Package 'desirability2'

May 11, 2023

Title Desirability Functions for Multiparameter Optimization

Version 0.0.1

Description In-line functions for multivariate optimization via desirability functions (Derringer and Suich, 1980, <doi:10.1080/00224065.1980.11980968>) with easy use within 'dplyr' pipelines.

License MIT + file LICENSE

URL https://desirability2.tidymodels.org, https://github.com/tidymodels/desirability2

Depends R (>= 2.10)

Imports glue, purrr, rlang, stats, tibble

Suggests covr, dplyr, ggplot2, spelling, testthat (>= 3.0.0)

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.3

NeedsCompilation no

Author Max Kuhn [aut, cre] (<https://orcid.org/0000-0003-2402-136X>), Posit Software, PBC [cph, fnd]

Maintainer Max Kuhn <max@posit.co>

Repository CRAN

Date/Publication 2023-05-11 09:10:02 UTC

R topics documented:

classification_results *Classification results*

Description

These data are a variation of a case study at `tidymodels.org` where a penalized regression model was used for a binary classification task. The outcome metrics in `classification_results` are the areas under the ROC and PR curve, log-likelihood, and the number of predictors selected for a given amount of penalization. Two tuning parameters, `mixture` and `penalty`, were varied across 300 conditions.

Value

`classification_results`
a tibble

Source

See the `example-data` directory in the package with code that is a variation of the analysis shown at https://www.tidymodels.org/start/case-study/.

Examples

```
data(classification_results)
```

d_overall *Determine overall desirability*

Description

Once desirability columns have been created, determine the overall desirability using a mean (geometric by default).

Usage

```
d_overall(..., geometric = TRUE, tolerance = 0)
```

Arguments

...	One or more unquoted expressions separated by commas. To choose multiple columns using selectors, dplyr::across() can be used (see the example below).
geometric	A logical for whether the geometric or arithmetic mean should be used to summarize the columns.
tolerance	A numeric value where values strictly less than this value are capped at the value. For example, if users wish to use the geometric mean without completely excluding settings, a value greater than zero can be used.

Value

A numeric vector.

See Also

d_max()

Examples

```
library(dplyr)

# Choose model tuning parameters that minimize the number of predictors used
# while maximizing the area under the ROC curve.

classification_results %>%
  mutate(
    d_feat = d_min(num_features, 1, 200),
    d_roc  = d_max(roc_auc, 0.5, 0.9),
    d_all  = d_overall(across(starts_with("d_")))
  ) %>%
  arrange(desc(d_all))

# Bias the ranking toward minimizing features by using a larger scale.

classification_results %>%
  mutate(
    d_feat = d_min(num_features, 1, 200, scale = 3),
    d_roc  = d_max(roc_auc, 0.5, 0.9),
    d_all  = d_overall(across(starts_with("d_")))
  ) %>%
  arrange(desc(d_all))
```

`inline_desirability` *Desirability functions for in-line computations*

Description

Desirability functions map some input to a `[0, 1]` scale where zero is unacceptable and one is most desirable. The mapping depends on the situation. For example, `d_max()` increases desirability with the input while `d_min()` does the opposite. See the plots in the examples to see more examples.

Currently, only the desirability functions defined by Derringer and Suich (1980) are implemented.

Usage

```
d_max(x, low, high, scale = 1, missing = NA_real_, use_data = FALSE)

d_min(x, low, high, scale = 1, missing = NA_real_, use_data = FALSE)

d_target(
  x,
  low,
  target,
  high,
  scale_low = 1,
  scale_high = 1,
  missing = NA_real_,
  use_data = FALSE
)

d_box(x, low, high, missing = NA_real_, use_data = FALSE)

d_custom(x, x_vals, desirability, missing = NA_real_)

d_category(x, categories, missing = NA_real_)
```

Arguments

`x`	A vector of data to compute the desirability function
`low, high, target`	Single numeric values that define the active ranges of desirability.
`scale, scale_low, scale_high`	A single numeric value to rescale the desirability function (each should be great than 0.0). Values >1.0 make the desirability more difficult to satisfy while smaller values make it easier (see the examples below). `scale_low` and `scale_high` do the same for target functions with `scale_low` affecting the range below the `target` value and `scale_high` affecting values greater than `target`.
`missing`	A single numeric value on `[0, 1]` (or `NA_real_`) that defines how missing values in `x` are mapped to the desirability score.

use_data — Should the low, middle, and/or high values be derived from the data (x) using the minimum, maximum, or median (respectively)?

x_vals, desirability — Numeric vectors of the same length that define the desirability results at specific values of x. Values below and above the data in x_vals are given values of zero and one, respectively.

categories — A named list of desirability values that match all possible categories to specific desirability values. Data that are not included in categories are given the value in missing.

Details

Each function translates the values to desirability on [0, 1].

