Lecture 5: Decision Trees - Part II

- ♢ Attributes with continuous values
- ♢ Attributes with multiple values
- ♢ Missing values
- ♢ Taking into account costs of attributes
- ♢ Using decision trees for regression

Attributes with continuous values

Example:

Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

A decision tree needs to perform tests on these attributes as well

What kind of test do we want?

Value of the attribute less than a cut point!

What cut points should we consider?

We need to consider only cut points where the class label changes!

Attributes with multiple values

If an attribute splits the data perfectly, it will always be preferred by information gain.

E.g. a unique ID for each data point!

But that has very poor generalization performance.

You would think pruning can help, but what can you do with a tree that just has one node?

Two solutions:

1. Use another criterion that is more fair
2. Ensure that all attributes have the same number of values

A better criterion: Gain ratio

For a set of instances S and an attribute A with v possible values

$$GainRatio(S, A) = \frac{Gain(S, A)}{SplitInformation(S, A)}$$

where

$$SplitInformation(S, A) = -\sum_{i=1}^{v} \frac{|S_v|}{|S|} \log_2 \frac{|S_v|}{|S|}$$

So for an attribute that splits the data into many partitions mostly uniformly, $SplitInformation$ will be high

Problem: It can actually become too high!

Solution: First use $Gain$, then use $GainRatio$ for attributes with $Gain$ above average

Many other metrics available!

Ensuring the same number of values

If an attribute A has $v > 2$ possible values, $Val_1..Val_v$, replace it by v Boolean attributes, $A_k, k = 1..v$, where:

$$A_k = \begin{cases} 1 \text{ if } A = Val_k \\ 0 \text{ otherwise} \end{cases}, \ \forall k = 1..v$$

This is called *1-of-v encoding*

Used more generally to encode learning data (e.g. in neural networks)

Missing Values during Classification

- "Most likely" value based on all the data that reaches the current node. "Most likely" means the most frequent attribute value
- Assign all possible values with some probability (usually proportional to the number of instances going down the same path. We will predict all the possible class labels with the appropriate probabilities too.

Missing Values during Tree Construction

1. Introduce an "unknown" value
2. Modify gain ratio to take into account probability of an attribute being known:

   $$Gain(S, A)P(A)$$

   where P(A) is the fraction of the instances that reached the node, in which the value was known

Costs of attributes

Include cost in the metric, e.g.

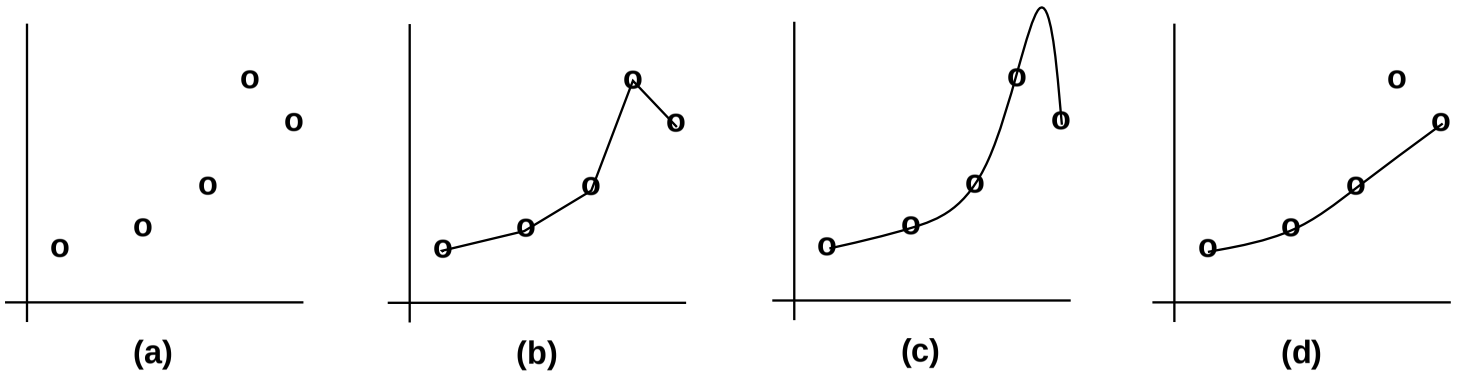
$$\frac{Gain^2(S, A)}{Cost(A)}$$

Mostly a problem in the medical domain

Two problems in inductive (supervised) learning

Assume somebody gives us examples of what we are trying to learn

Try to find a *target function* that fits the examples



If the output domain is discrete, this problem is called *classification*

If the output is continuous, the problem is known as *regression* of *function approximation*.

Decision trees for regression

Decision trees were designed for classification

How to use them for regression?

Piece-wise constant approximation! (CART, Breiman et. al, 1984)

Basic CART algorithm

Given a set of labeled training instances $x_1^i..x_m^i, y^i, i = 1..n$, where each label y^i is a real number:

1. Compute the average of all the labels: $\bar{y} = \frac{1}{n} \Sigma_{i=1}^{n} y^i$
2. Compute the mean squared error of the instances:

   $$\frac{1}{n} \sum_{i=1}^{n} (y^i - \bar{y})^2$$

3. If the error is below a desired threshold, create a leaf with the label $\bar{y}$
4. Otherwise pick the best attribute to split the data on
5. Add a node that tests the attribute
6. Split the training set according to the value of the attribute
7. Recurse on each subset of the training data

Choosing the "best" attribute

The same principle as in classification: we want the attribute that minimizes the error in each partition

The error in this case is the sum of the mean square errors from each partition.

If $v \in V$ are all the possible values of an attribute, and the corresponding partitions have N_v examples, then we want to minimize:

$$\sum_{v \in V} \frac{1}{N_v} \sum_{i=1}^{N_v} (y^i - \bar{y}^v)^2$$

Pruning and all the other methods can can be applied in this case too.