Overfitting in Decision Trees

- If a decision tree is fully grown, it may lose some generalization capability.
- This is a phenomenon known as *overfitting*.

Definition of Overfitting

Consider the error of hypothesis h. We let error on the training data be $\mathrm{error}_{train}(h)$ and error over the entire distribution D of data be $\mathrm{error}_D(h)$.

Then a hypothesis h "overfits" the training data if there is an alternative hypothesis, h', such that:

$$\mathrm{error}_{train}(h) < \mathrm{error}_{train}(h')$$
$$\mathrm{error}_D(h) < \mathrm{error}_D(h')$$

Model Overfitting

Errors committed by classification models are generally divided into two types:

1. **Training Errors**
   The number of misclassification errors committed on training records; also called resubstitution error.

2. **Generalization Errors**
   The expected error of the model on previously unseen records.

Causes of Overfitting

1. **Overfitting Due to Presence of Noise**
   Mislabeled instances may contradict the class labels of other similar records.

2. **Overfitting Due to Lack of Representative Instances**
   Lack of representative instances in the training data can prevent refinement of the learning algorithm.

3. **Overfitting and the Multiple Comparison Procedure**
   Failure to compensate for algorithms that explore a large number of alternatives can result in spurious fitting.

Overfitting Due to Noise: An Example

An example training set for classifying mammals. Asterisks denote mislabelings.

Name	**Body Temperature**	**Gives Birth**	**Four-legged**	**Hibernates**	**Class Label**
Porcupine	Warm-blooded	Yes	Yes	Yes	***Yes***
Cat	Warm-blooded	Yes	Yes	No	***Yes***
Bat	Warm-blooded	Yes	No	Yes	***No****
Whale	Warm-blooded	Yes	No	No	***No****
Salamander	Cold-blooded	No	Yes	Yes	***No***
Komodo dragon	Cold-blooded	No	Yes	No	***No***
Python	Cold-blooded	No	No	Yes	***No***
Salmon	Cold-blooded	No	No	No	***No***
Eagle	Warm-blooded	No	No	No	***No***
Guppy	Cold-blooded	Yes	No	No	***No***

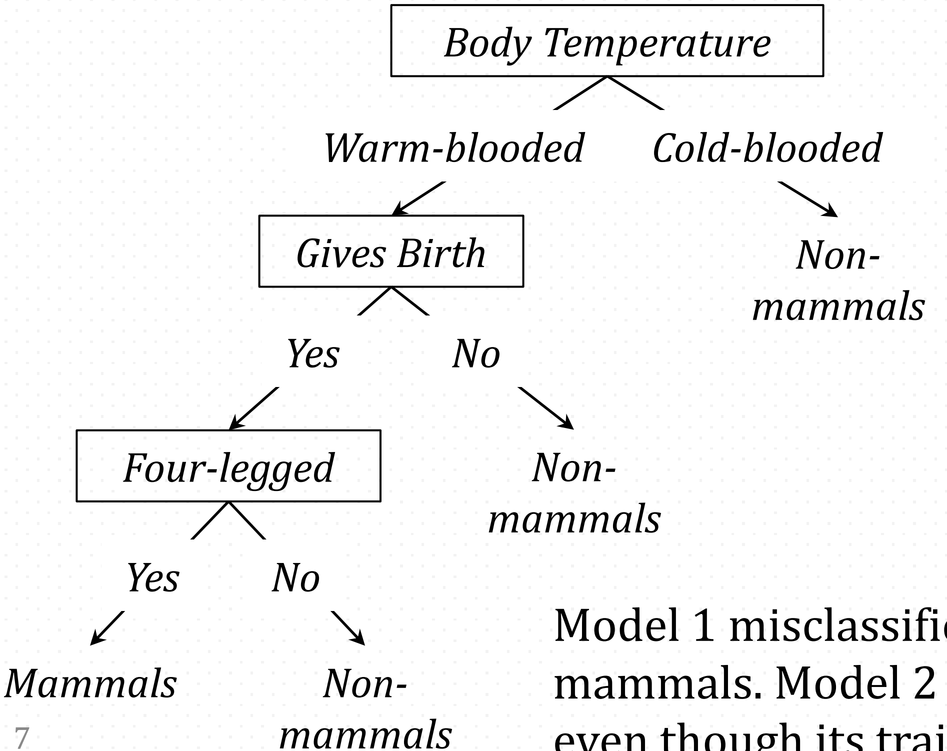
Overfitting Due to Noise

An example testing set for classifying mammals.

