Package ‘eudract’

January 9, 2026

Type Package

Title Creates Safety Results Summary in XML to Upload to EudraCT, or ClinicalTrials.gov

Date 2026-01-09

Version 1.1.0

URL `https://shug0131.github.io/eudraCT/`

BugReports `https://github.com/shug0131/eudraCT/issues`

Description The remit of the European Clinical Trials Data Base (EudraCT <`https://eudract.ema.europa.eu/`>), or ClinicalTrials.gov <`https://clinicaltrials.gov/`>, is to provide open access to summaries of all registered clinical trial results; thus aiming to prevent non-reporting of negative results and provide open-access to results to inform future research. The amount of information required and the format of the results, however, imposes a large extra workload at the end of studies on clinical trial units. In particular, the adverse-event-reporting component requires entering: each unique combination of treatment group and safety event; for every such event above, a further 4 pieces of information (body system, number of occurrences, number of subjects, number exposed) for non-serious events, plus an extra three pieces of data for serious adverse events (numbers of causally related events, deaths, causally related deaths). This package prepares the required statistics needed by EudraCT and formats them into the precise requirements to directly upload an XML file into the web portal, with no further data entry by hand.

License GPL-2

Language en-GB

Encoding UTF-8

LazyData true

Imports dplyr, ggplot2, httr, magrittr, patchwork, scales, tidyr, utils, xml2, xslt

Depends R (>= 4.1.0)

RoxygenNote 7.3.3

Suggests covr, knitr, rmarkdown, stringr, testthat, tools, vdiffr, pharmaverseadam

VignetteBuilder knitr

NeedsCompilation no

Author Simon Bond [cre],
Beatrice Pantaleo [aut]

Maintainer Simon Bond <`simon.bond7@nhs.net`>

Repository CRAN

Date/Publication 2026-01-09 14:00:02 UTC

Contents

`clintrials_gov_convert` *applies a conversion using xslt from a simple xml file to a ClinicalTrials.gov compatible file, and checks against the schema*

Description

applies a conversion using xslt from a simple xml file to a ClinicalTrials.gov compatible file, and checks against the schema

Usage

```
clintrials_gov_convert(
  input,
  original,
  output,
  xslt = system.file("extdata", "simpleToCtGov.xslt", package = "eudract"),
  schema_input = system.file("extdata", "simple.xsd", package = "eudract"),
```

```
  schema_results = system.file("extdata", "RRSUploadSchema.xsd", package = "eudract"),
  schema_output = system.file("extdata", "ProtocolRecordSchema.xsd", package = "eudract"),
   soc = system.file("extdata", "soc.xml", package = "eudract")
)
```

Arguments

`input`	a character string giving the file path to the simple xml file
`original`	a character string giving the file path to the study file downloaded from ClinicalTrials.gov
`output`	a character string naming the output file
`xslt`	a character string giving the file path to the xslt script. Defaults to the script provided in this package
`schema_input`	a character string giving the file path to the schema for the simple xml file. Defaults to the schema provided in this package
`schema_results`	a character string giving the file path to the schema for the results section of the output. A copy was downloaded and is provided in this package as the default.
`schema_output`	a character string giving the file path to the schema for the overall output. A copy was downloaded and is provided in this package as the default.
`soc`	a character string giving an xml file that contains the System Organ Class look-up table going from EudraCT numbers to ClinicalTrials words.

Value

the output from the validation against the schema. A new file is created as a side-effect, which is suitable to upload into ClinicalTrials.gov. This over-writes the file given in `original` with the additional safety events.

See Also

`safety_summary` `simple_safety_xml`

Examples

```
safety_statistics <- safety_summary(safety,
                                    exposed=c("Experimental"=60,"Control"=67))
simple <- tempfile(fileext = ".xml")
eudract <- tempfile(fileext = ".xml")
ct <- tempfile(fileext = ".xml")
simple_safety_xml(safety_statistics, simple)
eudract_convert(input=simple,
                output=eudract)
clintrials_gov_convert(input=simple,
                       original=system.file("extdata", "1234.xml", package ="eudract"),
                output=ct)
## Not run:
  # This needs a real user account to work
  clintrials_gov_upload(
```

```
    input=simple,
    orgname="CTU",
    username="Student",
    password="Guinness",
    studyid="1234"
    )


## End(Not run)
```

`clintrials_gov_upload` *applies a conversion using xslt from a simple xml file to a ClinicalTrials.gov compatible file, merges into a study record from the portal, and uploads the result.*

Description

applies a conversion using xslt from a simple xml file to a ClinicalTrials.gov compatible file, merges into a study record from the portal, and uploads the result.

