e-PG Pathshala

Subject : Computer Science

Paper: Machine Learning

Module: Classification and Regression Trees - II

Module No: CS/ML/16

Quadrant I – e-text

Welcome to the e-PG Pathshala Lecture Series on Machine Learning. In this module we will be continuing with Classification and Regression Trees where we will be discussing tree pruning and other aspects of Classification and Regression Trees.

Learning Objectives:

The learning objectives of this module are as follows:

- To explain the tree pruning step of the CART approach
- To discuss how CART selects the Optimal Tree
- To outline regression trees used for prediction
- To differentiate between Classification and Regression Trees
- To list the advantages and disadvantages of CART

16.1 CART Steps

Let us recall the various steps of the CART methodology. They are:

1. **Initialization:** Here wecreate a tree with one node containing all the training data.
2. The next step is **Tree growing or Recursive Partitioning.** In this step a splitting criterion is first selected and then the best splits are ranked.
3. **Stop tree building:**This is the step where we complete the tree construction which happens when every aspect of the dataset is visible in decision tree
4. **Tree Pruning –** This is the final step which prunes the tree after construction.

In the previous module we had discussed the first three steps of CART. In this module we will first discuss the tree pruning step of CART.

.

16.2 Tree Pruning

One of the important fundamental problems associated with decision trees is the issue of "overfitting" of the data which in turn results in poor generalization. This essentially means that the decision tree is so detailed that it can fit the training samples but fails to generalize when a hitherto unknown samples is presented that isleads to low predictive accuracy of new data.A solution to this problem is to prune the tree. Figure 16.1 shows an example of how a decision tree can be pruned.

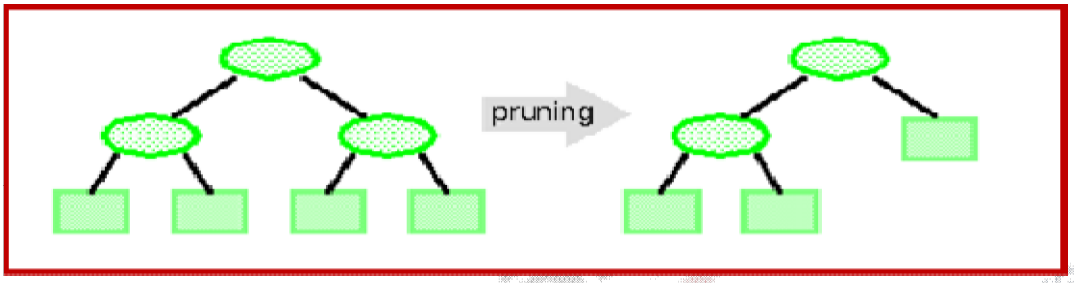

Figure 16.1 Tree Pruning

CART's philosophy is to let the decision tree grow to thefull extent, then carry out pruning that is cutting of some branches. The idea is to find that point at which the validation error begins to rise. We generate successively smaller trees by pruning leaves.
The pruning process yields a set of trees of different sizes and associated error rates. Of all the trees generated, two trees of are of particular interest: They are

- Minimum error tree: This tree has lowest error rate on validation data
- Best pruned tree: Use a set of data different from the training data to decide which is the "best pruned tree". This is the smallest tree within one standard error of minimum error. This adds a bonus for simplicity/ parsimony.

16.3 Selection of the Optimal Tree by CART

16.3.1 Growing & Pruning

One straight forward approach for CART is to stop growing the tree early instead of constructing the tree to the largest extent possible. The important question here is how to determine when to stop the growing. As we have already discussed CART just grows the tree all the way out then prunes it back. There are basically two ways in which post pruning can be carried out. Post pruning waits until the full decision tree has built and then prunes the attributes. Sequentially collapse nodes that result in the smallest change in purity or prune the "weakest link" that is prune the nodes that add least to overall accuracy of the tree. There are two techniques of pruning – sub-tree replacement and sub-tree raising (Figure 16.2). In sub-tree replacement the entire sub-tree is replaced by a single leaf node while in sub-tree raising where the entire sub-tree is raised onto another node.

