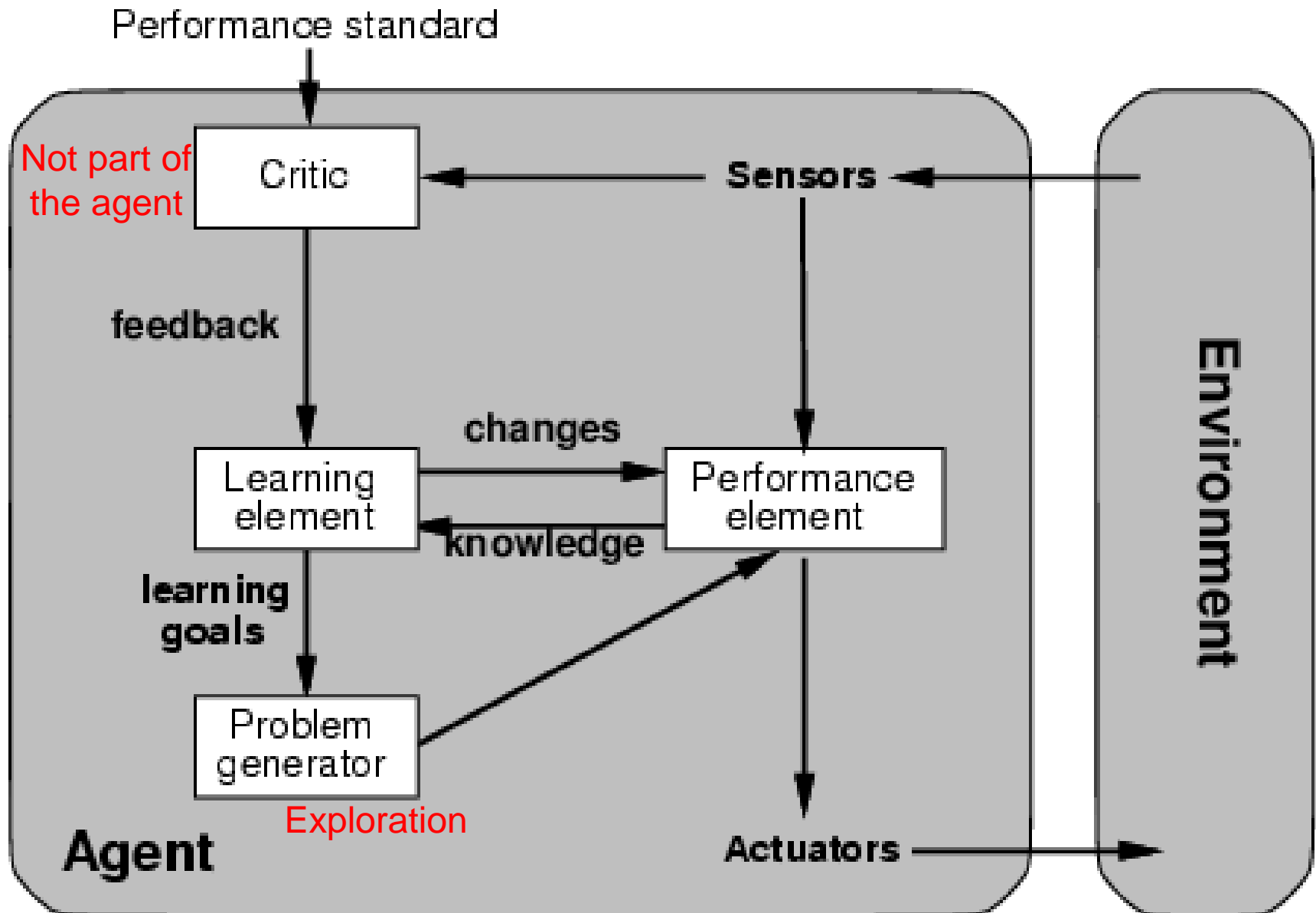# Learning

Purpose of Learning

- The agent can improve itself.

- The agent can adapt itself to different problems / scenarios.

- Why can't these improvements/abilities to handle different problems/scenarios be programmed in?

  - The designer can not program the agent for every possible problem/scenario that it may encounter.

  - The environment is stochastic or not fully observable.

  - The designer does not know how to program the agent for a task, except for letting the agent learn to solve it.

- Now think about some examples an agent can learn to do (i.e., not all its knowledge or action is hard-coded).

Learning Agent

Designing a Learning Agent

- Which components are to be affected by learning?

- What is already known (prior knowledge)?

- How are the components and the data represented:

  - Factored representation: A vector of attributes (numerical or categorical, continuous or discrete)

- What kind of feedback is available, if any?

Types of Learning

- **Unsupervised learning**: The agent needs to organize the object or data with no outside help and no feedback.

  - Clustering: To group objects into meaningful or useful "clusters".

- **Reinforcement leaning**: The available feedback is the outcome (reward or penalty). This can happen long after the decisions or actions that lead to the outcome.

  - Goal of learning: To minimize penalty and maximize reward.

- **Supervised learning**: The correct answers are given by an "teacher" during the learning process.

  - Goal of learning: To generate "correct" answers given the inputs.

Learning from Examples

- A form of supervised learning

- A set of example cases (training samples) and their corresponding correct/desired outputs are given.

- The goal is to form a mechanism that, when given an example case, will produce the correct output.

Example of Learning from Examples

Example problem: Learning to decide whether to wait for a table when arriving at a restaurant:

Should I keep waiting, or should I go to 7-eleven instead?

Example of Learning from Examples

The example cases:

Examples	Input Attributes										Output
	Alt.	Bar	Fri/Sat	Hun	Pat.	Price	Rain	Res.	Type	Time	Wait?
x_1	Y	N	N	Y	Some	\$\$\$	N	Y	French	0-10	Y
x_2	Y	N	N	Y	Full	\$	N	N	Thai	30-60	N
x_3	N	Y	N	N	Some	\$	N	N	Burger	0-10	Y
x_4	Y	N	Y	Y	Full	\$	Y	N	Thai	10-30	Y
x_5	Y	N	Y	N	Full	\$\$\$	N	Y	French	>60	N
x_6	N	Y	N	Y	Some	\$\$	Y	Y	Italian	0-10	Y
x_7	N	Y	N	N	None	\$	Y	N	Burger	0-10	N
x_8	N	N	N	Y	Some	\$\$	Y	Y	Thai	0-10	Y
x_9	N	Y	Y	N	Full	\$	Y	N	Burger	>60	N
x_{10}	Y	Y	Y	Y	Full	\$\$\$	N	Y	Italian	10-30	N
x_{11}	N	N	N	N	None	\$	N	N	Thai	0-10	N
x_{12}	Y	Y	Y	Y	Full	\$	N	N	Burger	30-60	Y

