Package 'customsteps'

December 3, 2018

Title Customizable Higher-Order Recipe Step Functions

Version 0.7.1.0

Description Customizable higher-order recipe step functions for the 'recipes' package. These step functions take 'prep' and 'bake' helper functions as inputs and create specifications of customized recipe steps as output.

URL https://github.com/smaakage85/customsteps

Depends R (>= 3.1.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

Imports recipes, dplyr, tibble, magrittr, purrr, tidyselect, methods, generics, rlang

RoxygenNote 6.1.0

NeedsCompilation no

Author Lars Kjeldgaard [aut, cre]

Maintainer Lars Kjeldgaard <lars_kjeldgaard@hotmail.com>

Repository CRAN

Date/Publication 2018-12-03 10:12:42 UTC

R topics documented:

step_custom_filter *Custom Filter*

Description

'step_custom_filter' creates a *specification* of a (higher-order) recipe step that will potentially remove variables using a custom filter function.

Usage

```
step_custom_filter(recipe, ..., role = NA, trained = FALSE,
  filter_function = NULL, options = NULL, removals = NULL,
  skip = FALSE, id = rand_id("custom_filter"))

## S3 method for class 'step_custom_filter'
tidy(x, ...)
```

Arguments

recipe	A recipe object. The step will be added to the sequence of operations for this recipe.
...	One or more selector functions to choose which variables that will evaluated by the filtering. See [recipes::selections()] for more details.
role	Not used by this step since no new variables are created.
trained	A logical to indicate if the quantities for preprocessing have been estimated.
filter_function	A custom filter function, that will diagnose problematic variables (see Details below).
options	A list of options that will be provided to the filter function as arguments (see Details below).
removals	A character string that contains the names of the columns that should be removed. These values are not determined until [recipes::prep.recipe()] is called.
skip	A logical. Should the step be skipped when the recipe is baked by bake.recipe()? While all operations are baked when prep.recipe() is run, some operations may not be able to be conducted on new data (e.g. processing the outcome variable(s)). Care should be taken when using skip = TRUE as it may affect the computations for subsequent operations
id	A character string that is unique to this step to identify it.
x	A 'step_custom_filter' object.

Details

This step diagnoses problematic variables according to a custom filter function. The filter function must meet the following requirements:

1. the function must at least take one argument ‘x‘: the subset of selected variables from the initial data set.
2. the function must return a vector with the names of the variables diagnosed as problematic.

All additional arguments to the custom filter function must be provided through the ’options’ argument.

Value

An updated version of ‘recipe‘ with the new step added to the sequence of existing steps (if any). For the ‘tidy‘ method, a tibble with columns ‘terms‘ which is the columns that will be removed as well as the step ‘id‘.

See Also

[recipes::recipe()] [recipes::prep.recipe()] [recipes::bake.recipe()]

Examples

```
library(magrittr)
library(tidyselect)
library(generics)
library(tibble)
library(purrr)
library(recipes)

# generate data.
df <- tibble(a = c(1, -999, 3,NA,NA),
             b = c(1,3, NA,NA,NA),
             c = c(1,-999,3,4,5),
             d = rep(1, 5),
             e = c(-999, -999, -999, -999, NA),
             f = rep(NA, 5))

# Create custom filter function to identify variables with a proportion of
# missing values above some threshold. The function treats # values provided
# with the 'other_values' argument as missings.

filter_missings <- function(x, threshold = 0.5, other_values = NULL) {

  # identify problematic variables.
  if (is.null(other_values)) {

    problematic_lgl <- map_lgl(x, ~ mean(is.na(.)) >= threshold)

  } else {
```

```
    problematic_lgl <- map_lgl(x, ~ mean(is.na(.) |
    . %in% other_values) >= threshold)

  }

  # return names of problematic variables.
  names(x)[problematic_lgl]

}

# create recipe.
rec <- recipe(df) %>%
  step_custom_filter(everything(),
                     filter_function = filter_missings,
                     options = list(threshold = 0.5, other_values = -999))

# prep recipe.
rec_prep <- prep(rec)

# bake recipe.
rec_baked <- bake(rec_prep, df)

# inspect output.
tidy(rec)
tidy(rec, number = 1)
tidy(rec_prep)
tidy(rec_prep, number = 1)
rec_baked
```

`step_custom_transformation` *Custom Transformation*

Description

‘step_custom_transformation‘ creates a *specification* of a higher order recipe step that will make a transformation of the input data from (custom) ‘prep‘ and ‘bake‘ helper functions.

