Package 'AdhereRViz'

October 12, 2022

Type Package

Title Adherence to Medications

Version 0.2.1

Author Dan Dediu [aut, cre],
Alexandra Dima [aut],
Samuel Allemann [aut]

Maintainer Dan Dediu <ddediu@gmail.com>

Description Interactive graphical user interface (GUI) for the package 'AdhereR', allowing the user to access different data sources, to explore the patterns of medication use therein, and the computation of various measures of adherence. It is implemented using Shiny and HTML/CSS/JavaScript.

URL https://github.com/ddediu/AdhereR

License GPL (>= 2)

Imports AdhereR (>= 0.7.1), data.table (>= 1.9), manipulate (>= 1.0), shiny (>= 1.0), shinyWidgets (>= 0.4.4), shinyjs (>= 1.0), V8 (>= 1.5), colourpicker (>= 1.0), viridisLite(>= 0.3), highlight (>= 0.4), clipr (>= 0.4), knitr (>= 1.20), DBI (>= 1.0), RMariaDB (>= 1.0.5), RSQLite (>= 2.1)

Depends R (>= 3.0)

Suggests rmarkdown (>= 1.1), readODS (>= 1.6), readxl (>= 1.2), haven (>= 2.0), R.rsp (>= 0.40)

VignetteBuilder knitr, R.rsp

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation no

Repository CRAN

Date/Publication 2022-06-24 08:00:07 UTC

R topics documented:

`plot_interactive_cma` *Interactive exploration and CMA computation.*

Description

Interactively plots the data for one of more patients, allowing the real-time exploration of the various CMAs and their parameters. It can use `Rstudio`'s `manipulate` library (deprecated) or `Shiny` (recommended).

Usage

```
plot_interactive_cma(
  data = NULL,
  ID = NULL,
  cma.class = c("simple", "per episode", "sliding window")[1],
  print.full.params = FALSE,
  ID.colname = NA,
  event.date.colname = NA,
  event.duration.colname = NA,
  event.daily.dose.colname = NA,
  medication.class.colname = NA,
  medication.groups = NULL,
  date.format = "%m/%d/%Y",
  followup.window.start.max = 5 * 365,
  followup.window.duration.max = 5 * 365,
  observation.window.start.max = followup.window.start.max,
  observation.window.duration.max = followup.window.duration.max,
  align.all.patients = FALSE,
  align.first.event.at.zero = FALSE,
  maximum.permissible.gap.max = 2 * 365,
  sliding.window.start.max = followup.window.start.max,
  sliding.window.duration.max = 2 * 365,
  sliding.window.step.duration.max = 2 * 365,
  backend = c("shiny", "rstudio")[1],
  use.system.browser = FALSE,
  get.colnames.fnc = function(d) names(d),
  get.patients.fnc = function(d, idcol) unique(d[[idcol]]),
 get.data.for.patients.fnc = function(patientid, d, idcol, cols = NA, maxrows = NA)
    d[d[[idcol]] %in% patientid, ],
  ...
)
```

Arguments

`data`	Usually a `data.frame` containing the events (prescribing or dispensing) used to compute the CMA. Must contain, at a minimum, the patient unique ID, the

event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters). Alternatively, this can be any other data source (for example, a connection to a database), in which case the user must redefine the arguments `get.colnames.fnc`, `get.patients.fnc` and `get.data.for.patients.fnc` appropriately. Currently, this works only when using Shiny for interactive rendering. For a working example, please see the vignette describing the interfacing with databases.

`ID` The ID (as given in the `ID.colname` column) of the patient whose data to interactively plot (if absent, pick the first one); please not that this an be interactively selected during plotting.

`cma.class` The type of CMAs to plot; can be "simple" (CMA0 to CMA9), "per episode", or "sliding window".

`print.full.params`
A *logical* specifying if the values of all the parameters used to generate the current plot should be printed in the console (if *TRUE*, it can generate extremely verbose output!).

`ID.colname` A *string*, the name of the column in `data` containing the unique patient ID, or `NA` if not defined.

`event.date.colname`
A *string*, the name of the column in `data` containing the start date of the event (in the format given in the `date.format` parameter), or `NA` if not defined.

`event.duration.colname`
A *string*, the name of the column in `data` containing the event duration (in days), or `NA` if not defined.

`event.daily.dose.colname`
A *string*, the name of the column in `data` containing the prescribed daily dose, or `NA` if not defined.

`medication.class.colname`
A *string*, the name of the column in `data` containing the classes/types/groups of medication, or `NA` if not defined.

`medication.groups`
A *named character vector* defining the medication classes, the name of a column in the data the defines the groups, or `NULL` if none (the default).

`date.format` A *string* giving the format of the dates used in the `data` and the other parameters; see the `format` parameters of the `as.Date` function for details (NB, this concerns only the dates given as strings and not as `Date` objects).

`followup.window.start.max`
The maximum number of days when the follow-up window can start.

`followup.window.duration.max`
The maximum duration of the follow-up window in days.

`observation.window.start.max`
The maximum number of days when the observation window can start.

`observation.window.duration.max`
The maximum duration of the observation window in days.

`align.all.patients`
: Should the patients be aligend?

`align.first.event.at.zero`
: Should the first event be put at zero?

`maximum.permissible.gap.max`
: The maximum permissible gap in days.

`sliding.window.start.max`
: The maximum number of days when the sliding windows can start.

`sliding.window.duration.max`
: The maximum duration of the sliding windows in days.

`sliding.window.step.duration.max`
: The maximum sliding window step in days.

`backend`
: The plotting backend to use; "shiny" (the default) tries to use the Shiny framework, while "rstudio" uses the manipulate RStudio capability.

`use.system.browser`
: For shiny, use the system browser?

`get.colnames.fnc`
: A *function* taking as parameter the data source and returning the column names. Must be overridden when the data source is not derived from a `data.frame`.

`get.patients.fnc`
: A *function* taking as parameter the data source and the patient ID column name, and returns the list of all patient IDs. Must be overridden when the data source is not derived from a `data.frame`.

`get.data.for.patients.fnc`
: A *function* taking as parameter a (set of) patient ID(s), the data source, and the patient ID column name, and returns the list of all patient IDs. Must be overridden when the data source is not derived from a `data.frame`.

`...`
: Extra arguments.

Details

The `manipulate` is kept for backward compatibility only, as it is much more limited than `Shiny` and will receive no new development in the future. `Shiny` currently allows the use of any other data source besides a default (and usual) `data.frame` (or derived), such a connection to an `SQL` database. In this case, the user *must* redefine the three argument functions `get.colnames.fnc`, `get.patients.fnc` and `get.data.for.patients.fnc` which collectively define an interface for listing the column names, all the patient IDs, and for retreiving the actual data for a (set of) patient ID(s). A fully worked example is described in the vignette detailing the access to standard databases storaging the patient information. For more info please see the vignette.

Value

Nothing

See Also

The vignette *AdhereR: Interactive plotting (and more) with Shiny*.

Examples

```
## Not run:
library(AdhereR);
plot_interactive_cma(med.events,
                     ID.colname="PATIENT_ID",
                     event.date.colname="DATE",
                     event.duration.colname="DURATION",
                     event.daily.dose.colname="PERDAY",
                     medication.class.colname="CATEGORY");
## End(Not run)
```

Index