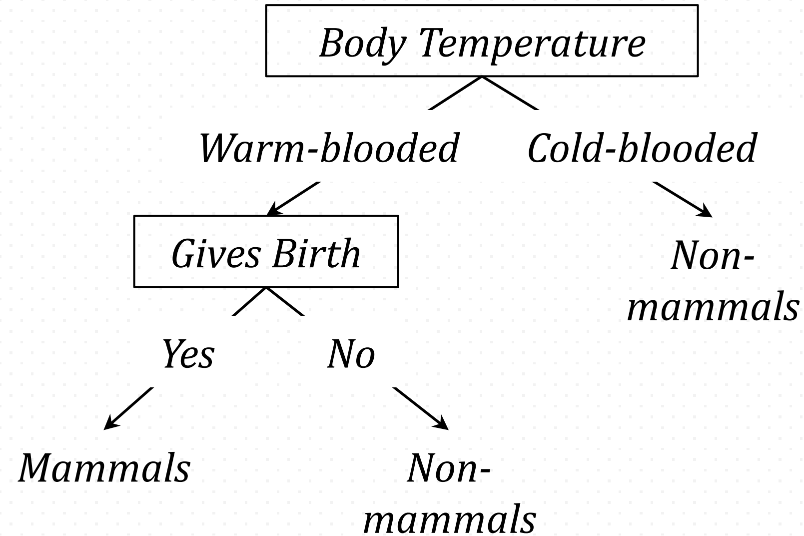
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
Human	Warm-blooded	Yes	No	No	***Yes***
Pigeon	Warm-blooded	No	No	No	***No***
Elephant	Warm-blooded	Yes	Yes	No	***Yes***
Leopard shark	Cold-blooded	Yes	No	No	***No***
Turtle	Cold-blooded	No	Yes	No	***No***
Penguin	Cold-blooded	No	No	No	***No***
Eel	Cold-blooded	No	No	No	***No***
Dolphin	Warm-blooded	Yes	No	No	***Yes***
Spiny anteater	Warm-blooded	No	Yes	Yes	***Yes***
Gila monster	Cold-blooded	No	Yes	Yes	***No***

Overfitting Due to Noise

Model 1



Model 2



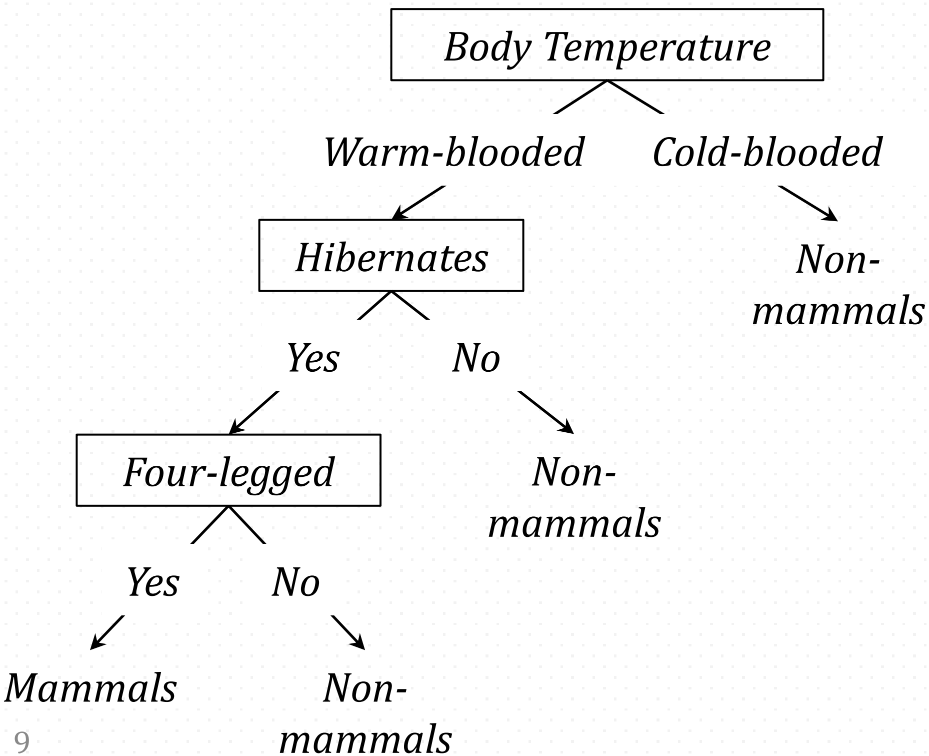
Model 1 misclassifies humans and dolphins as non-mammals. Model 2 has a lower test error rate (10%) even though its training error rate is higher (20%).

