Package: epidict (via r-universe)

August 18, 2024

Title Epidemiology data dictionaries and random data generators

Version 0.0.0.9001

Description The 'R4EPIs' project <https://R4epis.netlify.com> seeks to provide a set of standardized tools for analysis of outbreak and survey data in humanitarian aid settings. This package currently provides standardized data dictionaries from MSF OCA for four outbreak scenarios (Acute Jaundice Syndrome, Cholera, Measles, Meningitis) and three surveys (Retrospective mortality and access to care, Malnutrition, and Vaccination coverage). In addition, a data generator from these dictionaries is provided.

URL https://r4epis.netlify.com, https://github.com/R4EPI/epidict, https://r4epi.github.io/epidict, https://r4epi.github.io/epidict/

License GPL-3

Imports tibble, readxl, stats, utils, dplyr, tidyr, rlang

Suggests testthat (>= 2.1.0), matchmaker, covr, knitr, rmarkdown, DT

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

VignetteBuilder knitr

Repository https://r4epi.r-universe.dev

RemoteUrl https://github.com/r4epi/epidict

RemoteRef HEAD

RemoteSha 9cf5a5315c7bb2c4d3c9ab986ddd5a0534e8cd65

Contents

---

gen_data *Generate random linelist or survey data*

Description

Based on a dictionary generator like `msf_dict()` or `msf_dict_survey()`, this function will generate a randomized data set based on values defined in the dictionaries. The randomized dataset produced should mimic an excel export from DHIS2 for outbreaks and a Kobo export for surveys.

Usage

```
gen_data(
  dictionary,
  varnames = "data_element_shortname",
  numcases = 300,
  org = "MSF"
)
```

Arguments

`dictionary`	Specify which dictionary you would like to use.
`varnames`	Specify name of column that contains variable names. If `dictionary` is a survey, `varnames` needs to be "name"‘.
`numcases`	Specify the number of cases you want (default is 300)
`org`	the organization the dictionary belongs to. Currently, only MSF exists. In the future, dictionaries from WHO and other organizations may become available.

Value

a data frame with cases in rows and variables in columns. The number of columns will vary from dictionary to dictionary, so please use the dictionary functions to generate a corresponding dictionary.

Examples

```
if (require("dplyr") & require("matchmaker")) {
  withAutoprint({

    # You will often want to use MSF dictionaries to translate codes to human-
    # readable variables. Here, we generate a data set of 20 cases:
    dat <- gen_data(
      dictionary = "Cholera",
      varnames = "data_element_shortname",
      numcases = 20,
      org = "MSF"
    )
```

```
    print(dat)

    # We want the expanded dictionary, so we will select `compact = FALSE`
    dict <- msf_dict(disease = "Cholera", long = TRUE, compact = FALSE, tibble = TRUE)
    print(dict)

    # Now we can use matchmaker to filter the data:
    dat_clean <- matchmaker::match_df(dat, dict,
      from = "option_code",
      to = "option_name",
      by = "data_element_shortname",
      order = "option_order_in_set"
    )
    print(dat_clean)

  })
}
```

`msf_dict` *MSF data dictionaries and dummy datasets*

Description

These function produces MSF OCA dictionaries based on DHIS2 (for outbreaks) and Kobo (for surveys) data sets defining the data element name, code, short names, types, and key/value pairs for translating the codes into human-readable format.

Usage

```
msf_dict(
  disease,
  name = "MSF-outbreak-dict.xlsx",
  tibble = TRUE,
  compact = TRUE,
  long = TRUE
)

msf_dict_survey(
  disease,
  name = "MSF-survey-dict.xlsx",
  tibble = TRUE,
  compact = TRUE,
  long = TRUE,
  template = TRUE
)
```

