Package: kpitools (via r-universe)

October 15, 2024

Title Tools for creating key performance indicator reports for clinical trial

Version 0.2.3

Description Assessing performance of clinical trials can assist identify problems earlier in the trial than might be possible without it and help to improve trial quality. Tools for the creating performance indicator reports are however uncommon. 'kpitools' aims to provide tools to create such reports.

License GPL (>= 3)

Encoding UTF-8

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Suggests knitr, lubridate, markdown, patchwork, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends dplyr, ggplot2, magrittr, purrr, rlang

Imports stringr, tidyr

VignetteBuilder knitr

Repository https://ctu-bern.r-universe.dev

RemoteUrl https://github.com/CTU-Bern/kpitools

RemoteRef HEAD

RemoteSha ce22b8964c968799eab1c0352e983369bb7cf8a1

Contents

`as.kpilist` *Convert a list to a kpilist*

Description

Convert a list to a kpilist

Usage

```
as.kpilist(x)
```

Arguments

`x`	list of kpi objects

Value

a kpilist

Examples

```
l <- lapply(c("drat", "hp", "qsec"), function(x){
 kpi(mtcars,
     var = x,
     by = c("am", "cyl"),
     kpi_fn = kpi_fn_median)
})
as.kpilist(l)
```

c.kpi *Concatenate* kpi *objects*

Description

Concatenate kpi objects

Usage

```
## S3 method for class 'kpi'
c(...)
```

Arguments

...	kpi or kpilist objects

Value

kpilist object

Examples

```
kpi1 <- mtcars %>%
  kpi(var = "mpg", by = c("am", "cyl"), txt = "MPG",
      kpi_fn = kpi_fn_median)
kpi2 <- mtcars %>%
  kpi(var = "drat", by = c("am", "cyl"), txt = "DRAT",
      kpi_fn = kpi_fn_median)
l <- c(kpi1, kpi2)
kpi3 <- mtcars %>%
  mutate(cylgt4 = cyl > 4) %>%
  kpi(var = "cylgt4", by = c("am", "cyl"), txt = "Cylinders",
      kpi_fn = kpi_fn_perc)
l2 <- c(l, kpi3)
```

fab_dow *Day of week figure(s)*

Description

In a normal setting it may be that observations that occur at the weekend are indicative of data fabrication. fab_dow (short for fabrication, day of week), produces a plot that may help to identify problems. Customs vary in different countries, so that should be accounted for when interpreting these figures.

Usage

```
fab_dow(
  data,
  var,
  by = NULL,
  dow_fmt = "%a",
  output = c("facet", "list"),
  col = "grey",
  fill = "grey",
  ...
)
```

Arguments

data	cdata frame containing var (and, optionally, by) variable(s)
var	string. Name of variable containing relevant dates or datetimes (will be coerced to date via as.Date)
by	string. Name of variable denoting grouping
dow_fmt	format for day of week
output	output format facet combines figures via ggplot2::facet_wrap, list returns a list of ggplot2 plots
col	colour to use for bar lines
fill	colour to use for bar fill
...	options passed to facet_wrap (see examples)

Value

list or ggplot2 object

Examples

```
set.seed(234)
dat <- data.frame(
  x = Sys.Date() + sample(-20:19, 40, TRUE),
  by = c(rep(1, 10), rep(2, 30))
)
dat %>% fab_dow("x")
dat %>% fab_dow("x", "by")
# free x scale
dat %>% fab_dow("x", "by", scales = "free_x")
# different colour bars
dat %>% fab_dow("x", fill = "orange")
# list of plots
dat %>% fab_dow("x", "by", output = "list")
# change colours
dat %>% fab_dow("x", col = "purple", fill = "pink")
```

`fab_tod` *Time of day figure(s)*

Description

In a normal setting it may be that observations that occur at night are indicative of data fabrication. `fab_tod` (short for fabrication, time of day), produces a plot that may help to identify problems. Customs vary in different countries, so that should be accounted for when interpreting these figures.

Usage

```
fab_tod(
  data,
  var,
  by = NULL,
  dow_fmt = "%a",
  output = c("list", "facet"),
  col_poly = "black",
  x_poly = c(8.5, 21.5),
  col_bars = "grey"
)
```

Arguments

`data`	data frame containing `var` (and, optionally, `by`) variable(s)
`var`	string. Name of variable containing relevant datetimes
`by`	string. Name of variable denoting grouping
`dow_fmt`	format for day of week
`output`	output format `facet` combines figures via ggplot2::facet_wrap, `list` returns a list of ggplot2 plots
`col_poly`	colour to use for the region indicating possible fabrication
`x_poly`	x coordinates for the start and end of the region indicating possible fabrication
`col_bars`	colour to use for bars indicating counts

Details

Due to a limitation of faceting plots with polar coordinates, faceted plots all have the same y coordinate (equivalent to fixed axes). To free the coordinate system, use the list output (default) and wrap them together using e.g. patchwork, possibly applying some customizations in advance.

