Self-Pruning Prediction Trees (Abstract)

Sally Goldman
Google Research
1600 Ampitheatre Parkway
Mountain View, CA 94043
sgoldman@google.com

Yoram Singer
Google Reserach
1600 Ampitheatre Parkway
Mountain View, CA 94043
singer@google.com

February 10, 2010

Decision trees and regression trees are well-studied and widely used in practice (Breiman et al., 1984; Quinlan, 1986). The learning process of decision trees typically consists of two parts: a growing phase in which nodes are added to the tree based on a prediction gain, and a pruning phase in which the tree size is reduced in order to guard from over-fitting and provide good generalization. A variety of pruning methods have been proposed and analyzed (see for instance (Kearns and Mansour, 1996; Mansour, 1997)). Most pruning methods however are not directly related to the growing phases.

In this paper we define a generalization of a decision tree that we call a *Self-pruning Prediction Tree* (SPT). Each node of a SPT is associated with a real valued prediction. Given an input instance, the prediction of a SPT is formed by summing the individual prediction at the nodes traversed from the root node to a leaf by applying a sequence of branching predicates. The probability associated with a target is obtained by applying an exponential model over the aforementioned sum. (See below for a formal definition.) Alternatively, a SPT can be viewed as a piece-wise *constant* function from the input space into the reals. We associate with each SPT a notion of complexity that amounts to the total function variation using the latter view. The hierarchical structure of a SPT enables efficient computation of its variation value which facilitates an efficient pruning procedure that is tightly coupled with the growing phase. Our construction is novel and the resulting self-pruning algorithm has not been proposed before. Nonetheless, this work builds upon and distills ideas from statistics (Breiman et al., 1984), information theory (Willems et al., 1995), and learning theory (Helmbold and Schapire, 1997).

A prediction tree (PT) is a generalization decision tree in which each node x has associated with it both a variable v_x used for branching as in a standard decision tree, and a real value α_x. Here we focus on binary predictions, however our definition can be easily extended for regression problems. For any node x in the prediction tree, let P_x be the nodes along the path from the root to x, and let $b(x) = \sum_{i \in P_x} \alpha_i$. For any node, the prediction tree defines a probablity that an example reaching this node is positive. Specifically, for y the label of an example, and x an arbitrary node in the prediction tree, $\Pr(y = +1|x) = 1/(1 + e^{-b(x)})$ and $\Pr(y = -1|x) = 1/(1 + e^{b(x)})$.

For a prediction tree T we define the *variation complexity* $V(T)$ as $\sum_{x \in T} \left\| \alpha_{C(x)} \right\|_p$ where $C(x)$ is the set of children of x, and by convention $\alpha = 0$ for a null child. So for $p = 1$, $V(T) = \sum_{x \in T} \sum_{x' \in C(x)} \alpha_{x'} = \sum_{x \in T} \alpha_x$, and for $p = \infty$, $V(T) = \sum_{x \in T} (\max_{x' \in C(x)} \alpha_{x'})$. Let f_T be the function represented by prediction tree T. Our goal is to construct a tree that minimizes the logistic loss $\mathcal{L}$ with a ℓ_p regularization applied to the complexity of T. That is, we aim to minimize $\mathcal{L} = \sum_{i=1}^{m} \left[w_i \log \left(1 + e^{-y_i \cdot f_T(x)} \right) \right] + \lambda \left\| V(T) \right\|_p$ where m is the size of the training set, $y_i \in \{-1, +1\}$ is the label for example i, and w_i is the weight associated with example

i where we assume that $\sum_{i=1}^{m} w_i = 1$.

As in the standard greedy tree building algorithm, we select the variable to place in the root, and then recursively apply the same procedure to all newly added nodes. However, the optimization used to select the variable to place in the root simultaneously determines the value α_j for each the branches defined by the selected variable. For each branch for which $\alpha_j \neq 0$, this process is recursively applied. A key contribution of our algorithm is that the regularizer used in our obective function determines when to stop growing the tree. Furthermore, the regularization constant λ can be viewed as a control for the degree of "sparsity" for the prediction tree.

Consider a leaf node x that we are considering expanding. We select the variable v to replace leaf x by selecting the one which minimizes the loss. We now derive the loss obtained when x is replaced by the k-ary variable v, which in turn will create children with real-values $\alpha_1, \ldots, \alpha_k$. Let $w_{i,j} = w_i$ when example i goes to branch j (which has associated value α_j), and $w_{i,j} = 0$ otherwise. We compute the values of the α by minimizing the logistic loss $\mathcal{L}$ with a ℓ_p regularizer: $\mathcal{L} = \sum_{j=1}^{k} \sum_{i=1}^{m} w_{i,j} \log(1 + e^{-y_i(\alpha_j + b(x))}) + \lambda \left\|\alpha\right\|_p$.

Since we are focusing on the task of replacing node x, we let $b = b(x)$, and let $\mu_j^+ = \sum_{y_i \geq 0} w_{i,j}$ and let $\mu_j^- = \sum_{y_i < 0} w_{i,j}$. Thus $\mathcal{L} = \sum_{j=1}^{k} [\mu_j^+ \log(1 + e^{-(\alpha_j + b)}) + \mu_j^- \log(1 + e^{(\alpha_j + b)})] + \lambda \left\|\alpha\right\|_p$.

It can be shown that for dual $\mathcal{Q}$ where γ_j is the dual variable associated with α_j,

$$-\mathcal{Q} = \sum_j \mu_j H_2 \left(\frac{\mu_j^+ - \gamma_j}{\mu_j} \right) + b \sum_j \gamma_j$$

with the constraint that $\left\|\gamma\right\|_q \leq \lambda$ where q is the dual norm of p. Observe that when $\gamma_j = 0$, our optimization criteria becomes the standard weighted entropy-based information gain.

The key contribution of our work, in addition to the introduction of the the self-pruning prediction tree, is our nearly-linear time algorithm to efficiently compute the optimal αs for each node in the prediction tree using either the ℓ_1 regularizer ($p = 1$) or an ℓ_∞ regularizer ($p = \infty$) in defining the tree complexity. At a high-level, our algorithm works by computing the optimal solution $\gamma_1^*, \ldots, \gamma_k^*$ for the dual, and then maps back to the primal solution using $\alpha_j = \log \left(\frac{\mu_j^+ - \gamma_j}{\mu_j^- + \gamma_j} \right) - b$. Furthermore, we believe that our techniques is quite general, and can be extended for regression problems.

References

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth & Brooks, 1984.

David P. Helmbold and Robert E. Schapire. Predicting nearly as well as the best pruning of a decision tree. *Machine Learning*, 27(1):51–68, April 1997.

Michael Kearns and Yishay Mansour. A fast, bottom-up decision tree pruning algorithm with near-optimal generalization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1996.

Yishay Mansour. Pessimistic decision tree pruning based on tree size. In *ml97*, pages 195–201, 1997.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens. The context tree weighting method: basic properties. *IEEE Transactions on Information Theory*, 41(3):653–664, 1995.