Chapter 9

DECISION TREES

Lior Rokach
Department of Industrial Engineering
Tel-Aviv University
liorr@eng.tau.ac.il

Oded Maimon
Department of Industrial Engineering
Tel-Aviv University
maimon@eng.tau.ac.il

Abstract Decision Trees are considered to be one of the most popular approaches for representing classifiers. Researchers from various disciplines such as statistics, machine learning, pattern recognition, and Data Mining have dealt with the issue of growing a decision tree from available data. This paper presents an updated survey of current methods for constructing decision tree classifiers in a top-down manner. The chapter suggests a unified algorithmic framework for presenting these algorithms and describes various splitting criteria and pruning methodologies.

Keywords: Decision tree, Information Gain, Gini Index, Gain Ratio, Pruning, Minimum Description Length, C4.5, CART, Oblivious Decision Trees

1. Decision Trees

A decision tree is a classifier expressed as a recursive partition of the instance space. The decision tree consists of nodes that form a *rooted tree*, meaning it is a *directed tree* with a node called "root" that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an *internal* or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values. In the simplest and most frequent case, each test considers a single attribute, such that the instance space is partitioned according to the attribute's value. In the case of numeric attributes, the condition refers to a range.

Each leaf is assigned to one class representing the most appropriate target value. Alternatively, the leaf may hold a probability vector indicating the probability of the target attribute having a certain value. Instances are classified by navigating them from the root of the tree down to a leaf, according to the outcome of the tests along the path. Figure 9.1 describes a decision tree that reasons whether or not a potential customer will respond to a direct mailing. Internal nodes are represented as circles, whereas leaves are denoted as triangles. Note that this decision tree incorporates both nominal and numeric attributes. Given this classifier, the analyst can predict the response of a potential customer (by sorting it down the tree), and understand the behavioral characteristics of the entire potential customers population regarding direct mailing. Each node is labeled with the attribute it tests, and its branches are labeled with its corresponding values.

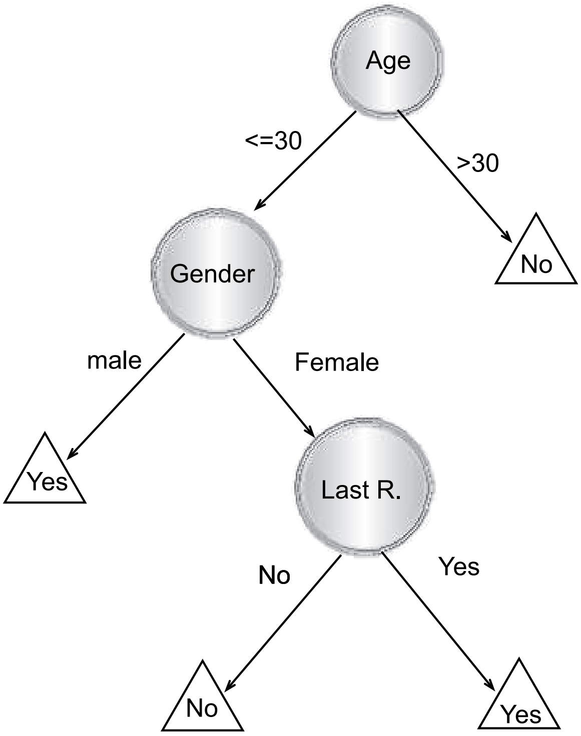

Figure 9.1. Decision Tree Presenting Response to Direct Mailing.

In case of numeric attributes, decision trees can be geometrically interpreted as a collection of hyperplanes, each orthogonal to one of the axes. Naturally, decision-makers prefer less complex decision trees, since they may be considered more comprehensible. Furthermore, according to Breiman *et al.* (1984) the tree complexity has a crucial effect on its accuracy. The tree complexity is explicitly controlled by the stopping criteria used and the pruning method employed. Usually the tree complexity is measured by one of the following metrics: the total number of nodes, total number of leaves, tree depth and number of attributes used. Decision tree induction is closely related to rule induction. Each path from the root of a decision tree to one of its leaves can be transformed into a rule simply by conjoining the tests along the path to form the antecedent part, and taking the leaf's class prediction as the class value. For example, one of the paths in Figure 9.1 can be transformed into the rule: "If customer age is is less than or equal to or equal to 30, and the gender of the customer is "Male" – then the customer will respond to the mail". The resulting rule set can then be simplified to improve its comprehensibility to a human user, and possibly its accuracy (Quinlan, 1987).

