

INTERACTIVE CONSTRUCTION AND ANALYSIS OF TREES

Simon Urbanek

Department of computer oriented statistics and data analysis
University of Augsburg, 86135 Augsburg, Germany
simon.urbanek@math.uni-augsburg.de

Abstract

Tree models are a popular tool for data analysis because of their interpretability and their role as an alternative to commonly used generalized linear models. Trees are easy to explain and therefore it is natural to think of ways of involving domain experts in the model-building process. Most current algorithms for tree construction generate results automatically and do not readily permit substantial user intervention. We propose a visual interactive approach to allow the user to query and amend the tree at any point to incorporate their knowledge in the model. We introduce new graphics to support this editing process. In effect, the user is able to interactively analyze the data and the model simultaneously. This method can be applied both to classification and regression trees. We have incorporated this method in our software for visualization and analysis of trees and forests, KLIMT.

1 Introduction

Classification and regression trees provide flexible and powerful models that can be easily communicated to experts in non-statistical domains. Although the final models are easy to interpret, the commonly used methods for model construction[?] are fully automatic and only locally optimal. All possible splits in each node are considered for each covariate and the split maximizing a criterion (such as the parent/children difference of the entropy or the Gini-index) is chosen. There are no ways to monitor or influence the tree construction process or to incorpo-

rate additional knowledge. In practical applications it is often necessary to take external constraints posed on the covariates into account. Those aspects are usually not respected in the tree construction.

There are also several known properties such as the instability of individual models or variable masking that make proper assessment of tree models sometimes difficult. Those properties could be analyzed already during tree construction, but such tools are missing so far. More recently these issues have been approached by various techniques such as bagging[?], boosting[?] or random forests[?]. The general idea is to construct multiple tree models by slight changes in the algorithm or the data. Effectively the goal is to not use the locally optimal split, but to randomize or iterate.

This shows that variations in the individual splits are not only possible but even desirable. The main questions are how to include additional knowledge in the construction process, how to decide whether a certain split is stable, and how to assess the value of a split.

The idea is to visualize what the tree construction algorithm “sees” and to allow user interaction at any point. This allows us to modify existing trees to assess their quality and stability, and to include expert knowledge where necessary. The method can be used to check automatically generated models and to generate tree models semi-automatically or fully manually.

In the next section we will describe the visualization methods that provide an insight into the possible split options in a node. We also illustrate various practical aspects of the methods used on real data.

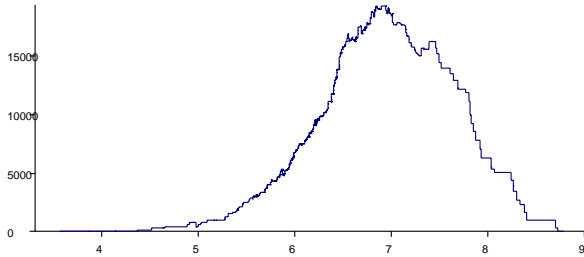


Figure 1: *Mountain plot* of the variable `rooms` for the root node of the Boston housing data with deviance as the measure of impurity (which is up to a constant equivalent to the entropy).

Finally we will conclude with some thought on future research.

2 Mountain plots and interactive splitting

The CART tree building algorithm operates on one node at a time. All covariates are examined and for each covariate all possible splits are considered. Each such split s results in two child nodes. A measure of impurity is calculated for each child and the parent. The decrease of impurity $\Delta i(s)$, that is the difference between the impurity of the parent node and the sum of impurities of the child nodes, is taken as the criterion for the quality of the split. Among all splits the one with the highest decrease of impurity is chosen. The corresponding child nodes are created and the selection procedure is repeated recursively.

Let us consider an ordinal or continuous covariate in a node. Then we can define a function $\Delta I : \mathbb{R} \rightarrow \mathbb{R}$ which for each point x takes the value $\Delta I(x) = \Delta i(s_x)$ where s_x is the split at the position x . The function is a step function which is constant in between the data points.

We can plot the function in a line plot where the x axis represents the covariate and the y axis represents $\Delta I(x)$. Such plot is shown in Fig. 1. At any given point x_i the corresponding y -value represents

the decrease of impurity if the split was placed at x_i . We gave the plot the name *mountain plot* due to its resemblance to a profile of a mountain or an entire mountain range. By definition $\Delta I(x)$ is zero for x outside the data range. The highest peak of such a mountain is the locally CART-optimal split for that variable.