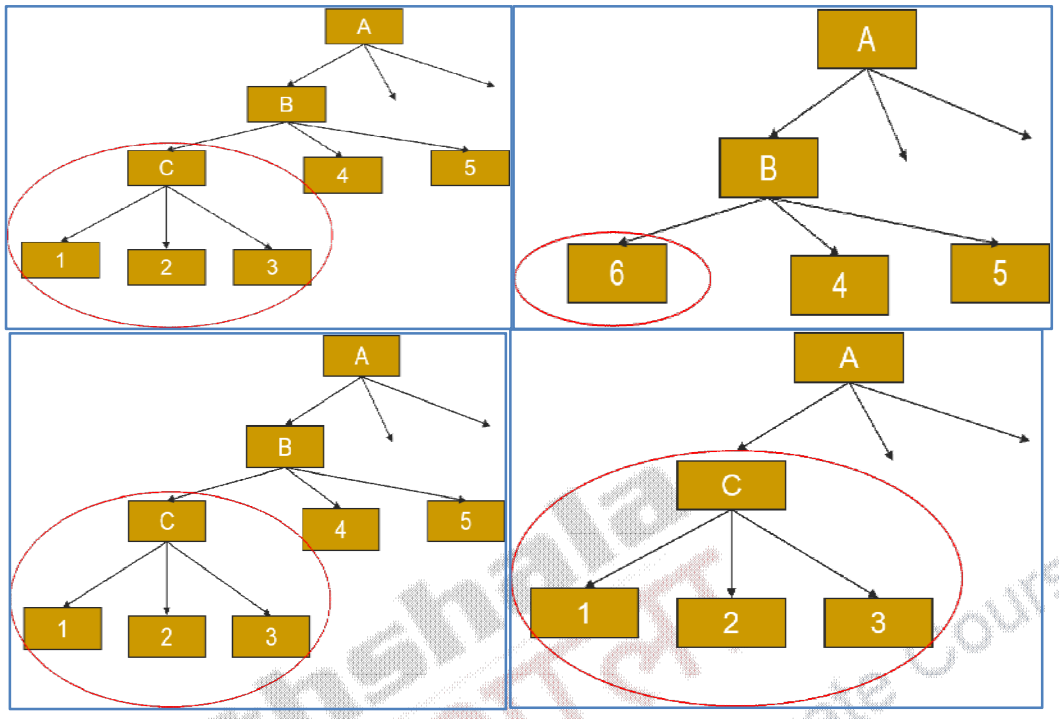

Figure 16.2 Methods of Tree Pruning

16.3.2Finding the Right Tree

"Inside every big tree is a small, perfect tree waiting to come out."--Dan Steinberg 2004. In an optimal tree training and test error rates should be similar. It is useful to compare not just overall error rate but also within node performance between training and test data. The optimal tree is selected by finding the optimal tradeoff of bias and variance. Now the question is how to find this optimal tree.At each pruning stage, multiple trees are possible and here we use *cost complexity* to choose the best tree at that stage.

16.3.3 Cost Complexity

Now let us define cost complexity. The following equation calculates the cost complexity CC(T) where T is the decision tree. *MC*is the proportion of misclassified records and α is the penalty factor attached to tree size usually set by the user. L is the number of leaves of the tree. Among trees of given size, we choose the one with lowest CC. We need to determine the CC for each size of tree.

CC(T) =*MC* + α*L*

Where MC = misclassification rate -Relative to # misclassifications in root node and *L* = # leaves (terminal nodes) .

You get a credit for lower misclassification(MC)but you also get a penalty for more leaves.Let T_0 be the biggest tree.We need to find sub-treesT_α of T_0 that minimizes CC(T). This is similar to finding an optimal trade-off betweengetting accurate results but avoiding excessive complexity.

16.3.4 Weakest-Link Pruning

Here we discuss pruning the nodes that add least to overall accuracy of the tree in other words we prune the weakest link. The sequence is determined all the way back to root node since there is a possibility that the complete tree needs to be pruned. Therefore we are sequentially pruning nodes that result in the smallest change in purity. This gives us a nested sequence of trees that are all sub-trees of T_0 that is the nested sequence of trees from root to leaf.This gives us a nested sequence of trees that are all sub-trees of some treeT_0.The sequence of sub-treesof T_0will also*have* $\boldsymbol{T_\alpha}$which is the sub-tree that minimizes CC. This simple strategy helps in finding the best tree that is we find the tree in the above sequence that minimizes misclassification rate. Contribution of a node to the overall tree is a function of both increase in accuracy and size of node. Normally the accuracy gain is weighted by the share of sample. In general, small nodes tend to get removed before large ones. αis a free parameter in CC(T)$\boldsymbol{= MC + \alpha L.}$ There is a one to one correspondence between α and size of tree. Now the issue is the value of αthat we should choose. If α=0 it means that maximum tree T_0 is best but if α=big it means thatwe will never get past the root node. The actual value of αlies somewhere in the middle and normally cross-validation is used to select the optimal value of α (size).