2. Algorithmic Framework for Decision Trees

Decision tree inducers are algorithms that automatically construct a decision tree from a given dataset. Typically the goal is to find the optimal decision tree by minimizing the generalization error. However, other target functions can be also defined, for instance, minimizing the number of nodes or minimizing the average depth.

Induction of an optimal decision tree from a given data is considered to be a hard task. It has been shown that finding a minimal decision tree consistent with the training set is NP–hard (Hancock *et al.*, 1996). Moreover, it has been shown that constructing a minimal binary tree with respect to the expected number of tests required for classifying an unseen instance is NP–complete (Hyafil and Rivest, 1976). Even finding the minimal equivalent decision tree for a given decision tree (Zantema and Bodlaender, 2000) or building the optimal decision tree from decision tables is known to be NP–hard (Naumov, 1991).

The above results indicate that using optimal decision tree algorithms is feasible only in small problems. Consequently, heuristics methods are required for solving the problem. Roughly speaking, these methods can be divided into two groups: top–down and bottom–up with clear preference in the literature to the first group.

There are various top–down decision trees inducers such as ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993), CART (Breiman *et al.*, 1984). Some consist of two conceptual phases: growing and pruning (C4.5 and CART). Other inducers perform only the growing phase.

Figure 9.2 presents a typical algorithmic framework for top–down inducing of a decision tree using growing and pruning. Note that these algorithms are greedy by nature and construct the decision tree in a top–down, recursive manner (also known as "divide and conquer"). In each iteration, the algorithm considers the partition of the training set using the outcome of a discrete function of the input attributes. The selection of the most appropriate function is made according to some splitting measures. After the selection of an appropriate split, each node further subdivides the training set into smaller subsets, until no split gains sufficient splitting measure or a stopping criteria is satisfied.

3. Univariate Splitting Criteria

3.1 Overview

In most of the cases, the discrete splitting functions are univariate. Univariate means that an internal node is split according to the value of a single attribute. Consequently, the inducer searches for the best attribute upon which to split. There are various univariate criteria. These criteria can be characterized in different ways, such as:

- According to the origin of the measure: information theory, dependence, and distance.

- According to the measure structure: impurity based criteria, normalized impurity based criteria and Binary criteria.

The following section describes the most common criteria in the literature.

3.2 Impurity-based Criteria

Given a random variable x with k discrete values, distributed according to $P = (p_1, p_2, \ldots, p_k)$, an impurity measure is a function ϕ:[0, 1]^k $\rightarrow R$ that satisfies the following conditions:

- ϕ (P)$\geq$0

- ϕ (P) is minimum if $\exists$i such that component p$_i$ = 1.

- ϕ (P) is maximum if $\forall$i, 1 $\leq$ i $\leq$ k, p$_i$ = 1/k.

- ϕ (P) is symmetric with respect to components of P.

- ϕ (P) is smooth (differentiable everywhere) in its range.

```
TreeGrowing (S,A,y)
Where:
S - Training Set
A - Input Feature Set
y - Target Feature
Create a new tree T with a single root node.
IF One of the Stopping Criteria is fulfilled THEN
    Mark the root node in T as a leaf with the most
    common value of y in S as a label.
ELSE
    Find a discrete function f(A) of the input
        attributes values such that splitting S
        according to f(A)'s outcomes ($v_1$,...,$v_n$) gains
        the best splitting metric.
    IF best splitting metric $>$ treshold THEN
        Label t with f(A)
        FOR each outcome $v_i$ of f(A):
            Set Subtree$_i$= TreeGrowing ($\sigma_{f(A)=v_i}S$,A,y).
            Connect the root node of $t_T$ to Subtree$_i$ with
                    an edge that is labelled as $v_i$
        END FOR
    ELSE
        Mark the root node in T as a leaf with the most
            common value of y in S as a label.
    END IF
END IF
RETURN T
TreePruning (S,T,y)
Where:
S - Training Set
y - Target Feature
T - The tree to be pruned
DO
    Select a node t in T such that pruning it
        maximally improve some evaluation criteria
    IF t$\neq$Ø THEN T=pruned(T,t)
UNTIL t=Ø
RETURN T
```

Figure 9.2. Top-Down Algorithmic Framework for Decision Trees Induction.