We want to illustrate several properties of the mountain plots in some practical examples. We will use examples from the Boston housing data used in Breiman’s book on trees (regression trees) and the Meningitis dataset which features attributes of patients suffering from the meningitis disease (classification trees).

Since the x axis of the mountain plot corresponds to the data axis, it is possible to combine a mountain plot with the corresponding plot showing the split variable and the target variable. Such combination for a classification tree is shown in Fig. 2.

The red solid line denotes the optimal split chosen by the CART algorithm. Obviously there are four other splits represented by dotted lines, which are very close to the performance of the optimal split. The visualization allows us to see possible alternatives at a glance. In the illustrated case the alternatives are very similar in terms of the impurity criterion and are well worth considering. It is very likely that bagging and boosting would feature all four values in their iterations. It could be that the alternative splits may be better founded by the expert knowledge and it would be wise to assess their performance. In other cases the mountain range may feature a single clear peak indicating a more stable split.

The mountain plots are not restricted to comparisons among competing splits of one covariate, they can be used to compare several covariates at once. It is possible to use the same scale on the y axis for all mountain plots and place them side-by-side, as illustrated in Fig. 3. The figure features two mountain plots, one for the `Rooms` and one for the `LowStat` variable of the Boston housing dataset. It is evident that the `LowStat` variable performs almost as well as the `Rooms` variable.

Mountain plots are useful especially in the process of interactively building or modifying splits in a tree. They can be used as a tool for monitoring options and

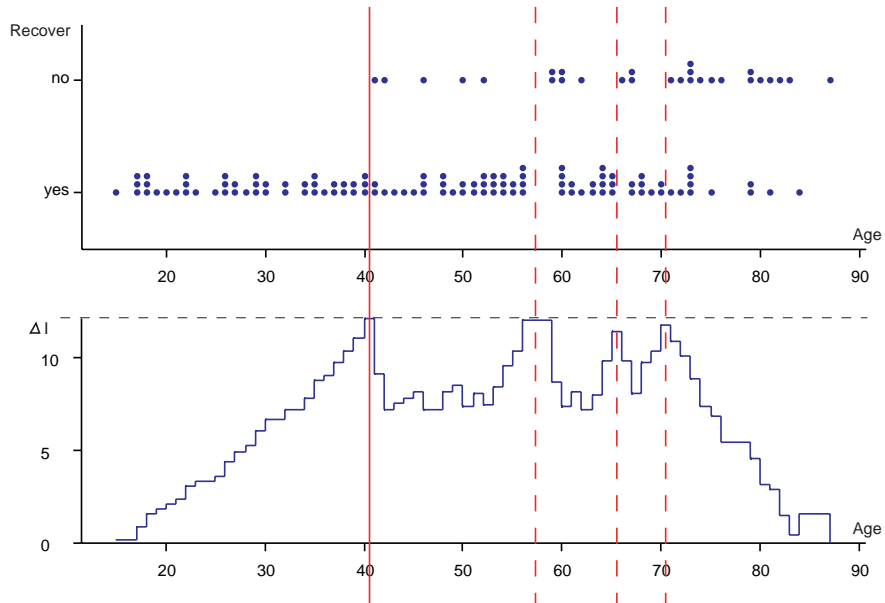


Figure 2: Stacked dotplot side-by-side of the Age variable and the target variable Recover along with the corresponding mountain plot of the root node for the variable Age from the Meningitis dataset.