Decision Tree

The input-output relations are organized into a tree. Each node divides its set of samples into subsets according to a particular input attribute.
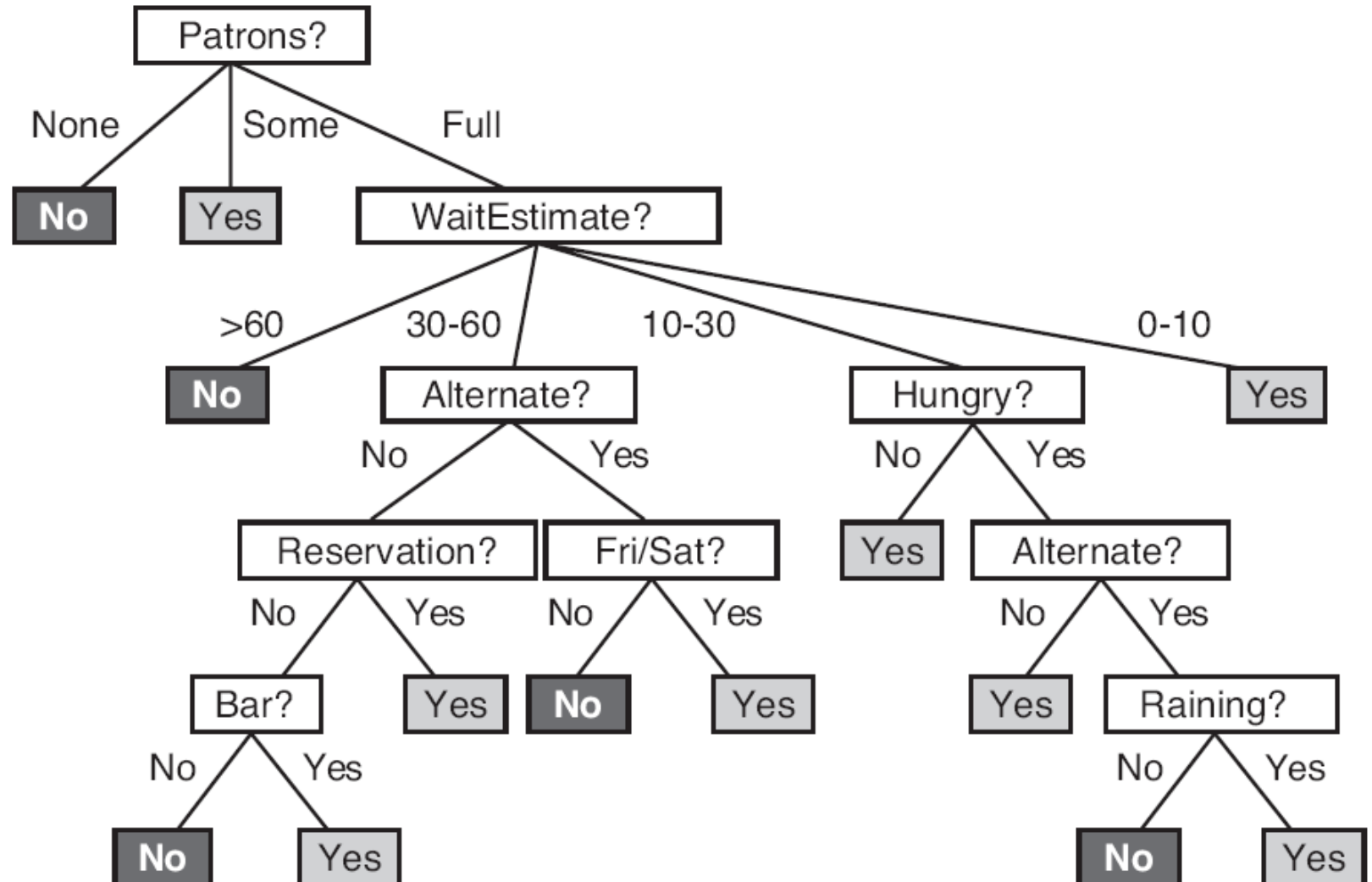
Decision Tree Learning

- Each tree node is associated with a set of labeled samples.

- In the beginning, we generate only the root node, which is associated with all the samples.

- The following steps are applied to each new node:

  - Select an attribute as the basis for splitting the samples of the current node.

  - Divide the values of the selected attribute in the set of samples into several groups. A new tree node is generated for each of these groups.

- The key is <u>how to split the samples</u>. (How to select an attribute, and how to partition the values of the selected attribute.)

Choosing Attribute Tests: The Idea

Idea: a good attribute splits the samples into subsets that (ideally) contains samples of the same labels.

Q: Which of the following two attributes are better?
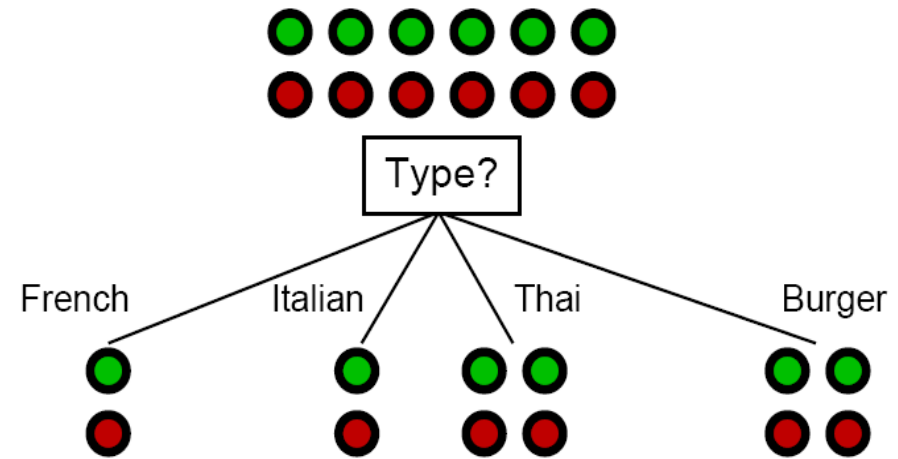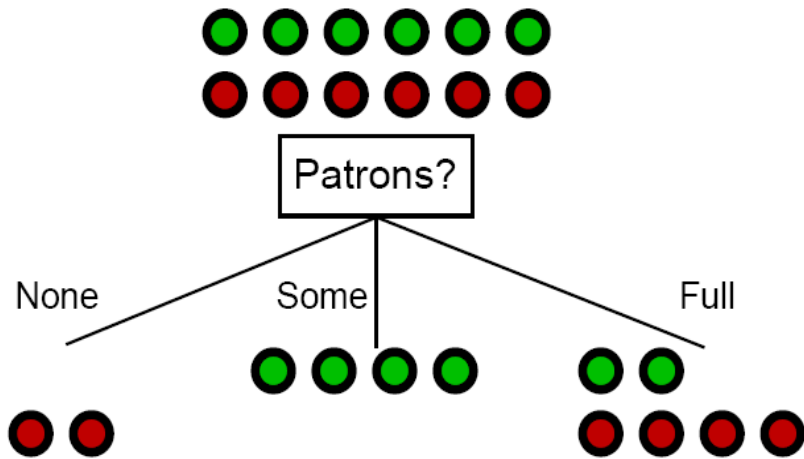
Choosing Attribute Tests

- The goal of splitting is so that the sample labels (desired outputs) of the child nodes are more "pure" (or are "less uncertain") than those in the parent node.

- How to measure the "impurity" or "uncertainty" of the label distribution of a node? Some typical ones:

  - Shannon's Entropy: $H(V) = -\sum_k P(v_k) \log_2 P(v_k)$

  - Gini Index: $G(V) = 1 - \sum_k P(v_k)^2$

Choosing Attribute Tests

- The goal of splitting is so that the sample labels (desired outputs) of the child nodes are more "pure" (or are "less uncertain") than those in the parent node.

- Some representative metrics of "impurity" or "uncertainty":

  - Shannon's Entropy: $$H = -\sum_k p_k \log_2 p_k$$

  - Gini Index: $$G = 1 - \sum_k p_k^2$$

    Here p_k is the ratio of items with label k.
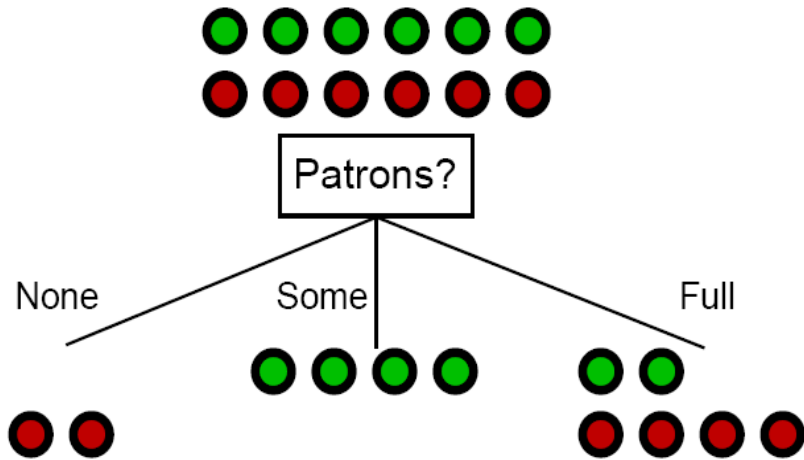
Choosing Attribute Tests

Now, consider an attribute with m possible values. Splitting a node (A) with N items on this attribute will result in m child nodes (C_1, C_2, … C_m) with N_1, N_2, … N_m items, respectively. The idea is to select the attribute that gives the most reduction in uncertainty/impurity (entropy used here):

$$\Delta H = H(A) - \sum_{i=1}^{m} \frac{N_i}{N} H(C_i)$$

For entropy, this is also called the *information gain*.