Arguments

disease	Specify which disease you would like to use. • msf_dict() supports "AJS", "Cholera", "Measles", "Meningitis" • msf_dict_survey() supports "Mortality", "Nutrition", "Vaccination_long" and "Vaccination_short" (only used in surveys if template = TRUE)
name	the name of the dictionary stored in the package. • msf_dict_survey() supports Kobo dictionaries not stored within this package, to use these: specify nameas path to .xlsx file and set the template = False
tibble	Return data dictionary as a tidyverse tibble (default is TRUE)
compact	if TRUE (default), then a nested data frame is returned where each row represents a single variable and a nested data frame column called "options", which can be expanded with tidyr::unnest(). This only works if long = TRUE.
long	If TRUE (default), the returned data dictionary is in long format with each option getting one row. If FALSE, then two data frames are returned, one with variables and the other with content options. @param template Only used for msf_dict_survey(). If TRUE (default) the returned data dictionary is a generic MSF OCA ERB pre-approved dictionary. If FALSE allows you to read in your own Kobo dictionary by defining a path in name.
template	(for survey dictionaries): if TRUE read in a generic dictionary based on the MSF OCA ERB pre-approved template. However you can also specify your own dictionary if this differs substantially, by setting template = FALSE and defining a path in name.

See Also

matchmaker::match_df() gen_data() msf_dict_survey()

Examples

```
if (require("dplyr") & require("matchmaker")) {
  withAutoprint({
    # You will often want to use MSF dictionaries to translate codes to human-
    # readable variables. Here, we generate a data set of 20 cases:
    dat <- gen_data(
      dictionary = "Cholera",
      varnames = "data_element_shortname",
      numcases = 20,
      org = "MSF"
    )
    print(dat)

    # We want the expanded dictionary, so we will select `compact = FALSE`
    dict <- msf_dict(disease = "Cholera", long = TRUE, compact = FALSE, tibble = TRUE)
    print(dict)
```

```
    # Now we can use matchmaker to filter the data:
    dat_clean <- matchmaker::match_df(dat, dict,
      from = "option_code",
      to = "option_name",
      by = "data_element_shortname",
      order = "option_order_in_set"
    )
    print(dat_clean)
  })
}
```

`msf_dict_rename_helper` *Helper for aligning your data to a standardised dictionary or your own dictionary.*

Description

Helper for aligning your data to a standardised dictionary or your own dictionary.

Usage

```
msf_dict_rename_helper(
  disease,
  name,
  varnames = "data_element_shortname",
  varnames_type,
  rmd,
  template = TRUE,
  copy_to_clipboard = TRUE
)
```

Arguments

`disease`	Specify which disease you would like to use. Currently supports "Cholera", "Measles", "Meningitis", "AJS", "Mortality", "Nutrition", "Vaccination_short" and "Vaccination_long".
`name`	The name of the dictionary stored in the package. The default will use dictionaries from the package. However you can also use dictionaries not stored within this package, to use these: specify `name`as path to .xlsx file and set the `template = False` - nb. this needs to be a dataframe containing `varnames` and `varnames_type`. You will also need to specify a path to `rmd`.
`varnames`	The name of column that contains variable names. The default set to "data_element_shortname". If `dictionary` is a survey ("Mortality", "Nutrition", "Vaccination_short" or "Vaccination_long") `varnames` needs to be "name"‘. Otherwise if using your own dictionary then specify.

`varnames_type` The name of column that contains the variable type. The default will use "data_element_valuetype" for DHIS2 and "type" for Kobo dictionaries. If you specify your own dictionary then this needs to be the same length as `varnames` in your dictionary.

`rmd` The Rmarkdown template which you would like to compare to. Default is will use those included in the package. However you can also use Rmarkdowns not stored within this package, to use these: specify `rmd`as path to .rmd file and set `template = False`; nb. you will need to specify a path to a file in `name` which contains `varnames` and `varnames_type`.

`template` If `TRUE` (default) read in a generic dictionary and Rmarkdown based on the MSF OCA ERB pre-approved template. However you can also specify your own dictionary if this differs substantially, by setting `template = FALSE`.

`copy_to_clipboard`
if `TRUE` (default), the rename template will be copied to the user's clipboard with `clipr::write_clip()`. If `FALSE`, the rename template will be printed to the user's console.

Value

A dplyr command used to rename columns in your data frame according to the dictionary

Index