Overview

- Research talk wrap-up
- Theorem prover tips
- Learning decision trees

Final exam: Monday, May 6th at 10:30am until 12:30pm in Rm. 126 HRBB.

Prj2 Tips

- Make variables unique across different clauses.
- Within a single clause, a variable may appear several times.
- Across different clauses, a constant or function may appear several times (be careful with function arguments though).
- Test your prover with really simple theorems. Gradually increase the number of clauses.

\[ \begin{align*}
(1 & ((P X)) \text{ NIL}) \\
(2 & \text{ NIL} \quad ((P \ (A)))) \\
\end{align*} \]

- All three theorems (howling hound, roadrunner, and customs) are provable.

Inductive Learning

- Given example pairs \((x, f(x))\), return a function \(h\) that approximates the function \(f\):
  - pure inductive inference, or induction.
- The function \(h\) is called a hypothesis.

Inductive Learning and Bias

Given (a) as the training data, we can come up with several different hypotheses: (b) to (d)

- selection of one hypothesis over another is called a bias.
  - exact match to training data
  - prefer imprecise but smooth approximation
  - etc.
• learn to approximate **discrete-valued** target functions.

• step-by-step decision making (disjunction of conjunctions)

• applications: medical diagnosis, assess credit risk of loan applicants, etc.

---

**Decision Trees: What They Represent (cont’d)**

- In other words, for each instance (or example), there are attributes (Patrons, Hungry, etc.) and each instance have a full attribute value assignment.
- For a given instance, it is classified into different discrete classes by the decision tree.
- For training, many (instance, class) pairs are used.

---

**Decision Trees: What They Represent**

Wait or not (**Yes/No**)? The decision tree above corresponds to:

\[
(Patrons = \text{Full} \land Hungry = \text{No} \land Type = \text{French})
\]
\[
(Patrons = \text{Full} \land Hungry = \text{No} \land Type = \text{Thai} \land Fri/Sat = \text{Yes})
\]
\[
(Patrons = \text{Full} \land Hungry = \text{No} \land Type = \text{Burger})
\]

Decision trees represent disjunction of conjunctions.

---

**Constructing Decision Trees from Examples**

- Given a set of examples (training set), both **positive** and **negative**, the task is to construct a decision tree that describes a concise decision path.
- Using the resulting decision tree, we want to **classify** new instances of examples (either as **yes** or **no**).
Constructing Decision Trees: Trivial Solution

- A trivial solution is to explicitly construct paths for each given example.
- The problem with this approach is that it is not able to deal with situations where, some attribute values are missing or new kinds of situations arise.
- Consider that some attributes may not count much toward the final classification.

Finding a Concise Decision Tree

- Memorizing all cases may not be the best way.
- We want to extract a decision pattern that can describe a large number of cases in a concise way.
- Such an inductive bias is called Ockham's razor: The most likely hypothesis is the simplest one that is consistent with all observations.
- In terms of a decision tree, we want to make as few tests before reaching a decision, i.e. the depth of the tree should be shallow.

Finding a Concise Decision Tree (cont’d)

- Basic idea: pick up attributes that can clearly separate positive and negative cases.
- These attributes are more important than others: the final classification heavily depend on the value of these attributes.
### Decision Tree Learning Algorithm

**Function** `DECISION-TREE-LEARNING(examples, attributes, default)` **Returns** a decision tree

**Inputs:**
- `examples`, set of examples
- `attributes`, set of attributes
- `default`, default value for the goal predicate

1. If `examples` is empty, return `default`.
2. If all examples have the same classification, return the classification.
3. If `attributes` is empty, return `MAJORITY-VALUE(examples)`.
4. Choose an attribute `best` to test and a new decision tree `tree` with root test `best`
5. For each value `v_i` of `best`
   - Let `examples_i` be elements of `examples` with `best = v_i`
   - Let `subtree` be `DECISION-TREE-LEARNING(examples_i, attributes - best, MAJORITY-VALUE(examples_i))`
   - Add a branch to `tree` with label `v_i` and subtree `subtree`

**Accuracy of Decision Trees**

![Accuracy Graph]

- Divide examples into training and test sets.
- Train using the training set.
- Measure accuracy of resulting decision tree on the test set.

- Some attributes are not tested at all.
- Odd paths can be generated (Thai food branch).
- Sometimes the tree can be incorrect for new examples (exceptional cases).

**Choosing the Best Attribute to Test First**

Use Shannon’s information theory to choose the attribute that gives the maximum information gain.

- Pick an attribute such that the information gain (or entropy reduction) is maximized.
- Entropy measures the average surprisal of events. Less probable events are more surprising.
**Entropy and Information Gain**

\[
Entropy(E) = \sum_{i \in C} -P_i \log_2(P_i)
\]

\[
Gain(E, A) = Entropy(E) - \sum_{v \in Values(A)} \frac{|E_v|}{|E|} Entropy(E_v)
\]

- \(E\): set of examples
- \(A\): a single attribute
- \(E_v\): set of examples where attribute \(A = v\).
- \(|S|\): cardinality of set \(S\).

**Key Points**

Decision tree learning:

- What is the embodied principle (or bias)?
- How to choose the best attribute? Given a set of examples, choose the best attribute to test first.
- What are the issues? noise, overfitting, etc.

**Issues in Decision Tree Learning**

- Noise and overfitting
- Missing attribute values from examples
- Multivalued attributes with large number of possible values
- Continuous-valued attributes.

**Next Time**

- Monday: final exam review
- Tuesday: redefined day – general Q and A (attendance not required). Recommended reading.