UnbalancedClass_Ex

Ross Jacobucci

June 12, 2015

This is just a short example of how to use two different packages in R for dealing with highly unbalanced classes.

The first is using the ROSE package, which has a nice intro:

http://journal.r-project.org/archive/2014-1/menardi-lunardon-torelli.pdf

And publication outlining technique:

http://link.springer.com/article/10.1007%2Fs10618-012-0295-5

```
library(ROSE)
```

```
## Loaded ROSE 0.0-3
```

```
library(rpart)
library(ISLR)
data(Default)


table(Default$default)
```

```
##
##   No  Yes
## 9667  333
```

```
1-333/(9667+333) # what we would get if predicting everyone as not defaulting
```

```
## [1] 0.9667
```

Easy to see that there are highly unbalanced classes, and we are going to do very well according to accuracy, but our resulting model will not provide any information above and beyond what we could do by just guessing everyone will not default, which obviously is not helpful to banks…

ROSE

```
# create new dataset with balanced classes
#note: if 1's were the smaller class, use method="over
data.bal.un <- ovun.sample(default ~ ., data = Default, method = "under",
                           p = 0.5, seed = 1)$data
table(data.bal.un$default)
```

```
##
##  No Yes
## 321 333
```

```r
# now could either run model on new dataset with balanced classes, or


ROSE.hold <- ROSE.eval(default ~ ., data = data.bal.un, learner = rpart,
                       acc.measure="auc",method.assess = "holdout",
                       extr.pred = function(obj)obj[,2], seed = 1)
ROSE.hold
```

```
##
## Call:
## ROSE.eval(formula = default ~ ., data = data.bal.un, learner = rpart,
##     acc.measure = "auc", extr.pred = function(obj) obj[, 2],
##     method.assess = "holdout", seed = 1)
##
## Holdout estimate of auc: 0.890
```

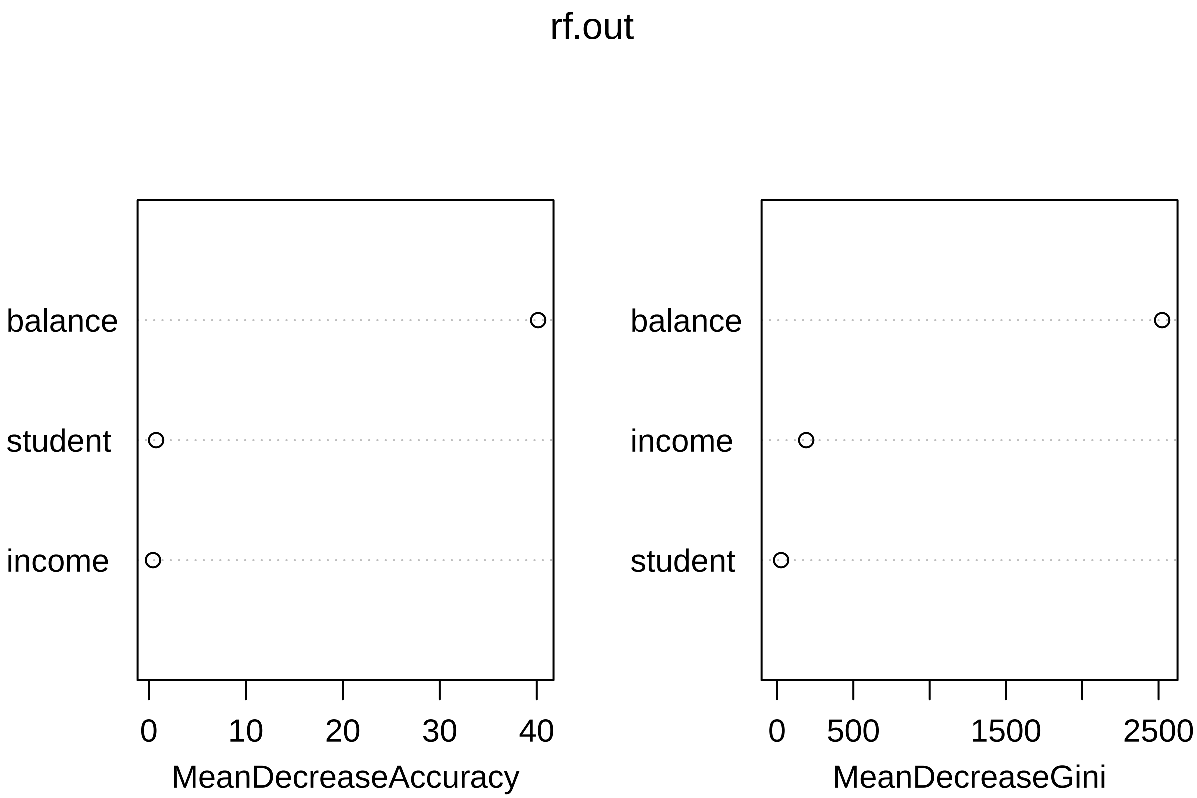
Now that we have created a balanced sample (note, it doesn't have to be 50% sampling of classes), we can use the ROSE.eval function to run trees with rpart, and determine how we are doing on a holdout sample. Note this is just an easier way to test the results on a test dataset, without having to explicitly create a separate dataset.

randomForest

```r
# use randomForest to do classification with balanced classes
library(randomForest); library(caret)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
## Loading required package: lattice
## Loading required package: ggplot2
```

```r
# look at "classwt" argument
rf.out <- randomForest(as.factor(default) ~ ., data=Default,classwt=c(0.5,0.5),importance=TRUE)
varImpPlot(rf.out)
```



