# CSC242: Intro to AI

Lecture 17

## Learning from Examples

# Learning from Examples

# Learning

# Why Learn?

- Can’t anticipate all possible situations that the agent might find themselves in
- Cannot anticipate all changes that might occur over time
- Don’t know how to program it other than by learning!

# Dimensions of Learning

Unsupervised $\xleftrightarrow{\text{Teacher}}$ Supervised

Active $\xleftrightarrow{\text{Learner}}$ Passive

Delayed $\xleftrightarrow{\text{Feedback}}$ Immediate

# Dimensions of Learning

Teacher

Unsupervised ⟷ Supervised

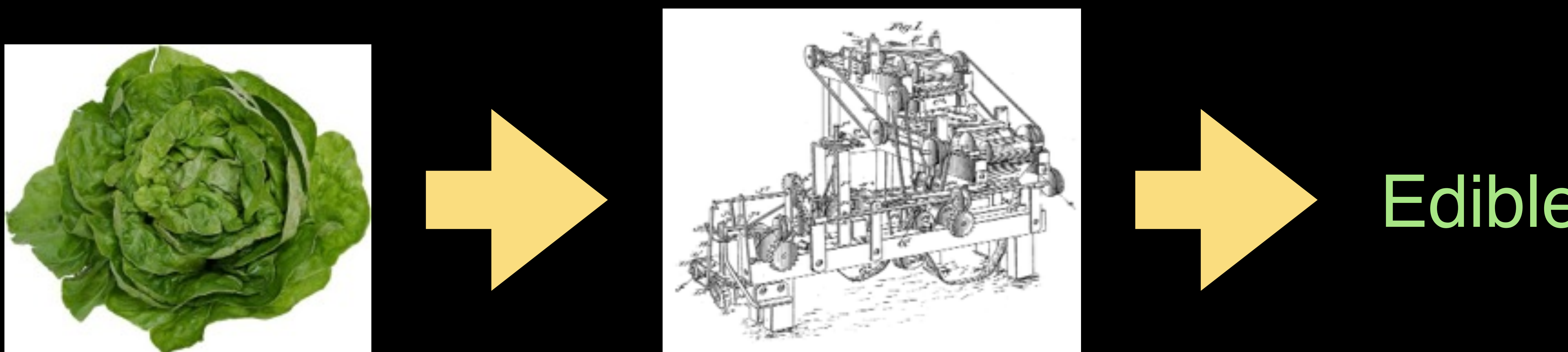
Learner

Active ⟷ Passive

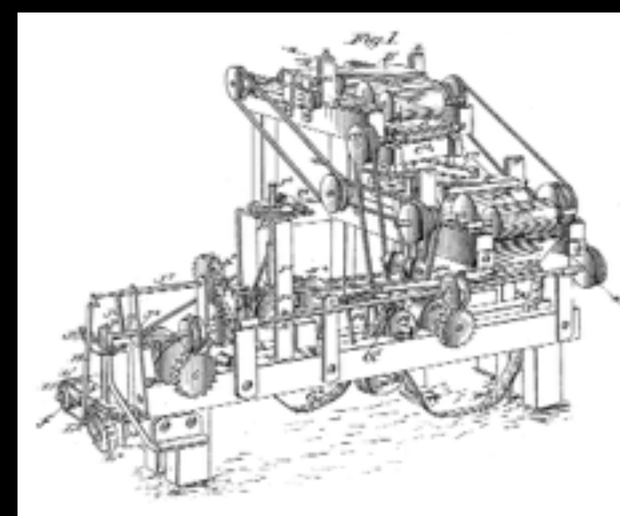
Feedback

Delayed ⟷ Immediate

# A Classifier

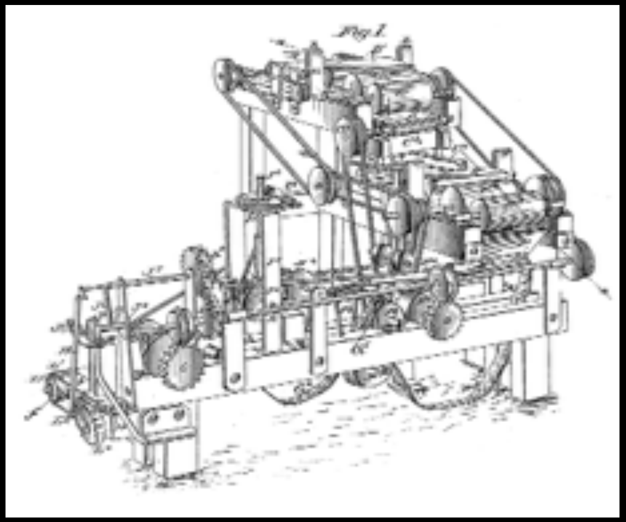


# A Classifier



Edible

# A Classifier



Poison

# Learning a Classifier

Training Data

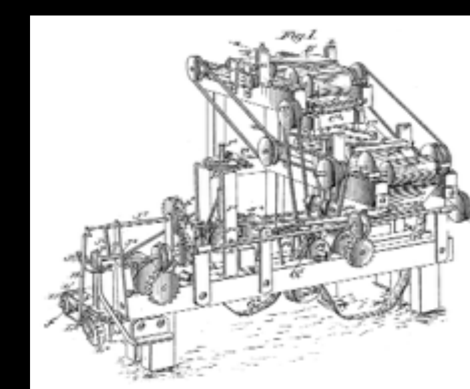
Edible

Edible

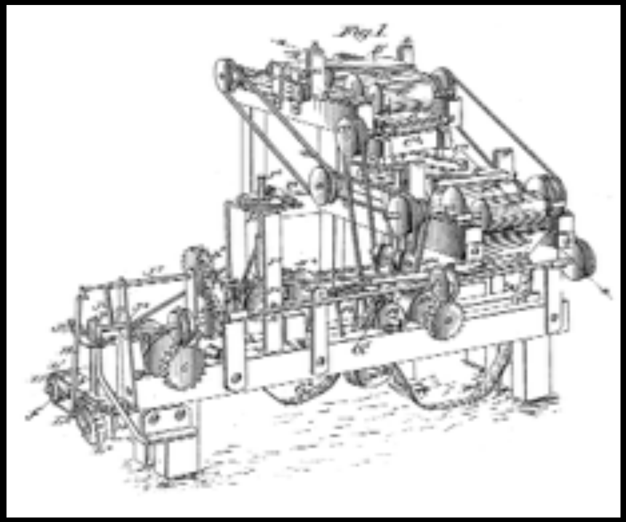
Poison

Learner

Classifier



# Generalization



Poison

