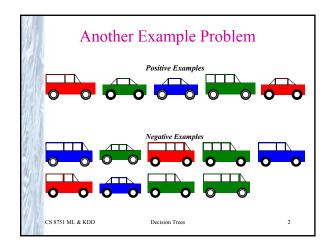
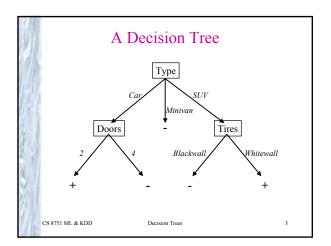
Decision Trees

- · Decision tree representation
- · ID3 learning algorithm
- Entropy, Information gain
- Overfitting

CS 8751 ML & KDD

Decision Trees





Decision Trees Decision tree representation • Each internal node tests an attribute • Each branch corresponds to an attribute value • Each leaf node assigns a classification How would you represent: • ∧,∨, XOR • (A∧B)∨(C∧¬D∧E) • M of N CS 8751 ML & KDD Decision Trees 4

When to Consider Decision Trees

- · Instances describable by attribute-value pairs
- · Target function is discrete valued
- · Disjunctive hypothesis may be required
- · Possibly noisy training data

Examples

- Equipment or medical diagnosis
- · Credit risk analysis
- Modeling calendar scheduling preferences

CS 8751 ML & KDD

Decision Trees

Top-Down Induction of Decision Trees

Main loop:

- 1. A = the "best" decision attribute for next *node*
- 2. Assign A as decision attribute for node
- 3. For each value of A, create descendant of node
- 4. Divide training examples among child nodes
- If training examples perfectly classified, STOP
 Else iterate over new leaf nodes

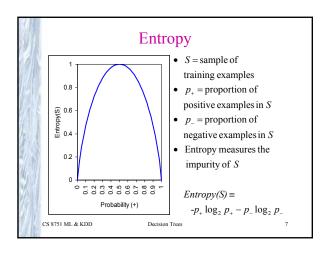
Which attribute is best?





CS 8751 ML & KDD

Decision Trees



Entropy

Entropy(S) = expected number of bits need to encode
 class (+ or -) of randomly drawn member of S
 (using an optimal, shortest-length code)

Why?

Information theory: optimal length code assigns $-log_2p$ bits to message having probability p

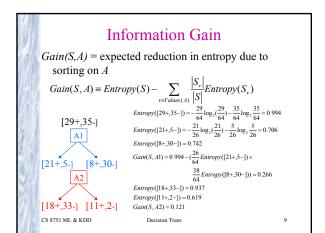
So, expected number of bits to encode + or - of random member of *S*:

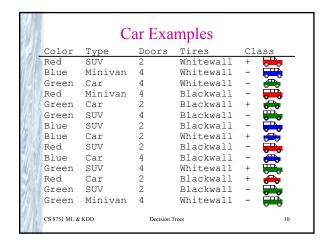
$$-p_{+}\log_{2}p_{+}-p_{-}\log_{2}p_{-}$$

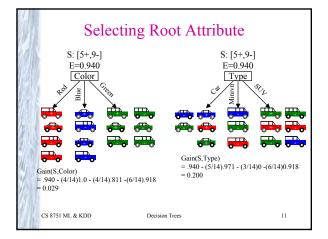
$$Entropy(S) \equiv -p_{+} \log_2 p_{+} - p_{-} \log_2 p_{-}$$

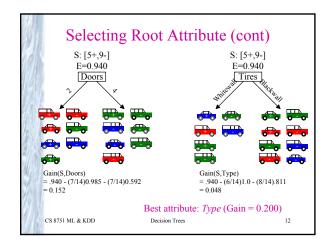
CS 8751 ML & KDD

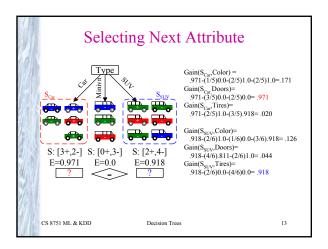
Decision Trees

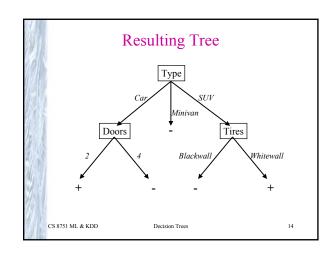


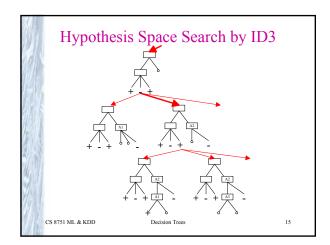












Hypothesis Space Search by ID3

- Hypothesis space is complete!
 - Target function is in there (but will we find it?)
- Outputs a single hypothesis (which one?)
 - Cannot play 20 questions
- · No back tracing
 - Local minima possible
- · Statistically-based search choices
 - Robust to noisy data
- Inductive bias: approximately "prefer shortest tree"

CS 8751 ML & KDD Decision Trees 16

Inductive Bias in ID3

Note H is the power set of instances X

Unbiased?

