Package ‘themis’

January 13, 2020

Title Extra Recipes Steps for Dealing with Unbalanced Data

Version 0.1.0


License MIT + file LICENSE

Suggests testthat (>= 2.1.0), covr, ggplot2, modeldata

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Depends R (>= 2.10), recipes (>= 0.1.4)

Imports tibble, purrr, withr, generics, dplyr, rlang, tidyselect (>= 0.2.5), ROSE, unbalanced, RANN, dials

URL https://github.com/tidymodels/themis

BugReports https://github.com/tidymodels/themis/issues

NeedsCompilation no

Author Emil Hvitfeldt [aut, cre] (<https://orcid.org/0000-0002-0679-1945>)

Maintainer Emil Hvitfeldt <emilhhvitfeldt@gmail.com>

Repository CRAN

Date/Publication 2020-01-13 15:30:02 UTC
R topics documented:

    all_neighbors ........................................... 2
    circle_example ........................................... 2
    step_adasyn .............................................. 3
    step_bsmote .............................................. 5
    step_downsample .......................................... 8
    step_nearmiss ............................................ 10
    step_rose ................................................ 12
    step_smote ............................................... 15
    step_tomek ............................................... 17
    step_upsample ............................................ 19

Index

---

all_neighbors    Parameter to determine which neighbors to use

Description

    Used in step_bsmote().

Usage

    all_neighbors(values = c(TRUE, FALSE))

Arguments

    values       A vector of possible values (TRUE or FALSE).

Examples

    all_neighbors()

---

circle_example    Synthetic Dataset with a circle

Description

    A random dataset with two classes one of which is inside a circle. Used for examples to show how the different methods handles borders.

Usage

    circle_example
Format

A data frame with 200 rows and 3 variables:

- **x** Numeric.
- **y** Numeric.
- **class** Factor, values "Circle" and "Rest".

Description

`step_adasyn` creates a *specification* of a recipe step that generates synthetic positive instances using ADASYN algorithm.

Usage

```r
step_adasyn(
  recipe,
  ..., 
  role = NA, 
  trained = FALSE, 
  column = NULL, 
  over_ratio = 1, 
  neighbors = 5, 
  skip = TRUE, 
  seed = sample.int(10^5, 1), 
  id = rand_id("adasyn")
)
```

## S3 method for class 'step_adasyn'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in *single factor variable*. For the tidy method, these are not currently used.
- **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.
- **column** A character string of the variable name that will be populated (eventually) by the ... selectors.
over_ratio  A numeric value for the ratio of the majority-to-minority frequencies. The default value (1) means that all other levels are sampled up to have the same frequency as the most occurring level. A value of 0.5 would mean that the minority levels will have (at most) (approximately) half as many rows than the majority level.

neighbors  An integer. Number of nearest neighbours that are used to generate the new examples of the minority class.

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.

seed  An integer that will be used as the seed when applied.

id  A character string that is unique to this step to identify it.

x  A `step_adasyn` object.

Details

All columns in the data are sampled and returned by `juice()` and `bake()`.

All columns used in this step must be numeric with no missing data.

When used in modeling, users should strongly consider using the option `skip = TRUE` so that the extra sampling is not conducted outside of the training set.

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 variable used to sample.

References


Examples

```r
library(recipes)
library(modeldata)
data(okc)

sort(table(okc$Class, useNA = "always"))

ds_rec <- recipe(Class ~ age + height, data = okc) %>%
  step_meanimpute(all_predictors()) %>%
  step_adasyn(Class) %>%
  prep()

sort(table(juice(ds_rec)$Class, useNA = "always"))
```
# since 'skip' defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = okc)
table(baked_okc$Class, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
  geom_point() +
  labs(title = "Without ADASYN")

recipe(class ~ ., data = circle_example) %>%
  step_adasyn(class) %>%
  prep() %>%
  juice() %>%
  ggplot(aes(x, y, color = class)) +
  geom_point() +
  labs(title = "With ADASYN")

---

**step_bsmote**

*Apply borderline-SMOTE algorithm*

**Description**

*step_bsmote* creates a *specification* of a recipe step that generates new examples of the minority class using nearest neighbors of these cases in the border region between classes.