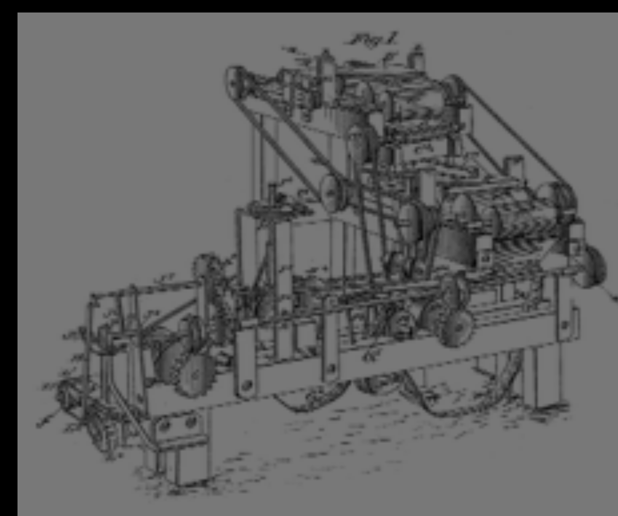
- Ability to classify items that were never seen before
- Going beyond simple memorization

# Features

- img9201.jpg
- (color=green,
  leafs_per_stem=3,
  leaf_edge=jagged)

- The observable properties of the things to classified
- Also called "attributes"

# Labels



Poison

- Classification: Symbols
- Regression: Numbers

# Hypothesis Space

Training Data

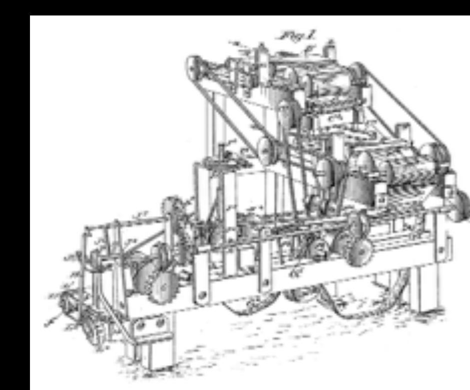
Edible

Edible

Poison

Learner

Classifier



- Space of possible outputs of the learning system
- Polynomial functions, decision trees, neural networks, …

# Learning Functions from Examples

# Function Learning

- There is some function $y = f(x)$
- We don't know $f$
- We want to learn a function $h$ that approximates the true function $f$

Hypothesis

# Function Learning

- There is some function $y = f(x)$
- We don't know $f$
- We want to learn a function $h$ that approximates the true function $f$
- Learning is a search through the space of possible hypotheses for one that will perform well