Choosing Attribute Tests

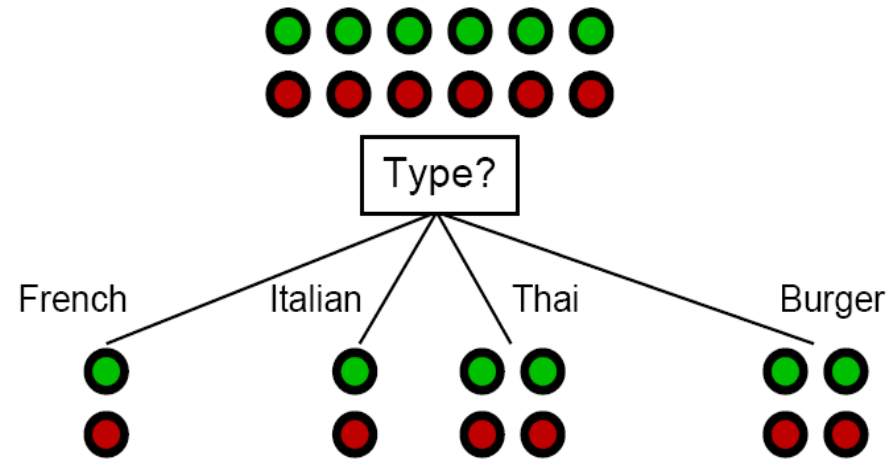Now we can try to choose an attribute again:

$$H(A) = 2\left(-\frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(C_1) = H(C_2) = 0$$

$$H(C_3) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.80$$

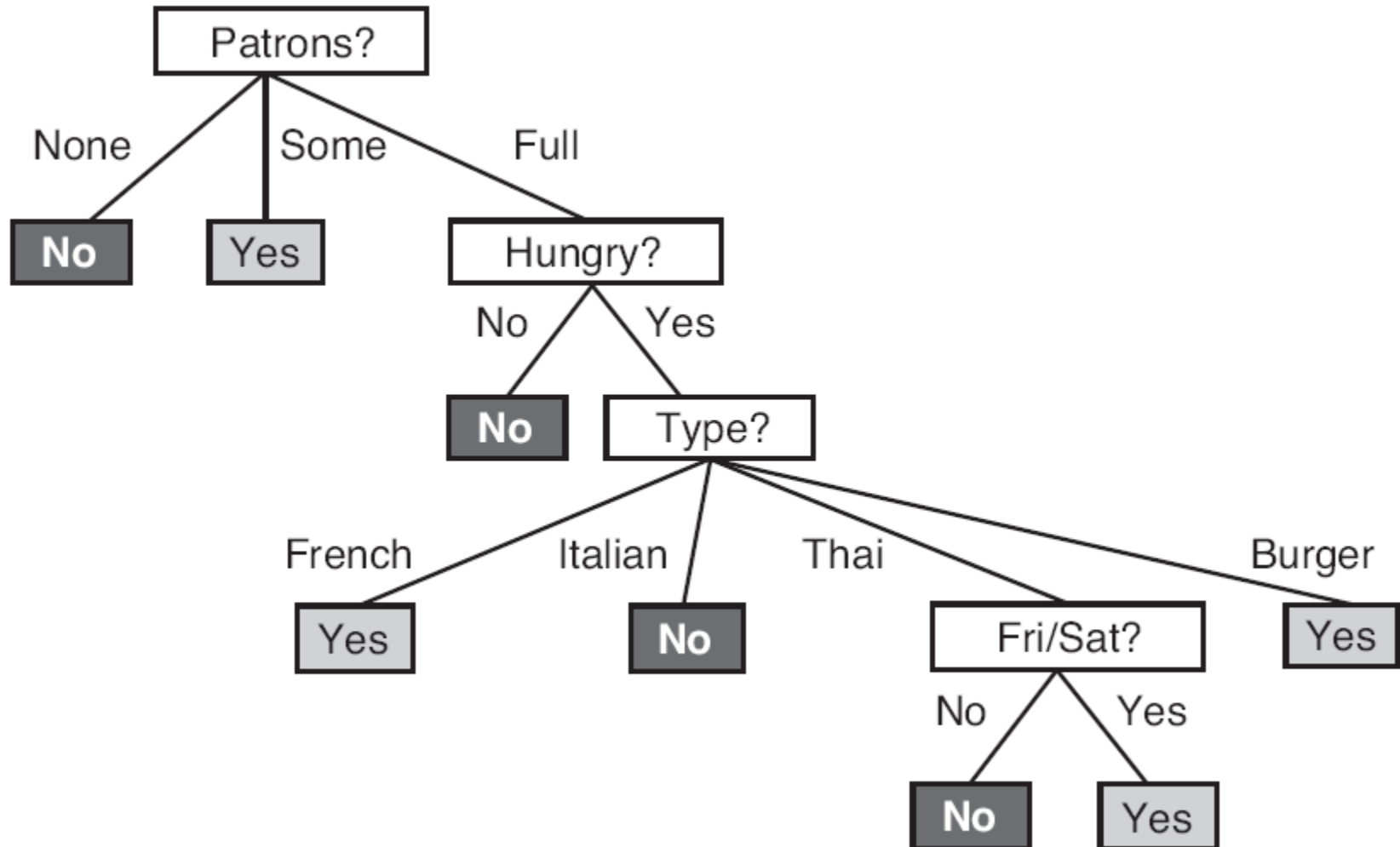$$\Delta H = 1 - (2/12)*0 - (4/12)*0 - (6/12)*0.80$$
$$= 0.60$$

$$H(C_1) = H(C_2) = H(C_3) = H(C_4) = 1$$

$$\Delta H = 0$$

Choosing Attribute Tests

This tree is substantially smaller than the first one, but it gives the same outputs for all the training samples.

Supervised Learning Overview

■ Main usages of supervised learning:

● **Classification**: The expected output is one of several categories. Such an agent / program is called a **classifier**.

● **Regression**: The expected output is a numerical value. Example: temperature in weather forecast.

■ Both involve some input/output mapping: $f : x \rightarrow y$
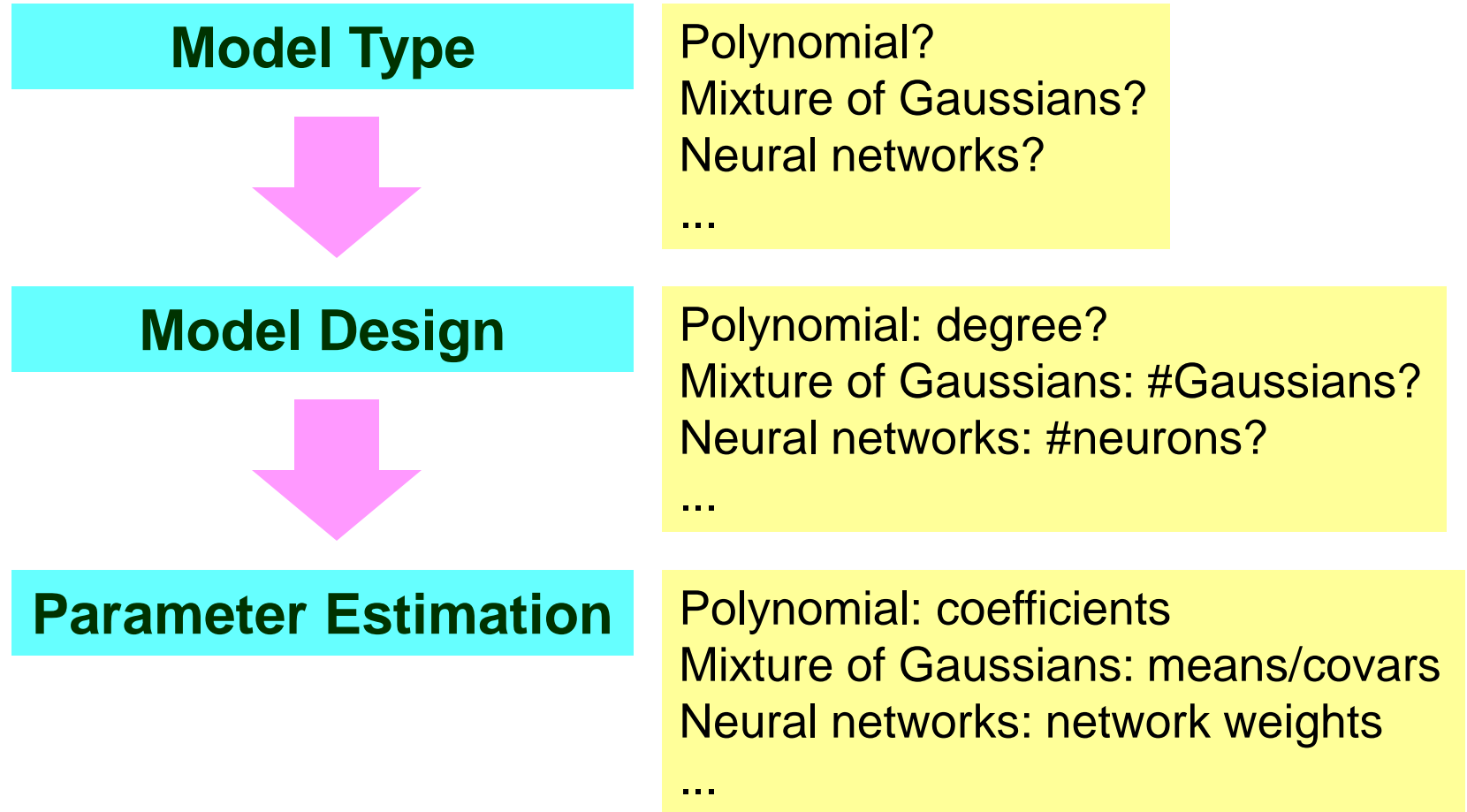
Supervised Learning Overview

Training and testing:

- Training data: Collect samples in the form of $(x \rightarrow y)$.