**Usage**

```r
step_bsmote(
  recipe,
  ...,
  role = NA,
  trained = FALSE,
  column = NULL,
  over_ratio = 1,
  neighbors = 5,
  all_neighbors = FALSE,
  skip = TRUE,
  seed = sample.int(10^5, 1),
  id = rand_id("bsmote")
)
```

## S3 method for class 'step_bsmote'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in a single factor variable. For the tidy method, these are not currently used.
- **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.
- **column**: A character string of the variable name that will be populated (eventually) by the ... selectors.
- **over_ratio**: A numeric value for the ratio of the majority-to-minority frequencies. The default value (1) means that all other levels are sampled up to have the same frequency as the most occurring level. A value of 0.5 would mean that the minority levels will have (at most) (approximately) half as many rows than the majority level.
- **neighbors**: An integer. Number of nearest neighbours that are used to generate the new examples of the minority class.
- **all_neighbors**: Type of two borderline-SMOTE method. Defaults to FALSE. See details.
- **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.
- **seed**: An integer that will be used as the seed when smote-ing.
- **id**: A character string that is unique to this step to identify it.
- **x**: A `step_bsmote` object.

Details

This method works the same way as `step_smote()`, expect that instead of generating points around every point of of the minority class each point is first being classified into the boxes "danger" and "not". For each point the k nearest neighbors is calculated. If all the neighbors comes from a different class it is labeled noise and put into the "not" box. If more then half of the neighbours comes from a different class it is labeled "danger.

If all_neighbors = FALSE then points will be generated between nearest neighbours in its own class. If all_neighbors = TRUE then points will be generated between any nearest neighbours. See examples for visualization.

The parameter neighbors controls the way the new examples are created. For each currently existing minority class example X new examples will be created (this is controlled by the parameter over_ratio as mentioned above). These examples will be generated by using the information from the neighbors nearest neighbours of each example of the minority class. The parameter neighbors controls how many of these neighbours are used.

All columns in the data are sampled and returned by `juice()` and `bake()`.
All columns used in this step must be numeric with no missing data.
When used in modeling, users should strongly consider using the option `skip = TRUE` so that the extra sampling is not conducted outside of the training set.

**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 variable used to sample.

**References**


**Examples**

```r
library(recipes)
library(modeldata)
data(credit_data)

sort(table(credit_data$Status, useNA = "always"))

d_rec <- recipe(Status ~ Age + Income + Assets, data = credit_data) %>%
  step_meanimpute(all_predictors()) %>%
  step_bsmote(Status) %>%
  prep()

sort(table(juice(d_rec)$Status, useNA = "always"))

# since `skip` defaults to TRUE, baking the step has no effect

table(baked_okc$Status, useNA = "always")

d_rec2 <- recipe(Status ~ Age + Income + Assets, data = credit_data) %>%
  step_meanimpute(all_predictors()) %>%
  step_bsmote(Status, over_ratio = 0.2) %>%
  prep()

table(juice(d_rec2)$Status, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
  geom_point() +
  labs(title = "Without SMOTE")

recipe(class ~ ., data = circle_example) %>%
  step_bsmote(class, all_neighbors = FALSE) %>%
  prep() %>%
  juice() %>%
  ggplot(aes(x, y, color = class)) +
  geom_point() +
```
step_downsample

Down-Sample a Data Set Based on a Factor Variable

Description

step_downsample creates a specification of a recipe step that will remove rows of a data set to make the occurrence of levels in a specific factor level level equal.

Usage

step_downsample(
  recipe,
  ..., 
  under_ratio = 1,
  ratio = NA,
  role = NA,
  trained = FALSE,
  column = NULL,
  target = NA,
  skip = TRUE,
  seed = sample.int(10^5, 1),
  id = rand_id("downsample")
)

## S3 method for class 'step_downsample'
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 variable is used to sample the data. See selections() for more details. The selection should result in single factor variable. For the tidy method, these are not currently used.
under_ratio  A numeric value for the ratio of the minority-to-majority frequencies. The default value (1) means that all other levels are sampled down to have the same frequency as the least occurring level. A value of 2 would mean that the majority levels will have (at most) (approximately) twice as many rows than the minority level.

ratio        Deprecated argument; same as under_ratio
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.
column       A character string of the variable name that will be populated (eventually) by the ... selectors.
target       An integer that will be used to subsample. This should not be set by the user and will be populated by prep.
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
seed         An integer that will be used as the seed when downsampling.
id           A character string that is unique to this step to identify it.
x            A step_downsample object.