Overfitting Due to Lack of Samples

An example training set for classifying mammals.

Name	**Body Temperature**	**Gives Birth**	**Four-legged**	**Hibernates**	**Class Label**
Salamander	Cold-blooded	No	Yes	Yes	***No***
Guppy	Cold-blooded	Yes	No	No	***No***
Eagle	Warm-blooded	No	No	No	***No***
Poorwill	Warm-blooded	No	No	Yes	***No***
Platypus	Warm-blooded	No	Yes	Yes	***Yes***

Overfitting Due to Lack of Samples



Although the model's training error is zero, its error rate on the test set if 30%.

Humans, elephants, and dolphins are misclassified because the decision tree classifies all warm-blooded vertebrates that do not hibernate as non-mammals. The tree arrives at this classification decision because there is only one training records, which is an eagle, with such characteristics.

Model Overfitting

A good model must not only fit the training data well but also accurately classify records it has never seen.

In other words, a good model must have *low training error* **and** *low generalization error*.

Model Overfitting

A good model must not only fit the training data well but also accurately classify records it has never seen.

In other words, a good model must have *low training error* **and** *low generalization error*.

Occam's Razor

"Everything should be made as simple as possible, but no simpler."

All other things being equal, simple theories are preferable to complex ones.

Occam's Razor

But *why* prefer a short hypothesis?

1. There are fewer short hypotheses than long hypotheses.
2. A short hypothesis that fits the data is unlikely to be a coincidence.
3. A long hypothesis that fits the data might be a coincidence.

Minimum Description Length Principle

- A formalization of Occam's razor.
- The Idea:

The best hypothesis for a given set of data is the one that leads to the best compression of the data.

How do we measure "compression"?

MDL: Inuitive Explanation

Occam's razor: prefer the shortest hypothesis.

MDL: prefer the hypothesis h that minimizes the space required to describe a theory plus the space required to describe the theory's mistakes.

MDL: Formal Explanation

Occam's razor: prefer the shortest hypothesis.

MDL: prefer the hypothesis h that minimizes

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}}\, L_{C_1}(h) + L_{C_2}(D|h)$$

where L_{C_x} is the description length of x under encoding C.

MDL Example

Let H be a set of decision trees (hypotheses) and D be a set of training data labels. Then,

$L_{C_1}(h)$ is the number of bits to describe tree h.

$L_{C_2}(D|h)$ is the number of bits to describe D given h.

- Note that $L_{C_2}(D|h) = 0$ if all training instances are classified perfectly by h. It need only describe exceptions.

Hence h_{MDL} trades-off tree size for training errors.

MDL for Classification Models

- The hypothesis is the classification model and the description length is the combined description of the model and its errors on the training data.
- Using the MDL principle, we seek a classifier with *shortest* description.
- Used this way, the MDL principle is a *model selection criterion*—a way to select between potential models or hypotheses.

Model Selection Criteria

Model selection criteria attempt to find a good compromise between:

a) The model's complexity
b) The model's prediction accuracy on unseen data

- Reasoning: a good model is a simple model that achieves high accuracy on the given data
- Also known as Occam's Razor: the best theory is the smallest one that describes all the facts

Elegance vs. Errors

Consider the following two theories of some data:

Theory 1: very simple, elegant theory that explains the data almost perfectly

Theory 2: significantly more complex theory that reproduces the data without mistakes

Theory 1 is probably preferable.