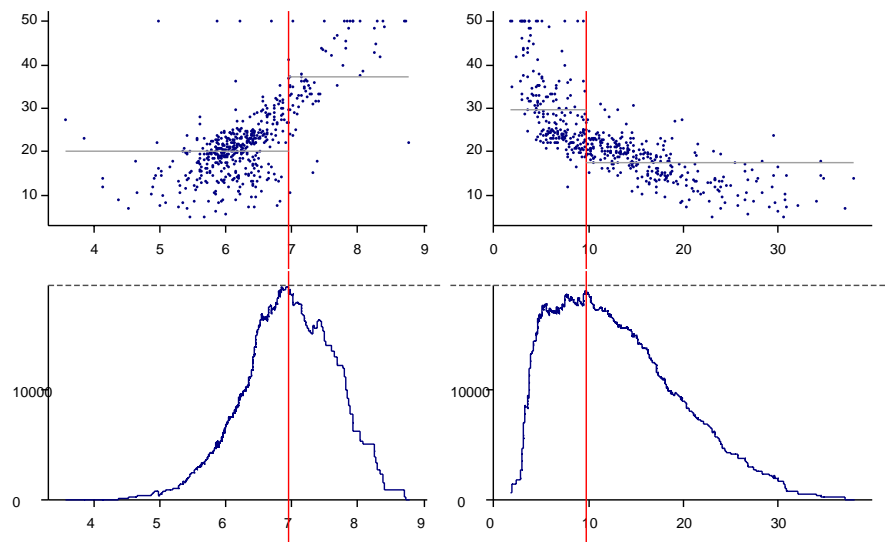


Figure 3: Two mountain plots of the variables Rooms and LowStat and the corresponding scatterplots vs the target variable. The split on Rooms just beats the LowStat split in the decrease of impurity.

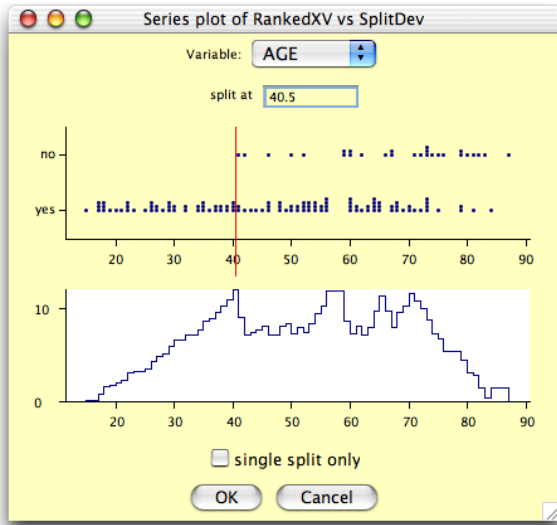


Figure 4: Interactive split editor in KLIMIT.

choices in each node. In an interactive framework it is possible to manually choose a different split directly in the plot and either grow the remaining tree automatically or continue in the investigation manually. This allows the incorporation of additional knowledge in the splitting process and the evaluation of alternate choices proposed by domain experts.

Such an interactive framework is implemented in the KLIMIT[?], software for visualization and analysis of tree models, which also provides facilities for the comparison of tree model ensembles. The corresponding split editor is shown in Fig. 4. The interface allows the user to select an arbitrary node in a tree, optionally starting from a full root node, and to cycle through the different variables available in the split. The corresponding mountain plot is displayed along with a scatterplot or dotplot of the target variable versus the splitting variable.

The user can choose to change the split and either create two child nodes or to start automatic growing of subtrees starting with the manually selected split. The resulting tree is kept along with the original tree for further analysis. It is then possible to compare the

classification behavior or the distribution of cases in terminal nodes of such a newly created tree relative to the previous tree models.

3 Conclusion

Mountain plots provide a way of analyzing individual splits in a tree. The idea is to visualize what the CART algorithm ‘sees’ in the process of finding the locally optimal split. Alternate split values or variables can be checked at a glance, providing information about the stability of the split.

Along with possibility to interactively influence the splits, the mountain plots provide a tool to incorporate expert knowledge in the model building process and to apply constraints relating to the splitting variables. The resulting trees can be analyzed and compared with the algorithmically optimal tree in order to find weaknesses or strengths of a particular model or model group.

The presented methods are applicable to any classification and regression trees. Mountain plots in current form assume ordinal or continuous variables. The extension to categorical variables is straightforward by taking all possible combinations of categories as the basis for the plot. Due to the fact that the individual splits cannot be ordered, the fixed elegant ‘walk’ across each variable is not possible.

Our future research concentrates on extending the split editor to categorical variables and to provide a statistically based measure of how ‘significant’ the difference between two splits is. A further aim is to provide a way of looking simultaneously at multiple split levels, so that we can visualize even more than what the univariate locally-optimizing algorithms can handle. This would provide an even better understanding of the structure in the data and more flexible possibilities in the construction of tree models.

References

KLIMIT-Project: <http://www.klimt-project.com>