Decision Trees

Aykut Erdem
Dept. of Computer Engineering
Hacettepe University

Today
• Decision Trees
• Tree construction
• Overfitting
• Pruning
• Real-valued inputs

Machine Learning in the ER

Can we predict infection?

Many crucial decisions about a patient’s care are made here!
Can we predict infection?

• Previous automatic approaches based on simple criteria:
  - Temperature < 96.8 °F or > 100.4 °F
  - Heart rate > 90 beats/min
  - Respiratory rate > 20 breaths/min

• Too simplified... e.g., heart rate depends on age!

Predicting infection using decision trees

Another example: Mushrooms

• These are the attributes we have for each patient:
  - Temperature
  - Heart rate (HR)
  - Respiratory rate (RR)
  - Age
  - Acuity and pain level
  - Diastolic and systolic blood pressure (DBP, SBP)
  - Oxygen Saturation (SaO2)

• We have these attributes + label (infection) for 200,000 patients!
• Let’s learn to classify infection
Mushroom features

1. cap-shape: bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
2. cap-surface: fibrous=f, grooves=g, scaly=y, smooth=s
3. cap-color:
   - brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
4. bruises?: bruises=t, no=f
5. odor: almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
6. gill-attachment:
   - attached=a, descending=d, free=f, notched=n
7. ...

Two mushrooms

\[
x_1 = x, s, n, t, p, f, c, n, k, e, e, s, w, w, p, w, o, p, k, s, u
\]
\[
y_1 = p
\]
\[
x_2 = x, s, y, t, a, f, c, b, k, e, c, s, w, w, p, w, o, p, n, n, g
\]
\[
y_2 = e
\]

1. cap-shape: bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
2. cap-surface: fibrous=f, grooves=g, scaly=y, smooth=s
3. cap-color:
   - brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
4. ...

Automobile Miles-per-gallon prediction

<table>
<thead>
<tr>
<th>MPG</th>
<th>Cylinders</th>
<th>Displacement</th>
<th>Horsepower</th>
<th>Weight</th>
<th>Acceleration</th>
<th>Model Year</th>
<th>Maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>75 to 78</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75 to 74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4 medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>high</td>
<td>75 to 74</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>medium</td>
<td>75 to 74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75 to 74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>75 to 74</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>medium</td>
<td>75 to 74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>medium</td>
<td>high</td>
<td>low</td>
<td>medium</td>
<td>75 to 74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>medium</td>
<td>79 to 83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>79 to 83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>75 to 78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>79 to 83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>low</td>
<td>75 to 78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>79 to 83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>75 to 78</td>
<td>europe</td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>79 to 83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>5 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75 to 78</td>
<td>europe</td>
</tr>
</tbody>
</table>

Hypotheses: decision trees \( f: X \rightarrow Y \)

- Each internal node tests an attribute \( x_i \)
- Each branch assigns an attribute value \( x_i = v \)
- Each leaf assigns a class \( y \)
- To classify input \( x \): traverse the tree from root to leaf, output the labeled \( y \)

Human interpretable!
Many trees can represent the same concept. But, not all trees will have the same size. For Boolean functions, path to leaf gives truth table row. But, could require exponentially many nodes. The simplest (smallest) decision tree is an NP-complete problem. Learning the simplest decision tree is hard. Resort to a greedy heuristic: Start from empty decision tree. Split on next best attribute (feature). Recurse.
**A Decision Stump**

- Internal node question: “what is the number of cylinders”?
- Leaves: classify by majority vote

**Key idea: Greedily learn trees using recursion**

- Take the Original Dataset.
- And partition it according to the value of the attribute we split on

**Recursive Step**

- Build tree from These records..

**Second level of tree**

- Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)
1. Do not split when all examples have the same label.

2. Can not split when we run out of questions.

**Measuring uncertainty**

- Good split if we are more certain about classification after split
  - Deterministic good (all true or all false)
  - Uniform distribution bad
  - What about distributions in between?

  \[
  P(Y=A) = \frac{1}{2} \quad P(Y=B) = \frac{1}{4} \quad P(Y=C) = \frac{1}{8} \quad P(Y=D) = \frac{1}{8}
  \]

  \[
  P(Y=A) = \frac{1}{4} \quad P(Y=B) = \frac{1}{4} \quad P(Y=C) = \frac{1}{4} \quad P(Y=D) = \frac{1}{4}
  \]

**Entropy**

- Entropy $H(Y)$ of a random variable $Y$

  \[
  H(Y) = - \sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)
  \]

- More uncertainty, more entropy!

- Information Theory interpretation: $H(Y)$ is the expected number of bits needed to encode a randomly drawn value of $Y$ (under most efficient code)
High, Low Entropy

• “High Entropy”
  - Y is from a uniform like distribution
  - Flat histogram
  - Values sampled from it are less predictable

• “Low Entropy”
  - Y is from a varied (peaks and valleys) distribution
  - Histogram has many lows and highs
  - Values sampled from it are more predictable

Entropy Example

\[
H(Y) = - \sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)
\]

\[
P(Y=t) = \frac{5}{6}
\]

\[
P(Y=f) = \frac{1}{6}
\]

\[
H(Y) = - \frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} = 0.65
\]

Conditional Entropy

\[
H(Y \mid X) = - \sum_{j=1}^{n} P(X = x_j) \sum_{i=1}^{k} P(Y = y_i \mid X = x_j) \log_2 P(Y = y_i \mid X = x_j)
\]

Information gain

• Decrease in entropy (uncertainty) after splitting

\[
IG(X) = H(Y) - H(Y \mid X)
\]

In our running example:

\[
IG(X_t) = H(Y) - H(Y \mid X_t)
\]

\[
= 0.65 - 0.33
\]

\[
IG(X_t) > 0 \rightarrow \text{we prefer the split!}
\]
Learning decision trees

- Start from empty decision tree
- Split on next best attribute (feature)
  - Use, for example, information gain to select attribute:
    \[ \arg \max_i IG(X_i) = \arg \max_i H(Y) - H(Y | X_i) \]
- Recurse

When to stop?