  - For such a pair of $(x \rightarrow y)$, y is often called the ground-truth or target output of x.

  - If y represents a category (class), then it is also called the class label (or simply the label) of x.

- Training: Derive an estimated model of the mapping f using the training data.

- Testing: Using the derive model, determine y for some given x (normally not in the training data).

Supervised Learning Overview

- There are numerous possible forms (model types) to represent the mapping. Examples:
  - Rule Sets
  - Polynomials
  - Gaussian Mixtures
  - k Nearest Neighbors
  - Decision Trees
  - Neural Networks
  - …

- There are many possible forms for representing the mapping of a given set of training data. This means that the actual form used is a <u>design choice</u>.

- For each given form of representation, there are <u>parameters</u> to be determined according to the training data.
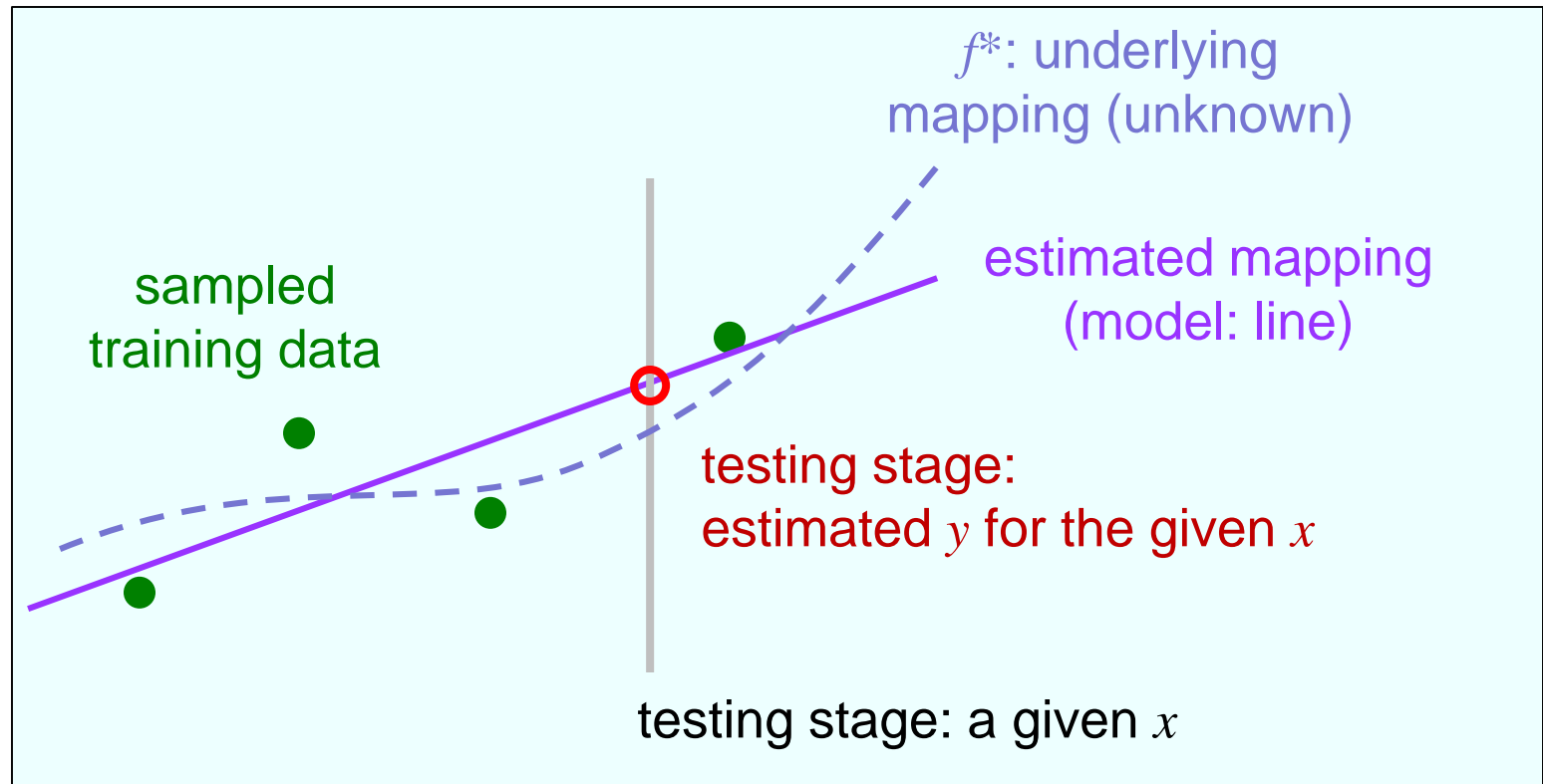
Supervised Learning Overview

■ Stages of learning a model for $f : x \rightarrow y$:

Model Type	Polynomial? Mixture of Gaussians? Neural networks? ...
Model Design	Polynomial: degree? Mixture of Gaussians: #Gaussians? Neural networks: #neurons? ...
Parameter Estimation	Polynomial: coefficients Mixture of Gaussians: means/covars Neural networks: network weights ...

Training and Testing

Here we will use regression as an example.

Bias-Variance Dilemma

Desired properties of the estimated mapping:

- Low **Bias** (goodness of fitting):
  - The estimated mapping should fit to the training data as well as possible.
  - Goal: To capture the behaviors of the underlying mapping as well as possible.
- Low **Variance** (high confidence):
  - Assume that we have several estimated mappings, each derived from a different set of training data.
  - For a given x, each mapping gives an estimations for y.
  - The estimations for y should by as similar to each other as possible.
  - Goal: To be confident of the estimation of y.

Bias-Variance Dilemma

For supervised learning, we always need to consider the trade-off between bias and variance.

- Reason 1:

  Sampled Value =
  Signal (from underlying mapping)
  + Noise

  Noise is a reality of life that you have to deal with.

- Reason 2: We can only have a finite number of training samples. (If we have infinite number of training samples, the noise can be averaged out.)

Bias-Variance Dilemma

Now, difficulties for supervised learning:

- To reduce **Bias**

  ➔ Use more complex models

  ➔ BOTH signal and noise are learned

  ➔ Overfitting

- To reduce **Variance**

  ➔ Try to avoid learning noise (variance is caused by noise)