Details

Down-sampling is intended to be performed on the training set alone. For this reason, the default is skip = TRUE. It is advisable to use prep(recipe, retain = TRUE) when preparing the recipe; in this way juice() can be used to obtain the down-sampled version of the data.

If there are missing values in the factor variable that is used to define the sampling, missing data are selected at random in the same way that the other factor levels are sampled. Missing values are not used to determine the amount of data in the minority level

For any data with factor levels occurring with the same frequency as the minority level, all data will be retained.

All columns in the data are sampled and returned by juice() and bake().

Keep in mind that the location of down-sampling in the step may have effects. For example, if centering and scaling, it is not clear whether those operations should be conducted before or after rows are removed.

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 variable used to sample.

Examples

library(recipes)
library(modeldata)
data(okc)

sort(table(okc$diet, useNA = "always"))

ds_rec <- recipe(~ ., data = okc) %>%
  step_downsample(diet) %>%
  prep(training = okc, retain = TRUE)

sort(table(juice(ds_rec)$diet, useNA = "always"))

# since 'skip' defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = okc)

table(baked_okc$diet, useNA = "always")

---

**step_nearmiss**

*Under-sampling by removing points near other classes.*

**Description**

`step_nearmiss` creates a specification of a recipe step that removes majority class instances by undersampling points in the majority class based on their distance to other points in the same class.

**Usage**

```r
step_nearmiss(
  recipe,
  ..., 
  role = NA,
  trained = FALSE,
  column = NULL,
  under_ratio = 1,
  neighbors = 5,
  skip = TRUE,
  seed = sample.int(10^5, 1),
  id = rand_id("nearmiss")
)
```

## S3 method for class 'step_nearmiss'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in single factor variable. For the `tidy` method, these are not currently used.
- `role` Not used by this step since no new variables are created.
step_nearmiss

trained A logical to indicate if the quantities for preprocessing have been estimated.
column A character string of the variable name that will be populated (eventually) by the ... selectors.
under_ratio A numeric value for the ratio of the minority-to-majority frequencies. The default value (1) means that all other levels are sampled down to have the same frequency as the least occurring level. A value of 2 would mean that the majority levels will have (at most) (approximately) twice as many rows than the minority level.
neighbors An integer. Number of nearest neighbours that are used to generate the new examples of the minority class.
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
seed An integer that will be used as the seed when applied.
id A character string that is unique to this step to identify it.
x A step_nearmiss object.

Details

This methods retained the points form the majority classes which has the smallest mean distance to the k nearest points in the other classes.
All columns in the data are sampled and returned by juice() and bake().
All columns used in this step must be numeric with no missing data.
When used in modeling, users should strongly consider using the option skip = TRUE so that the extra sampling is not conducted outside of the training set.

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 variable used to sample.

References


Examples

library(recipes)
library(modeldata)
data(okc)

sort(table(okc$Class, useNA = "always"))

ds_rec <- recipe(Class ~ age + height, data = okc) %>%
step_rose %>%
step_meanimpute(all_predictors()) %>%
step_nearmiss(Class) %>%
prep()

sort(table(juice(ds_rec)$Class, useNA = "always"))

# since 'skip' defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = okc)
table(baked_okc$Class, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
  geom_point() +
  labs(title = "Without NEARMISS") +
  xlim(c(1, 15)) +
  ylim(c(1, 15))

recipe(class ~ ., data = circle_example) %>%
  step_nearmiss(class) %>%
  prep() %>%
  juice() %>%
  ggplot(aes(x, y, color = class)) +
  geom_point() +
  labs(title = "With NEARMISS") +
  xlim(c(1, 15)) +
  ylim(c(1, 15))

---

## step_rose

**Apply ROSE algorithm**

### Description

*step_rose* creates a *specification* of a recipe step that generates sample of synthetic data by enlarging the features space of minority and majority class example. Using *ROSE::ROSE()*.