Value

list or ggplot2 object

Examples

```
set.seed(234)
dat <- data.frame(
  x = lubridate::ymd_h("2020-05-01 14") + 60^2*sample(0:20, 40, TRUE),
  by = c(rep(1, 10), rep(2, 30))
)
dat %>% fab_tod("x")
dat %>% fab_tod("x") + theme_kpitools()
dat %>% fab_tod("x", "by")
#faceted of plots
dat %>% fab_tod("x", "by", output = "facet")
#with patchwork
patchwork::wrap_plots(dat %>% fab_tod("x", "by"))
```

---

kpi *Create KPI tables*

Description

Create KPI tables

Usage

```
kpi(
  data,
  var,
  by = NULL,
  kpi_fn = kpi_fn_mean,
  txt = "",
  n_iqr = 2,
  breakpoints = NULL,
  risklabels = risklabs(breakpoints),
  riskcolors = riskcols(breakpoints),
  direction = c("increasing", "decreasing"),
  raw_cut = FALSE,
  keep_data = FALSE
)
```

Arguments

data	a data frame
var	the variable to summarize
by	optional variable(s) to group over
kpi_fn	summary function
txt	a descriptive text

n_iqr	number of IQRs below/above the lower/upper quartiles that should be considered outliers
breakpoints	cut points (if KPIs use a traffic light system)
risklabels	labels for the cut points. By default, variations on low/moderate/high are used
riskcolors	colors for the cut points. By default, variations on green/yellow/red are used
direction	seriousness relative to breakpoints
raw_cut	add a group variable without applying risklabels
keep_data	keep raw data or not

Value

a list with either 1 or (length(by) + 1) lists.

Examples

```
kpi_test <- mtcars %>%
  mutate(cylgt4 = cyl > 4) %>%
  kpi(var = "mpg",
      breakpoints = c(0, 22, 50),
      by = c("am", "cyl"),
      txt = "MPG",
      kpi_fn = kpi_fn_median)
```

---

kpi_accumulate

Accumulate kpilists into KPIs per site lists The KPIs themselves are all well and good for e.g. a report where you walk through each individual KPI and present all of the info there, but they're not ideal if you want all of the KPIs for a given site or country or the overall study in a single table. kpi_accumulate *does this conversion*

Description

Accumulate kpilists into KPIs per site lists The KPIs themselves are all well and good for e.g. a report where you walk through each individual KPI and present all of the info there, but they're not ideal if you want all of the KPIs for a given site or country or the overall study in a single table. kpi_accumulate does this conversion

Usage

```
kpi_accumulate(kpilist, by = NULL, split = TRUE)
```

Arguments

kpilist	list of KPIs
by	which by variable from the kpi call to accumulate
split	logical. Whether to split the output by the levels of the by variable(s)

Examples

```
kpi1 <- mtcars %>%
  kpi(var = "mpg", by = c("am", "cyl"), txt = "MPG",
      kpi_fn = kpi_fn_median, breakpoints = c(0, 20, 30, 50))
kpi2 <- mtcars %>%
  kpi(var = "drat", by = c("am", "cyl"), txt = "DRAT",
      kpi_fn = kpi_fn_median, breakpoints = c(0, 3, 4, 50))
l <- c(kpi1, kpi2)
kpi3 <- mtcars %>%
  mutate(cylgt4 = cyl > 4) %>%
  kpi(var = "cylgt4", by = c("am", "cyl"), txt = "Cylinders",
      kpi_fn = kpi_fn_perc, , breakpoints = c(0, 30, 50, 100))
l2 <- c(l, kpi3)
kpi_accumulate(l2)
# only the cyl level
kpi_accumulate(l2, by = "cyl")
# only the study/overall level
kpi_accumulate(l2, by = "overall")
# no splitting
kpi_accumulate(l2, split = FALSE)
```

---

kpi_fns *Get a list of KPI summary functions provided by* kpitools.

Description

Get a list of KPI summary functions provided by kpitools.

