Package 'WinRatio'

July 21, 2025

Type Package

Title Win Ratio for Prioritized Outcomes and 95% Confidence Interval

Version 1.0

Date 2020-11-16

Author Kevin Duarte,
Joao Pedro Ferreira

Maintainer Kevin Duarte <k.duarte@chru-nancy.fr>

Description Calculate the win ratio for prioritized outcomes and the 95% confidence interval based on Bebu and Lachin (2016) <doi:10.1093/biostatistics/kxv032>. Three type of outcomes can be analyzed: survival ``failure-time'' events, repeated survival ``failure-time'' events and continuous or ordinal ``non-failure time'' events that are captured at specific time-points in the study.

License GPL (>= 2)

Depends R (>= 4.0.0), tidyverse (>= 1.3.0)

Imports Rcpp (>= 1.0.5), dplyr, magrittr, tidyselect

Suggests survival

LinkingTo Rcpp

RoxygenNote 7.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-11-23 08:10:02 UTC

Contents

WinRatio-package   *Win Ratio for Prioritized Outcomes and 95% Confidence Interval*

Description

Calculate the win ratio for prioritized outcomes and the 95% confidence interval based on Bebu and Lachin (2016). Three type of outcomes can be analyzed: survival "failure-time" events, repeated survival "failure-time" events and continuous or ordinal "non-failure time" events that are captured at specific time-points in the study.

Author(s)

Kevin Duarte, Joao Pedro Ferreira

Maintainer: Kevin Duarte <k.duarte@chru-nancy.fr>

References

Pocock S.J., Ariti C.A., Collier T.J., Wang D. The win ratio: a new approach to the analysis of composite endpoints in clinical trials based on clinical priorities. *Eur Heart Journal*. 2012; **33**:176-182. DOI: 10.1093/eurheartj/ehr352.

Bebu I. and Lachin J.M. Large sample inference for a win ratio analysis of a composite outcome based on prioritized components. *Biostatistics*. 2016; **17**:178-187. DOI: 10.1093/biostatistics/kxv032.

---

is.binary   *Binary Vectors*

Description

is.binary is used to test if a vector is of binary type, i.e. if a vector contains only 0 and/or 1 values.

Usage

```
is.binary(x)
```

Arguments

x   a vector to be tested.

Value

is.binary returns TRUE or FALSE depending on whether a vector is of binary type or not.

Examples

```
x1 <- runif(100, 0, 1)
x2 <- ifelse(x1 < 0.5, 1, 0)
is.binary(x1)
is.binary(x2)
```

---

summary.WinRatio   *Summarizing Win Ratio Results*

Description

This function is used for summarizing the results for class `'WinRatio'` objects.

Usage

```
## S3 method for class 'WinRatio'
summary(object, ..., digits = 2)
```

Arguments

object	an object of class `'WinRatio'`.
...	additional arguments affecting the summary produced.
digits	the number of significant digits to use when printing win ratio and 95% confidence interval.

Examples

```
## For examples see example(winratio)
```

---

winratio   *Win Ratio for Prioritized Outcomes and 95% Confidence Interval*

Description

Calculate the win ratio for prioritized outcomes and the 95% confidence interval based on Bebu and Lachin (2016). Three type of outcomes can be analyzed: survival "failure-time" events, repeated survival "failure-time" events and continuous or ordinal "non-failure time" events that are captured at specific time-points in the study.

Usage

```
winratio(id, trt, active = NULL, outcomes, fu, data, keep.matrix = FALSE)
```

Arguments

id	a string indicating the patient ID variable. The patient ID variable must not contain missing values or duplicates.
trt	a string indicating the treatment variable. The variable can be of any type (binary, numeric, character, factor) and must imperatively have 2 unique values/levels.
active	a numeric or string value used to define the active treatment group for the calculation of the win ratio. If active = NULL (default value), active group is automatically determined according to the type of treatment variable:

- binary variable: active group = 1.
- numeric variable: active group = maximum value.
- character variable: active group = last string after sorting the character variable in ascending order.
- factor variable: active group = last level.

outcomes	a list used to define all outcomes in order of priority. The first element to be defined must be the outcome considered as being of greater clinical importance, usually this outcome is a fatal event. Each element must be a character vector of length 3 or a character list of length 3 depending on the type of outcome (survival event, repeated survival event, continuous or ordinal event):