Note that if the probability vector has a component of 1 (the variable x gets only one value), then the variable is defined as pure. On the other hand, if all components are equal, the level of impurity reaches maximum.

Given a training set S, the probability vector of the target attribute y is defined as:

$$P_y(S) = \left(\frac{|\sigma_{y=c_1} S|}{|S|}, \ldots, \frac{\left|\sigma_{y=c_{|dom(y)|}} S\right|}{|S|} \right)$$

The goodness–of–split due to discrete attribute a_i is defined as reduction in impurity of the target attribute after partitioning S according to the values $\mathrm{v_{i,j}} \in dom(a_i)$:

$$\Delta\Phi(a_i, S) = \phi(P_y(S)) - \sum_{j=1}^{|dom(a_i)|} \frac{|\sigma_{a_i=v_{i,j}} S|}{|S|} \cdot \phi(P_y(\sigma_{a_i=v_{i,j}} S))$$

3.3 Information Gain

Information gain is an impurity-based criterion that uses the entropy measure (origin from information theory) as the impurity measure (Quinlan, 1987).

$$InformationGain(a_i, S) =$$
$$Entropy(y, S) - \sum_{v_{i,j} \in dom(a_i)} \frac{|\sigma_{a_i=v_{i,j}} S|}{|S|} \cdot Entropy(y, \sigma_{a_i=v_{i,j}} S)$$

where:

$$Entropy(y, S) = \sum_{c_j \in dom(y)} -\frac{\left|\sigma_{y=c_j} S\right|}{|S|} \cdot \log_2 \frac{\left|\sigma_{y=c_j} S\right|}{|S|}$$

3.4 Gini Index

Gini index is an impurity-based criterion that measures the divergences between the probability distributions of the target attribute's values. The Gini index has been used in various works such as (Breiman *et al.*, 1984) and (Gelfand *et al.*, 1991) and it is defined as:

$$Gini(y, S) = 1 - \sum_{c_j \in dom(y)} \left(\frac{\left|\sigma_{y=c_j} S\right|}{|S|} \right)^2$$

Consequently the evaluation criterion for selecting the attribute a_i is defined as:

$$GiniGain(a_i, S) = Gini(y, S) - \sum_{v_{i,j} \in dom(a_i)} \frac{\left|\sigma_{a_i=v_{i,j}} S\right|}{|S|} \cdot Gini(y, \sigma_{a_i=v_{i,j}} S)$$

3.5 Likelihood-Ratio Chi–Squared Statistics

The likelihood–ratio is defined as (Attneave, 1959)

$$G^2(a_i, S) = 2 \cdot \ln(2) \cdot |S| \cdot InformationGain(a_i, S)$$

This ratio is useful for measuring the statistical significance of the information gain criterion. The zero hypothesis (H_0) is that the input attribute and the target attribute are conditionally independent. If H_0 holds, the test statistic is distributed as χ^2 with degrees of freedom equal to: $(dom(a_i) - 1) \cdot (dom(y) - 1)$.

3.6 DKM Criterion

The DKM criterion is an impurity-based splitting criterion designed for binary class attributes (Dietterich *et al.*, 1996) and (Kearns and Mansour, 1999). The impurity-based function is defined as:

$$DKM(y, S) = 2 \cdot \sqrt{\left(\frac{|\sigma_{y=c_1} S|}{|S|}\right) \cdot \left(\frac{|\sigma_{y=c_2} S|}{|S|}\right)}$$

It has been theoretically proved (Kearns and Mansour, 1999) that this criterion requires smaller trees for obtaining a certain error than other impurity based criteria (information gain and Gini index).

3.7 Normalized Impurity Based Criteria

The impurity-based criterion described above is biased towards attributes with larger domain values. Namely, it prefers input attributes with many values over attributes with less values (Quinlan, 1986). For instance, an input attribute that represents the national security number will probably get the highest information gain. However, adding this attribute to a decision tree will result in a poor generalized accuracy. For that reason, it is useful to "normalize" the impurity based measures, as described in the following sections.

