Fully grown decision trees in random forests

Several sources suggest it's ok to fully grow the decision trees in a RF (e.g., Leo Breiman's article and Elements of Statistical Learning, p. 596).

I don't understand the following. Suppose that due to noise, a single data point x of class A ended up somewhere deep inside class B (in terms of its position in the space of features). Every tree that includes x will have a leaf that contains just x alone (because it's fully grown, so it won't stop until each node can precisely identify the class on the training data set; and because x is so isolated from other points of class A, that it can't ever be joined with them in a contiguous region of space carved by a tree). Roughly two-thirds of the trees will include x. Therefore, it seems the majority vote would always be to classify any points close x as if they belong to class A - even though this is clearly overfitting.

A similar argument can be made for any noise that caused a few points of one class to end up in the region that should be assigned to another class.

How is it, then, that fully grown trees inside a RF aren't causing major overfitting?

random-forest cart overfitting

asked Nov 14 '15 at 11:52
max
369 1 12

1 I will admit to being confused by the first sentence of this question. RFs don't "grow" a single tree but many trees. The structure that results from a single tree is lost in the process of developing the forest and aggregating the results across lots of "mini-trees." Therefore, the ensemble nature of the RF answer doesn't represent a "model" that would even be vulnerable to "overfitting." – DJohnson Nov 14 '15 at 12:25

Yes, RF has multiple trees, and each of these trees is fully grown. The structure isn't lost at all, each tree works like usual, the ensemble just counts how many trees predicted each category, and selects the one with the highest count. – max Nov 14 '15 at 18:09

...for practical use of RF, I agree with DJohnson – Soren Havelund Welling Nov 16 '15 at 10:50

1 Answer

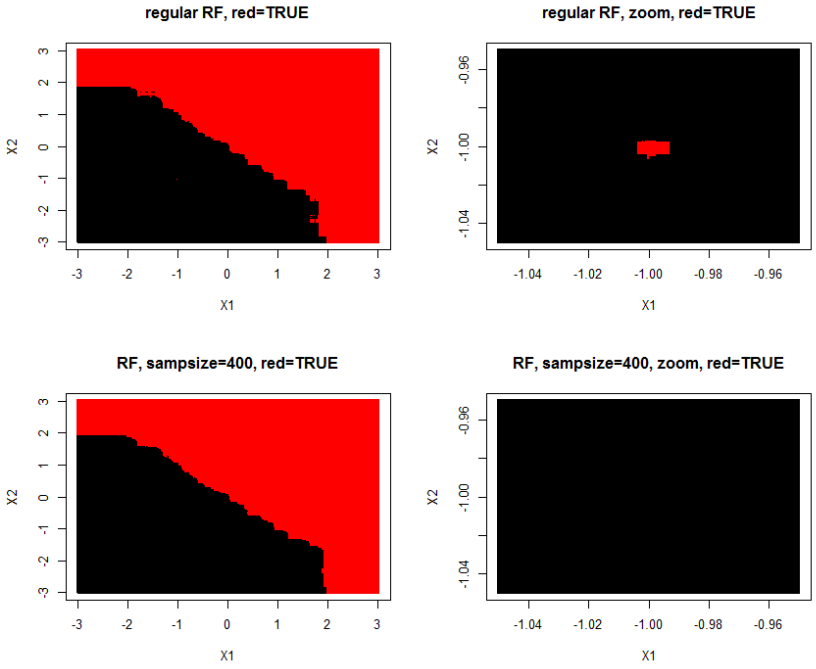
Yes, even a single A-outlier (sample of class A) placed in the middle of many B examples (in a feature space) would affect the structure of the forest. The trained forest model will predict new samples as A, when these are placed very close to the A-outlier. But the density of neighboring B examples will decrease size of this "predict A"-island in the "predict B"-ocean. But the "predict A"-island will not disappear. For noisy classification, e.g. Contraceptive Method Choice, the default random forest can be improved by lowering tried varaibles in each split(mtry) and bootstrap sample size(sampsize). If sampsize is e.g. 40% of training size, then any '*single sample prediction island*' will completely drown if surrounded by only counter examples, as it only will be present in 40% of the trees.