### Usage

```r
step_rose(
  recipe,
  ...,
  role = NA,
  trained = FALSE,
  column = NULL,
  over_ratio = 1,
  minority_prop = 0.5,
  minority_smoothness = 1,
  majority_smoothness = 1,
)```
step_rose

skip = TRUE,
seed = sample.int(10^5, 1),
id = rand_id("rose")
)

## S3 method for class 'step_rose'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in single factor variable. For the tidy method, these are not currently used.
- **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.
- **column**: A character string of the variable name that will be populated (eventually) by the ... selectors.
- **over_ratio**: A numeric value for the ratio of the majority-to-minority frequencies. The default value (1) means that all other levels are sampled up to have the same frequency as the most occurring level. A value of 0.5 would mean that the minority levels will have (at most) (approximately) half as many rows than the majority level.
- **minority_prop**: A numeric. Determines the of over-sampling of the minority class. Defaults to 0.5.
- **minority_smoothness**: A numeric. Shrink factor to be multiplied by the smoothing parameters to estimate the conditional kernel density of the minority class. Defaults to 1.
- **majority_smoothness**: A numeric. Shrink factor to be multiplied by the smoothing parameters to estimate the conditional kernel density of the majority class. Defaults to 1.
- **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.
- **seed**: An integer that will be used as the seed when rose-ing.
- **id**: A character string that is unique to this step to identify it.
- **x**: A step_rose object.

Details

The factor variable used to balance around must only have 2 levels.
The ROSE algorithm works by selecting an observation belonging to class k and generates new examples in its neighborhood determined by some matrix $H_k$. Smaller values of these arguments have the effect of shrinking the entries of the corresponding smoothing matrix $H_k$. Shrinking would be a cautious choice if there is a concern that excessively large neighborhoods could lead to blur the boundaries between the regions of the feature space associated with each class.

All columns in the data are sampled and returned by `juice()` and `bake()`.

All columns used in this step must be numeric.

When used in modeling, users should strongly consider using the option `skip = TRUE` so that the extra sampling is not conducted outside of the training set.

**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 variable used to sample.

**References**


**Examples**

```r
library(recipes)
library(modeldata)
data(okc)

sort(table(okc$Class, useNA = "always"))

d_s_rec <- recipe(Class ~ age + height, data = okc) %>%
  step_rose(Class) %>%
  prep()

sort(table(juice(ds_rec)$Class, useNA = "always"))

# since `skip` defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = okc)
table(baked_okc$Class, useNA = "always")

d_s_rec2 <- recipe(Class ~ age + height, data = okc) %>%
  step_rose(Class, minority_prop = 0.3) %>%
  prep()

table(juice(ds_rec2)$Class, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
  geom_point() +
```
**step_smote**

`labs(title = "Without ROSE")`

`recipe(class ~ ., data = circle_example) %>%`
`  step_rose(class) %>%`
`  prep() %>%`
`  juice() %>%`
`  ggplot(aes(x, y, color = class)) +`
`  geom_point() +`
`  labs(title = "With ROSE")`

---

**Apply SMOTE algorithm**

**Description**

*step_smote* creates a *specification* of a recipe step that generate new examples of the minority class using nearest neighbors of these cases.

**Usage**

```r
step_smote(
  recipe,
  ..., # One or more selector functions to choose which variable is used to sample the data. See selections() for more details. The selection should result in a single factor variable. For the tidy method, these are not currently used.
  role = NA,
  trained = FALSE,
  column = NULL,
  over_ratio = 1,
  neighbors = 5,
  skip = TRUE,
  seed = sample.int(10^5, 1),
  id = rand_id("smote")
)
```

```r
## S3 method for class 'step_smote'
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 variable is used to sample the data. See selections() for more details. The selection should result in a single factor variable. For the tidy method, these are not currently used.
- `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.
**column**
A character string of the variable name that will be populated (eventually) by the ... selectors.

**over_ratio**
A numeric value for the ratio of the majority-to-minority frequencies. The default value (1) means that all other levels are sampled up to have the same frequency as the most occurring level. A value of 0.5 would mean that the minority levels will have (at most) (approximately) half as many rows than the majority level.

