Decision Trees & Limits of Learning

CMSC 422

Marine Carpuat
marine@cs.umd.edu

Credit: some examples & figures by Tom Mitchell
Today’s Topics

• Decision trees
  – What is the inductive bias?
  – Generalization issues: overfitting/underfitting

• Practical concerns: dealing with data
  – Train/dev/test sets
  – From raw data to well-defined examples
DECISION TREES
Recap: An example training set

<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>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
Recap: A decision tree to decide whether to play tennis
Recap: Function Approximation with Decision Trees

Problem setting

- **Set of possible instances** \( X \)
  - Each instance \( x \in X \) is a feature vector \( x = [x_1, \ldots, x_D] \)
- **Unknown target function** \( f: X \rightarrow Y \)
  - \( Y \) is discrete valued
- **Set of function hypotheses** \( H = \{ h \mid h: X \rightarrow Y \} \)
  - Each hypothesis \( h \) is a decision tree

Input

- **Training examples** \( \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\} \) of unknown target function \( f \)

Output

- Hypothesis \( h \in H \) that best approximates target function \( f \)
Decision Trees

- What is a decision tree?
- How to learn a decision tree from data?
- What is the inductive bias?
- Generalization?
  - Overfitting/underfitting
  - Selecting train/dev/test data
Inductive bias in decision tree learning

- Our learning algorithm performs heuristic search through space of decision trees
- It stops at smallest acceptable tree
- Why do we prefer small trees?
  - Occam’s razor: prefer the simplest hypothesis that fits the data
Why prefer short hypotheses?

• Pros
  – Fewer short hypotheses than long ones
    • A short hypothesis that fits the data is less likely to be a statistical coincidence

• Cons
  – What’s so special about short hypotheses?
Evaluating the learned hypothesis $h$

- Assume
  - we’ve learned a tree $h$ using the top-down induction algorithm
  - It fits the training data perfectly

- Are we done? Can we guarantee we have found a good hypothesis?
Recall: Formalizing Induction

- Given
  - a loss function $l$
  - a sample from some unknown data distribution $D$

- Our task is to compute a function $f$ that has low expected error over $D$ with respect to $l$.

$$
\mathbb{E}_{(x,y) \sim D} \{l(y, f(x))\} = \sum_{(x,y)} D(x, y) l(y, f(x))
$$
Training error is not sufficient

• We care about **generalization** to new examples

• A tree can classify training data perfectly, yet classify new examples incorrectly
  – Because training examples are only a sample of data distribution
    • a feature might correlate with class by coincidence
  – Because training examples could be noisy
    • e.g., accident in labeling
Let’s add a noisy training example. How does this affect the learned decision tree?

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
</tr>
<tr>
<td>D15</td>
<td>Sunny</td>
<td>Hot</td>
<td>Normal</td>
<td>Strong</td>
</tr>
</tbody>
</table>
Overfitting

• Consider a hypothesis $h$ and its:
  – Error rate over training data $error_{train}(h)$
  – True error rate over all data $error_{true}(h)$

• We say $h$ overfits the training data if
  \[ error_{train}(h) < error_{true}(h) \]

• Amount of overfitting =
  \[ error_{true}(h) - error_{train}(h) \]
Evaluating on test data

• Problem: we don’t know $error_{true}(h)$!

• Solution:
  – we set aside a test set
    • some examples that will be used for evaluation
  – we don’t look at them during training!
  – after learning a decision tree, we calculate $error_{test}(h)$
Measuring effect of overfitting in decision trees
Overfitting

• Another way of putting it

• A hypothesis $h$ is said to overfit the training data, if there is another hypothesis $h'$, such that
  – $h$ has a smaller error than $h'$ on the training data
  – but $h$ has larger error on the test data than $h'$. 
Underfitting/Overfitting

• Underfitting
  – Learning algorithm had the opportunity to learn more from training data, but didn’t

• Overfitting
  – Learning algorithm paid too much attention to idiosyncracies of the training data; the resulting tree doesn’t generalize
Practical impact on decision tree learning

• What we want:
  – A decision tree that neither underfits nor overfits
  – Because it is expected to do best in the future

• How can we encourage that behavior?
  – Set a maximum tree depth D
Decision Trees

• What is a decision tree?
• How to learn a decision tree from data?
• What is the inductive bias?
  – Occam’s razor: preference for short trees
• Generalization?
  – Overfitting/underfitting
Your thoughts?

What are the pros and cons of decision trees?
DEALING WITH DATA
1 robocop is an intelligent science fiction thriller and social satire, one with class and style. The film, set in old Detroit in the year 1991, stars Peter Weller as Murphy, a lieutenant on the city's police force. 1991's Detroit suffers from rampant crime and a police department run by a private contractor (Security Concepts Inc.). The employees (the cops) are threatening to strike. To make matters worse, a savage group of cop-killers has been terrorizing the city. [...]

Do the folks at Disney have no common decency? They have resurrected yet another cartoon and turned it into a live-action hodgepodge of expensive special effects, embarrassing writing and kid-friendly slapstick. Wasn't Mr. Magoo enough, people? Obviously not. Inspector gadget is not what I would call ideal family entertainment. [...]

How would you define input vectors $x$ to represent each example? What features would you use?
Train/dev/test sets

In practice, we always split examples into 3 distinct sets

• **Training set**
  – Used to learn the **parameters** of the ML model
  – e.g., what are the nodes and branches of the decision tree

• **Development set**
  – aka tuning set, aka validation set, aka held-out data)
  – Used to learn **hyperparameters**
    • Parameter that controls other parameters of the model
    • e.g., max depth of decision tree

• **Test set**
  – Used to evaluate how well we’re doing on new unseen examples
Cardinal rule of machine learning:

Never ever touch your test data!
Summary: what you should know

**Decision Trees**
- What is a decision tree, and how to induce it from data

**Fundamental Machine Learning Concepts**
- Difference between memorization and generalization
- What inductive bias is, and what is its role in learning
- What underfitting and overfitting means
- How to take a task and cast it as a learning problem

**Why you should never ever touch your test data!!**