Package ‘admiralophtha’

June 12, 2023

Type Package

Title ADaM in R Asset Library - Ophthalmology

Version 0.2.0

Description Aids the programming of Clinical Data Standards Interchange Consortium (CDISC) compliant Ophthalmology Analysis Data Model (ADaM) datasets in R. ADaM datasets are a mandatory part of any New Drug or Biologics License Application submitted to the United States Food and Drug Administration (FDA). Analysis derivations are implemented in accordance with the “Analysis Data Model Implementation Guide” (CDISC Analysis Data Model Team, 2021, <https://www.cdisc.org/standards/foundational/adam/adamig-v1-3-release-package>).

License Apache License (>= 2)

BugReports https://github.com/pharmaverse/admiralophtha/issues

URL https://pharmaverse.github.io/admiralophtha/,
https://github.com/pharmaverse/admiralophtha/

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.3

Depends R (>= 3.5)

Imports admiral, admiraldev, dplyr, lubridate, magrittr, purrr, rlang, stringr, hms, tidy, tidyselect, lifecycle

Suggests admiral.test, devtools, diffdf, lintr, pkgdown, testthat, knitr, methods, miniUI, rmarkdown, roxygen2, spelling, tibble, usethis, covr, DT

VignetteBuilder knitr

NeedsCompilation no

Author Edoardo Mancini [aut, cre], Ritika Aggarwal [aut], Jane Gao [aut],
**Description**

An example Best Corrected Visual Acuity (BCVA) analysis dataset

**Usage**

```r
admiralophtha_adbcva
```

**Format**

An object of class `tbl_df` (inherits from `tbl.data.frame`) with 7672 rows and 116 columns.

**Source**

See Also

Other datasets: *admiralophtha_adoe, admiralophtha_advfq, admiralophtha_ex, admiralophtha_qs*

---

*admiralophtha_adoe*  
*Ophthalmology Exam Analysis Dataset*

**Description**

An example Ophthalmology Exam Analysis dataset

**Usage**

`admiralophtha_adoe`

**Format**

An object of class `tbl_df` (inherits from `tbl, data.frame`) with 7672 rows and 98 columns.

**Source**


See Also

Other datasets: *admiralophtha_adbcva, admiralophtha_advfq, admiralophtha_ex, admiralophtha_qs*

---

*admiralophtha_advfq*  
*Visual Function Questionnaire Analysis Dataset*

**Description**

An example Visual Function Questionnaire (VFQ) analysis dataset

**Usage**

`admiralophtha_advfq`

**Format**

An object of class `tbl_df` (inherits from `tbl, data.frame`) with 28798 rows and 41 columns.

**Source**

See Also

Other datasets: admiralophtha_adbcva, admiralophtha_adoe, admiralophtha_ex, admiralophtha_qs

---

**admiralophtha_ex**  
*Exposure Ophthalmology SDTM Dataset*

**Description**

An example Exposure SDTM dataset with ophthalmology-specific variables such as EXLOC and EXLAT

**Usage**

`admiralophtha_ex`

**Format**

An object of class `tbl_df` (inherits from `tbl, data.frame`) with 591 rows and 19 columns.

**Source**

Constructed using `ex` from the `{admiral}` package

---

See Also

Other datasets: admiralophtha_adbcva, admiralophtha_adoe, admiralophtha_advfq, admiralophtha_qs

---

**admiralophtha_qs**  
*Ophthalmology Questionnaires SDTM Dataset*

**Description**

An example Questionnaires SDTM dataset with ophthalmology-specific questionnaire of NEI VFQ-25

**Usage**

`admiralophtha_qs`

**Format**

An object of class `tbl_df` (inherits from `tbl, data.frame`) with 197671 rows and 20 columns.

**Source**

Constructed using `qs` from the `{admiral}` package
convert_etdrs_to_logmar

See Also

Other datasets: admiralophtha_adbcva, admiralophtha_adoe, admiralophtha_advfq, admiralophtha_ex

convert_etdrs_to_logmar

ETDRS → LogMAR conversion

Description

Convert ETDRS score to LogMAR units

Usage

convert_etdrs_to_logmar(value)

Arguments

value object containing ETDRS score to convert to logMAR

Details

ETDRS value converted to logMAR as logMAR = -0.02 * ETDRS + 1.7

Value

The input value converted converted to logMAR units

Author(s)

