Random Forest - Overview

How do I handle data when the number of variables is very high?

Often a problem that needs to be tackled is so large or complex that we need a group of experts not just a single one to tackle it. Linux, for example, is such a complex system that building it took hundreds of experts.

What if we could harness the decision-making power and the subject matter expertise of many experts and use it in Data Science? There is such a technique called Random Forests which uses collective decision-making to improve on the outcome possible with a single decision maker. In this approach each software "expert" uses a tree-based algorithm to do their bit and then a collection of such trees is used to compute or evolve a model that is better than the output of any one expert.

Decision Trees from data.

First let's understand how the individual tree-based decision making works.

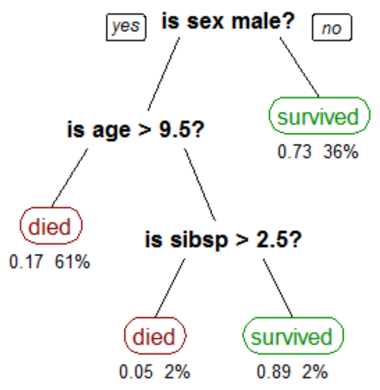
Consider a pool of college applicants applying to Super Exclusive Institute Of Tech. The average SAT score for admission has historically been 2,200. And the average GPA 4.9. We are given the application info on 1,000 applicants. We are asked to create a model that will allow us to predict students most likely to be admitted. How do we go about doing this?

One approach would be to first divide the applicants into those that have SAT score over 2,200 and then call this the "more likely" group. Then to further test for the GPA in this group and split it into two based on GPA less than or equal to 4.9 vs GPA over 4.9. We call the former subgroup "most likely" and the latter a "high maybe".

Then we do the same thing to the group with SAT score below 2,200 calling the high GPA subgroup a "maybe", and the low GPA subgroup a "probably not". This seems reasonable but there are a number of questions that arise.

- What if we split on GPA first and then SAT scores - would we get the same groupings? i.e. what is the best way to split?
- What if we used more criteria such as essay evaluation scores, extra curriculars, awards and distinctions in sports etc. i.e. how many attributes should we use to create splits and what are the most significant attributes.
- We were given averages, but what about the spread, what about outliers? i.e. how does the distribution of attributes affect misclassification?

A decision tree uses the intrinsic structure of the data to make these splits.



This graphic from Wikipedia [1] represents intrinsic structure of the "Titanic" dataset, data on the survivors of the Titanic disater.

The information in a decision tree format. The numbers next to each node are the probability of survival and the % of the observations that were assigned to (classified as) the category represented by this node. Each left branch corresponds to a "yes" answer, the right one a "no". Each green node represents "survived", each red one "did not survive".

The number of spouses or siblings aboard is recorded as "sibsp".

As is well known you had a much smaller chance of surviving if you were male and in the less expensive berths. You had a much greater chance of surviving if you were an infant or female and in the most expensive berths.

Measures of "goodness" for decision trees

The science of Decision Trees quantifies all this using measures called Information Gain and Entropy. Essentially we want to have just the right amount of splits so that we don't keep splitting a group once we have the "best" split. So how do we know when one way to split is better than another. Obviously if one split leads to a clean partition into 'admit' vs. 'reject' then it's good. But what if we split on say essay scores right at the top. We might get groups that have wide variation in GPA, STA in both halves. So we really haven't improved our ability to predict much because both groups seem equally mixed.

This kind of variation in a set indicates a higher "entropy" while a set with all identical members has very low or zero "entropy". So, when we split a set we want the halves to be more distinct from each other and the members in the group to be more like each other – i.e. we want entropy to go down as we keep splitting. So if we use an approach that doesn't reduce the entropy by much it is probably not a good attribute or a good value to split on.

If we take a Decision Tree that has been created and we reverse the process, then when we combine two nodes, we will increase entropy or variation as groups get combined, The gain in entropy is called Information Gain. So the best splits are those which give the best Information Gain when reversed.

This is all very loose but has a strong mathematical foundation that is used to construct the modeling software that creates such "decision tree" models.

When given a set of samples with many attributes, a decision tree model will identify the attributes that are best to split on and the values of those attributes that we should use to do the splitting. It will then print out a number of parameters, including number of attributes used to split, which ones, and Information Gain,... etc.

So how does modeling software decide the best tree? It tries every one and compares Information Gain for each and then picks the best one.

#Now for the Forest Decision Trees present a simple clean conceptual model to understand classification by an iterative procedure. However, in practice, a single Decision Tree is not very useful for real world problems involving a large number of variables and moderate to large sized data. For this we need heavy artillery. A group of "experts" constituting a Random Forest.

But what is an "expert" in this scenario?

If we consider our "expert" to have in their head a decision tree modeler and we assemble say 100 such experts, then, loosely speaking, we have the makings of a Random Forest. We want a collection of experts to decide our result, expecting that the result will be much better than a single one. So we will need some way to decide how to collate and sort through the "opinions".

If you recall the Olympic Gymnastic competitions or diving competitions where a panel of judges scores a participant, you might remember that the top and bottom scores are dropped and the rest are averaged. A Random Forest algorithm uses such techniques to eliminate some of the opinions but might randomly drop some percentage and then rerun the "competition", doing this each time and then averaging the result after say 100 such trials.

Why use such complicated techniques?

Well for one, they are more accurate, as mathematicaly provable. But also because, when we have 10s or 100s of attributes Random Forests are able to surface the most significant ones and use these in their modeling without any extra effort on our part. So what's the catch? This comes at some computational cost so our model may run for many minutes instead of a few seconds even with a few thousand samples, since orders of magnitude more calculations are being done. However there are many more benefits for this one cost. Random Forests are much more tolerant of missing values, bad data, and outliers, and can handle mixed data types, numerical and categorical.

We will explore a rich data set generated from the accelerometer and gyroscope of mobile phones, and use it to understand various activities of the user - such as sitting, standing, walking etc., based on particular combinations of the data attributes. Our data has more than 500 such attributes and the data is also messy and rich so this is a good candiadte for combining domain knowledge with the power of Random Forests in the exploration and analysis to follow.

Exercise

Research the following terms and learn how they relate Decision Trees to Random Forests

- Boostrapping
- Bootstrap Aggregation
- Bagging
- Boosting
- Out of Bag Errors

References [1] Decision Trees http://en.wikipedia.org/wiki/Decision_tree_learning