Classification and Trees*

Luc Devroye

School of Computer Science, McGill University
Montreal, Canada H3A 2K6

This paper is dedicated to the memory of Pierre Devijver.

Abstract. Breiman, Friedman, Gordon and Stone recognized that tree classifiers would be very valuable to practicing statisticians. Their CART algorithm became very popular indeed. Designing tree-based classifiers, however, has its pitfalls. It is easy to make them too simple or too complicated so that Bayes risk consistency is compromised. In this talk, we explore the relationship between algorithmic complexity of tree-based methods and performance.

Extended Abstract

In scientific applications, the dual objective of a classification method is to classify and explain. It is this argument that makes partition methods interesting—these are methods in which the space is split up into disjoint sets. On each set, classification is performed by a simple majority vote. More formally, if $(X, Y) \in R^d \times \{0, 1\}$ is the unknown underlying distribution of a datum (X) and its class (Y), and the data consist of independent identically distributed copies of (X, Y), namely $D_n = ((X_1, Y_1), \ldots, (X_n, Y_n))$, then a classifier is an estimator $g_n(X, D_n) \in \{0, 1\}$ of Y, and the probability of error is

$$L_n = \mathbb{P}\{g_n(X, D_n) \neq Y | D_n\}.$$

The Bayes error $L^* = \inf_g \mathbb{P}\{g(X) \neq Y\}$ is the smallest error one can hope to obtain. If g were known, then we could consider consider the partition into $A = \{x \in R^d : g(x) = 1\}$ (the unknown Bayes discriminant set) and its complement. And indeed, most classifiers can be considered as partition classifiers for the partition (A_n, A_n^c), where $A_n = \{x \in R^d : g_n(x) = 1\}$, where A_n is an approximation of A.

What matters however is the explanatory aspect—how can the partition be described and constructed? For example, histograms based on regular grids could be considered, but they suffer from several drawbacks—first and foremost, they ignore the possible clustering in the data, and secondly, they can hardly be called instructional tools for explaining data. Thirdly, even modest dimensions quickly lead to histograms with underpopulated cells.

* The author's research was sponsored by NSERC Grant A3456.

E.R. Hancock et al. (Eds.): SSPR & SPR 2010, LNCS 6218, pp. 40–44, 2010.

This has led many researchers to consider smart and simple partitions. The linear discriminant, and the perceptron (Rosenblatt, 1962), are based upon partitions by hyperplanes. The question is whether we have universality, i.e., does $L_n \to L^*$ in probability as $n \to \infty$ for any distribution of (X, Y)? The answer is negative if one linear discriminant is used—how can it hope to get close to the unknown Bayes discriminant set A, which can be of arbitrary form? But the answer is affirmative, provided that linear discriminants are cleverly used as building blocks.

Buoyed by the intriguing comparison between brain function and learning machines, early methods of classification often involved combinations of linear discriminants. For example, in the committee machine (Nilsson, 1965), many linear discriminants are considered, each delivering a vote to each halfspace (to class "one" on one side, and to the "zero" class on the other side). For a particular X, its votes are totalled to make a final decision. Neural networks can be considered as smooth generalizations of this simple machine. Grid histograms can be considered as generalizations of committee machines in which all separating hyperplanes are aligned with the axes and regularly spaced—they have more degrees of freedom though. While not important in high dimensions, grid histograms do have one salient feature—they lead to universally consistent rules provided that the grid cell sizes shrink to zero with n and the average number of points per cell tends to infinity with n. The question then is whether committee machines are universally consistent. For example, we optimize k hyperplanes by minimizing the errors on the data, and if $k \to \infty$ and $k = o(n)$, then one would expect universal consistency. However, this is unknown (see Problem 30.6 in Devroye, Györfi and Lugosi, 1996).

Partitions based on hyperplanes are called arrangements. All arrangements in turn can be emulated by trees in which decisions are made by verifying signs of linear expressions. This leads naturally to tree classifiers. Each node in such a classifier makes a decision based on whether x is in certain set (ball, halfspace, simplex, hyperrectangle, and so forth) or not. Leaves in the tree correspond to sets in a partition.

Tree classifiers come in many flavors. One can cross-categorize by the style of partition. At the top of the list are the linear partitions perpendicular to the axes, which we shall call orthogonal cuts. They were preferred in the popular CART method (Breiman, Friedman, Olshen, Stone, 1984) because of the easy way in which classifiers can be explained, one variable (coordinate) at a time. Trees obtained by consecutive orthogonal cuts are called k-d trees in the computer science literature. Linear cuts lead to so-called hyperplane search trees—they too were proposed in the 1970s. Occasionally, one finds partitions by membership in simplices and rectangles.

More fundamental is the type of information used to create the tree-based partition. If only the X_i's are used, the partition can at best attempt to keep close points together, hoping that the joint distribution of (X, Y) shows some smoothness. Yet, ignoring the Y_i's has its advantages. For example, one obtains universal consistency under the following simple (and optimal) conditions. Let

C be the cell of the partition in which a random datum X falls, and let its diameter and cardinality be D and N, respectively. Then $D \to 0$ and $N \to \infty$ in probability suffice (Devroye, Györfi and Lugosi, 1996, p. 94). An example includes the median tree partition—split each coordinate in turn at the median of the X_i's, until each leaf cell has about k points. Then $k \to \infty$ and $k = o(n)$, and the existence of a density for X are the only conditions needed (Devroye, Györfi and Lugosi, 1996, p. 323).

However, ignoring the Y_i's is against human nature. For data on the real line ($d = 1$), there is an optimal binary split that minimizes the error on the data itself, which we shall call a Stoller (after Stoller, 1954). The Stoller split can be used in any direction, and indeed, one could consider the best linear or orthogonal split for $d > 1$. However, even today, we are missing simple theorems with easy-to-check conditions for universal consistency when trees, or partitions, are based on D_n in general. There are indeed many pitfalls that lead to inconsistency.