Usage

```
clintrials_gov_upload(
  input,
  orgname,
  username,
  password,
  studyid,
  url = "https://register.clinicaltrials.gov/",
  check = interactive(),
  output = "study_file.xml",
  backup = "bak_study_file.xml",
  xslt = system.file("extdata", "simpleToCtGov.xslt", package = "eudract"),
  schema_input = system.file("extdata", "simple.xsd", package = "eudract"),
 schema_results = system.file("extdata", "RRSUploadSchema.xsd", package = "eudract"),
 schema_output = system.file("extdata", "ProtocolRecordSchema.xsd", package = "eudract"),
  soc = system.file("extdata", "soc.xml", package = "eudract")
)
```

Arguments

`input`	a character string giving the file path to the simple xml file
`orgname`	a character string giving the organisation name used to log in
`username`	a character string giving the user-name used to log in
`password`	a character string giving the password used to log in
`studyid`	a character string given the unique study id within the portal

`url`	a character string giving the URL of the website to log in. Defaults to the live site, but the testing site is the alternative.
`check`	a logical that will check on the command line if you want to proceed as results will be overwritten. Defaults to TRUE.
`output`	a character string naming the output xml file
`backup`	a character string naming the copy of the original data that is created.
`xslt`	a character string giving the file path to the xslt script. Defaults to the script provided in this package
`schema_input`	a character string giving the file path to the schema for the simple xml file. Defaults to the schema provided in this package
`schema_results`	a character string giving the file path to the schema for the results section of the output. A copy was downloaded and is provided in this package as the default.
`schema_output`	a character string giving the file path to the schema for the overall output. A copy was downloaded and is provided in this package as the default.
`soc`	a character string giving an xml file that contains the System Organ Class look-up table going from EudraCT numbers to ClinicalTrials words.

Value

Invisibly returns the results from the two API with the portal `response`, `POST` . A new file is created as a side-effect, which is uploaded into ClinicalTrials.gov. This over-writes the original safety data online with the additional safety events. A backup copy of the original data is also saved.

See Also

`safety_summary` `simple_safety_xml` [ClinicalTrials.gov manual](https://prsinfo.clinicaltrials.gov/prs-users-guide.html#section10)

Examples

```
safety_statistics <- safety_summary(safety,
                                    exposed=c("Experimental"=60,"Control"=67))
simple <- tempfile(fileext = ".xml")
eudract <- tempfile(fileext = ".xml")
ct <- tempfile(fileext = ".xml")
simple_safety_xml(safety_statistics, simple)
eudract_convert(input=simple,
                output=eudract)
clintrials_gov_convert(input=simple,
                       original=system.file("extdata", "1234.xml", package ="eudract"),
                output=ct)
## Not run:
  # This needs a real user account to work
  clintrials_gov_upload(
    input=simple,
    orgname="CTU",
    username="Student",
    password="Guinness",
```

```
    studyid="1234"
    )


## End(Not run)
```

`create.safety_summary` *function that creates a safety_summary object from individual data.frames*

Description

function that creates a safety_summary object from individual data.frames

Usage

```
create.safety_summary(group, non_serious, serious)
```

Arguments

`group`	a data frame that contains the group-level statistics
`non_serious`	a data frame that contains the non-serious term-group level statistics
`serious`	a data frame that contains the serious term-group level statistics

Value

a safety_summary object

`dot_plot` *creates a dot-plot of safety data showing the absolute and relative risks*

Description

creates a dot-plot of safety data showing the absolute and relative risks

Usage

```
dot_plot(
  safety,
  type = c("non_serious", "serious"),
  reference = safety$GROUP$title[1],
  size = 95,
  text_width = 10,
  base = 2,
  valid_estimates = TRUE
)
```

Arguments

`safety`	an object created by `safety_summary` or by `relative_risk`, in case you want to re-order or filter the choice of rows.
`type`	a choice of "non_serious" (default) or "serious" as to which type of AE to report on
`reference`	character vector naming the reference arm for the calculations. Defaults to the first row of the safety$GROUP.
`size`	a number between 0-100, giving the size of the confidence interval. Default is 95.
`text_width`	Integer giving a target width to which the labels are wrapped. Defaults to 10.
`base`	numeric value to which a log scale uses as tick marks. Suggest powers of 2, or 5.
`valid_estimates`	a logical, which determines if only terms with valid estimates of relative risk are included in the table. The alternative is to include terms with zeroes.