- First split looks good! But, when do we stop?

Base Case

One

Don't split a node if all matching records have the same output value

Two

Don't split a node if data points are identical on remaining attributes
Base Cases: An idea

- **Base Case One:** If all records in current data subset have the same output then don’t recurse
- **Base Case Two:** If all records have exactly the same set of input attributes then don’t recurse

Proposed Base Case 3:
If all attributes have small information gain then don’t recurse

- This is not a good idea

If we omit proposed case 3:

\[ y = a \text{ XOR } b \]

The resulting decision tree:

Instead, perform pruning after building a tree

The problem with proposed case 3

\[ y = a \text{ XOR } b \]

The information gains:

<table>
<thead>
<tr>
<th>Input</th>
<th>Value</th>
<th>Distribution</th>
<th>Info Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Decision trees will overfit

![Decision tree graph](image-url)
Decision trees will overfit

- Standard decision trees have no learning bias
  - Training set error is always zero!
  - (If there is no label noise)
  - Lots of variance
  - Must introduce some bias towards simpler trees
- Many strategies for picking simpler trees
  - Fixed depth
  - Fixed number of leaves
- Random forests

Real-valued inputs

- What should we do if some of the inputs are real-valued?
- What should we do if some of the inputs are real-valued?

Threshold splits

- Binary tree: split on attribute X at value t
  - One branch: X < t
  - Other branch: X ≥ t
- Requires small change
  - Allow repeated splits on same variable along a path

Hopeless: hypothesis with such a high branching factor will shatter any dataset and overfit
The set of possible thresholds

- Binary tree, split on attribute $X$
  - One branch: $X < t$
  - Other branch: $X \geq t$
- Search through possible values of $t$
  - Seems hard!!!
- But only a finite number of $t$’s are important:

  \[
  x_1, x_2, \ldots, x_m
  \]
- Sort data according to $X$ into \{ $x_i (x_{i+1} - x_i) / 2$ \}
- Moreover, only splits between examples from different classes matter!

Picking the best threshold

- Suppose $X$ is real valued with threshold $t$
- Want $IG(Y \mid X:t)$, the information gain for $Y$ when testing if $X$ is greater than or less than $t$
- Define:
  - $H(Y \mid X:t) = p(X<t)H(Y \mid X < t) + p(X=t)H(Y \mid X = t)$
  - $IG(Y \mid X:t) = H(Y) - H(Y \mid X:t)$
  - $IG^*(Y \mid X) = \max_{t} IG(Y \mid X:t)$
- Use: $IG^*(Y \mid X)$ for continuous variables

Example with MPG

<table>
<thead>
<tr>
<th>mpg values:</th>
<th>bad</th>
<th>good</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cylinders</td>
<td>&lt; 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6+</td>
<td></td>
</tr>
<tr>
<td>displacement</td>
<td>&lt; 188</td>
<td></td>
</tr>
<tr>
<td></td>
<td>188+</td>
<td></td>
</tr>
<tr>
<td>horsepower</td>
<td>&lt; 94</td>
<td></td>
</tr>
<tr>
<td></td>
<td>94+</td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>&lt; 2789</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2789+</td>
<td></td>
</tr>
<tr>
<td>acceleration</td>
<td>&lt; 18.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>18.2+</td>
<td></td>
</tr>
<tr>
<td>make</td>
<td>&lt; 81</td>
<td></td>
</tr>
<tr>
<td></td>
<td>81+</td>
<td></td>
</tr>
<tr>
<td>origin</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mpg values:</th>
<th>bad</th>
<th>good</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cylinders</td>
<td>&lt; 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6+</td>
<td></td>
</tr>
<tr>
<td>displacement</td>
<td>&lt; 188</td>
<td></td>
</tr>
<tr>
<td></td>
<td>188+</td>
<td></td>
</tr>
<tr>
<td>horsepower</td>
<td>&lt; 94</td>
<td></td>
</tr>
<tr>
<td></td>
<td>94+</td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>&lt; 2789</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2789+</td>
<td></td>
</tr>
<tr>
<td>acceleration</td>
<td>&lt; 18.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>18.2+</td>
<td></td>
</tr>
<tr>
<td>make</td>
<td>&lt; 81</td>
<td></td>
</tr>
<tr>
<td></td>
<td>81+</td>
<td></td>
</tr>
<tr>
<td>origin</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td></td>
</tr>
</tbody>
</table>

Example tree for our continuous dataset

- Suppose $X$ is real valued with threshold $t$
- Want $IG(Y \mid X:t)$, the information gain for $Y$ when testing if $X$ is greater than or less than $t$
- Define:
  - $H(Y \mid X:t) = p(X<t)H(Y \mid X < t) + p(X=t)H(Y \mid X = t)$
  - $IG(Y \mid X:t) = H(Y) - H(Y \mid X:t)$
  - $IG^*(Y \mid X) = \max_{t} IG(Y \mid X:t)$
- Use: $IG^*(Y \mid X)$ for continuous variables
What you need to know about decision trees

- Decision trees are one of the most popular ML tools
  - Easy to understand, implement, and use
  - Computationally cheap (to solve heuristically)
- Information gain to select attributes (ID3, C4.5,...)
- Presented for classification, can be used for regression and density estimation too
- Decision trees will overfit!!!
  - Must use tricks to find “simple trees”, e.g.,
    - Fixed depth/Early stopping
    - Pruning
  - Or, use ensembles of different trees (random forests)