Decision tree

Basic idea

Decision tree learning is one of the most widely used and practical methods for inductive inference. It is a method for approximating discrete-valued functions that is robust to noisy data and capable of learning disjunctive expressions. This chapter

Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Each node in the

In general, decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances. Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions. For example, the decision tree shown in Figure 3.1 corresponds to the expression

$$(Outlook = Sunny \wedge Humidity = Normal)$$
$$\vee \quad (Outlook = Overcast)$$
$$\vee \quad (Outlook = Rain \wedge Wind = Weak)$$

Demo

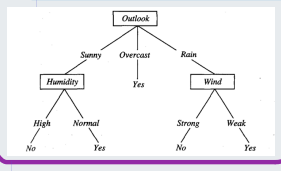


FIGURE 3.1
A decision tree for the concept *PlayTennis*. An example is classified by sorting it through the tree to the appropriate leaf node, then returning the classification associated with this leaf (in this case, *Yes* or *No*). This tree classifies Saturday mornings according to whether or not they are suitable for playing tennis.

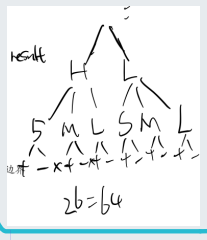
Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute.

Bias

Hard bias(search space)

- *Instances are represented by attribute-value pairs.* Instances are described by a fixed set of attributes (e.g., *Temperature*) and their values (e.g., *Hot*). The easiest situation for decision tree learning is when each attribute takes on a small number of disjoint possible values (e.g., *Hot*, *Mild*, *Cold*). However,
- *Disjunctive descriptions may be required.* As noted above, decision trees naturally represent disjunctive expressions.
- *The training data may contain errors.* Decision tree learning methods are robust to errors, both errors in classifications of the training examples and errors in the attribute values that describe these examples.
- *The training data may contain missing attribute values.* Decision tree methods can be used even when some training examples have unknown values

Hypotheses space



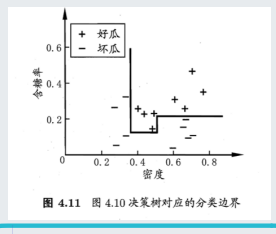


图 4.11 图 4.10 决策树对应的分类边界

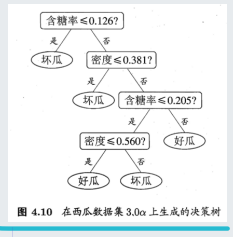


图 4.10 在西瓜数据集 3.0α 上生成的决策树

Soft bias(search strategy/ preference)

Occam's razor

Occam's razor: Prefer the simplest hypothesis that fits the data.

soft bias will result in smaller trees, but is too short-sighted since a seemingly worthless split early on in the tree might be followed by a very good split

a better strategy is to grow a very large tree T0, and then prune it back in order to obtain a subtree

Tree pruning

1.traditional

estimate its test error using cross-validation or the validation set approach

select a subtree that subtree leads to the lowest test error rate

2.cost complexity pruning/weakest link pruning

For each value of α there corresponds a subtree $T \subset T_0$ such that

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \qquad (8.4)$$

$|T|$ indicates the number of terminal nodes of the tree T

The tuning parameter α controls a trade-off between the subtree's complexity and its fit to the training data

Depends on algorithm assumptions

Splitting criteria/loss function

Information gain

entropy

In order to define information gain precisely, we begin by defining a measure commonly used in information theory, called *entropy*, that characterizes the (im)purity of an arbitrary collection of examples. Given a collection S, containing positive and negative examples of some target concept, the entropy of S relative to this boolean classification is

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus} \qquad (3.1)$$

the training data. The measure we will use, called *information gain*, is simply the expected reduction in entropy caused by partitioning the examples according to this attribute. More precisely, the information gain, $Gain(S, A)$ of an attribute A,

relative to a collection of examples S, is defined as

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \qquad (3.4)$$

where $Values(A)$ is the set of all possible values for attribute A, and S_v is the subset of S for which attribute A has value v (i.e., $S_v = \{s \in S | A(s) = v\}$). Note

Gain ratio

The *GainRatio* measure is defined in terms of the earlier *Gain* measure, as well as this *SplitInformation*, as follows

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)} \qquad (3.6)$$

Split information

successfully is the *gain ratio* (Quinlan 1986). The gain ratio measure penalizes attributes such as *Date* by incorporating a term, called *split information*, that is sensitive to how broadly and uniformly the attribute splits the data:

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \qquad (3.5)$$

Notice that the *SplitInformation* term discourages the selection of attributes with many uniformly distributed values. For example, consider a collection of n ex-

Variance reduction

$R_1, \ldots, R_J$? In theory, the regions could have any shape. However, we choose to divide the predictor space into high-dimensional rectangles, or boxes, for simplicity and for ease of interpretation of the resulting predictive model. The goal is to find boxes $R_1, \ldots, R_J$ that minimize the RSS, given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2, \qquad (8.1)$$

is often employed in cases where the target variable is continuous (regression tree)

Gini impurity

$$\mathrm{Gini}(D) = \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'}$$
$$= 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2 .$$

直观来说, $\mathrm{Gini}(D)$ 反映了从数据集 D 中随机抽取两个样本, 其类别标记不一致的概率. 因此, $\mathrm{Gini}(D)$ 越小, 则数据集 D 的纯度越高.

Gini index

$$\mathrm{Gini\_index}(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Gini}(D^v) .$$

Algorithm

ID3

Procedure

Our basic algorithm, ID3, learns decision trees by constructing them top-down, beginning with the question "which attribute should be tested at the root of the tree?" To answer this question, each instance attribute is evaluated using a statistical test to determine how well it alone classifies the training examples. The best attribute is selected and used as the test at the root node of the tree. A descendant of the root node is then created for each possible value of this attribute, and the training examples are sorted to the appropriate descendant node (i.e., down the branch corresponding to the example's value for this attribute). The entire process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree. This forms a greedy search for an acceptable decision tree, in which the algorithm never backtracks to reconsider earlier choices. A simplified version of the algorithm, specialized to learning boolean-valued functions (i.e., concept learning), is described in Table 3.1.

Soft bias

Greedy

Accept best search at each single step

A closer approximation to the inductive bias of ID3: Shorter trees are preferred over longer trees. Trees that place high information gain attributes close to the root are preferred over those that do not.

C4.5

Soft bias

Uses gain ratio

Due with continuous value

Bi-partition

取值来对结点进行划分. 此时, 连续属性离散化技术可派上用场. 最简单的策略是采用二分法(bi-partition)对连续属性进行处理, 这正是 C4.5 决策树算法中采用的机制 [Quinlan, 1993].

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leqslant i \leqslant n-1 \right\}, \qquad (4.7)$$

即把区间 $[a^i, a^{i+1})$ 的中位点 $\frac{a^i + a^{i+1}}{2}$ 作为候选划分点. 然后, 我们就可像离散属性值一样来考察这些划分点, 选取最优的划分点进行样本集合的划分. 例如, 可对式(4.2)稍加改造:

$$\mathrm{Gain}(D, a) = \max_{t \in T_a} \mathrm{Gain}(D, a, t)$$
$$= \max_{t \in T_a} \mathrm{Ent}(D) - \sum_{\lambda \in \{-,+\}} \frac{|D_t^\lambda|}{|D|} \mathrm{Ent}(D_t^\lambda), \qquad (4.8)$$

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

Gain(D, 色泽) = 0.109; Gain(D, 根蒂) = 0.143;
Gain(D, 敲声) = 0.141; Gain(D, 纹理) = 0.381;
Gain(D, 脐部) = 0.289; Gain(D, 触感) = 0.006;
Gain(D, 密度) = 0.262; Gain(D, 含糖率) = 0.349.

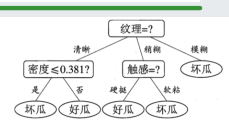


图 4.8 在西瓜数据集 3.0 上基于信息增益生成的决策树

Example

Pros & cons

Pros

Easy to interpret

The human interpretability of a tree model such as CART is often seen as its major strength. However, in practice it is found that the particular tree structure that is learned is very sensitive to the details of the data set, so that a small change to the training data can result in a very different set of splits (Hastie *et al.*, 2001).

Cons

There are other problems with tree-based methods of the kind considered in this section. One is that the splits are aligned with the axes of the feature space, which may be very suboptimal. For instance, to separate two classes whose optimal decision boundary runs at 45 degrees to the axes would need a large number of axis-parallel splits of the input space as compared to a single non-axis-aligned split. Furthermore, the splits in a decision tree are hard, so that each region of input space is associated with one, and only one, leaf node model. The last issue is particularly problematic in regression where we are typically aiming to model smooth functions, and yet the tree model produces piecewise-constant predictions with discontinuities at the split boundaries.