- for survival events, the element is a character vector c(x, "s", y), where x is a string indicating the name of the event variable, "s" indicating that this outcome is a "failure-time" event, and y is a string indicating the name of the time-to-event variable.
- for repeated survival events, note k the maximum number of events observed during follow-up, the element is a character list list(x, "r", y), where x = c(x1, ..., xk) is a character vector of length k indicating the name of event variables, "r" indicating that this outcome is a "failure-time" repeated event, and y = c(y1, ..., yk) is a character vector of length k indicating the name of time-to-event variables corresponding to the event variables.
- for continuous or ordinal events, the element is a character vector c(x, "c", d), where x is a string indicating the name of the continuous or ordinal variable, "c" indicating that this outcome is a continuous or ordinal event, and d is a string with two options "<" or ">" indicating the direction of deterioration or worsening.

fu	a string indicating the name of the follow-up time variable.
data	a data frame containing all the variables listed in the id, trt, outcomes and fu arguments.
keep.matrix	a logical value indicating if the 'win-loss' matrix is kept. Default value is FALSE.

Details

'win-loss' matrix: Let n1 the number of patients in the active group and n0 the number of patients in the other group. The 'win-loss' matrix is a matrix M of dimension n1 x n0 with its element M[i,j] is defined as:

   `M[i,j] = 1` if i wins against j on 1st outcome,

   `M[i,j] = -1` if i loses against j on 1st outcome,

   `M[i,j] = k` if i wins on outcome k after ties with j on outcomes 1 to (k-1),

   `M[i,j] = -k` if i loses on outcome k after ties with j on outcomes 1 to (k-1),

   `M[i,j] = 0` if i and j ties on all outcomes.

Value

`call`	a list with all arguments entered `list(id, trt, active, outcomes, fu, data,` `keep.matrix)`.
`group1`	the value/level of the active group (group 1).
`group0`	the value/level of the other group (group 0).
`n1`	the number of subjects in group 1.
`n0`	the number of subjects in group 0.
`n`	the total number of subjects.
`wins`	a vector of numbers of 'winners' for each outcome.
`loss`	a vector of numbers of 'losers' for each outcome.
`total.wins`	the total number of 'winners'.
`total.loss`	the total number of 'losers'.
`total.ties`	the total number of ties.
`wr`	the win ratio.
`v`	the estimated variance of win ratio.
`z`	the value of the test statistic.
`p.value`	the p-value of the test.
`wr.lower`	the lower end of the 95% confidence interval.
`wr.upper`	the upper end of the 95% confidence interval.
`wr.matrix`	the 'win-loss' matrix (only if `keep.matrix = T`). See details above.

Examples

```
# Load survival package to use the dataset 'bladder1'
library(survival)

# Creation of dataset 'df' with 3 outcomes:
#   Outcome 1: death (survival event)
#   Outcome 2: cancer recurrence (repeated survival event)
#   Outcome 3: size of largest initial tumour (continuous event)
data1 <- bladder1  %>%
  mutate(trt = if_else(treatment == "placebo", "Placebo", "Treatment")) %>%
  group_by(id) %>%
  mutate(death = if_else(max(status) %in% c(2, 3), 1, 0),
         t2death = max(stop)) %>%
  ungroup() %>%
  select(id, trt, death, t2death, number, size) %>%
```

```
  unique()

data2 <- bladder1 %>%
  filter(status == 1) %>%
  select(id, t2recurr = stop) %>%
  mutate(recurr = 1) %>%
  arrange(id, t2recurr) %>%
  group_by(id) %>%
  mutate(nrecurr = row_number()) %>%
  ungroup() %>%
  full_join((data1 %>% select(id)), by = "id") %>%
  complete(id, nrecurr) %>%
  filter(!is.na(nrecurr)) %>%
  full_join((data1 %>% select(id, t2death)), by = "id") %>%
  mutate(
    recurr = replace(recurr, is.na(recurr), 0),
    t2recurr = if_else(is.na(t2recurr), t2death, t2recurr)
  ) %>%
  select(id, nrecurr, recurr, t2recurr)

data3 <- data2 %>%
  pivot_wider(id_cols = "id", names_from =  nrecurr,
              values_from = recurr, names_prefix  = "recurr")

data4 <- data2 %>%
  pivot_wider(id_cols = "id", names_from =  nrecurr,
              values_from = t2recurr, names_prefix  = "t2recurr")

data5 <- full_join(data3, data4, by = "id")

df <- full_join(data1, data5, by = "id")

# Calculate the win ratio
wr <- winratio(id = "id", trt = "trt", active = "Treatment",
               outcomes = list(outc1 = c("death", "s", "t2death"),
                       outc2 = list(paste0("recurr", 1:9), "r", paste0("t2recurr", 1:9)),
                             outc3 = c("size", "c", ">")),
               fu = "t2death", data = df)
summary(wr)
```

Index