**neighbors**
An integer. Number of nearest neighbours that are used to generate the new examples of the minority class.

**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.

**seed**
An integer that will be used as the seed when smote-ing.

**id**
A character string that is unique to this step to identify it.

**x**
A `step_smote` object.

**Details**
The parameter `neighbors` controls the way the new examples are created. For each currently existing minority class example X new examples will be created (this is controlled by the parameter `over_ratio` as mentioned above). These examples will be generated by using the information from the `neighbors` nearest neighbours of each example of the minority class. The parameter `neighbors` controls how many of these neighbours are used.

All columns in the data are sampled and returned by `juice()` and `bake()`.
All columns used in this step must be numeric with no missing data.

When used in modeling, users should strongly consider using the option `skip = TRUE` so that the extra sampling is *not* conducted outside of the training set.

**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 variable used to sample.

**References**

**Examples**
```r
library(recipes)
library(modeldata)
data(credit_data)

sort(table(credit_data$Status, useNA = "always"))
```
ds_rec <- recipe(Status ~ Age + Income + Assets, data = credit_data) %>%
  step_meanimpute(all_predictors()) %>%
  step_smote(Status) %>%
  prep()

sort(table(juice(ds_rec)$Status, useNA = "always"))

# since 'skip' defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = credit_data)

baked_okc$Status

ds_rec2 <- recipe(Status ~ Age + Income + Assets, data = credit_data) %>%
  step_meanimpute(all_predictors()) %>%
  step_smote(Status, over_ratio = 0.2) %>%
  prep()

table(juice(ds_rec2)$Status, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
  geom_point() +
  labs(title = "Without SMOTE")

recipe(class ~ ., data = circle_example) %>%
  step_smote(class) %>%
  prep() %>%
  juice() %>%
  ggplot(aes(x, y, color = class)) +
  geom_point() +
  labs(title = "With SMOTE")

---

**step_tomek**

*Under-sampling by removing Tomek's links.*

**Description**

*step_tomek* creates a specification of a recipe step that removes majority class instances of Tomek links. Using `unbalanced::ubTomek()`.

**Usage**

```r
step_tomek(
  recipe,
  ..., 
  role = NA,
  trained = FALSE,
  column = NULL,
)```
skip = TRUE,
seed = sample.int(10^5, 1),
id = rand_id("tomek")
)
## S3 method for class 'step_tomek'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in a single factor variable. For the tidy method, these are not currently used.
- **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.
- **column**: A character string of the variable name that will be populated (eventually) by the ... selectors.
- **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.
- **seed**: An integer that will be used as the seed when applied.
- **id**: A character string that is unique to this step to identify it.
- **x**: A step_tomek object.

Details

The factor variable used to balance around must only have 2 levels. All other variables must be numerics with no missing data.

A tomek link is defined as a pair of points from different classes and are each others nearest neighbors.

All columns in the data are sampled and returned by `juice()` and `bake()`.

When used in modeling, users should strongly consider using the option `skip = TRUE` so that the extra sampling is not conducted outside of the training set.

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 variable used to sample.

References

Examples

```r
library(recipes)
library(modeldata)
data(okc)

sort(table(okc$Class, useNA = "always"))

ds_rec <- recipe(Class ~ age + height, data = okc) %>%
    step_meanimpute(all_predictors()) %>%
    step_tomek(Class) %>%
    prep()

sort(table(juice(ds_rec)$Class, useNA = "always"))

# since 'skip' defaults to TRUE, baking the step has no effect
baked_okc <- bake(ds_rec, new_data = okc)
table(baked_okc$Class, useNA = "always")

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
    geom_point() +
    labs(title = "Without Tomek") +
    xlim(c(1, 15)) +
    ylim(c(1, 15))

recipe(class ~ ., data = circle_example) %>%
    step_tomek(class) %>%
    prep() %>%
    juice() %>%
    ggplot(aes(x, y, color = class)) +
    geom_point() +
    labs(title = "With Tomek") +
    xlim(c(1, 15)) +
    ylim(c(1, 15))
```

---

**step_upsample**  
Up-Sample a Data Set Based on a Factor Variable

**Description**

*step_upsample* creates a *specification* of a recipe step that will replicate rows of a data set to make the occurrence of levels in a specific factor level equal.

**Usage**