Usage

```
step_custom_transformation(recipe, ..., role = "predictor",
  trained = FALSE, prep_function = NULL, prep_options = NULL,
  prep_output = NULL, bake_function = NULL, bake_options = NULL,
  bake_how = "bind_cols", selected_vars = NULL, skip = FALSE,
  id = rand_id("custom_transformation"))

## S3 method for class 'step_custom_transformation'
tidy(x, ...)
```

Arguments

recipe	A recipe object. The step will be added to the sequence of operations for this recipe.
...	One or more selector functions to choose which variables are affected by the step. See [recipes::selections()] for more details. The names of the selected variables will be stored in the ‘selected_vars‘ argument.
role	For model terms created by this step, what analysis role should they be assigned? By default, the function assumes that the new columns will be used as predictors in a model.
trained	A logical to indicate if the quantities for preprocessing have been estimated.
prep_function	A function. This is a helper function for the [recipes::prep.recipe()] method. It will be invoked, when the recipe is ’prepped’ by [recipes::prep.recipe()]. The function MUST satisfy the following conditions: (1) the function must take an argument ‘x‘: the the subset of selected variables (‘selected_vars‘) from the initial data set, (2) the function MUST return the (required) estimated parameters that can be later applied to other data sets. This output can be of any appropriate type and shape. Leave ‘prep_function‘ as NULL, if the preparation of new data sets does not depend on parameters learned on the initial data set.
prep_options	A list with (any) additional arguments for the prep helper function call EXCEPT for the ‘x‘ argument. Leave as NULL, if no ‘prep_function‘ is given.
prep_output	Output from prep helper (‘prep_function‘) function call consisting of the estimated parameters from the initial data set set, that will be applied to other data sets. Results are not computed until [recipes::prep.recipe()] is called.
bake_function	A function. This is a helper function for the ’bake’ method. It will be invoked, when the recipe is ’baked’ by ‘bake.recipe()‘. The function MUST satisfy the following conditions: (1) the function must take an argument ‘x‘: the new data set, that the transformation will be applied to, (2) IF the preparation of new data sets depends on parameters learned on the initial data set, the function must take the argument ‘prep_output‘: the output from the prep helper fct (‘prep_function‘), (3) the output from from the function should be the transformed variables. The output must be of a type and shape, that allows it to be binded column wise to the new data set after converting it to a ‘tibble‘.
bake_options	A list with (any) arguments for the ‘bake_function‘ function call EXCEPT for the ‘x‘ and ‘prep_output‘ arguments.
bake_how	A character. How should the transformed variables be appended to the new data set? Choose from options (1) ‘bind_cols‘: simply bind the transformed variables to the new data set or (2) ‘replace‘: replace the selected variables (‘selected vars‘) from the new data set with the transformed variables.
selected_vars	A character string that contains the names of the selected variables. These values are not determined until [recipes::prep.recipe()] is called.
skip	A logical. Should the step be skipped when the recipe is baked by [recipes::bake.recipe()]? While all operations are baked when [recipes::prep.recipe()] is run, some operations may not be able to be conducted on new data (e.g. processing the outcome variable(s)). Care should be taken when using ‘skip = TRUE‘ as it may affect the computations for subsequent operations.

id	A character string that is unique to this step to identify it.
x	A ‘step_custom_transformation‘ object.

Value

An updated version of ‘recipe‘ with the new step added to the sequence of existing steps (if any). For the ‘tidy‘ method, a ‘tibble‘ with columns ‘terms‘ (the selectors or variables selected) as well as the step ‘id‘.

See Also

[recipes::recipe()] [recipes::prep.recipe()] [recipes::bake.recipe()]

Examples

```
library(dplyr)
library(purrr)
library(tibble)
library(recipes)
library(generics)

# divide 'mtcars' into two data sets.
cars_initial <- mtcars[1:16, ]
cars_new <- mtcars[17:nrow(mtcars), ]

# define prep helper function, that computes means and standard deviations
# for (an arbitrary number of) numeric variables.
compute_means_sd <- function(x) {

 map(.x = x, ~ list(mean = mean(.x), sd = sd(.x)))

}

# define bake helper function, that centers numeric variables to have
# a mean of 'alpha' and scale them to have a standard deviation of
# 'beta'.
center_scale <- function(x, prep_output, alpha, beta) {

  # extract only the relevant variables from the new data set.
  new_data <- select(x, names(prep_output))

  # apply transformation to each of these variables.
  # variables are centered around 'alpha' and scaled to have a standard
  # deviation of 'beta'.
  map2(.x = new_data,
       .y = prep_output,
       ~ alpha + (.x - .y$mean) * beta / .y$sd)

}

# create recipe.
rec <- recipe(cars_initial) %>%
```

```
  step_custom_transformation(mpg, disp,
                             prep_function = compute_means_sd,
                             bake_function = center_scale,
                             bake_options = list(alpha = 0, beta = 1),
                             bake_how = "replace")

# prep recipe.
rec_prep <- prep(rec)

# bake recipe.
rec_baked <- bake(rec_prep, cars_new)
rec_baked

# inspect output.
rec
rec_baked
tidy(rec)
tidy(rec, 1)
tidy(rec_prep)
tidy(rec_prep, 1)
```

Index