In this, class weights need to add to one Example weights No's as 0.5, Yes's as 0.5 By not setting weights, implicity setting to classwt=c(0.96,0.04)

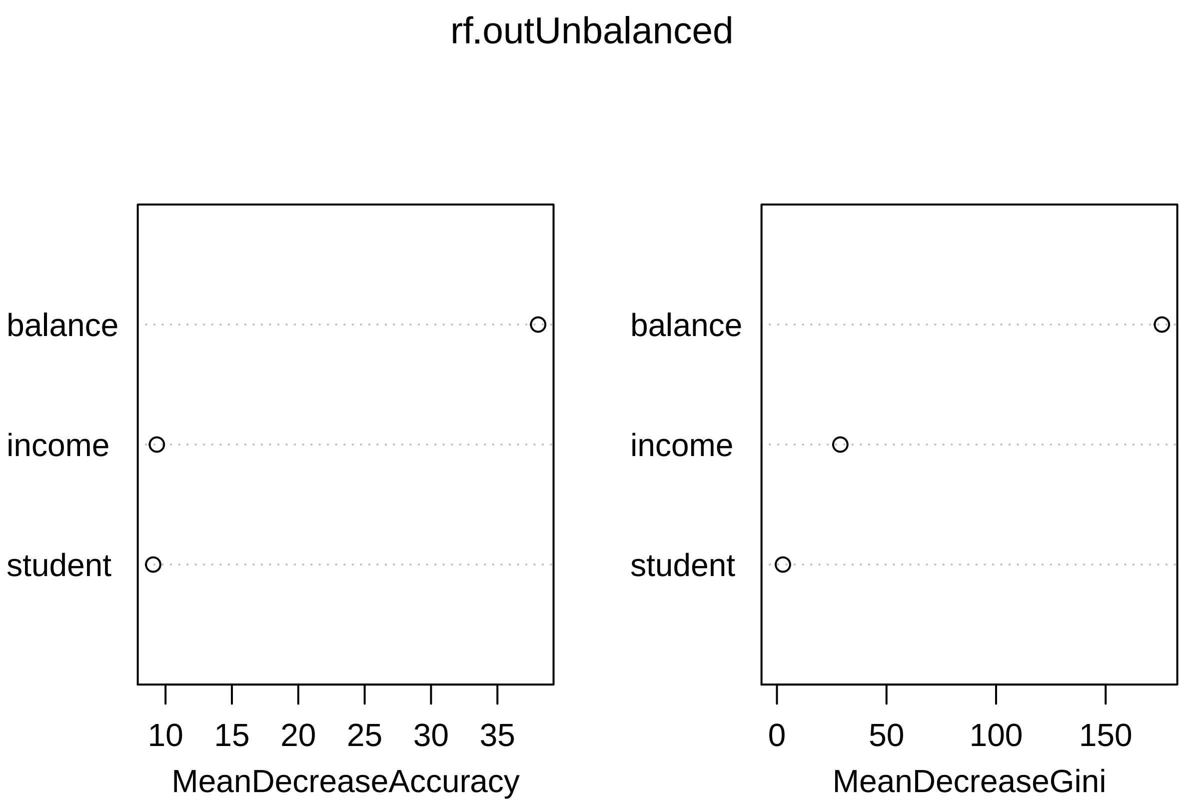
```r
pred1 <- predict(rf.out,newdata=Default)
confusionMatrix(pred1,Default$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  8471   32
##        Yes 1196  301
##
##                Accuracy : 0.8772
##                  95% CI : (0.8706, 0.8836)
##     No Information Rate : 0.9667
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2903
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8763
##             Specificity : 0.9039
##          Pos Pred Value : 0.9962
##          Neg Pred Value : 0.2011
##              Prevalence : 0.9667
##          Detection Rate : 0.8471
##    Detection Prevalence : 0.8503
```

```
##        Balanced Accuracy : 0.8901
##
##         'Positive' Class : No
##
```

Compare to just treating as unbalanced with no correction

```r
rf.outUnbalanced <- randomForest(as.factor(default) ~ ., data=Default,importance=TRUE)
varImpPlot(rf.outUnbalanced)
```




```r
pred2 <- predict(rf.outUnbalanced,newdata=Default)
confusionMatrix(pred2,Default$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  9645  221
##        Yes   22  112
##
##                Accuracy : 0.9757
##                  95% CI : (0.9725, 0.9786)
##     No Information Rate : 0.9667
##     P-Value [Acc > NIR] : 8.77e-08
##
##                   Kappa : 0.4695
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
##             Sensitivity : 0.9977
##             Specificity : 0.3363
##          Pos Pred Value : 0.9776
##          Neg Pred Value : 0.8358
##              Prevalence : 0.9667
##          Detection Rate : 0.9645
##    Detection Prevalence : 0.9866
##       Balanced Accuracy : 0.6670
##
##        'Positive' Class : No
##
```

By balancing classes, we do worse according to measures such as accuracy, however, we are able to glean some information that could actually be useful in predicting propensity to default on a loan.

caret

Another way to to create a new dataset is from the caret package

```
dat.up <- upSample(Default[,2:4], Default[,1], yname = "default")
# also downSample() if classes were switched
table(dat.up$default)
```

```
##
##   No  Yes
## 9667 9667
```