16.3.5 How to Cross-Validate

Cross-validation (CV) is usually used to select the optimal decision tree. This aspect is built into the CART algorithm. This method is essential to the method; and not an add-on used to reduce computational time. As already discussed the basic idea is to “grow the tree” out as far as one can and then “prune back”. Cross-validation helps us to decide when to stop pruning.

Before we go further let us first generally understand K-fold cross validation. In this method, the data set is divided into *k* subsets, and the holdout method is repeated *k* times. Each time, one of the *k* subsets is used as the test set and the other *k-1* subsets together form the training set. The k-trials are carried out and the average error across all the trials is computed. The main advantage of this method is that the way in which the test and training set are divided is immaterial since each data point gets a chance to be in the test set exactly once, and gets to be in a training set *k-1* times.

Now when we apply the 10-fold cross validation to find the optimum value of αwe grow the tree T_0on all the data. Now we break the data into 10 equal-size pieces. Then we grow a tree on 90% of the data. Now for testing we drop the remaining 10% (test data) down the nested trees corresponding to each value of α.For each αwe add up errors in all 10 of the test data sets, keeping track of theα corresponding to lowest test error. This will correspond to one of the nested trees T_k«T_0.

16.4 Regression trees

A regression tree can be seen as a constant functionused to partition the input attributespace. This is a tree-based modeling methodology for *continuous* target variable. Every record in a node is assigned such that in some way it models a *step function*(Figure 16.3).

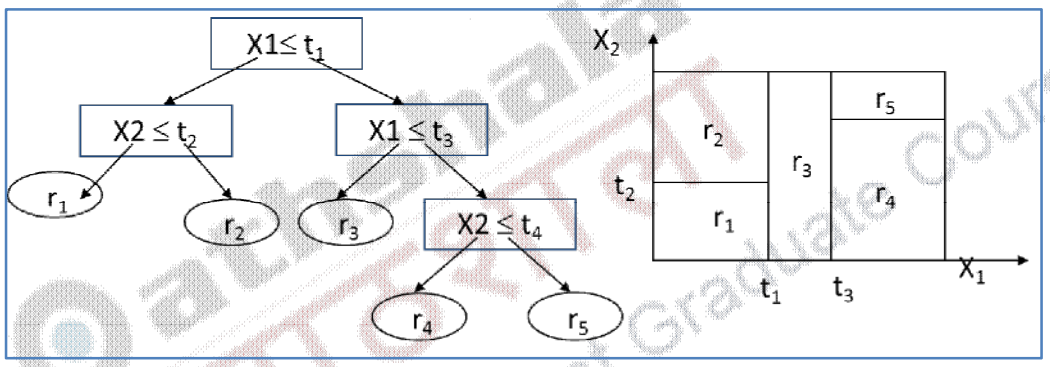

Figure 16.3 Example of a Regression Tree

16.4.1 Regression Trees for Prediction

A Regression tree is similar to a classification tree, except that the Y the target variable takes ordered values and a regression model is fitted to each node to give the predicted values of Y. We select the variable value (X=t_1) that produces the greatest "separation" in the target variable."Separation" is defined in many ways.Regression Treeswith continuous target variable normally use sum of squared errors.
In general, we find split that produces greatest separation in $\sum$**[y – E(y)]^2.** In other words we find nodes with minimal *within variance and* therefore greatest *between variance.* In other words Regression Trees partition the space into *M* regions: R_1, R_2, …, R_M.

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

$$where \quad , \qquad \hat{c}_m = average \quad (y_i \mid x_i \in R_m)$$

The best partition is the one that minimizes the sum of squared error:

$$\sum_{i=1}^{N} (y_i - f(x_i))^2$$

However finding the global minimum is computationally infeasible. Therefore a greedy algorithm is used at each level that chooses variable *j* and value *s* as:

$$\arg\min_{j,s} [\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2]$$

However the greedy algorithm makes the tree unstable. In other words, the error made at the upper level will be propagated to the lower level

The tree for regression uses exactly the same model as classification tree but with a number in each leaf instead of a classwhere in this case we are predicting whether a player will play or not (Figure 16.4).