Rachel Linacre

Examples

library(tibble)
library(dplyr)
library(admiral)
library(admiraldev)

adbcva <- tribble(
  ~STUDYID, ~USUBJID, ~AVAL,
  "XXX001", "P01", 5,
  "XXX001", "P02", 10,
  "XXX001", "P03", 15,
  "XXX001", "P04", 20,
  "XXX001", "P05", 25
)

adbcva <- adbcva %>% mutate(AVAL = convert_etdrs_to_logmar(AVAL))
**convert_logmar_to_etdrs**

*LogMAR -> ETDRS conversion*

**Description**

Convert LogMAR score to ETDRS units

**Usage**

`convert_logmar_to_etdrs(value)`

**Arguments**

- `value`: object containing logMAR score to convert to ETDRS

**Details**

logMAR value converted to ETDRS as ETDRS = \(-\frac{(\logMAR - 1.7)}{0.02}\)

**Value**

The input value converted to ETDRS units

**Author(s)**

Nandini R Thampi

**Examples**

```r
library(tibble)
library(dplyr)
library(admiral)

oe <- tribble(~STUDYID, ~USUBJID, ~OETESTCD, ~OEMETHOD, ~OESTRESN,
   "XXX001", "P01", "VACSCORE", "logMAR EYE CHART", 1.08,
   "XXX001", "P02", "VACSCORE", "logMAR EYE CHART", 1.66,
   "XXX001", "P03", "VACSCORE", "logMAR EYE CHART", 1.60,
   "XXX001", "P04", "VACSCORE", "ETDRS EYE CHART", 57,
   "XXX001", "P05", "VACSCORE", "ETDRS EYE CHART", 1)

adbcva <- oe %>%
  filter(OETESTCD == "VACSCORE" & toupper(OEMETHOD) == "LOGMAR EYE CHART") %>%
  mutate(OESTRESN = convert_logmar_to_etdrs(OESTRESN))
```
Derive Affected Eye

**Description**

Derive Affected Eye (AFEYE) in occurrence datasets

**Usage**

```
derive_var_afeye(dataset_occ, loc_var, lat_var, loc_vals = "EYE")
```

**Arguments**

- `dataset_occ`: Input Occurrence dataset
- `loc_var`: Location variable
- `lat_var`: Laterality variable
- `loc_vals`: xxLOC values for which AFEYE is derived

**Details**

Affected Eye is derived in the occurrence dataset using laterality and Study Eye. This assumes Study Eye has already been added from ADSL.

**Value**

The input occurrence dataset with Affected Eye (AFEYE) added.

**Author(s)**

Lucy Palmen

**Examples**

```
library(tibble)
library(admiral)

adae <- tribble(
  ~STUDYID, ~USUBJID, ~STUDYEYE, ~AELOC, ~AELAT,
  "XXX001", "P01", "RIGHT", "EYE", "RIGHT",
  "XXX001", "P01", "RIGHT", "EYE", "LEFT",
  "XXX001", "P01", "RIGHT", "EYE", "",
  "XXX001", "P01", "RIGHT", "", "RIGHT",
  "XXX001", "P02", "LEFT", "", "",
  "XXX001", "P02", "LEFT", "EYE", "LEFT",
  "XXX001", "P04", "BILATERAL", "EYE", "RIGHT",
  "XXX001", "P05", "RIGHT", "EYE", "RIGHT",
  "XXX001", "P05", "RIGHT", "EYE", "BILATERAL",
  "XXX001", "P06", "BILATERAL", "", ""
)
`derive_var_bcvacritxfl`

`derive_var_bcvacritxfl`  
*Adds CRITx CRITxFL pairs to ADBCVA dataset*

**Description**

Adds a criterion variables CRITx and their corresponding flags CRITxFL to an ADBCVA dataset.

**Usage**