Details

This is essentially a list of two ggplot objects joined together in a list, named as "left.panel" and "right.panel". They can each be individually edited if needed

Value

a graphical object that shows the estimates and CI of relative and absolute risk.

See Also

`safety_summary` `relative_risk` [relative_risks()]

Examples

```
safety_statistics <- safety_summary(safety,
           exposed=c("Experimental"=60,"Control"=67))
head( relative_risk(safety_statistics, type="serious") )
fig <- dot_plot(safety_statistics, type="non_serious", base=4)
fig
fig$left.panel <- fig$left.panel + ggplot2::labs(title="Absolute Risk")
fig
temp <- tempfile(fileext=".png")
png(filename = temp)
print(fig)
dev.off()
```

`eudract_convert` *applies a conversion using xslt from a simple xml file to a eudract compatible file, and checks against the schema*

Description

applies a conversion using xslt from a simple xml file to a eudract compatible file, and checks against the schema

Usage

```
eudract_convert(
  input,
  output,
  xslt = system.file("extdata", "simpleToEudraCT.xslt", package = "eudract"),
  schema_input = system.file("extdata", "simple.xsd", package = "eudract"),
 schema_output = system.file("extdata", "adverseEvents.xsd", package = "eudract")
)
```

Arguments

`input`	a character string giving the file path to the simple xml file
`output`	a character string naming the output file
`xslt`	a character string giving the file path to the xslt script. Defaults to the script provided in this package
`schema_input`	a character string giving the file path to the schema for the simple xml file. Defaults to the schema provided in this package
`schema_output`	a character string giving the file path to the schema. A copy was downloaded and is provided in this package as the default.

Value

the output from the validation against the schema. A new file is created as a side-effect, which is suitable to upload into eudraCT.

See Also

`safety_summary` `simple_safety_xml`

Examples

```
safety_statistics <- safety_summary(safety,
                                    exposed=c("Experimental"=60,"Control"=67))
simple <- tempfile(fileext = ".xml")
eudract <- tempfile(fileext = ".xml")
ct <- tempfile(fileext = ".xml")
simple_safety_xml(safety_statistics, simple)
```

```
eudract_convert(input=simple,
                output=eudract)
clintrials_gov_convert(input=simple,
                       original=system.file("extdata", "1234.xml", package ="eudract"),
                output=ct)
## Not run:
  # This needs a real user account to work
  clintrials_gov_upload(
    input=simple,
    orgname="CTU",
    username="Student",
    password="Guinness",
    studyid="1234"
    )


## End(Not run)
```

`incidence_table` *provide standard structured tables to report incidence rates of AEs by arm*

Description

provide standard structured tables to report incidence rates of AEs by arm

Usage

```
incidence_table(safety, type = c("non_serious", "serious"), percent_round = 0)
```

Arguments

`safety`	an object created by `safety_summary`
`type`	a choice of "non_serious" (default) or "serious" as to which type of AE to report on
`percent_round`	integer giving the number of decimal places to round the incidence percentage. Default of 0. Maybe you need more if there is a large sample size and a rare event of interest

Value

a data.frame that can be directly printed as a table to a report. Each arm has its own column and the text contains "p% (r, o)", where r is the number of participants with the term, o is the number of occurrences, and p a percentage of participants with the term.

See Also

`safety_summary`

Examples

```
safety_statistics <- safety_summary(safety,
           exposed=c("Experimental"=60,"Control"=67))
head( incidence_table(safety_statistics, type="serious") )
```