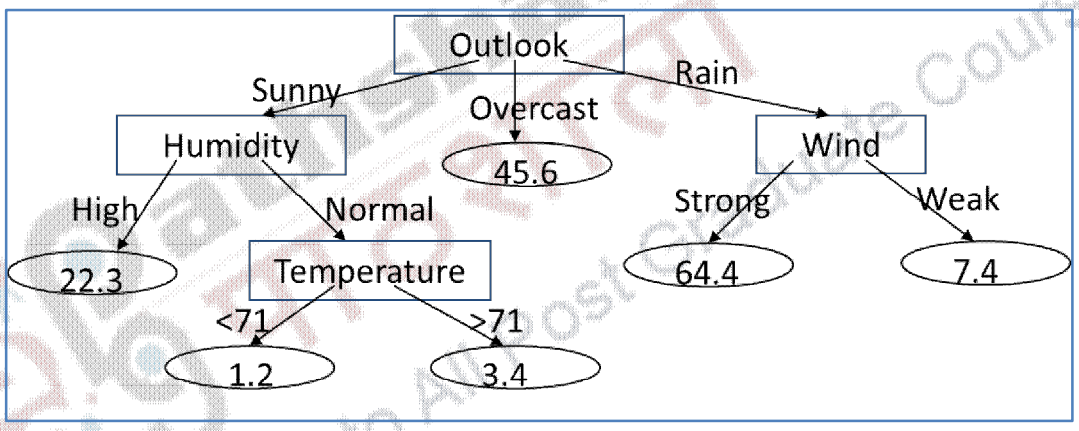

Figure 16.4 Another Example of a Regression Tree

16.4.2 Regression Tree – how large should we grow the tree?

As was discussed for classification trees here again we need to trade-off between bias and variance. If we have a very large tree there will be overfit (low bias, high variance) but however a small tree (low variance, high bias) might not capture the structure of the attribute values. The general strategies used are:

- split only when we can decrease the error (usually short-sighted)
- Cost-complexity pruning – discussed in the previous sections (preferred)

16.4.3 Regression tree pruning

Exactly the same algorithms that are used for classification trees can be used for regression trees and we can use both pre-pruning and post-pruning.In post-pruning, the tree that minimizes the squared error is selected.In practice, pruning is more important in regression because full trees are much more

complex (often all objects have different output values and hence the full tree has as many leaves as there are objects in the learning sample)

16.5 Classification Trees vs. Regression Trees

Classification trees are tree-based modeling for discrete target variable**.** In contrast with regression trees, various measures of *purity* for splitting such as Gini, entropy, “twoing” etc. are used. On the other hand regression trees are used for modelling continuous target variable. The splitting criteria used for regression trees is generally based on minimizing the sum of squared errors. The goodness of fit measure used to find the best tree in the case of classification trees are prior probabilities and misclassification costs and are used as model tuning parameters. In the case of regression trees no priors or misclassification costs are used and the same measure (namely sum of squared errors) is used as goodness of fit measure.

16.6 Advantages of CART

One of the main advantages of CART is that it can cope with any data structure or type. The classification tree has a simple and understandable form. CART is able to use conditional information effectively. CART techniques are invariant under transformations of the variables and are robust with respect to outliers that is they are not sensitive to outliers in *predictive* variables. Classification trees give an estimate of the misclassification rate and moreover is nonparametric that is makes no probabilistic assumptions. CART automatically performs variable selection and can use any combination of continuous and or discrete variables. CART has the ability to automatically bin massively categorical variables into a few categories -zip code, business class, make/model…etc.. It discovers “interactions” among variables and is good for “rules” search. CART is able to handle missing values automatically using the concept of “surrogate splits”. In general CART provides a good way to explore as well as visualize data.

16.7 Disadvantages of CART

CART does not use combinations of variables and does not consider dependencies between variables. Trees can be deceptive if the variable is not included because it was obscured by another variable. Tree structures may be unstable since a change in the samples may give different trees. Tree is optimal at each split that is making the tree locally optimal may not always result in global optimum.CART does a poor job of modeling *linear structure.* The model is a *step function*, not a continuous score and therefore if a tree has 10 nodes, it can only take on 10 possible values. It might take a large tree to find a good fit. However if the tree is large and complex, it may be hard to interpret. Moreover during pruning data gets chopped thinner at each split. As already explained

there may be instability of model structure where correlated variables may cause random data fluctuations that could result in entirely different trees.

16.7 Uses of CART

CART is used for building predictive models as an alternative to Generalized Linear Models (GLMs), neural nets, etc. It can also be used for exploratory data analysis where a tree can be built on nearly any data set with minimal data preparation. CART can be used for variable selection where it can rank variables and act as an alternative to stepwise regression.

Summary

- Classification and Regression Trees are an easily understandable and transparent method for predicting or classifying new records
- A tree is a graphical representation of a set of rules
- Trees must be pruned to avoid over-fitting of the training data
- As trees do not make any assumptions about the data structure, they usually require large samples