Lecture 4 of 42

Decision Trees

Wednesday, 30 January 2008

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.cis.ksu.edu/~bhsu

Readings:
Sections 3.1-3.5, Mitchell
Chapter 18, Russell and Norvig
MLC++, Kohavi et al

Lecture Outline

• Read 3.1-3.5, Mitchell; Chapter 18, Russell and Norvig; Kohavi et al paper
• Handout: “Data Mining with MLC++”, Kohavi et al
• Suggested Exercises: 18.3, Russell and Norvig; 3.1, Mitchell
• Decision Trees (DTs)
  – Examples of decision trees
  – Models: when to use
• Entropy and Information Gain
• ID3 Algorithm
  – Top-down induction of decision trees
    • Calculating reduction in entropy (information gain)
    • Using information gain in construction of tree
  – Relation of ID3 to hypothesis space search
    – Inductive bias in ID3
• Using MLC++ (Machine Learning Library in C++)
• Next: More Biases (Occam’s Razor); Managing DT Induction
### Decision Trees

- **Classifiers**
  - Instances (unlabeled examples): represented as attribute ("feature") vectors
- **Internal Nodes: Tests for Attribute Values**
  - Typical: equality test (e.g., “Wind = ?”)
  - Inequality, other tests possible
- **Branches: Attribute Values**
  - One-to-one correspondence (e.g., “Wind = Strong”, “Wind = Light”)
- **Leaves: Assigned Classifications (Class Labels)**

![Decision Tree for Concept PlayTennis](image)

### Boolean Decision Trees

- **Boolean Functions**
  - Representational power: universal set (i.e., can express any boolean function)
  - Q: Why?
    - A: Can be rewritten as rules in Disjunctive Normal Form (DNF)
    - Example below: \((Sunny \land Normal-Humidity) \lor Overcast \lor (Rain \land Light-Wind)\)
- **Other Boolean Concepts (over Boolean Instance Spaces)**
  - \(\land, \lor, \oplus\) (XOR)
  - \((A \land B) \lor (C \land \neg D \land E)\)
  - \(m\text{-of-}n\)

![Boolean Decision Tree for Concept PlayTennis](image)
A Tree to Predict C-Section Risk

- Learned from Medical Records of 1000 Women
- Negative Examples are Cesarean Sections
  - Prior distribution: [833+, 167-] 0.83+, 0.17-
  - Fetal-Presentation = 0: [822+, 167-] 0.90+, 0.10-
    - Primiparous = 0: [399+, 13-] 0.97+, 0.03-
    - Primiparous = 1: [368+, 68-] 0.84+, 0.16-
      - Fetal-Distress = 0: [334+, 17-] 0.88+, 0.12-
        - Birth-Weight ≥ 3349
        - Birth-Weight < 3349
      - Fetal-Distress = 1: [34+, 21-] 0.62+, 0.38-
  - Previous-C-Section = 0: [767+, 81-] 0.90+, 0.10-
  - Previous-C-Section = 1: [55+, 35-] 0.61+, 0.39-
    - Fetal-Presentation = 2: [3+, 29-] 0.11+, 0.89-
    - Fetal-Presentation = 3: [8+, 22-] 0.27+, 0.73-

When to Consider Using 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
  - Risk analysis
    - Credit, loans
    - Insurance
    - Consumer fraud
    - Employee fraud
  - Modeling calendar scheduling preferences (predicting quality of candidate time)
Decision Trees and Decision Boundaries

- Instances Usually Represented Using Discrete Valued Attributes
  - Typical types
    - Nominal (red, yellow, green)
    - Quantized (low, medium, high)
  - Handling numerical values
    - Discretization, a form of vector quantization (e.g., histogramming)
    - Using thresholds for splitting nodes

Example: Dividing Instance Space into Axis-Parallel Rectangles

```
+ + +  
+ + -  
- + -  
 1 3 5 7
```

Decision Tree Learning: Top-Down Induction (ID3)