  ➔ Use less complex models

  ➔ Signal is not sufficiently learned

  ➔ Underfitting

Fitting and Model Complexity

Using polynomial regression as an example:
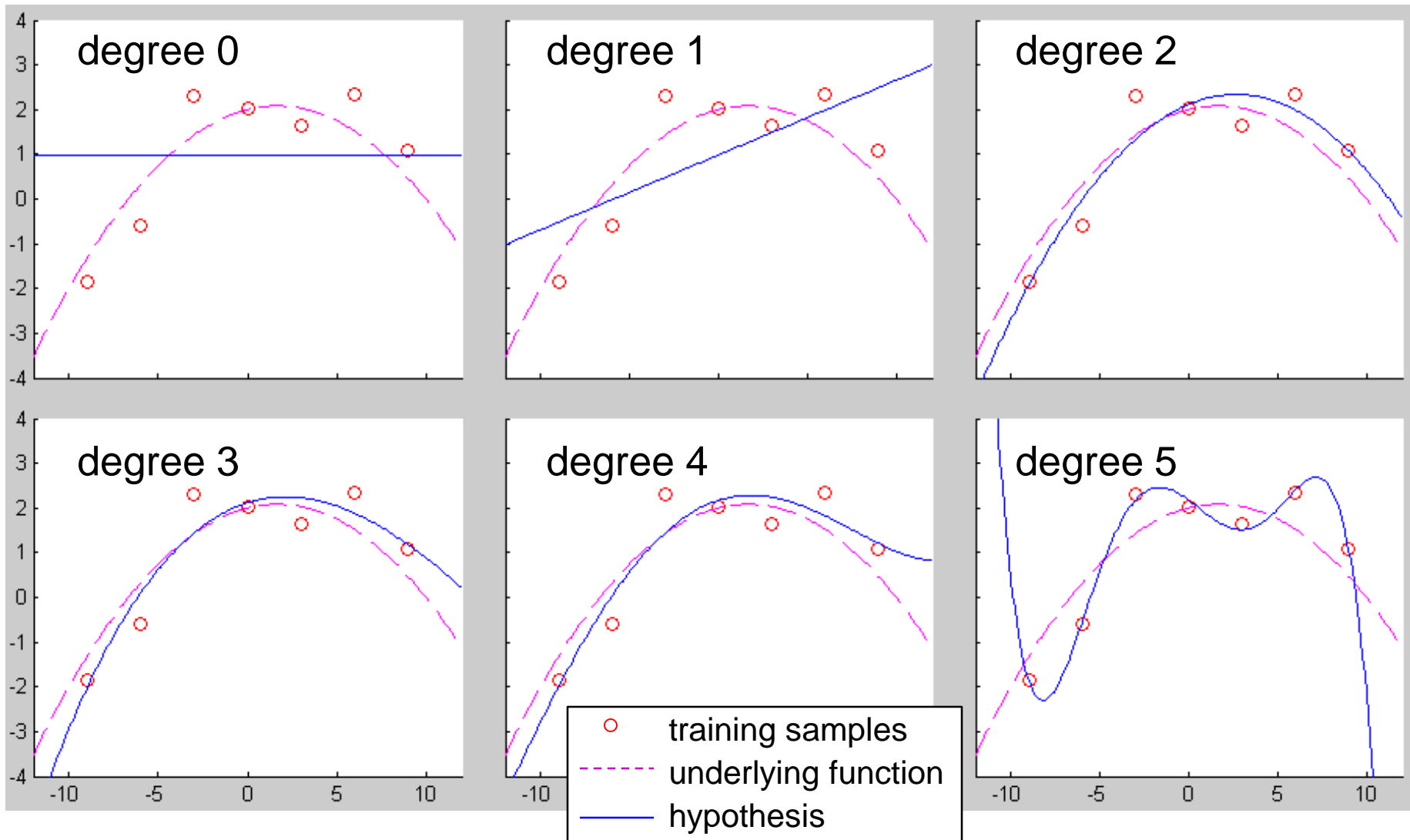
What to Do Now?

Several representative approaches:

- Increase the number of training samples.

- Try to determine the suitable level of model complexity.

- Regularization (the "suppression" of complexity during training)

- Consensus-based methods (ensembles)

Amount of Training Data

- The usual case: The more training data, the better.

- Rule of thumb (not always suitable):

  (# training samples) $\geq$ 10 x (# free parameters in model)

- Possible difficulties:

  - Data availability

  - Increased computational cost

- When there are few training data, some classifier types (e.g., support vector machines) generally work better than others.

Amount of Training Data

- Example (polynomial regression):
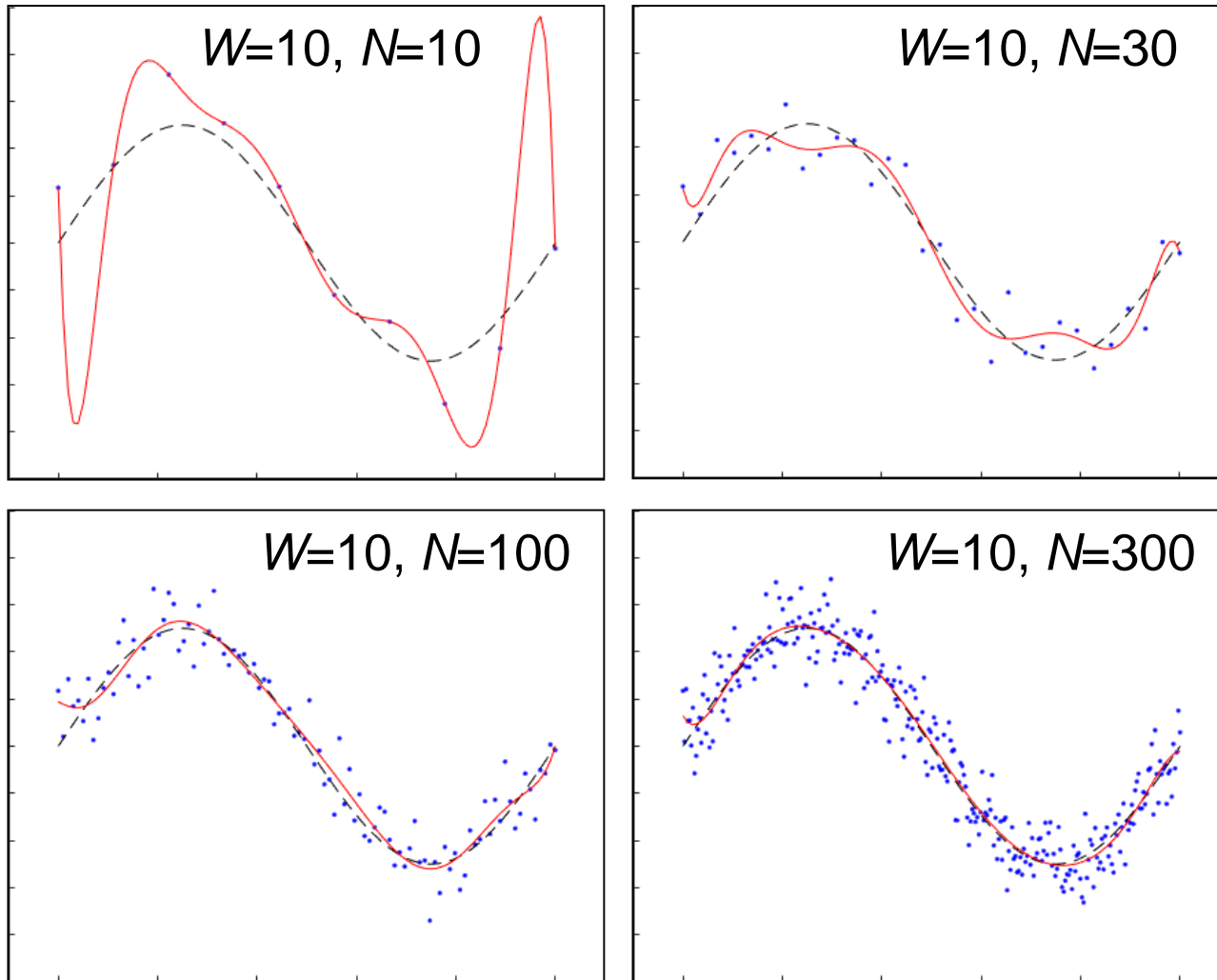
How Much Model Complexity?

- **Ockham's Razor**: Among different ways to explain the data, choose the simplest one.

- Example methods: **cross-validation**, post-processing model pruning.

- Many of these methods utilize **validation data** (training samples not used for model estimation) to check the **generalization** ability of the built model.