EDIT: If sample replacement is true, then more like 33% of trees.

```
mean(replicate(1000,length(unique(sample(1:1000,400,rep=T)))))
```

I made a simulation of the problem (A=TRUE,B=FALSE) where one (A/TRUE)sample is injected within many (B/FALSE) samples. Hereby is created a tiny A-island in the B ocean. The area of the A-island is so small, it has no influence on the overall prediction performance. Lowering sample size makes the island disappear.

1000 samples with two features X_1 and X_2 are of class "true/A" if $y_i = X_1 + X_2 >= 0$
Features are drawn from $N(0, 1)$




```
library(randomForest)
par(mfrow=c(2,2))
set.seed(1)

#make data
X = data.frame(replicate(2,rnorm(1000)))
y = factor(apply(X,1,sum) >=0) #create 2D class problem
X[1,] = c(-1,-1); y[1]='TRUE' #insert one TRUE outlier inside 'FALSE-land'

#train default forest
rf = randomForest(X,y)

#make test grid(250X250) from -3 to 3,
Xtest = expand.grid(replicate(2,seq(-3,3,le=250),simplify = FALSE))
Xtest[1,] = c(-1,-1) #insert the exact same coordinate of train outlier
Xtest = data.frame(Xtest); names(Xtest) = c("X1","X2")
plot(Xtest,col=predict(rf,Xtest),pch=15,cex=0.5,main="regular RF, red=TRUE")

#zoom in on area surrouding outlier
Xtest = expand.grid(replicate(2,seq(-1.05,-.95,le=250),simplify = FALSE))
Xtest = data.frame(Xtest); names(Xtest) = c("X1","X2")
plot(Xtest,col=predict(rf,Xtest),pch=15,cex=0.5,main= "regular RF, zoom, red=TRUE")

#train extra robust RF
rf = randomForest(X,y,sampsize = 400)
Xtest = expand.grid(replicate(2,seq(-3,3,le=250),simplify = FALSE))
Xtest[1,] = c(-1,-1)
Xtest = data.frame(Xtest); names(Xtest) = c("X1","X2")
plot(Xtest,col=predict(rf,Xtest),pch=15,cex=0.5,main="RF, sampsize=400, red=TRUE")

Xtest = expand.grid(replicate(2,seq(-1.05,-.95,le=250),simplify = FALSE))
Xtest = data.frame(Xtest); names(Xtest) = c("X1","X2")
plot(Xtest,col=predict(rf,Xtest),pch=15,cex=0.5,main="RF, sampsize=400, zoom, red=TRUE")
```

edited Nov 16 '15 at 10:46

answered Nov 16 '15 at 10:27

Soren Havelund Welling
2,871 5 17

Lowering sample size, but still with replacements, correct? In that case, only 40% * ~2/3, or roughly 25% of the original sample is used. I would have assumed that once you use less than 75% of the original sample, due to replacement less than half the trees will see the point, and so the island will disappear - but maybe I'm missing some detail. And about lowering `mtry` : it won't make the island disappear, but would reduce the noise in general, right? – max Nov 16 '15 at 10:31

40% without replacement, 33% with replacement. try simulate with mean(replicate(1000,length(unique(sample(1:1000,400,rep=T))))) – Soren Havelund Welling Nov 16 '15 at 10:39

lowering mtry will not make islands disappear. But for high dimensional data I guess it will make the islands smaller. – Soren Havelund Welling Nov 16 '15 at 10:42

In the case of binary classification, let's say the number of trees that see any given observation falls below 50% (so if you use replacements, you can start with maybe ~60%). Wouldn't that be guaranteed to remove the island if it was formed by a single observation? It would seem the majority vote would be for the other class at that point? – max Nov 16 '15 at 18:20

So in practice don't worry about it :) If you have 1000 samples and 500 trees and sample 600 for each tree with replacement. You are likely to have still some ~10 samples who happend to get picked for more than 250 trees. You can simulate the distribution with this one-liner: plot(table(table(unlist(replicate(500,unique(sample(1:1000,600,rep=T)))))))) – Soren Havelund Welling Nov 16 '15 at 22:15

Add Another Answer