`plot.dot_plot` *plot methods for dot_plot object*

Description

plot methods for dot_plot object

Usage

```
## S3 method for class 'dot_plot'
plot(x, ...)
```

Arguments

`x`	dot_plot object
`...`	other arguments for generic methods

`print.dot_plot` *print methods for dot_plot object*

Description

print methods for dot_plot object

Usage

```
## S3 method for class 'dot_plot'
print(x, ...)
```

Arguments

`x`	dot_plot object
`...`	other arguments for generic methods

`relative_risk` *Calculate relative risks to be reported or plotted as dot plot*

Description

Calculate relative risks to be reported or plotted as dot plot

Usage

```
relative_risk(
  safety,
  type = c("non_serious", "serious"),
  reference = safety$GROUP$title[1],
  size = 95
)

relative_risk_table(
  safety,
  type = c("non_serious", "serious"),
  reference = safety$GROUP$title[1],
  size = 95,
  digits = 3,
  valid_estimates = TRUE
)

order_filter(rel_risk, threshold = 10)
```

Arguments

`safety`	an object created by `safety_summary`
`type`	a choice of "non_serious" (default) or "serious" as to which type of AE to report on
`reference`	character vector naming the reference arm for the calculations. Defaults to the first row of the safety$GROUP.
`size`	a number between 0-100, giving the size of the confidence interval. Default is 95.
`digits`	integer giving the number of significant figures to report to. Default of 3.
`valid_estimates`	a logical, which determines if only terms with valid estimates of relative risk are included in the table. The alternative is to include terms with zeroes.
`rel_risk`	a relative risk object
`threshold`	a threshold on the percent scale, the max percentage for a term the incidence rate needs to exceed

Value

`relative_risk` returns of list of four items.

- "relative_risk" a data.frame that has the relative risk estimate and confidence intervals.
- "percentage" a data.frame with absolute percentages.
- "GROUP" a copy from the original `safety_summary` object.
- "reference" naming the reference group used to calculate relative risks

No adjustment made to deal with zeroes. This is suitable input for the `dot_plot` function, and in most cases will not be used directly, but may potentially be modified with filtration, or editing of terms, see `order_filter`.

`relative_risk_table` returns a data frame that is suitable for printing to a report, giving relative risks

`order_filter` returns a revised relative risk object, with the terms concatenated with SOC if there are any duplicates, then ordered by relative risk, into a factor, and filtered to only those terms with an incidence rate above the threshold.

See Also

`safety_summary` `dot_plot`

Examples

```
safety_statistics <- safety_summary(safety,
           exposed=c("Experimental"=60,"Control"=67))
head( relative_risk(safety_statistics, type="serious") )
relative_risk_table(safety_statistics, type="serious")
rr <- relative_risk(safety_statistics)
rr2 <- order_filter(rr, threshold=2)
dot_plot(rr2)
```

`safety` *Example of safety data*

Description

A dataset containing some example data of safety event in raw source format

Usage

```
safety
```

Format

a data frame with 8 columns and 16 rows

pt meddra preferred term code

subjid a unique subject identifier

related a logical indicating if the event is related to the treatment

soc the meddra code for the System Organ Class

fatal a numerical 0/1 to indicate if the event was fatal

serious a numerical 0/1 to indicate if the event was serious

group the treatment group for the subject

term a text description of the event. Needs to be matching 1-1 with the pt code

Details

The data contains one row per patient-event. So the numbers exposed in each arm cannot be inferred from these data, as patients with no events will not be included in these data.

The variable names and formats are those required by `safety_summary`. The variable `pt` is not strictly required. An alternative to `soc` would be the equivalent character string from `soc_code`

`safety_summary` *Calculate frequency tables from a rectangular data frame with one row per subject-event*

Description

Calculate frequency tables from a rectangular data frame with one row per subject-event

Usage

```
safety_summary(
  data,
  exposed,
  excess_deaths = 0,
  freq_threshold = 0,
  soc_index = c("meddra", "soc_term"),
  na.action = na.fail
)
```

Arguments

`data`	a data set containing the following columns: `subjid`, `term`, `soc`, `serious`, `related`, `fatal`, `group`. See `safety` for more details.

`exposed`	a numeric vector giving the numbers of subjects exposed in each group. This needs to be supplied directly by the user, and cannot be inferred from the input `data` with one row per patient-event. To ensure the ordering is correct either, name the vector with names matching the values in `data$group`, or ensure that the data$group is an ordered factor, or relying on alphabetical ordering of the values in `data$group`
`excess_deaths`	a numeric vector giving the number of extra deaths not reported within `data`. Defaults to 0.
`freq_threshold`	a value on a percentage scale at which to remove events if the incidence falls below. Defaults to 0
`soc_index`	a character vector either "meddra" or "soc_term", which is used to identify if the soc variable in data gives the numerical meddra code or the description in English.
`na.action`	a function that indicates what should happen if the data contain missing values. The default is `na.fail` as both repositories will not accept any missing values in the upload. Alternatives could be `na.omit`, `na.exclude`, or `na.pass`.