3.8 Gain Ratio

The gain ratio "normalizes" the information gain as follows (Quinlan, 1993):

$$GainRatio(a_i, S) = \frac{InformationGain(a_i, S)}{Entropy(a_i, S)}$$

Note that this ratio is not defined when the denominator is zero. Also the ratio may tend to favor attributes for which the denominator is very small. Consequently, it is suggested in two stages. First the information gain is calculated for all attributes. As a consequence, taking into consideration only attributes that have performed at least as good as the average information gain, the attribute that has obtained the best ratio gain is selected. It has been shown that the gain ratio tends to outperform simple information gain criteria, both from the accuracy aspect, as well as from classifier complexity aspects (Quinlan, 1988).

3.9 Distance Measure

The distance measure, like the gain ratio, normalizes the impurity measure. However, it suggests normalizing it in a different way (Lopez de Mantras, 1991):

$$\frac{\Delta\Phi(a_i,S)}{-\sum\limits_{v_{i,j}\in dom(a_i)}\sum\limits_{c_k\in dom(y)}\frac{|\sigma_{a_i=v_{i,j}\ AND\ y=c_k}S|}{|S|}\cdot\log_2\frac{|\sigma_{a_i=v_{i,j}\ AND\ y=c_k}S|}{|S|}}$$

3.10 Binary Criteria

The binary criteria are used for creating binary decision trees. These measures are based on division of the input attribute domain into two sub-domains.

Let $\beta(a_i, dom_1(a_i), dom_2(a_i), S)$ denote the binary criterion value for attribute a_i over sample S when $dom_1(a_i)$ and $dom_2(a_i)$ are its corresponding subdomains. The value obtained for the optimal division of the attribute domain into two mutually exclusive and exhaustive sub-domains is used for comparing attributes.

3.11 Twoing Criterion

The gini index may encounter problems when the domain of the target attribute is relatively wide (Breiman *et al.*, 1984). In this case it is possible to employ binary criterion called twoing criterion. This criterion is defined as:

$$\begin{gathered}twoing(a_i, dom_1(a_i), dom_2(a_i), S) = \\ 0.25\cdot\frac{\left|\sigma_{a_i\in dom_1(a_i)}S\right|}{|S|}\cdot\frac{\left|\sigma_{a_i\in dom_2(a_i)}S\right|}{|S|}\cdot \\ \left(\sum_{c_i\in dom(y)}\left|\frac{\left|\sigma_{a_i\in dom_1(a_i)\ AND\ y=c_i}S\right|}{\left|\sigma_{a_i\in dom_1(a_i)}S\right|}-\frac{\left|\sigma_{a_i\in dom_2(a_i)\ AND\ y=c_i}S\right|}{\left|\sigma_{a_i\in dom_2(a_i)}S\right|}\right|\right)^2\end{gathered}$$

When the target attribute is binary, the gini and twoing criteria are equivalent. For multi–class problems, the twoing criteria prefer attributes with evenly divided splits.

3.12 Orthogonal (ORT) Criterion

The ORT criterion was presented by Fayyad and Irani (1992). This binary criterion is defined as:

$$ORT(a_i, dom_1(a_i), dom_2(a_i), S) = 1 - cos\theta(P_{y,1}, P_{y,2})$$

where $\theta(P_{y,1}, P_{y,2})$ is the angle between two vectors $P_{y,1}$ and $P_{y,2}$. These vectors represent the probability distribution of the target attribute in the partitions $\sigma_{a_i \in dom_1(a_i)} S$ and $\sigma_{a_i \in dom_2(a_i)} S$ respectively.

It has been shown that this criterion performs better than the information gain and the Gini index for specific problem constellations.

3.13 Kolmogorov–Smirnov Criterion

A binary criterion that uses Kolmogorov–Smirnov distance has been proposed in Friedman (1977) and Rounds (1980). Assuming a binary target attribute, namely $dom(y) = \{c_1, c_2\}$, the criterion is defined as:

$$KS(a_i, dom_1(a_i), dom_2(a_i), S) =$$

$$\left| \frac{\left|\sigma_{a_i \in dom_1(a_i)\ AND\ y=c_1} S\right|}{\left|\sigma_{y=c_1} S\right|} - \frac{\left|\sigma_{a_i \in dom_1(a_i)\ AND\ y=c_2} S\right|}{\left|\sigma_{y=c_2} S\right|} \right|$$

This measure was extended in (Utgoff and Clouse, 1996) to handle target attributes with multiple classes and missing data values. Their results indicate that the suggested method outperforms the gain ratio criteria.