- Algorithm Build-DT (Examples, Attributes)
  - IF all examples have the same label THEN RETURN (leaf node with label)
  - ELSE
    - IF set of attributes is empty THEN RETURN (leaf with majority label)
    - ELSE
      - Choose best attribute $A$ as root
      - FOR each value $v$ of $A$
        - Create a branch out of the root for the condition $A = v$
        - IF $\{x \in $ Examples: $x.A = v\} = \emptyset$ THEN RETURN (leaf with majority label)
        - ELSE Build-DT ($\{x \in $ Examples: $x.A = v\}$, Attributes $\sim \{A\}$)
  - But Which Attribute Is Best?
Broadening the Applicability of Decision Trees

• Assumptions in Previous Algorithm
  – Discrete output
    • Real-valued outputs are possible
    • Regression trees [Breiman et al, 1984]
  – Discrete input
    • Quantization methods
    • Inequalities at nodes instead of equality tests (see rectangle example)

• Scaling Up
  – Critical in knowledge discovery and database mining (KDD) from very large databases (VLDB)
  – Good news: efficient algorithms exist for processing many examples
  – Bad news: much harder when there are too many attributes

• Other Desired Tolerances
  – Noisy data (classification noise = incorrect labels; attribute noise = inaccurate or imprecise data)
  – Missing attribute values

Choosing the “Best” Root Attribute

• Objective
  – Construct a decision tree that is as small as possible (Occam’s Razor)
  – Subject to: consistency with labels on training data

• Obstacles
  – Finding the minimal consistent hypothesis (i.e., decision tree) is NP-hard (D’oh!)
  – Recursive algorithm (Build-DT)
    • A greedy heuristic search for a simple tree
    • Cannot guarantee optimality (D’oh!)

• Main Decision: Next Attribute to Condition On
  – Want: attributes that split examples into sets that are relatively pure in one label
  – Result: closer to a leaf node
  – Most popular heuristic
    • Developed by J. R. Quinlan
    • Based on information gain
    • Used in ID3 algorithm
Entropy: Intuitive Notion

- A Measure of Uncertainty
  - The Quantity
    - Purity: how close a set of instances is to having just one label
    - Impurity (disorder): how close it is to total uncertainty over labels
  - The Measure: Entropy
    - Directly proportional to impurity, uncertainty, irregularity, surprise
    - Inversely proportional to purity, certainty, regularity, redundancy

- Example
  - For simplicity, assume \( H = \{0, 1\} \), distributed according to \( Pr(y) \)
    - Can have (more than 2) discrete class labels
    - Continuous random variables: differential entropy
  - Optimal purity for \( y \): either
    - \( Pr(y = 0) = 1, Pr(y = 1) = 0 \)
    - \( Pr(y = 1) = 1, Pr(y = 0) = 0 \)
  - What is the least pure probability distribution?
    - \( Pr(y = 0) = 0.5, Pr(y = 1) = 0.5 \)
    - Corresponds to maximum impurity/uncertainty/irregularity/surprise
    - Property of entropy: concave function ("concave downward")

Entropy: Information Theoretic Definition

- Components
  - \( D \): a set of examples \( \langle x_1, c(x_1) \rangle, \langle x_2, c(x_2) \rangle, \ldots, \langle x_m, c(x_m) \rangle \)
  - \( p_+ = Pr(c(x) = +), p_- = Pr(c(x) = -) \)

- Definition
  - \( H \) is defined over a probability density function \( p \)
  - \( D \) contains examples whose frequency of + and - labels indicates \( p_+ \) and \( p_- \) for the observed data
  - The entropy of \( D \) relative to \( c \) is:
    \[
    H(D) = -p_+ \log_b(p_+) - p_- \log_b(p_-)
    \]

- What Units is \( H \) Measured In?
  - Depends on the base \( b \) of the log (bits for \( b = 2 \), nats for \( b = e \), etc.)
  - A single bit is required to encode each example in the worst case (\( p_+ = 0.5 \))
  - If there is less uncertainty (e.g., \( p_+ = 0.8 \)), we can use less than 1 bit each
Information Gain: Information Theoretic Definition

- **Partitioning on Attribute Values**
  - Recall: a partition of $D$ is a collection of disjoint subsets whose union is $D$
  - Goal: measure the uncertainty removed by splitting on the value of attribute $A$

- **Definition**
  - The information gain of $D$ relative to attribute $A$ is the expected reduction in entropy due to splitting ("sorting") on $A$:
    \[
    \text{Gain}(D, A) = -H(D) - \sum_{v \in \text{values}(A)} \frac{|D_v|}{|D|} H(D_v)
    \]
  - Idea: partition on $A$; scale entropy to the size of each subset $D_v$