Value

a list of three dataframes: GROUP, SERIOUS, NON_SERIOUS. Each contains the summary statistics required by EudraCT, and is suitable for export.

See Also

`eudract_convert` `simple_safety_xml`

Examples

```
safety_statistics <- safety_summary(safety,
                                    exposed=c("Experimental"=60,"Control"=67))
simple <- tempfile(fileext = ".xml")
eudract <- tempfile(fileext = ".xml")
ct <- tempfile(fileext = ".xml")
simple_safety_xml(safety_statistics, simple)
eudract_convert(input=simple,
                output=eudract)
clintrials_gov_convert(input=simple,
                       original=system.file("extdata", "1234.xml", package ="eudract"),
                output=ct)
## Not run:
  # This needs a real user account to work
  clintrials_gov_upload(
    input=simple,
    orgname="CTU",
    username="Student",
    password="Guinness",
    studyid="1234"
    )
```

```
## End(Not run)
```

`safety_summary_adam` *Calculate frequency tables based on ADaM data.*

Description

Calculate frequency tables based on ADaM data.

Usage

```
safety_summary_adam(
  adsl,
  adae,
  freq_threshold = 0,
  na.action = na.fail,
  related_terms = c("POSSIBLE", "PROBABLE", "DEFINITELY", "Y")
)
```

Arguments

`adsl`	ADaM Subject-level analysis data set
`adae`	ADaM Adverse Event data set
`freq_threshold`	a value on a percentage scale at which to remove events if the incidence falls below. Defaults to 0
`na.action`	a function that indicates what should happen if the data contain missing values. The default is `na.fail` as both repositories will not accept any missing values in the upload. Alternatives could be `na.omit`, `na.exclude`, or `na.pass`.
`related_terms`	The set of terms used in `adae$AEREL` to identify related events.

Details

It is worth highlighting that the CDISC standards need to be followed. There must be an `ARM` variable in the `adsl` data. Screen failures must be filtered out. It is up to the user to manipulate the input data as required.

Value

a list of three data frames: GROUP, SERIOUS, NON_SERIOUS. Each contains the summary statistics required by EudraCT, and is suitable for export.

See Also

`safety_summary` `eudract_convert` `simple_safety_xml`

Examples

```
safety_summary_adam(
 pharmaverseadam::adsl |> dplyr::filter(ARM != "Screen Failure"),
 pharmaverseadam::adae)
```

`simple_safety_xml` *creates a simple xml file from the input of a safety_summary object*

Description

creates a simple xml file from the input of a safety_summary object

Usage

```
simple_safety_xml(
  x,
  file,
  schema = system.file("extdata", "simple.xsd", package = "eudract")
)
```

Arguments

`x`	an object of class `safety_summary`, as created by `safety_summary`.
`file`	a character string name the file to be created
`schema`	a character string giving the file path to the schema for the output xml file. Defaults to the schema provided in this package.

Value

no output is returned, but a file is created as a side-effect.

See Also

`eudract_convert` `safety_summary`

Examples

```
safety_statistics <- safety_summary(safety,
                                    exposed=c("Experimental"=60,"Control"=67))
simple <- tempfile(fileext = ".xml")
eudract <- tempfile(fileext = ".xml")
ct <- tempfile(fileext = ".xml")
simple_safety_xml(safety_statistics, simple)
eudract_convert(input=simple,
                output=eudract)
clintrials_gov_convert(input=simple,
```

```
                        original=system.file("extdata", "1234.xml", package ="eudract"),
                 output=ct)
## Not run:
  # This needs a real user account to work
  clintrials_gov_upload(
    input=simple,
    orgname="CTU",
    username="Student",
    password="Guinness",
    studyid="1234"
    )


## End(Not run)
```

`soc_code` *System Organ Class coding*

Description

A dataset containing text descriptions and medDRA and EudraCT codes for each system organ class

Usage

```
soc_code
```

Format

a data frame with 3 columns and 27 rows

soc_term	a text description
eutctId	the eudraCT coding
meddra	the meddra code

Source

https://www.meddra.org/, https://spor.ema.europa.eu/rmswi/#/

Index