Elegance vs. Errors Example

Canonical example: Kepler's laws of planetary motion.

- Actually *less* accurate the Copernicus's latest refinement of the Ptolemaic theory of epicycles.
- But far *simpler*.

"I have cleared the Augean stables of astronomy of cycles and spirals, and left behind me a single cartload of dung."
–Johannes Kepler

From Theory to Practice

Let's look at how to turn these ideas of model selection criteria into practice.

Avoiding Overfitting in Decision Trees

- Stop growing the tree when the data split is not statistically significant
- Grow the full tree, then prune
  - Do we really needs all the “small” leaves with perfect coverage?

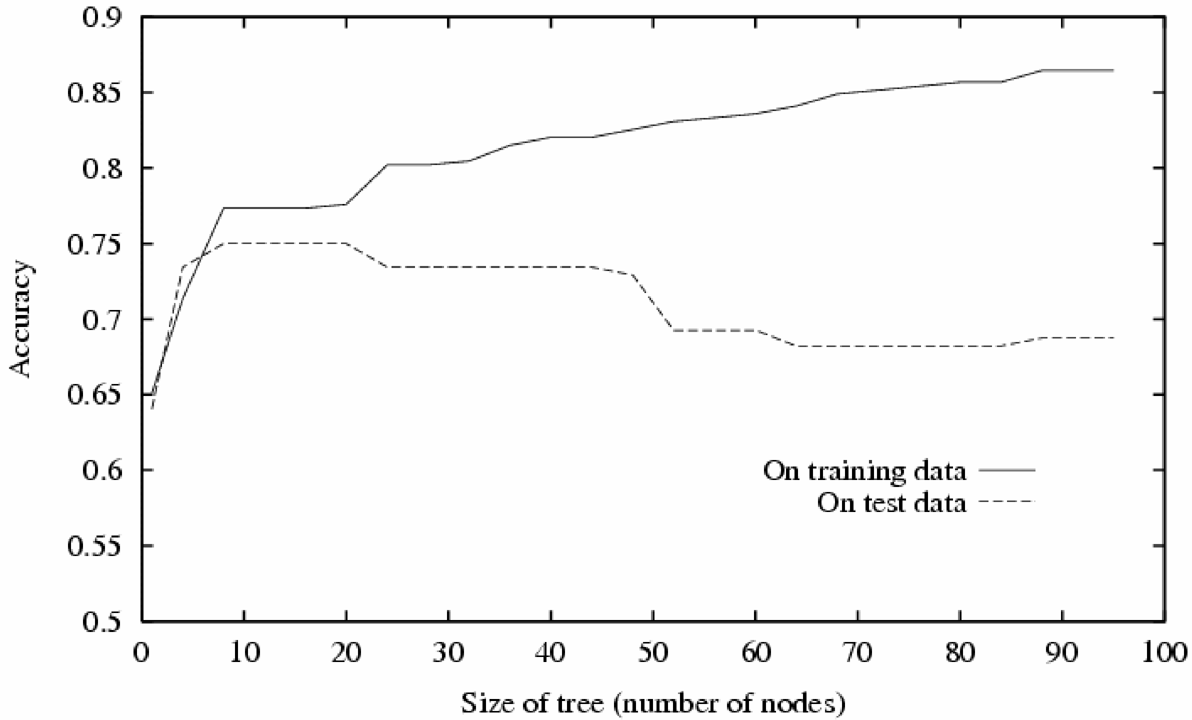
Avoiding Overfitting in Decision Trees

- How to select
  - Measure performance over training data (and include some estimates for generalization)
  - Measure performance over separate validation data
  - Use Minimum Description Length Principle (MDL)
    - Minimize, $size(tree) + size\big(misclassiciation(tree)\big)$

Decision Tree Pruning Methodologies

- Pre-pruning (top-down)
  - Stopping criteria while growing the tree
- Post-pruning (bottom-up)
  - Grow the tree, then prune
  - More popular

Decision Tree Overfitting



Decision Tree Pre-Pruning

- Stop the algorithm before it becomes a fully-grown tree
- Typical stopping conditions for a node
  - Stop if all instances belong to the same class
  - Stop if all the feature values are the same

Decision Tree Pre-Pruning

- More restrictive conditions
  - Stop if the number of instances is less than some use-specified threshold
  - Stop if the class distribution of instances are independent of the available features
    - Stop if expanding the current node does not improve impurity.