```r
derive_var_bcvacritxfl(
  dataset_adbcva,  # Input dataset (ADBCVA).
  paramcds = NULL,  # Vector of PARAMCD values for which to derive CRITx and CRITxFL.
  basetype = NULL,  # BASETYPE value for which to derive CRITx and CRITxFL.
  bcva_ranges = NULL,  # List containing one or more numeric vectors of length 2. For each vector c(a,b) in bcva_ranges, a pair of variables CRITx, CRITxFL is created with the condition: a <= CHG <= b. If criterion flags of that type are not required, then leave as NULL.
  bcva_uplims = NULL,  # List containing one or more numeric elements. For each element a in bcva_uplims, a pair of variables CRITx, CRITxFL is created with the condition: CHG <= a. If criterion flags of that type are not required, then leave as NULL.
  bcva_lowlims = NULL,  # List containing one or more numeric elements. For each element b in bcva_lowlims, a pair of variables CRITx, CRITxFL is created with the condition: CHG >= b. If criterion flags of that type are not required, then leave as NULL.
  additional_text = "",  # string containing additional text to append to CRITx
  critxfl_index = NULL  # positive integer detailing the first value of x to use in "CRITxFL". If not supplied, the function takes the first available value of x, counting up from x = 1.
)
```

**Arguments**

- `dataset_adbcva`: Input dataset (ADBCVA).
- `paramcds`: Vector of PARAMCD values for which to derive CRITx and CRITxFL.
- `basetype`: BASETYPE value for which to derive CRITx and CRITxFL.
- `bcva_ranges`: List containing one or more numeric vectors of length 2. For each vector c(a,b) in bcva_ranges, a pair of variables CRITx, CRITxFL is created with the condition: a <= CHG <= b. If criterion flags of that type are not required, then leave as NULL.
- `bcva_uplims`: List containing one or more numeric elements. For each element a in bcva_uplims, a pair of variables CRITx, CRITxFL is created with the condition: CHG <= a. If criterion flags of that type are not required, then leave as NULL.
- `bcva_lowlims`: List containing one or more numeric elements. For each element b in bcva_lowlims, a pair of variables CRITx, CRITxFL is created with the condition: CHG >= b. If criterion flags of that type are not required, then leave as NULL.
- `additional_text`: string containing additional text to append to CRITx
- `critxfl_index`: positive integer detailing the first value of x to use in "CRITxFL". If not supplied, the function takes the first available value of x, counting up from x = 1.
derive_var_bcvacritxfl

Details

This function works by calling derive_var_bcvacritxfl once for each of the elements in bcva_ranges, bcva_uplims and bcva_lowlims. NOTE: if CHG is equal to NA, then the resulting criterion flag is also marked as NA.

Value

The input ADBCVA dataset with additional column pairs CRITx, CRITxFL.

Author(s)

Edoardo Mancini

Examples

library(tibble)
library(admiral)
library(admiraldev)


derive_var_bcvacritxfl(dataset_adbcva = adbcva1, paramcds = c("SBCVA", "FBCVA"), basetype = NULL, bcva_ranges = list(c(0, 5), c(-5, -1), c(10, 15)), bcva_uplims = list(5, 10), bcva_lowlims = list(8), additional_text = "")

derive_var_studyeye

Description
Derive Study Eye (STUDYEYE) in the ADSL dataset

Usage
derive_var_studyeye(dataset_adsl, dataset_sc, sctestcd_value = "FOCID")

Arguments
dataset_adsl  ADSL input dataset
dataset_sc   SC input dataset
sctestcd_value  SCTESTCD value flagging Study Eye selection records. Default: "FOCID".

Details
Study Eye is derived in ADSL using the "Study Eye selection" records in the SC SDTM dataset.

Value
The input ADSL dataset with an additional column named STUDYEYE

Author(s)
Edoardo Mancini
Examples

```r
library(tibble)
library(admiral)

adsl <- tribble(~STUDYID, ~USUBJID, 
                 "XXX001", "P01", 
                 "XXX001", "P02", 
                 "XXX001", "P03", 
                 "XXX001", "P04", 
                 "XXX001", "P05"
)

sc <- tribble(~STUDYID, ~USUBJID, ~SCTESTCD, ~SCSTRESC, 
               "XXX001", "P01", "FOCID", "OS", 
               "XXX001", "P02", "ACOHORT", "COHORT1", 
               "XXX001", "P02", "FOCID", "OD", 
               "XXX001", "P04", "ACOHORT", "COHORT3", 
               "XXX001", "P05", "FOCID", "OU", 
               "XXX001", "P06", "FOCID", "OS"
)

derive_var_studyeye(adsl, sc)
```
Index

* datasets
  admiralophtha_adbcva, 2
  admiralophtha_adoe, 3
  admiralophtha_advfq, 3
  admiralophtha_ex, 4
  admiralophtha_qs, 4
* der_ads
  derive_var_studyeye, 10
* der_occds
  derive_var_afeye, 7
* der_ophtha
  derive_var_bcvacritxfl, 8
* utils_fmt
  convert_etdrs_to_logmar, 5
  convert_logmar_to_etdrs, 6

admiralophtha_adbcva, 2, 3–5
admiralophtha_adoe, 3, 3, 4, 5
admiralophtha_advfq, 3, 3, 4, 5
admiralophtha_ex, 3, 4, 4, 5
admiralophtha_qs, 3, 4, 4

convert_etdrs_to_logmar, 5
convert_logmar_to_etdrs, 6

derive_var_afeye, 7
derive_var_bcvacritxfl, 8
derive_var_studyeye, 10