

How to balance unbalanced classification 1:1 with SMOTE in R

Asked 3 years, 10 months ago Active 1 year, 4 months ago Viewed 13k times

▲
4
▼
★
2
🕒

I am doing binary classification and my current target class is composed of: Bad: 3126 Good:25038

So I want the number of Bad (minority) examples to equal the number of Good examples (1:1). So Bad needs to increase by ~8x (extra 21912 SMOTEd instances) and not increase the majority (Good). The code I am trying will not keep the number of Good constant, as currently.

Code I have tried:

Example 1:

```
library(DMwR)
smoted_data <- SMOTE(targetclass~., data, perc.over=700, perc.under=0, k=5,
  learner=NULL)
```

Example 1 output: Bad:25008 Good:0

Example 2:

```
smoted_data <- SMOTE(targetclass~., data, perc.over=700, k=5, learner=NULL)
```

Example 2 output: Bad: 25008 Good:43764

Example 3:

```
smoted_data <- SMOTE(targetclass~., data, perc.over=700, perc.under=100, k=5,
  learner=NULL)
```

Example 3 output: Bad: 25008 Good: 21882



edited Oct 22 '18 at 7:22



smci

20.7k ● 9 ● 88 ● 121

asked Apr 15 '16 at 15:48



CJava

149 ● 1 ● 3 ● 13

4 Answers

5

To achieve a 1:1 balance using `SMOTE`, you want to do this:

```
library(DMwR)
smoted_data <- SMOTE(targetclass~., data, perc.over=100)
```

I have to admit it doesn't seem obvious from the built-in documentation, but if you read the original documentation, it states:

The parameters `perc.over` and `perc.under` control the amount of over-sampling of the minority class and under-sampling of the majority classes, respectively.

`perc.over` will typically be a number above 100. For each case in the original data set belonging to the minority class, `perc.over/100` new examples of that class will be created. If `perc.over` is a value below 100 then a single case will be generated for a randomly selected proportion (given by `perc.over/100`) of the cases belonging to the minority class on the original data set.

So when `perc.over` is 100, you essentially creating 1 new example ($100/100 = 1$).

The default of `perc.under` is 200, and that is what you want to keep.

The parameter `perc.under` controls the proportion of cases of the majority class that will be randomly selected for the final "balanced" data set. This proportion is calculated with respect to the number of newly generated minority class cases.

```
prop.table(table(smoted_data$targetclass))
# returns 0.5 0.5
```

answered Jun 7 '18 at 10:32



onlyphantom

3,202 ● 2 ● 20 ● 35



2



You can try using the ROSE package in R.

A research article with example is available [here](#)

answered Dec 27 '16 at 9:35



Arvind

313 ● 1 ● 2 ● 11

Thanks for this. I have been struggling with other packages out there. This one works as expected – [Jasmine](#) Jul 2 '18 at 10:58



1



You should use a perc.under of 114.423. Since $(700/100) \times 3126 \times (114.423/100) = 25038.04$.

But note that since SMOTE does a random undersampling for the majority class, this way you would get a new data with duplicates in the majority class. That is to say, your new data will have 25038 GOOD samples, but they are not the same 25038 GOOD samples with the original data. Some GOOD samples will not be included and some will be duplicated in the newly generated data.

answered Apr 4 '17 at 18:56



Yan

387 ● 1 ● 14



0



I recommend you to use the `bimba` package under development by me. It is not yet available on CRAN but you can easily install it from github.

You can find instructions on how to install it on its github page:
<https://github.com/RomeroBarata/bimba>

The only restriction on the data for the use of the `SMOTE` function implemented in `bimba` is that the predictors must be numeric and the target must be both the last column of the data frame that holds your data and have only two values.

As long as your data abide by these restrictions, using the `SMOTE` function is easy:

```
library(bimba)
smoted_data <- SMOTE(data, perc_min = 50, k = 5)
```

Where `perc_min` specifies the desired percentage of the minority class after over-sampling (in that case `perc_min = 50` balance the classes). Note that the majority class is not under-sampled as in the `DMwR` package.

answered Jan 26 '18 at 13:48



[Romero Morais](#)

61 ● 3