Validation
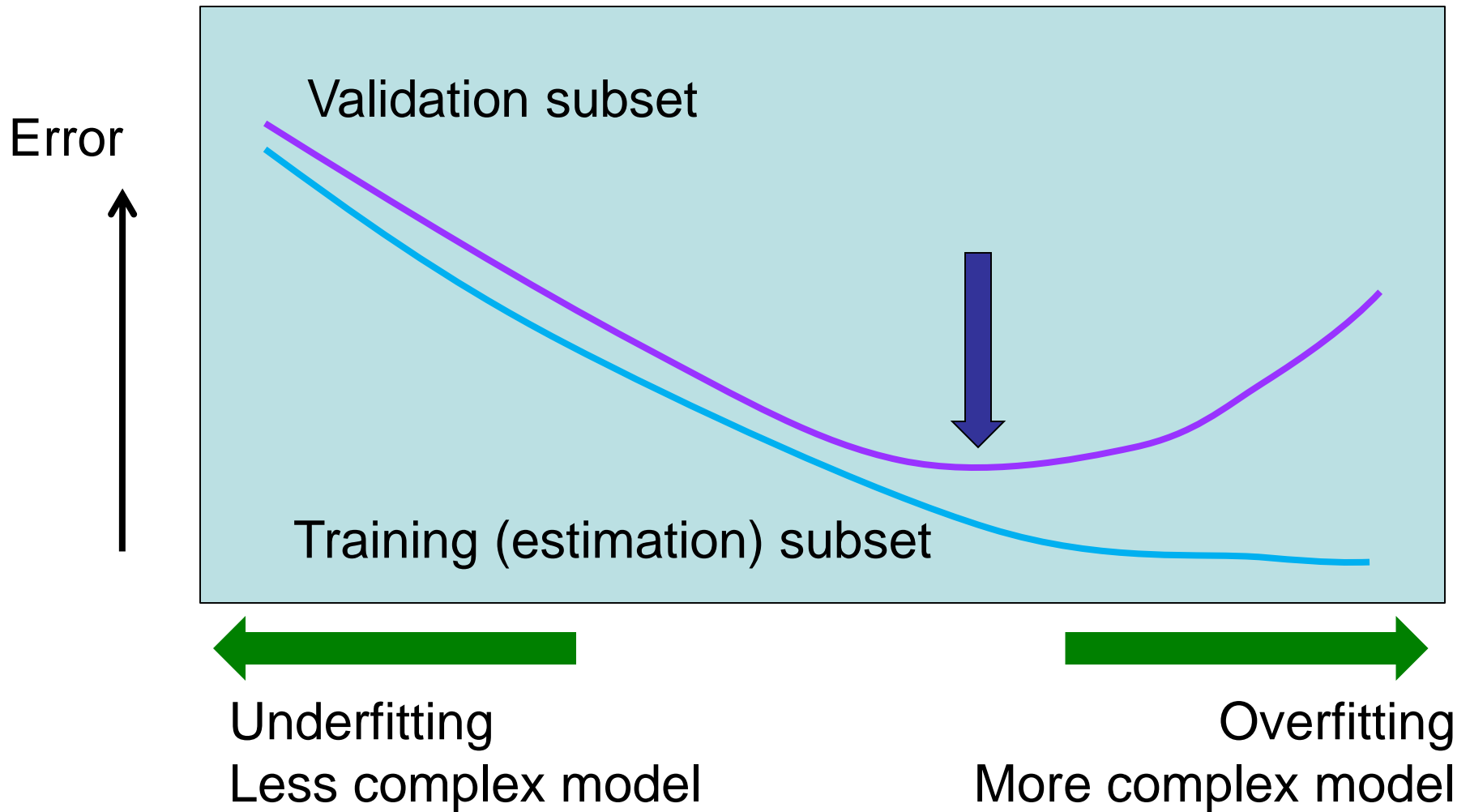
Cross-Validation

- The N labeled samples are divided into K subsets of approximately equal sizes ($K>1$). They should have similar distributions.

- The training process (model parameter estimation) is run K times (K trials).

- In the Kth trial, the Kth subset is used for validation, and the other subsets are used for training.

- The overall performance is the combination of the performance on all the validation subsets.

Cross-Validation

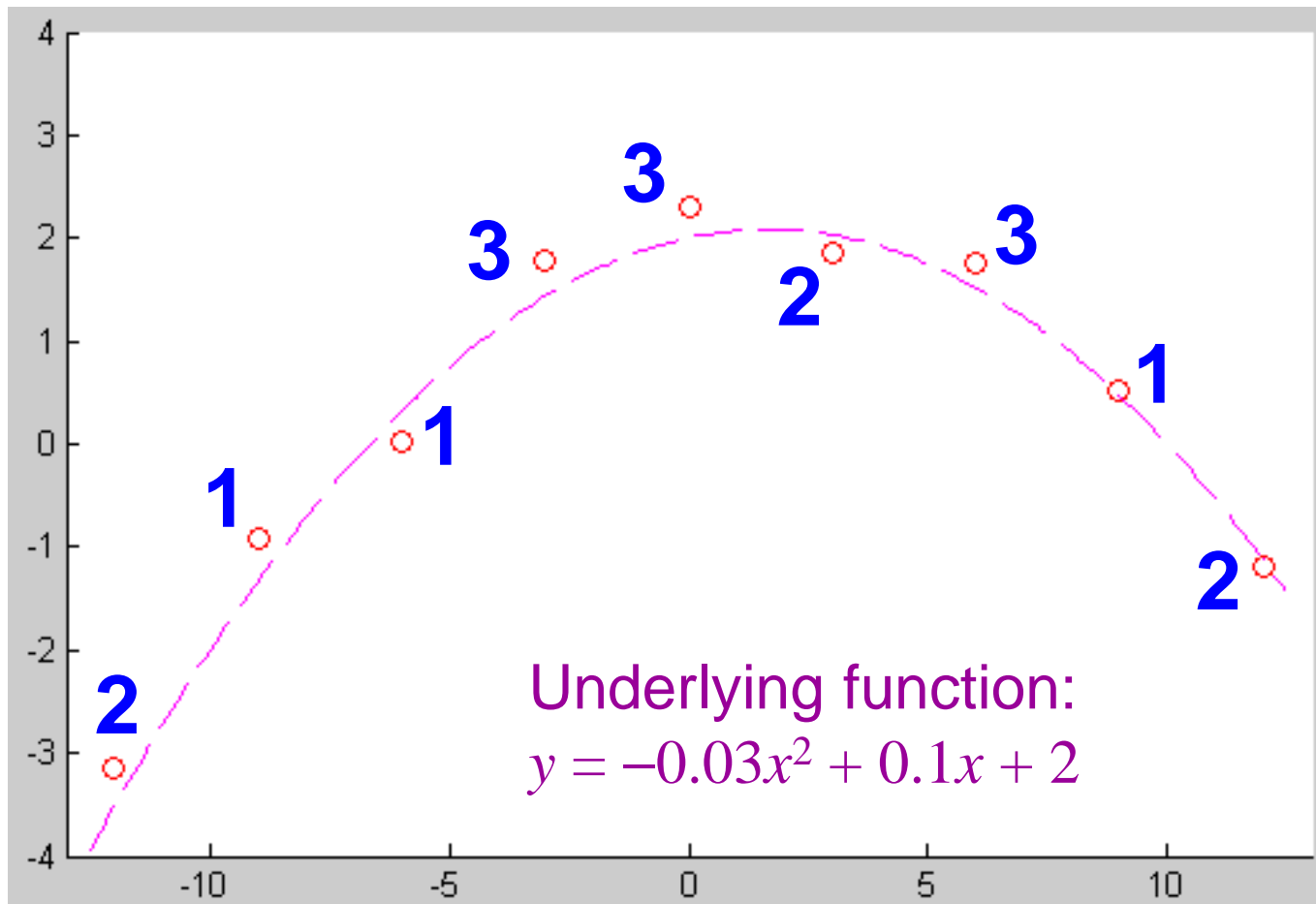4-fold cross-validation: parameter estimation   validation