Thirdly, and perhaps most importantly, tree classifiers can be categorized by the algorithm used in their construction. Consider first top-down constructions, in which we keep splitting leaves until we are satisfied. One can optimize a criterion at each step, choosing a leaf that is most promising to split. However, selecting a leaf that yields the best orthogonal Stoller split at each step is not good enough—it is generally not consistent (Devroye, Györfi and Lugosi, 1996, p. 335). One can however, remedy this by finding the best Stoller split of a leaf using hyperrectangles as separators (Devroye, Györfi and Lugosi, 1996, chapter 20.13). This is the greedy approach to design. At the other end of the spectrum is the one-shot design of a tree of a given complexity (number of nodes) using optimization, such as minimization of the error on the data. This is phenomenally expensive, but its consistency is usually easy to guaranteee thanks to powerful inequalities for the empirical measures of sets initially derived by Vapnik and Chervonenkis (1971).

Bottom-up strategies first make a fine partition, e.g., a partition in which each final cell has one X_i. Then, in a second step, cells are recombined in a given fashion. CART follows this approach.

Ensemble methods, popular in machine learning, are learning algorithms that construct a set of many individual classifiers (called base learners) and combine them to classify new data points by taking a weighted or unweighted vote of their predictions. It is now well-known that ensembles are often much more accurate than the individual classifiers that make them up. The success of ensemble algorithms on many benchmark data sets has raised considerable interest in understanding why such methods succeed and identifying circumstances in which they can be expected to produce good results. These methods differ in the way the base learner is fit and combined. For example, bagging (Breiman, 1996) proceeds by generating bootstrap samples from the original data set, constructing a classifier from each bootstrap sample, and voting to combine. In boosting (Freund and Shapire, 1996) and arcing algorithms (Breiman, 1991) the successive classifiers are constructed by giving increased weight to those points that have been frequently misclassified, and the classifiers are combined using weighted

voting. For a comprehensive review of ensemble methods, we refer the reader to Dietterich (2000).

Breiman (2001) provides a general framework for tree ensembles called "random forests". Each tree depends on the values of a random vector sampled independently and with the same distribution for all trees. Thus, a random forest is a classifier that consists of many decision trees and outputs the class that is the mode of the classes output by individual trees.

Random forests have been shown to give excellent performance on a number of practical problems. They work fast, generally exhibit a substantial performance improvement over single tree classifiers such as CART, and yield generalization error rates that compare favorably to the best statistical and machine learning methods.

Different random forests differ in how randomness is introduced in the tree building process, ranging from extreme random splitting strategies (Breiman (2000), Cutler and Zhao (2001)) to more involved data-dependent strategies (see Amit and Geman (1997), Breiman (2001), or Dietterich (2000)). Some attempts to investigate the consistency of random forests are by Breiman (2000, 2004), and Lin and Jeon (2006), who establish a connection between random forests and adaptive nearest neighbor methods.

Many examples of Breiman-style random forests are analyzed by Biau, Devroye and Lugosi (2008), and Biau and Devroye (2008). For example, sample k data from D_n uniformly at random, make a random k-d tree in a certain way and split until X has exactly one X_i in its cell. Record the vote Y_i. Now repeat many times and estimate Y by a majority rule. Without repetition, there is no consistency, but averaging the votes leads under some conditions to consistent rules.

References

[1] Amit, Y., Geman, D.: Shape quantization and recognition with randomized trees. Neural Computation 9, 1545–1588 (1997)

[2] Biau, G., Devroye, L., Lugosi, G.: Consistency of random forests and other averaging classifiers. Journal of Machine Learning Research 9, 2015–2033 (2008)

[3] Biau, G., Devroye, L.: On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification. Technical Report (2008)

[4] Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)

[5] Breiman, L.: Arcing classifiers. The Annals of Statistics 24, 801–849 (1998)

[6] Breiman, L.: Some infinite theory for predictor ensembles. Technical Report 577, Statistics Department, UC Berkeley (2000), `http://www.stat.berkeley.edu/~breiman`

[7] Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)

[8] Breiman, L.: Consistency for a simple model of random forests. Technical Report 670, Statistics Department, UC Berkeley (2004), `http://www.stat.berkeley.edu/~breiman`

[9] Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. CRC Press, Boca Raton (1984)

[10] Cutler, A., Zhao, G.: Pert – Perfect random tree ensembles. Computing Science and Statistics 33, 490–497 (2001)
[11] Devroye, L., Györfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. Springer, New York (1996)
[12] Dietterich, T.G.: An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Machine Learning 40, 139–157 (2000)
[13] Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
[14] Freund, Y., Shapire, R.: Experiments with a new boosting algorithm. In: Saitta, L. (ed.) Machine Learning: Proceedings of the 13th International Conference, pp. 148–156. Morgan Kaufmann, San Francisco (1996)
[15] Lin, Y., Jeon, Y.: Random forests and adaptive nearest neighbors. Journal of the American Statistical Association 101, 578–590 (2006)
[16] Nilsson, N.J.: Learning Machines: Foundations of Trainable Pattern Classifying Systems. McGraw-Hill, New York (1965)
[17] Rosenblatt, F.: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington (1962)
[18] Stoller, D.S.: Univariate two-population distribution-free discrimination. Journal of the American Statistical Association 49, 770–777 (1954)
[19] Vapnik, V.N., Chervonenkis, A.: On the uniform convergence of relative frequencies of events to their probabilities. Theory of Probability and its Applications 16, 264–280 (1971)