- **Which Attribute Is Best?**

### An Illustrative Example

**Training Examples for Concept PlayTennis**

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>High</td>
<td>Light</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

- $ID3 = \text{Build-DT using Gain}()$
- How Will $ID3$ Construct A Decision Tree?
Constructing A Decision Tree for PlayTennis using ID3 [1]

- Selecting The Root Attribute

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>Overcast</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

  - Prior (unconditioned) distribution: 9+, 5-
    - \( H(D) = \frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right) \) bits = 0.94 bits
    - \( H(D, \text{Humidity} = \text{High}) = \frac{3}{7} \log_2 \left( \frac{3}{7} \right) - \frac{4}{7} \log_2 \left( \frac{4}{7} \right) \) = 0.985 bits
    - \( H(D, \text{Humidity} = \text{Normal}) = \frac{6}{7} \log_2 \left( \frac{6}{7} \right) - \frac{1}{7} \log_2 \left( \frac{1}{7} \right) \) = 0.592 bits
    - \( \text{Gain}(D, \text{Humidity}) = 0.94 - \frac{7}{14} \times 0.985 + \frac{7}{14} \times 0.592 = 0.151 \) bits
    - \( \text{Gain}(D, \text{Wind}) = 0.94 - \frac{8}{14} \times 0.811 + \frac{6}{14} \times 1.0 = 0.048 \) bits

  - Selecting The Next Attribute (Root of Subtree)
    - Continue until every example is included in path or purity = 100%
    - What does purity = 100% mean?
    - Can \( \text{Gain}(D, A) < 0 \)?

\[
\text{Gain}(D, A) = H(D) - \sum_{v \in \text{values}(A)} \frac{|D_v|}{|D|} \cdot H(D_v)
\]
Constructing A Decision Tree for PlayTennis using ID3 [3]

- Selecting The Next Attribute (Root of Subtree)
  - Convention: \( \lg (0/a) = 0 \)
  - \( \text{Gain}(D_{\text{Sunny}}, \text{Humidity}) = 0.97 - (3/5) \times 0 - (2/5) \times 0 = 0.97 \) bits
  - \( \text{Gain}(D_{\text{Sunny}}, \text{Wind}) = 0.97 - (2/5) \times 1 - (3/5) \times 0.92 = 0.02 \) bits
  - \( \text{Gain}(D_{\text{Sunny}}, \text{Temperature}) = 0.57 \) bits

- Top-Down Induction
  - For discrete-valued attributes, terminates in \( O(n) \) splits
  - Makes at most one pass through data set at each level (why?)

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Sun</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

Constructing A Decision Tree for PlayTennis using ID3 [4]

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Rain</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

Outlook?:

<table>
<thead>
<tr>
<th>Outlook</th>
<th>1,2,3,4,5,6,7,8,9,10,11,12,13,14 [9+,5]</th>
</tr>
</thead>
</table>

Humidity?:

<table>
<thead>
<tr>
<th>Humidity</th>
<th>3,7,12,13 [6+,0]</th>
</tr>
</thead>
</table>

Wind?:

<table>
<thead>
<tr>
<th>Wind</th>
<th>4,5,6,10,14 [5+,2]</th>
</tr>
</thead>
</table>

CIS 732: Machine Learning and Pattern Recognition
Hypothesis Space Search by ID3

- **Search Problem**
  - Conduct a search of the space of decision trees, which can represent all possible discrete functions
    - Pros: expressiveness; flexibility
    - Cons: computational complexity; large, incomprehensible trees (next time)
  - Objective: to find the best decision tree (minimal consistent tree)
  - Obstacle: finding this tree is \( \text{NP}-\text{hard} \)

- **Tradeoff**
  - Use heuristic (figure of merit that guides search)
  - Use greedy algorithm
  - Aka hill-climbing (gradient “descent”) without backtracking

- **Statistical Learning**
  - Decisions based on statistical descriptors \( p_+, p_- \) for subsamples \( D_j \)
  - In ID3, all data used
  - Robust to noisy data

Inductive Bias in ID3

- **Heuristic : Search :: Inductive Bias : Inductive Generalization**
  - \( H \) is the power set of instances in \( X \)
  - \( \Rightarrow \) Unbiased? Not really...
    - Preference for short trees (termination condition)
    - Preference for trees with high information gain attributes near the root
    - Gain\( (\cdot) \) : heuristic function that captures the inductive bias of ID3
  - **Bias in ID3**
    - Preference for some hypotheses is encoded in heuristic function
    - Compare: a restriction of hypothesis space \( H \) (previous discussion of propositional normal forms: \( k\)-CNF, etc.)