Trial 1:   #1   #2   #3   #4

Trial 2:   #1   #2   #3   #4

Trial 3:   #1   #2   #3   #4

Trial 4:   #1   #2   #3   #4
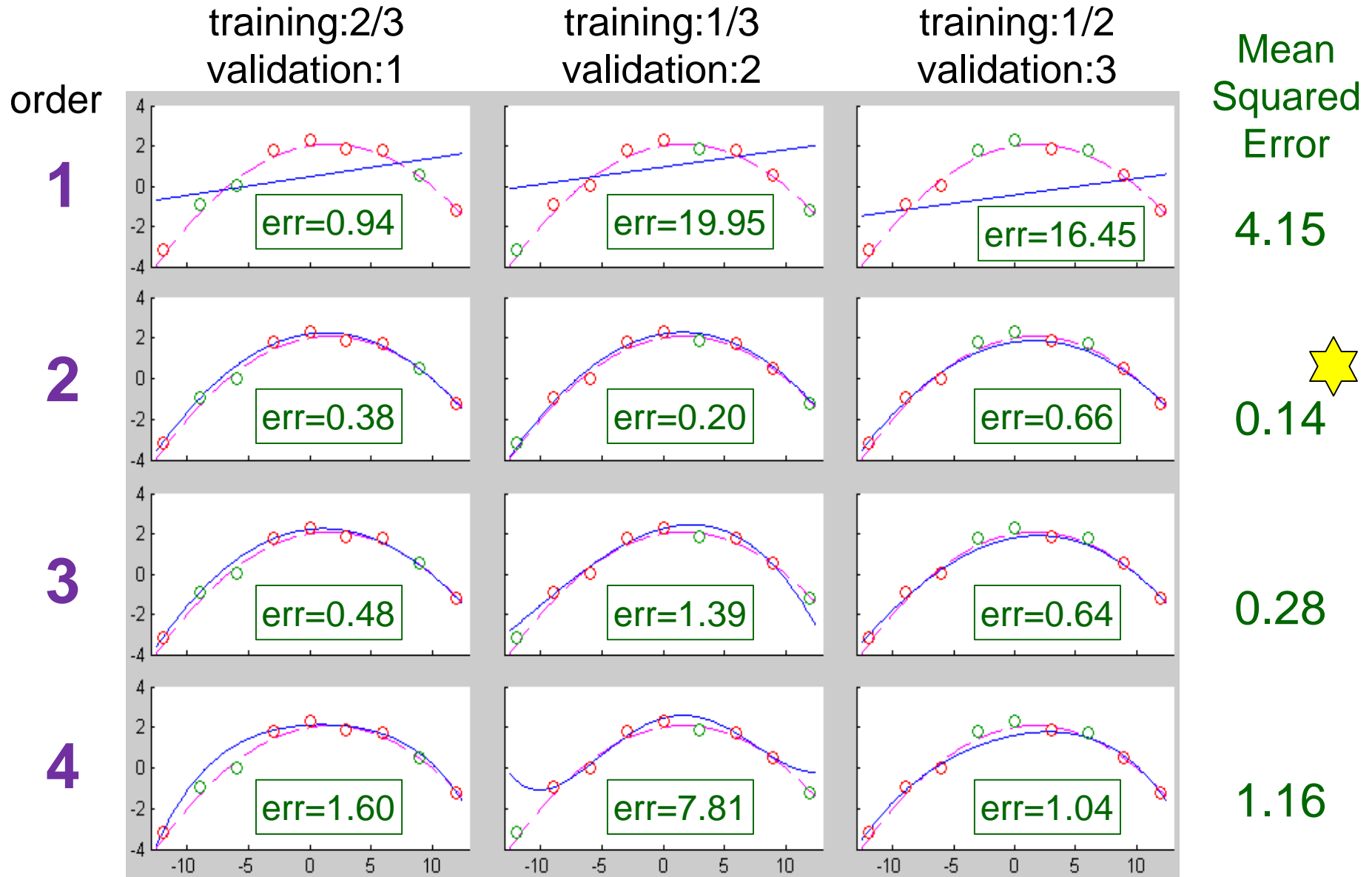
Extreme case: The leave-one-out method ($K=N$).

Model Selection by Cross-Validation

Example: polynomial regression:

9 training samples → 3 subsets

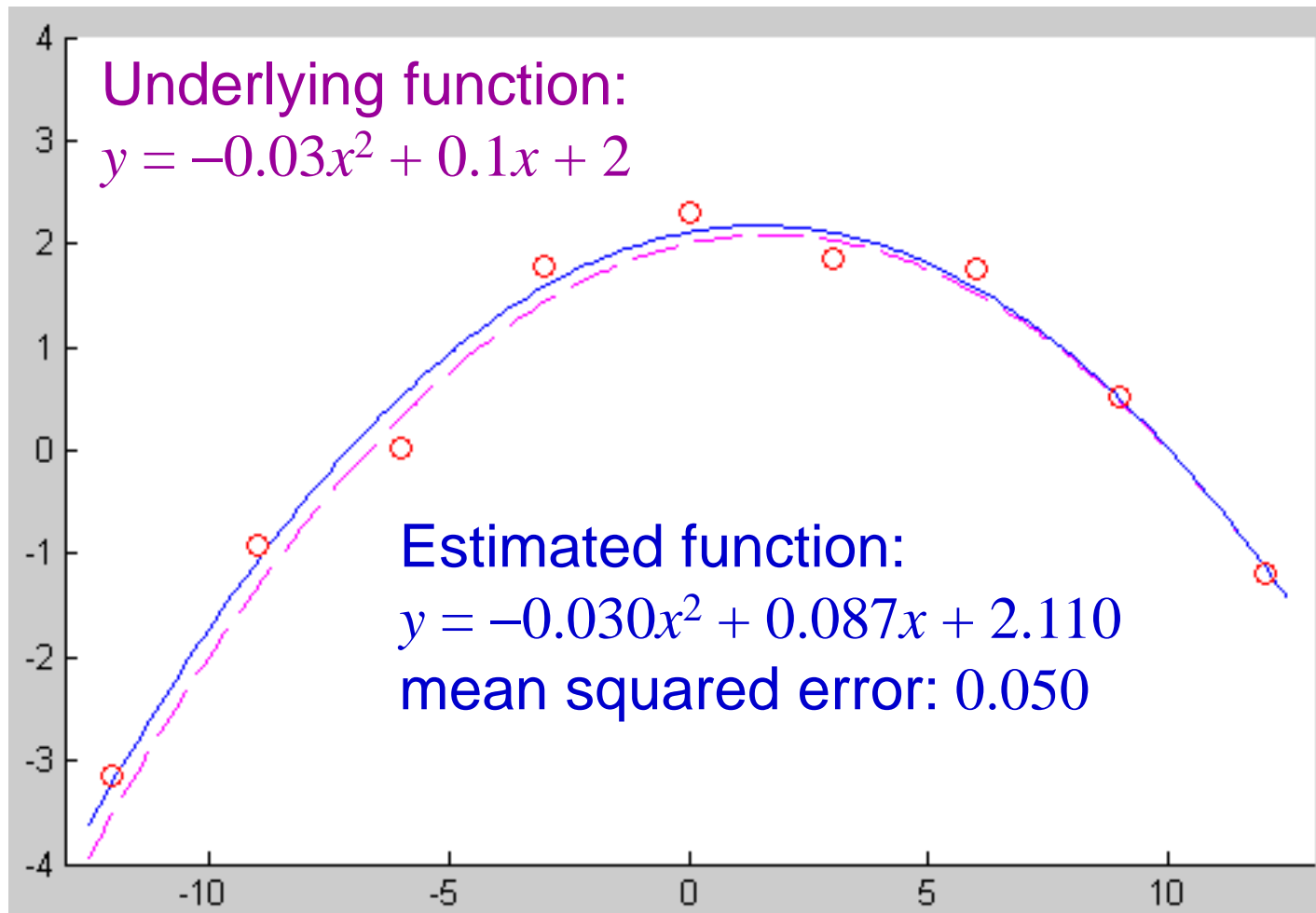Underlying function:
$y = -0.03x^2 + 0.1x + 2$

Model Selection by Cross-Validation

Model Selection by Cross-Validation

Finally, use the selected model and ALL the training samples for parameter optimization:

Underlying function:
$$y = -0.03x^2 + 0.1x + 2$$

Estimated function:
$$y = -0.030x^2 + 0.087x + 2.110$$
mean squared error: 0.050

Regularization

- Include the trade-off during the learning process:

  - **Regularization**: Modify the learning objective to minimize BOTH the bias and the model complexity. (Normally, the learning process only attempts to minimize bias.)

  - Example **cost function** (to be minimized) for polynomial regression:

To minimize:

$$E = \sum_i \left| y_i - \sum_{k=0}^{d} a_k x_i^k \right|^2 + \lambda \sum_{k=0}^{d} a_k^2$$

Fitting Error Regularization

Regularization

- Example (polynomial regression):
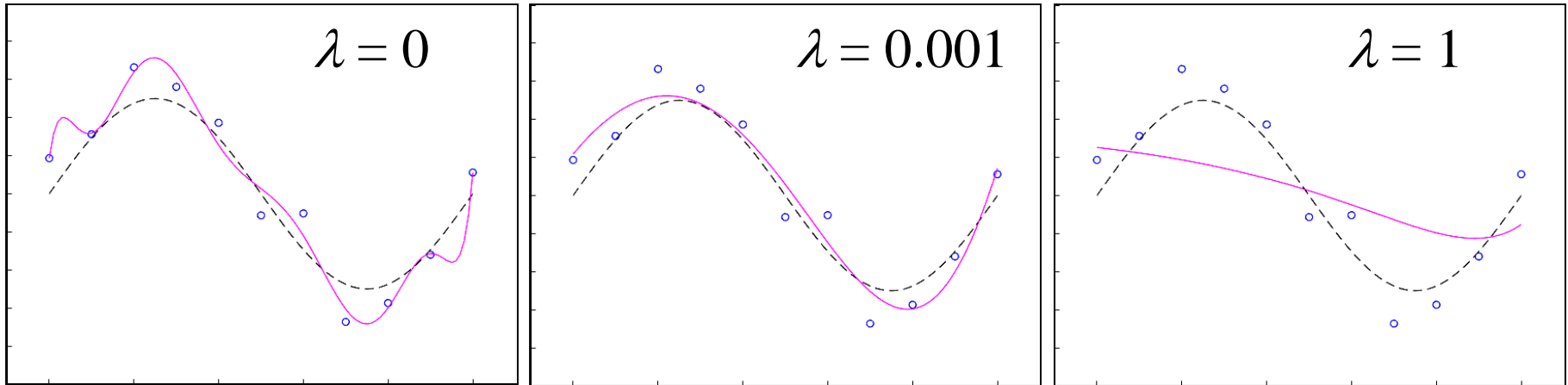
N=11, degree=9

Consensus Based Methods

- Build many somewhat different models (an **Ensemble**) for the same underlying mapping.