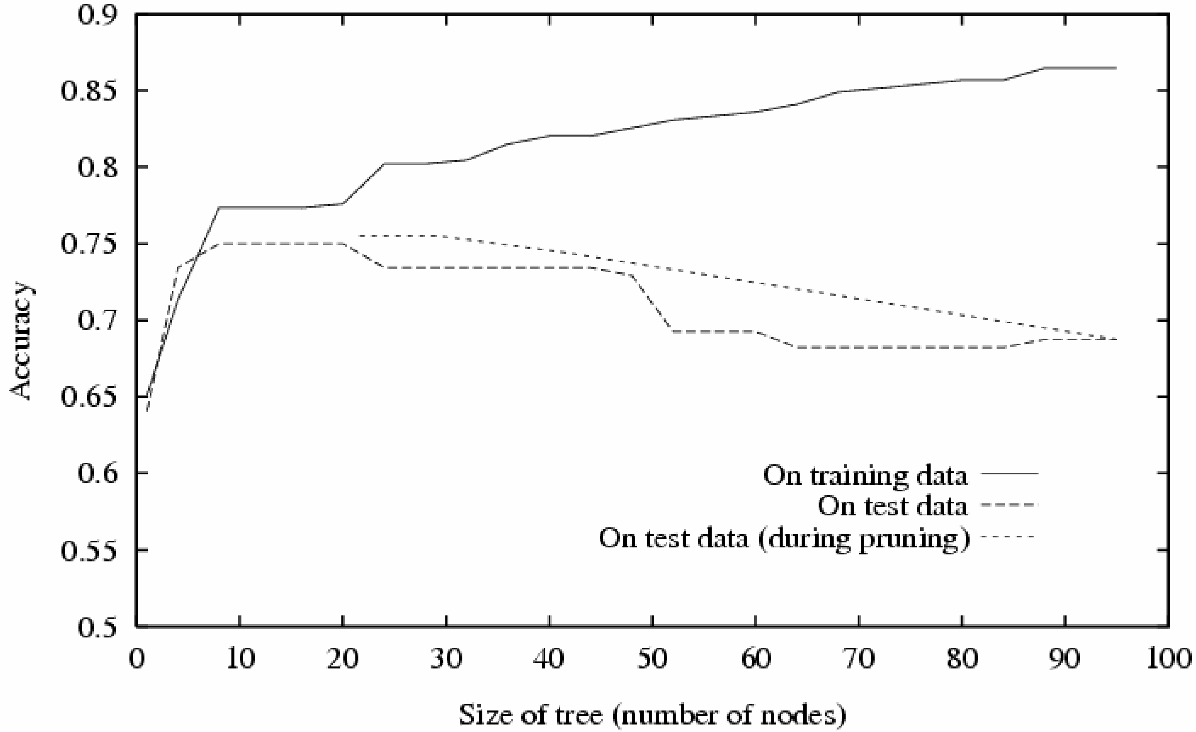
Decision Tree Post-Pruning

- Grow decision tree to its entirety
- Trim the nodes of the decision tree in a bottom-up fashion
- If generalization error improves after trimming, replace sub-tree by a leaf node
  - Class label of leaf node is determined from majority class of instances in the sub-tree
- Can use MDL for post-prunning

Decision Tree Post-Pruning

- Reduced Error Pruning
  - Split data into training and validation set
  - Remove one node at a time and evaluate the performance on the validation data
  - Remove the one that decreases the error
  - Usually produces the smallest version of a tree
  - But always requires a validation set

Decision Tree Pruning



Decision Trees: Pros and Cons

- Pros:
  - Fast in implementation
  - Works with all types of features
  - Insensitive to outliers
  - Few tuning parameters
  - Efficient in handling missing values
  - Interpretable model representation

Decision Trees: Pros and Cons

- Cons:
  - Not effective at approximating linear or smooth functions or boundaries
  - Trees always include high-order interactions
  - Large variance
    - Each split is conditional on all of its ancestor splits.