mean by group • <code>median</code>. Replace values with the median by group • <code>mode</code>. Replace values with the most common value by group • <code>replace</code>. Replace values with <code>replace_with</code>, which defaults to NA (i.e. mark as missing) • <code>min</code>. Replace with minimum value. • <code>max</code>. Replace with maximum value. • <code>sum</code>. Replace with sum of values.

Details

The argument `where` indicates which values are to be imputed. It can be specified as either a vector or as a function. Thus you can specify, for example, `is.na` to impute all missing values, or you can pass in a vector, if it depends on something else rather than just the current values of the variable to imputed. This design may change in future. In particular, if we want to impute implausible values and impute missing values separately, it's important that these steps are independent.

Value

A data frame of the same structure as data, with values imputed

impute_duration	<i>Replace missing or implausible prescription durations</i>
-----------------	--

Description

Instead of replacing missing stop dates, we impute the durations and then infer the stop dates from there.

Usage

```
impute_duration(
  data,
  method,
  where = is.na,
  group = c("patid", "start_date"),
  ...
)
```

Arguments

data	A data frame containing columns prodcod, pracid, patid
method	Method for imputing the values. See details.
where	Logical vector, or function applied to variable returning such a vector, indicating which elements to impute. Defaults to is.na
group	Level of structure for imputation. Defaults to whole study population.
...	Extra arguments, currently ignored

Details

We can fix clashing start dates by setting group to start_date and patid, i.e. average over groups with more than one member; any metric should return the original values if the group size is one.

Value

A data frame of the same structure as data, with values imputed

Examples

```
example_duration <- transform(example_therapy, duration = qty / ndd)
impute_duration(example_duration, method = 'mean', group = 'patid')
```

impute_ndd	<i>Replace implausible or missing numerical daily doses (NDD)</i>
------------	---

Description

Replace implausible or missing numerical daily doses (NDD)

Usage

```
impute_ndd(data, method, where = is.na, group = "population", ...)
```

Arguments

data	A data frame containing columns prodcode, pracid, patid
method	Method for imputing the values. See details.
where	Logical vector, or function applied to variable returning such a vector, indicating which elements to impute. Defaults to is.na
group	Level of structure for imputation. Defaults to whole study population.
...	Extra arguments, currently ignored

Value

A data frame of the same structure as data, with values imputed

Examples

```
impute_ndd(example_therapy, 'mean')
```

impute_qty	<i>Find implausible entries Replace implausible or missing prescription quantities</i>
------------	--

Description

Find implausible entries Replace implausible or missing prescription quantities

Usage

```
impute_qty(data, method, where = is.na, group = "population", ...)
```

Arguments

data	A data frame containing columns prodcode, pracid, patid
method	Method for imputing the values. See details.
where	Logical vector, or function applied to variable returning such a vector, indicating which elements to impute. Defaults to <code>is.na</code>
group	Level of structure for imputation. Defaults to whole study population.
...	Extra arguments, currently ignored

Value

A data frame of the same structure as data, with values imputed

Examples

```
impute_qty(example_therapy, 'mean')
```

isolate_overlaps	<i>Separating overlapping prescription periods</i>
------------------	--

Description

Run this function and then you can either simply discard overlapping intervals or shift them around using an appropriate algorithm.

Usage

```
isolate_overlaps(data)
```

Arguments

data	A data frame including variables patid, start_date, stop_date and prodcode
------	--

Details

The older implementation used `isolateoverlaps` from the `intervalaverage` package and `Overlap` from the `DescTools` package. Here we refactor it using functions from `tidyverse` instead.

Value

A data frame of patid, prodcode, start_date and stop_date, where intervals are either exactly overlapping or mutually non-overlapping (but not partially overlapping), such that the union of such intervals is equivalent to those originally provided in data

Note

This function currently doesn't use any keys except patid and prodcode. It may be desirable to add a row ID, for matching each partial interval back to the original interval from which it was derived. This may be relevant to models using weighted dosages.

See Also

intervalaverage::isolateoverlaps, [foverlaps](#)

Examples

```
set.seed(1)
overlapping_data <- data.frame(
  rowid = 1:20,
  patid = 1:2,
  prodcode = 'a',
  start_date = Sys.Date() + c(round(rexp(19, 1/7)), -20),
  qty = rpois(20, 64),
  ndd = sample(seq(.5, 12, by = .5), 20, replace = TRUE),
  stringsAsFactors = FALSE
)
overlapping_data <- transform(overlapping_data,
  stop_date = start_date + qty / ndd
)
isolate_overlaps(overlapping_data)
```

make_decisions

Human-friendly interface to the drug prep algorithm

Description

A helper function that allows specifying decision rules using English words rather than alphanumeric codes. Translates the rules into the corresponding codes and then passes them to [drug_prep](#) functions.