Usage

```
kpi_fns()
```

Value

character vector of functions

See Also

kpi_fn_

Examples

```
kpi_fns()
```

kpi_fn_n *KPI summary functions*

Description

These functions are not intended to be run as they are. They are intended to be passed as arguments to the kpi or kpi_calc functions. They summarize the data in the appropriate manner for the type of KPI. For example, the kpi_fn_prop counts the number of cases and total number of observations then calculates a proportion. kpi_fn_median simply calculates the median of the observations.

Usage

```
kpi_fn_n(.data)

kpi_fn_prop(.data)

kpi_fn_perc(.data)

kpi_fn_median(.data)

kpi_fn_mean(.data)

kpi_fn_iqr(.data)

kpi_fn_min(.data)

kpi_fn_max(.data)

kpi_fn_missing(.data)
```

Arguments

.data	data frame

Details

Functions should accept a dataframe with a var variable and return a dataframe with stat (other variables are optional, although an N variable allows for compatibility with downstream functions). All provided functions return stat, n_nonmiss and N, with some also returning n.

See the examples passing custom functions.

Examples

```
# mean
kpi(mtcars, "mpg", kpi_fn = kpi_fn_mean)
# median
kpi(mtcars, "mpg", kpi_fn = kpi_fn_median)
```

```
# interquartile range
kpi(mtcars, "mpg", kpi_fn = kpi_fn_iqr)
# minimum
kpi(mtcars, "mpg", kpi_fn = kpi_fn_min)
# maximum
kpi(mtcars, "mpg", kpi_fn = kpi_fn_max)
# proportion
kpi(mtcars, "am", kpi_fn = kpi_fn_prop)
# percentage
kpi(mtcars, "am", kpi_fn = kpi_fn_perc)
# number/sum
kpi(mtcars, "am", kpi_fn = kpi_fn_n)
```

`kpi_outlier` *Get the outliers*

Description

Get the outliers

Usage

```
kpi_outlier(kpitab, n_iqr = 2)
```

Arguments

`kpitab`	result from calc_kpi
`n_iqr`	number of IQRs below/above the lower/upper quartiles that should be considered outliers

Value

`kpitab` with just the outliers

Examples

```
# data(mtcars)
# mtcars %>%
#   kpi_calc("mpg", by = "am", kpi_fn = kpi_fn_median) %>%
#   kpi_outlier()
```

plot.kpi *Plot KPI objects*

Description

Plot KPI objects

Usage

```
## S3 method for class 'kpi'
plot(x, y = 1, col = "#E6002EFF", pch = 21, ...)
```

Arguments

x	result from kpi
y	ignored
col	colour for points
pch	point character
...	for possible future expansion

Value

list of ggplot objects

Examples

```
# defaults
kpi <- mtcars %>%
  kpi("mpg", by = c("am", "vs"), txt = "MPG")
plot(kpi)

# customizing the plots
plots <- plot(kpi)

plots$am +
  theme_bw() +
  labs(title = "Foo")
```

print.kpi *Print method for kpi objects*

Description

Print method for kpi objects

Usage

```
## S3 method for class 'kpi'
print(x, table = TRUE, outlier = TRUE, ...)
```

Arguments

x	kpi object
table	logical, whether to add a table stats by grouping variable(s) to the output
outlier	logical, whether to add a table of outliers to the output
...	not currently used

Value

output printed to the console

Examples

```
kpi <- mtcars %>%
  mutate(cylgt4 = cyl > 4) %>%
  kpi(var = "mpg", breakpoints = c(0, 22, 50), by = c("am", "cyl"), txt = "MPG",
      kpi_fn = kpi_fn_median)
print(kpi, table = TRUE, outlier = FALSE)
```

riskcols *Colors for KPIs cutoffs*

Description

Colors for KPIs cutoffs

Usage

```
riskcols(x)
```

Arguments

x	breakpoints

Value

string of `length(x) - 1` with suitable colors.

Examples

```
riskcols(1:4)
```

---

risklabs *Labels for KPIs with cutoffs*

Description

Labels for KPIs with cutoffs

Usage

```
risklabs(x)
```

Arguments

`x`	breakpoints

Value

string of `length(x) - 1` with suitable labels.

Examples

```
risklabs(1:4)
```

---

theme_kpitools *kpitools* `ggplot2` *theme*

Description

Theme based on `theme_bw` and removing y-axis and moving the legend to beneath the plot.

Usage

```
theme_kpitools()
```

Value

`ggplot2` theme object

Examples

```
kpi <- mtcars %>%
  kpi("mpg", by = "cyl", txt = "MPG")

# without the theme
plot(kpi)$cyl
# with the theme
plot(kpi)$cyl +
  theme_kpitools()
```

Index