Not really...

- Preference for short trees, and for those with high information gain attributes near the root
- Bias is a preference for some hypotheses, rather than a restriction of hypothesis space H
- Occam's razor: prefer the shortest hypothesis that fits the data

CS 8751 ML & KDD Decision Trees 17

Occam's Razor

Why prefer short hypotheses?

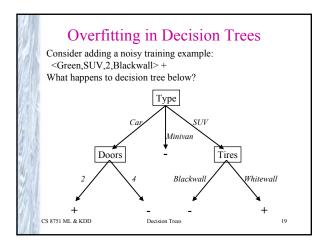
Argument in favor:

- Fewer short hypotheses than long hypotheses
- short hyp. that fits data unlikely to be coincidence
- long hyp. that fits data more likely to be coincidence

Argument opposed:

- There are many ways to define small sets of hypotheses
- e.g., all trees with a prime number of nodes that use attributes beginning with "Z"
- What is so special about small sets based on size of hypothesis?

CS 8751 ML & KDD Decision Trees 18

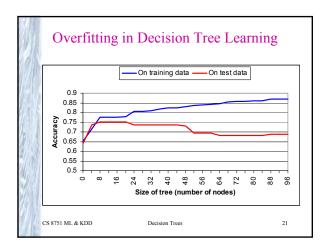


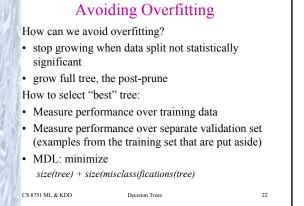
Overfitting Consider error of hypothesis h over training data: error_{train}(h) entire distribution D of data : error_D(h) Hypothesis $h \in H$ overfits the training data if there is an alternative hypothesis $h' \in H$ such that $error_{train}(h) < error_{train}(h')$ $error_D(h) > error_D(h')$

20

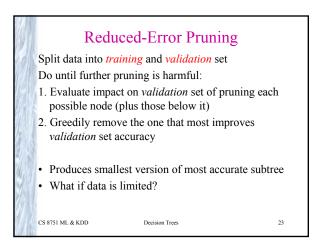
and

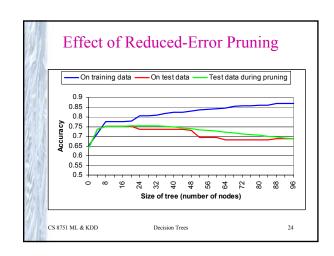
CS 8751 ML & KDD





Decision Trees





Decision Tree Post-Pruning

- A standard method in C4.5, C5.0
- Construct a complete tree
 - For each node estimate what the error might be with and without the node (needs a conservative estimate of error since based on training data)
 - Prune any node where the expected error stays the same
 - Greatly influenced by method for estimating likely errors

CS 8751 ML & KDD

Decision Trees

Rule Post-Pruning

- 1. Convert tree to equivalent set of rules
- 2. Prune each rule independently of others
- 3. Sort final rules into desired sequence for use

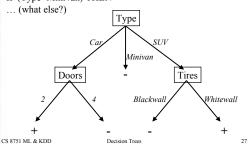
CS 8751 ML & KDD

Decision Trees

26

Converting a Tree to Rules

IF (Type=Car) AND (Doors=2) THEN + IF (Type=SUV) AND (Tires=Whitewall) THEN + IF (Type=Minivan) THEN -



Continuous Valued Attributes

Create one (or more) corresponding discrete attributes based on continuous

- (EngineSize = 325) = true or false
- (EngineSize \leq 330) = t or f (330 is "split" point)

How to pick best "split" point?

- 1. Sort continuous data
- 2. Look at points where class differs between two values
- 3. Pick the split point with the best gain

Why this one?

EngineSize: 285 (290 295) (310 (330) 330 345) 360 Class:

CS 8751 ML & KDD

Attributes with Many Values

Problem:

- If attribute has many values, Gain will select it
- Imagine if cars had *PurchaseDate* feature likely all would be different

One approach: use GainRatio instead

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

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

where S_i is subset of S for which A has value v_i

CS 8751 ML & KDD Decision Trees

Attributes with Costs

Consider

- medical diagnosis, BloodTest has cost \$150
- robotics, Width_from_1ft has cost 23 second

How to learn consistent tree with low expected cost?

Approaches: replace gain by

Tan and Schlimmer (1990) $Gain^2(S, A)$

Nunez (1988) $2^{Gain(S,A)}-1$ $(Cost(A)+1)^w$

where $w \in [0,1]$ and determines importance of cost

CS 8751 ML & KDD Decision Trees

30

Unknown Attribute Values

What if some examples missing values of A?

"?" in C4.5 data sets

Use training example anyway, sort through tree

- If node n tests A, assign most common value of A among other examples sorted to node n
- assign most common value of ${\cal A}$ among other examples with same target value
- assign probability p_i to each possible value v_i of A
 - assign fraction p_i of example to each descendant in tree

Classify new examples in same fashion

CS 8751 ML & KDD

Decision Trees

31