- The consensus should better represent the "signal" part, and the "noise" part is more likely to be averaged out.

- Result: Reduction of variance with minimal effect on bias.

- Common ways to create the ensemble of different models:

  - Random subset of attributes

  - Random subset of training samples

Back to Decision Trees

- Representative decision tree algorithms include:

- **ID3** (most similar to the method in the textbook)

  - Designed for categorical attributes

  - Post-processing pruning with validation data

- **C4.5** (a successor of ID3)

  - Categorical and numerical attributes

  - Use normalized information gain

- **CART** (**C**lassification **A**nd **R**egression **T**rees)

  - Categorical and numerical attributes

  - Branching factor is always two (binary tree)

Classification and Regression Trees

Node splitting criterion:

- Numerical attributes:
  - Select a threshold
  - Minimize total within-node variance
- Categorical attributes:
  - Need to find a way to partition the categories into two groups.
  - Multiple possible partitions if the items in a node are of more than two categories.
 - Minimize total Gini's impurity

Classification and Regression Trees

Output:

■ To avoid overfitting, the growth of the tree is usually limited. ➔ A leaf node is often "impure" in terms of the expected outputs of its items.

■ To generate an estimated output:

- For regression: Use the mean output of the items in the node.

- For classification: Use the majority vote of the items in the node.

Random Forests

- A representative example of ensemble learning.

- The final output is the average output of many randomized trees.

- **Bagging** = **b**ootstrap **agg**regat**ing** (*random sampling with replacement*)

- **Tree Bagging**:
  - Each tree is trained with a somewhat different set of training data (selected from a common pool of training data with bootstrap aggregation).

- **Feature/Attribute Bagging**:
  - Consider only a (small) subset of features/attributes at each node splitting.
  - Typically only $p^{1/2}$ of p features used (for classification).

Random Forests: An Example
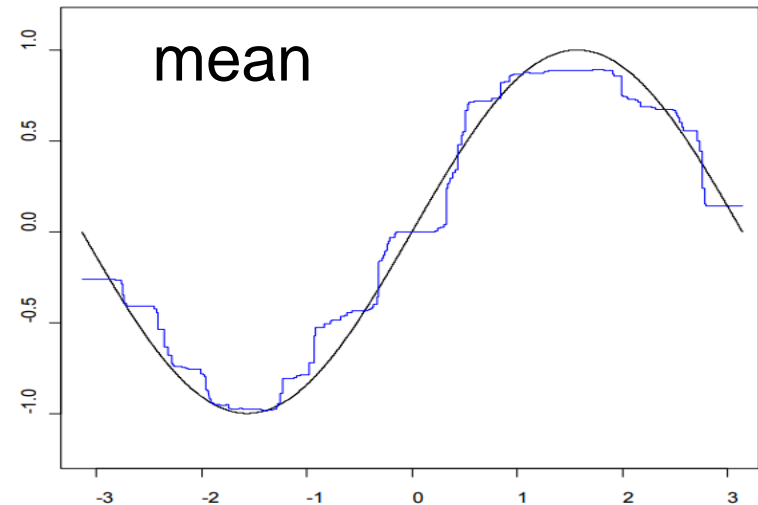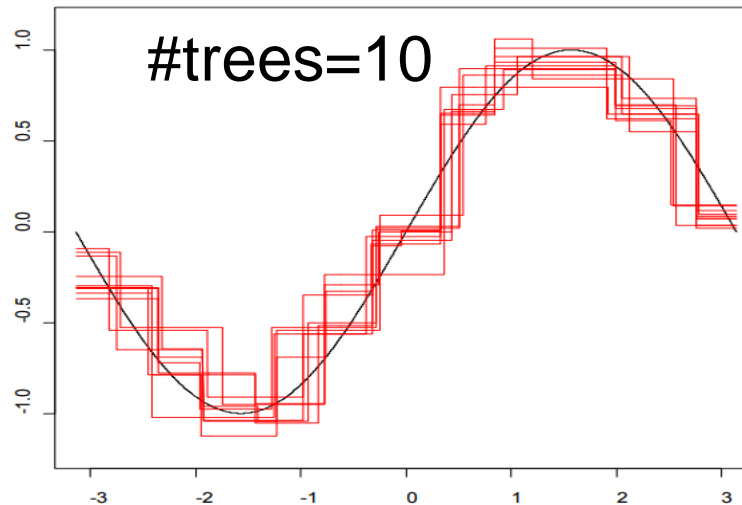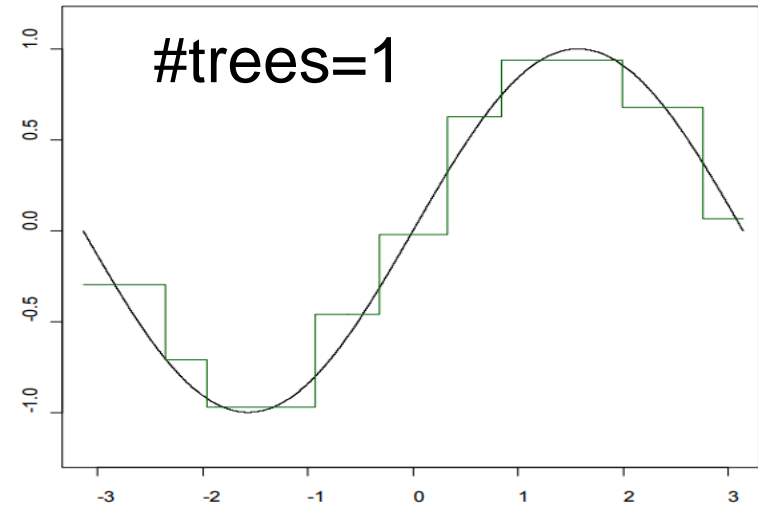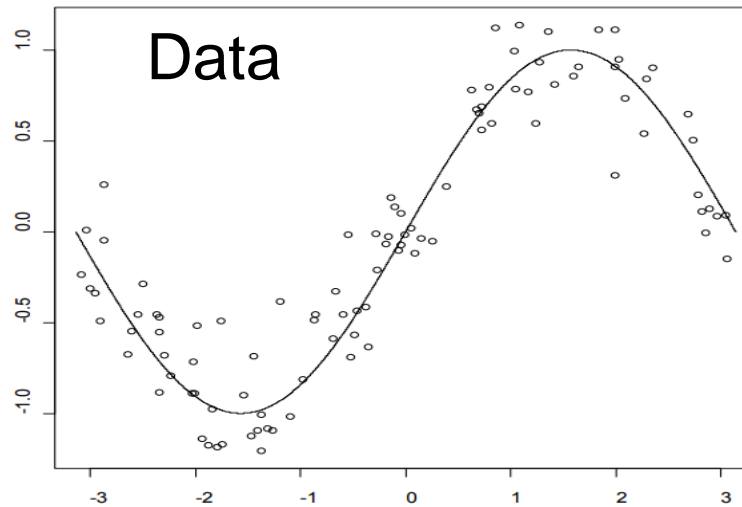
Random Forests: Generalization

- For each training sample x, there are some trees that do NOT consider it during tree building. It is called an out-of-bag sample for those trees.

- Use the trees for which x is out-of-bag to predict the output for x. (Use averaging for regression, and use majority vote for classification.)

- Repeat the above step for all training samples. Now we have the overall out-of-bag error (mean squared error for regression, classification error rate for classification) as a measure of generalization ability.

Random Forests: Generalization

Random forests do not overfit with increased number of trees.