Usage

```
make_decisions(
  implausible_qty,
  missing_qty,
  implausible_ndd,
  missing_ndd,
  implausible_duration,
  calculate_duration,
  missing_duration,
  clash_start,
  overlapping,
```

```
    small_gaps
  )
```

Arguments

implausible_qty implausible total drug quantities

missing_qty missing total drug quantities

implausible_ndd implausible daily dosage

missing_ndd missing daily dosage

implausible_duration overly-long prescription durations

calculate_duration formula or variable to compute prescription duration

missing_duration missing prescription duration

clash_start how to disambiguate prescriptions that start on the same date

overlapping how to handle prescription periods that overlap with one another

small_gaps how to handle short gaps between successive prescriptions

The argument **decision_phrases** may contain the following terms (without brackets, separated with spaces). Additional or incorrectly-named elements will be ignored.

implausible_qty (ignore|missing|mean|median|mode|next|previous) (individual|practice|population)

implausible_ndd (ignore|missing|mean|median|mode|next|previous) (individual|practice|population)

implausible_duration (ignore|missing|truncate) (6|12|24)

missing_qty (ignore|mean|median|mode|next|previous) (individual|practice|population)

missing_ndd (ignore|mean|median|mode|next|previous) (individual|practice|population)

missing_duration (ignore|mean) (individual|population|both)

calculate_duration (numdays|dose_duration|qty/ndd)

clash_start (ignore|mean|shortest|longest|sum)

overlapping (allow|shift)

small_gaps (ignore|close) (15|30|60)

Value

A character vector suitable for passing to the **decisions** argument of the [drug_prep](#) function.

Examples

```
make_decisions('ignore',
               'mean population',
               'missing',
               'mean practice',
               'truncate 6',
               'qty / ndd',
```

```
'mean individual',
'mean',
'allow',
'close 15')
```

min_max_dat	<i>Example plausible ranges for prescription quantity and dosage</i>
-------------	--

Description

Minimum and maximum plausible values for total prescription quantity and numerical daily dose of prescriptions given in the [cprd](#) dataset. Both datasets are hypothetical.

Usage

```
min_max_dat
```

Format

- A data frame with 2 rows and 5 variables:
- procode** CPRD unique code for the treatment selected by the GP
 - max_qty** maximum possible quantity to be prescribed for the product
 - min_qty** minimum possible quantity to be prescribed for the product
 - max_ndd** maximum possible number of daily dose to be prescribed for the product
 - min_ndd** minimum possible number of daily dose to be prescribed for the product ...

See Also

```
cprd
```

outside_range	<i>Do values fall outside a specified 'plausible' range?</i>
---------------	--

Description

A utility function for indicating if elements of a vector are implausible.

Usage

```
outside_range(x, lower, upper, open = TRUE)
```

Arguments

x	numeric vector
lower	minimum plausible value
upper	maximum plausible value
open	logical. If TRUE, values exactly equal to lower or upper are also considered implausible

Details

Though the function [between](#) already exists, it is not vectorised over the bounds.

shift_interval	<i>Shift time intervals until they no longer overlap</i>
----------------	--

Description

This is a function used by [decision_9](#).

Usage

```
shift_interval(x)
```

Arguments

x	a data frame containing variables start_date, stop_date and patid
---	---

Value

A data frame with time intervals moved such that they no longer overlap

Index

- * **datasets**
 - cprd, [5](#)
 - min_max_dat, [22](#)
- * **decision functions**
 - decision_1, [6](#)
 - decision_10, [7](#)
 - decision_2, [7](#)
 - decision_3, [8](#)
 - decision_4, [9](#)
 - decision_5, [10](#)
 - decision_6, [11](#)
 - decision_7, [12](#)
 - decision_8, [12](#)
 - decision_9, [13](#)
 - drug_prep, [14](#)
- between, [23](#)
- clean_duration, [2, 10](#)
- close_small_gaps, [3](#)
- compute_ndd, [4](#)
- cprd, [5, 22](#)
- decision_1, [6](#)
- decision_1(), [7–14](#)
- decision_10, [7](#)
- decision_10(), [6, 8–14](#)
- decision_2, [7](#)
- decision_2(), [6, 7, 9–14](#)
- decision_3, [8](#)
- decision_3(), [6–8, 10–14](#)
- decision_4, [9](#)
- decision_4(), [6–9, 11–14](#)
- decision_5, [10, 11](#)
- decision_5(), [6–14](#)
- decision_6, [11](#)
- decision_6(), [6–14](#)
- decision_7, [12](#)
- decision_7(), [6–11, 13, 14](#)
- decision_8, [12](#)
- decision_8(), [6–14](#)
- decision_9, [13, 23](#)
- decision_9(), [6–14](#)
- drug_prep, [14, 20, 21](#)
- drug_prep(), [6–13](#)
- example_therapy, [15](#)
- foverlaps, [20](#)
- get_mode, [15](#)
- impute, [16](#)
- impute_duration, [12, 17](#)
- impute_ndd, [8, 9, 16, 18](#)
- impute_qty, [6, 7, 16, 18](#)
- is.na, [16–19](#)
- isolate_overlaps, [19](#)
- make_decisions, [20](#)
- min_max_dat, [5, 22](#)
- outside_range, [22](#)
- shift_interval, [23](#)