- **Preference for Shortest Tree**
  - Prefer shortest tree that fits the data
  - An Occam’s Razor bias: shortest hypothesis that explains the observations
**MLC++:**

**A Machine Learning Library**

- **MLC++**
  - An object-oriented machine learning library
  - Contains a suite of inductive learning algorithms (including ID3)
  - Supports incorporation, reuse of other DT algorithms (C4.5, etc.)
  - Automation of statistical evaluation, cross-validation

- **Wrappers**
  - Optimization loops that iterate over inductive learning functions (inducers)
  - Used for performance tuning (finding subset of relevant attributes, etc.)

- **Combiners**
  - Optimization loops that iterate over or interleave inductive learning functions
  - Used for performance tuning (finding subset of relevant attributes, etc.)
  - Examples: bagging, boosting (later in this course) of ID3, C4.5

- **Graphical Display of Structures**
  - Visualization of DTs (AT&T dotty, SGI MineSet TreeViz)
  - General logic diagrams (projection visualization)

---

**Using MLC++**

- Refer to MLC++ references
  - Data mining paper (Kohavi, Sommerfeld, and Dougherty, 1996)
  - MLC++ user manual: Utilities 2.0 (Kohavi and Sommerfeld, 1996)
  - MLC++ tutorial (Kohavi, 1995)
  - Other development guides and tools on SGI MLC++ web site

- Online Documentation
  - Consult class web page after Homework 2 is handed out
  - MLC++ (Linux build) to be used for Homework 3
  - Related system: MineSet (commercial data mining edition of MLC++)
    - [http://www.sgi.com/software/mineset](http://www.sgi.com/software/mineset)
    - Many common algorithms
    - Common DT display format
    - Similar data formats

- Experimental Corpora (Data Sets)
  - UC Irvine Machine Learning Database Repository (MLDBR)
  - See [http://www.kdnuggets.com](http://www.kdnuggets.com) and class “Resources on the Web” page
Terminology

• Decision Trees (DTs)
  – Boolean DTs: target concept is binary-valued (i.e., Boolean-valued)
  – Building DTs
    • Histogramming: a method of vector quantization (encoding input using bins)
    • Discretization: converting continuous input into discrete (e.g., by histogramming)

• Entropy and Information Gain
  – Entropy $H(D)$ for a data set $D$ relative to an implicit concept
  – Information gain $Gain(D, A)$ for a data set partitioned by attribute $A$
  – Impurity, uncertainty, irregularity, surprise versus purity, certainty, regularity, redundancy

• Heuristic Search
  – Algorithm Build-DT: greedy search (hill-climbing without backtracking)
  – ID3 as Build-DT using the heuristic Gain()
  – Heuristic : Search :: Inductive Bias : Inductive Generalization

• MLC++ (Machine Learning Library in C++)
  – Data mining libraries (e.g., MLC++) and packages (e.g., MineSet)
  – Irvine Database: the Machine Learning Database Repository at UCI

Summary Points

• Decision Trees (DTs)
  – Can be boolean ($c(x) \in \{+, -\}$) or range over multiple classes
  – When to use DT-based models

• Generic Algorithm Build-DT: Top Down Induction
  – Calculating best attribute upon which to split
  – Recursive partitioning

• Entropy and Information Gain
  – Goal: to measure uncertainty removed by splitting on a candidate attribute $A$
    • Calculating information gain (change in entropy)
    • Using information gain in construction of tree
  – ID3 as Build-DT using Gain()

• ID3 as Hypothesis Space Search (in State Space of Decision Trees)

• Heuristic Search and Inductive Bias

• Data Mining using MLC++ (Machine Learning Library in C++)

• Next: More Biases (Occam’s Razor); Managing DT Induction