Equations:

Maximization:

- data > high: d = 1.0
- data < low: d = 0.0
- low <= data <= high: $d = \left(\frac{data-low}{high-low}\right)^{scale}$

Minimization:

- data > high: d = 0.0
- data < low: d = 1.0
- low <= data <= high: $d = \left(\frac{data=low}{low-high}\right)^{scale}$

Target:

- data > high: d = 0.0
- data < low: d = 0.0
- low <= data <= target: $d = \left(\frac{data-low}{target-low}\right)^{scale\_low}$
- target <= data <= high: $d = \left(\frac{data-high}{target-high}\right)^{scale\_high}$

Minimization:

- data > high: d = 0.0
- data < low: d = 0.0
- low <= data <= high:d = 1.0

Categories:

- data = level: d = 1.0
- data != level: d = 0.0

Custom:

For the sequence of values given to the function, d_custom() will return the desirability values that correspond to data matching values in x_vals. Otherwise, linear interpolation is used for values in-between.

Data-Based Values:

By default, most of the d_*() functions require specific user inputs for arguments such as low, target and high. When use_data = TRUE, the functions can use the minimum, median, and maximum values of the existing data to estimate those values (respectively) but *only when users do not specify them*.

Value

A numeric vector on `[0, 1]` where larger values are more desirable.

References

Derringer, G. and Suich, R. (1980), Simultaneous Optimization of Several Response Variables. *Journal of Quality Technology*, 12, 214-219.

See Also

`d_overall()`

Examples

```
library(dplyr)
library(ggplot2)

set.seed(1)
dat <- tibble(x = sort(runif(30)), y = sort(runif(30)))
d_max(dat$x[1:10], 0.1, 0.75)

dat %>%
  mutate(d_x = d_max(x, 0.1, 0.75))

set.seed(2)
tibble(z = sort(runif(100))) %>%
  mutate(
    no_scale = d_max(z, 0.1, 0.75),
    easier   = d_max(z, 0.1, 0.75, scale = 1/2)
  ) %>%
  ggplot(aes(x = z)) +
  geom_point(aes(y = no_scale)) +
  geom_line(aes(y = no_scale), alpha = .5) +
  geom_point(aes(y = easier), col = "blue") +
  geom_line(aes(y = easier), col = "blue", alpha = .5) +
  lims(x = 0:1, y = 0:1) +
  coord_fixed() +
  ylab("Desirability")

# ------------------------------------------------------------------------------
# Target example

dat %>%
  mutate(
    triangle = d_target(x, 0.1, 0.5, 0.9, scale_low = 2, scale_high = 1/2)
  ) %>%
  ggplot(aes(x = x, y = triangle)) +
  geom_point() +
  geom_line(alpha = .5) +
  lims(x = 0:1, y = 0:1) +
  coord_fixed() +
```

```
  ylab("Desirability")

# ------------------------------------------------------------------------------
# Box constraints

dat %>%
  mutate(box = d_box(x, 1/4, 3/4)) %>%
  ggplot(aes(x = x, y = box)) +
  geom_point() +
  geom_line(alpha = .5) +
  lims(x = 0:1, y = 0:1) +
  coord_fixed() +
  ylab("Desirability")

# ------------------------------------------------------------------------------
# Custom function

v_x <- seq(0, 1, length.out = 20)
v_d <- 1 - exp(-10 * abs(v_x - .5))

dat %>%
  mutate(v = d_custom(x, v_x, v_d)) %>%
  ggplot(aes(x = x, y = v)) +
  geom_point() +
  geom_line(alpha = .5) +
  lims(x = 0:1, y = 0:1) +
  coord_fixed() +
  ylab("Desirability")

# ------------------------------------------------------------------------------
# Qualitative data

set.seed(3)
groups <- sort(runif(10))
names(groups) <- letters[1:10]

tibble(x = letters[1:7]) %>%
  mutate(d = d_category(x, groups)) %>%
  ggplot(aes(x = x, y = d)) +
  geom_bar(stat = "identity") +
  lims(y = 0:1) +
  ylab("Desirability")

# ------------------------------------------------------------------------------
# Apply the same function to many columns at once (dplyr > 1.0)

dat %>%
  mutate(across(c(everything()), ~ d_min(., .2, .6), .names = "d_{col}"))

# ------------------------------------------------------------------------------
# Using current data

set.seed(9015)
```

```
tibble(z = c(0, sort(runif(20)), 1)) %>%
  mutate(
    user_specified = d_max(z, 0.1, 0.75),
    data_driven   = d_max(z, use_data = TRUE)
  ) %>%
  ggplot(aes(x = z)) +
  geom_point(aes(y = user_specified)) +
  geom_line(aes(y = user_specified), alpha = .5) +
  geom_point(aes(y = data_driven), col = "blue") +
  geom_line(aes(y = data_driven), col = "blue", alpha = .5) +
  lims(x = 0:1, y = 0:1) +
  coord_fixed() +
  ylab("Desirability")
```

Index