```r
step_upsample(
    recipe, 
```
...,  
over_ratio = 1,  
ratio = NA,  
role = NA,  
trained = FALSE,  
column = NULL,  
target = NA,  
skip = TRUE,  
seed = sample.int(10^5, 1),  
id = rand_id("upsample")
)

## S3 method for class 'step_upsample'
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 variable is used to sample the data. See `selections()` for more details. The selection should result in a single factor variable. For the `tidy` method, these are not currently used.

- **over_ratio**
  A numeric value for the ratio of the majority-to-minority frequencies. The default value (1) means that all other levels are sampled up to have the same frequency as the most occurring level. A value of 0.5 would mean that the minority levels will have (at most) (approximately) half as many rows than the majority level.

- **ratio**
  Deprecated argument; same as over_ratio.

- **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.

- **column**
  A character string of the variable name that will be populated (eventually) by the ... selectors.

- **target**
  An integer that will be used to subsample. This should not be set by the user and will be populated by prep.

- **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.

- **seed**
  An integer that will be used as the seed when upsampling.

- **id**
  A character string that is unique to this step to identify it.

- **x**
  A `step_upsample` object.
Details

Up-sampling is intended to be performed on the training set alone. For this reason, the default is `skip = TRUE`. It is advisable to use `prep(recipe, retain = TRUE)` when preparing the recipe; in this way `juice()` can be used to obtain the up-sampled version of the data.

If there are missing values in the factor variable that is used to define the sampling, missing data are selected at random in the same way that the other factor levels are sampled. Missing values are not used to determine the amount of data in the majority level (see example below).

For any data with factor levels occurring with the same frequency as the majority level, all data will be retained.

All columns in the data are sampled and returned by `juice()` and `bake()`.

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 variable used to sample.

Examples

```r
library(recipes)
library(modeldata)
data(okc)

orig <- table(okc$diet, useNA = "always")

sort(orig, decreasing = TRUE)

up_rec <- recipe(~ ., data = okc) %>%
  # Bring the minority levels up to about 200 each
  # 200/16562 is approx 0.0121
  step_upsample(diet, over_ratio = 0.0121) %>%
  prep(training = okc, retain = TRUE)

training <- table(juice(up_rec)$diet, useNA = "always")

# Since `skip` defaults to TRUE, baking the step has no effect
baked_okc <- bake(up_rec, new_data = okc)

baked <- table(baked_okc$diet, useNA = "always")

# Note that if the original data contained more rows than the
# target n (= ratio * majority_n), the data are left alone:
data.frame(
  level = names(orig),
  orig_freq = as.vector(orig),
  train_freq = as.vector(training),
  baked_freq = as.vector(baked)
)

library(ggplot2)

ggplot(circle_example, aes(x, y, color = class)) +
```

```r
geom_point() +
labs(title = "Without upsample")

recipe(class ~ ., data = circle_example) %>%
step_nearmiss(class) %>%
prep() %>%
juice() %>%
ggplot(aes(x, y, color = class)) +
geom_jitter() +
labs(title = "With upsample (with jittering)")
```
Index

*Topic datagen
  step_adasyn, 3
  step_bsmote, 5
  step_downsample, 8
  step_nearmiss, 10
  step_rose, 12
  step_smote, 15
  step_tomek, 17
  step_upsample, 19

*Topic datasets
  circle_example, 2

all_neighbors, 2

bake(), 4, 6, 9, 11, 14, 16, 18, 21
bake.recipe(), 4, 6, 9, 11, 13, 16, 18, 20

circle_example, 2

juice(), 4, 6, 9, 11, 14, 16, 18, 21

prep.recipe(), 4, 6, 9, 11, 13, 16, 18, 20

ROSE::ROSE(), 12

selections(), 3, 6, 8, 10, 13, 15, 18, 20
step_adasyn, 3
step_bsmote, 5
step_downsample, 8
step_nearmiss, 10
step_rose, 12
step_smote, 15
step_smote(), 6
step_tomek, 17
step_upsample, 19

tidy.step_adasyn (step_adasyn), 3
 tidy.step_bsmote (step_bsmote), 5
 tidy.step_downsample (step_downsample), 8
 tidy.step_nearmiss (step_nearmiss), 10

unbalanced::ubTomek(), 17