Random Forest Classification

Introduction

If you have played the child's game *20 Questions* where the answers are *Yes/No* (true/false) then this analogy may help you? One game of *20 Questions* is an example of one Decision Tree. Now imagine a person plays this game tens, hundreds or even thousands of times. After a number of games one should learn the *best* questions to ask and become faster and better at guessing the result. Random Forests are very much like this child's game which is played over and over.

A Random Forest classifier is a meta classifier built on nothing more than a large number of Decision Trees *grown* under certain conditions. Like a meta-analysis, the Random Forest uses a large set of previously constructed Decision Trees processed into a ensemble learning tool.

Let us start with the more basic explanation of a Classification and Regression Tree (CART) learning system. A Decision Tree is a set of nested *If. . . Then* logical questions which are designed to provide an efficient way of steering a course through a set of narrowing branches to come to a answer or description of a circumstance. The simplest decision tree has a root node which is the first question which branches or splits to only 2 offspring, daughter nodes or leaves. A leaf is simple the terminal point for the branching tree.

It is common to discuss decision trees in terms of how deep it may be built. Depending of the application, it may be advantageous to build a tree with a very small number of nodes or a fully articulated tree with every item on its own leaf allowing the decision maker to fully differentiate to every observation. For example a tree which stops after 1 or 2 nodes maybe produced in the hope of finding a strong classifier. Alternatively, fully defined trees with one item per one leaf may not classify well and may need to be pruned like a wild tree out of control. As you see the tree analogy has its roots in everyday circumstances.

The algorithm that I choose to work with utilizes the Ranger software package, which is a fast implementation of Random Forests, particularly suited for high dimensional data.[1]

Outline of Ranger Random Forest Algorithm

As mentioned earlier, a Random Forest Classifier is a Supervised learning technique. Therefore requiring a labeled pair of observations.

Data set: $(X_1, y_1), (X_2, y_2),\ ...,\ (X_N, y_N);\quad y \in \{1, ...,\ C\}$, where C is the number of classes

The Random Forest process start by choosing a sample from the original data set for a complete run. This process is called Bootstrapping. Bootstrapping uses a random number generator to sample from the set of original observations producing a new subset, i.e. drawing a sample of size N with replacement. The subset is then partitioned into further 2 sets. One set is approximately 66% in size of N and becomes the training set. The second set, which is 33% of N in size, is used as a testing set later on. Although it is possible to change this proportion in other software packages this value is 'factory set' in Ranger.

Once the training set is produced the splitting process may start. The first step in this process is to determine the number of features which will be used for optimum splitting. Using R, the hyperparameter is called the `mtry`. Using a simple grid type search it has been found that an optimum value of `mtry` is the square root of the number of features in the experiment, i.e. `mtry` $= \sqrt{M}$, M is the number of features.

The splitting process has two additional hyperparameters which can be set by the researcher. These hyperparameters are called `splitrule` and `min.node.size`. The `splitrule` determines which calculation will be used to measure the split purity. In general, there are two methods for calculating purity of splits with repsect to decision trees. One method is the *Gini coefficient* while the second common method is the *entropy coefficient.*

[1] https://cran.r-project.org/web/packages/ranger/index.html

Gini coefficient:

$$G_1 \ = \ 1 - \sum_{k=1}^{n}(X_k - X_{k-1})(Y_k + Y_{k-1})$$

Where:

- X_k is the cumulated proportion of the population variable, for $k = 0,\ldots,$ n, with $X_0 = 0$, $X_n = 1$.
- Y_k is the cumulated proportion of the income variable, for $k = 0,\ldots,$ n, with $Y_0 = 0$, $Y_n = 1$.
- Y_k should be indexed in non-decreasing order $(Y_k > Y_k - 1)$

entropy coefficient:

$$H(X) \ = \ -\sum_x P_x(x) \cdot log P_x(x)$$

Subsequently, utilizing the training set discussed above a complete Decision Tree is built to the largest extent possible, not employing pruning.

---

https://github.com/imbs-hl/ranger

CART trees

Tree-based and rule-based models are popular modeling tools for a number of reasons. First, they generate a set of conditions that are highly interpretable and are easy to implement.

Furthermore, these models can effectively handle missing data and implicitly conduct feature selection, characteristics that are desirable for many real-life modeling problems.

Two well-known weaknesses are (1) model instability (i.e., slight changes in the data can drastically change the structure of the tree or rules and, hence, the interpretation) and (2) less-than-optimal predictive performance.

If the relationship between predictors and the response cannot be adequately defined by rectangular subspaces of the predictors, then tree-based or rule-based models will have larger prediction error than other kinds of models.

Random Forest ranger Classification, Regression e1071, ranger, dplyr mtry, splitrule, min.node.size

---

meta estimator that fits a number of decision tree classifiers statistical analysis that combines the results of multiple scientific studies

- describe decision trees, nodes, leafs, daughter nodes, depth,
- gini entropy
- Trees are fast to learn and very fast for making predictions. They are also often accurate for a broad range of problems and do not require any special preparation for your data.

Classification - CART

Decision Tree Classification in R

See Youtube: Melvin L. big data

- Decision Trees using rpart library [https://cran.r-project.org/web/packages/rpart/index.html]

Limitations, considerations and alternatives: oldest Used for Regression or Classification (CART)
**Example uses Iris dataset from R.Fisher

```
* See5/C5.0 has been designed to analyze substantial databases containing thousands to millions of reco
* To maximize interpretability, See5/C5.0 classifiers are expressed as decision trees or sets of if-the
* is easy to use (easy to understand) and does not presume any special knowledge of Statistics or Machi
```