3.14 AUC–Splitting Criteria

The idea of using the AUC metric as a splitting criterion was recently proposed in (Ferri *et al.*, 2002). The attribute that obtains the maximal area under the convex hull of the ROC curve is selected. It has been shown that the AUC–based splitting criterion outperforms other splitting criteria both with respect to classification accuracy and area under the ROC curve. It is important to note that unlike impurity criteria, this criterion does not perform a comparison between the impurity of the parent node with the weighted impurity of the children after splitting.

3.15 Other Univariate Splitting Criteria

Additional univariate splitting criteria can be found in the literature, such as permutation statistics (Li and Dubes, 1986), mean posterior improvements (Taylor and Silverman, 1993) and hypergeometric distribution measures (Martin, 1997).

Table 9.1. Additional Decision Tree Inducers.

Algorithm	Description	Reference
CAL5	Designed specifically for numerical–valued attributes	Muller and Wysotzki (1994)
FACT	An earlier version of QUEST. Uses statistical tests to select an attribute for splitting each node and then uses discriminant analysis to find the split point.	Loh and Vanichsetakul (1988)
LMDT	Constructs a decision tree based on multivariate tests are linear combinations of the attributes.	Brodley and Utgoff (1995)
T1	A one–level decision tree that classifies instances using only one attribute. Missing values are treated as a "special value". Support both continuous an nominal attributes.	Holte (1993)
PUBLIC	Integrates the growing and pruning by using MDL cost in order to reduce the computational complexity.	Rastogi and Shim (2000)
MARS	A multiple regression function is approximated using linear splines and their tensor products.	Friedman (1991)

9. Advantages and Disadvantages of Decision Trees

Several advantages of the decision tree as a classification tool have been pointed out in the literature:

1. Decision trees are self–explanatory and when compacted they are also easy to follow. In other words if the decision tree has a reasonable number of leaves, it can be grasped by non–professional users. Furthermore decision trees can be converted to a set of rules. Thus, this representation is considered as comprehensible.

2. Decision trees can handle both nominal and numeric input attributes.

3. Decision tree representation is rich enough to represent any discrete–value classifier.

4. Decision trees are capable of handling datasets that may have errors.

5. Decision trees are capable of handling datasets that may have missing values.

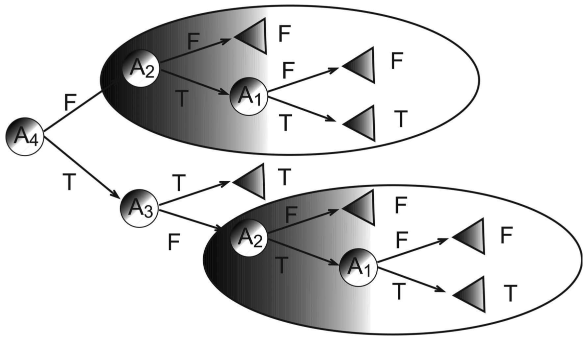

Figure 9.3. Illustration of Decision Tree with Replication.

6. Decision trees are considered to be a nonparametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure.

On the other hand, decision trees have such disadvantages as:

1. Most of the algorithms (like ID3 and C4.5) require that the target attribute will have only discrete values.

2. As decision trees use the "divide and conquer" method, they tend to perform well if a few highly relevant attributes exist, but less so if many complex interactions are present. One of the reasons for this is that other classifiers can compactly describe a classifier that would be very challenging to represent using a decision tree. A simple illustration of this phenomenon is the replication problem of decision trees (Pagallo and Huassler, 1990). Since most decision trees divide the instance space into mutually exclusive regions to represent a concept, in some cases the tree should contain several duplications of the same sub-tree in order to represent the classifier. For instance if the concept follows the following binary function: $y = (A_1 \cap A_2) \cup (A_3 \cap A_4)$ then the minimal univariate decision tree that represents this function is illustrated in Figure 9.3. Note that the tree contains two copies of the same subt-ree.

3. The greedy characteristic of decision trees leads to another disadvantage that should be pointed out. This is its over–sensitivity to the training set, to irrelevant attributes and to noise (Quinlan, 1993).