Hypothesis

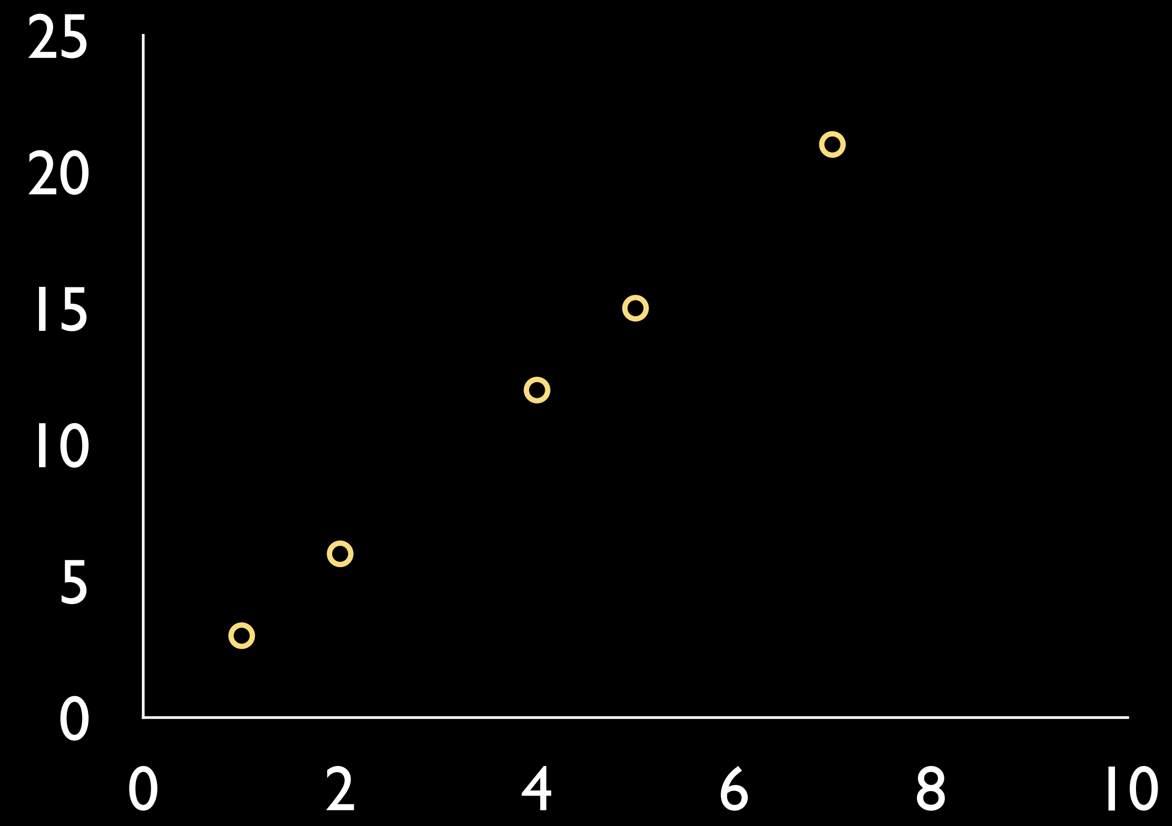
# Supervised Learning

- Given a training set of $N$ example input-output pairs:
  $$(x_1,\ y_1),\ (x_2,\ y_2),\ \ldots,\ (x_\mathrm{N},\ y_\mathrm{N})$$
  where each $y_j = f(x_j)$
- Discover function $h$ that approximates $f$
- Search through the space of possible hypotheses for one that will perform well

# Training Data

$f(x)=y$

| $x$ | $y$ |
|---|---|
| 1 | 3 |
| 2 | 6 |
| 4 | 12 |
| 5 | 15 |
| 7 | 21 |




$h(x)=?$

# Evaluating Accuracy

- Training set: $(x_1,\ y_1),\ (x_2,\ y_2),\ \ldots,\ (x_\mathrm{N},\ y_\mathrm{N})$
- Test set: Additional $(x_j,\ y_j)$ pairs distinct from training set
- Test accuracy of $h$ by comparing $h(x_j)$ to $y_j$ for $(x_j,\ y_j)$ from test set
- Generalization: Ability to handle examples in test set that were not in training test

## Training Data

$f(x) = y$

| $x$ | $y$ |
|---|---|
| 1 | 3 |
| 2 | 6 |
| 4 | 12 |
| 5 | 15 |
| 7 | 21 |

$h(x) = ?$

## Testing Data

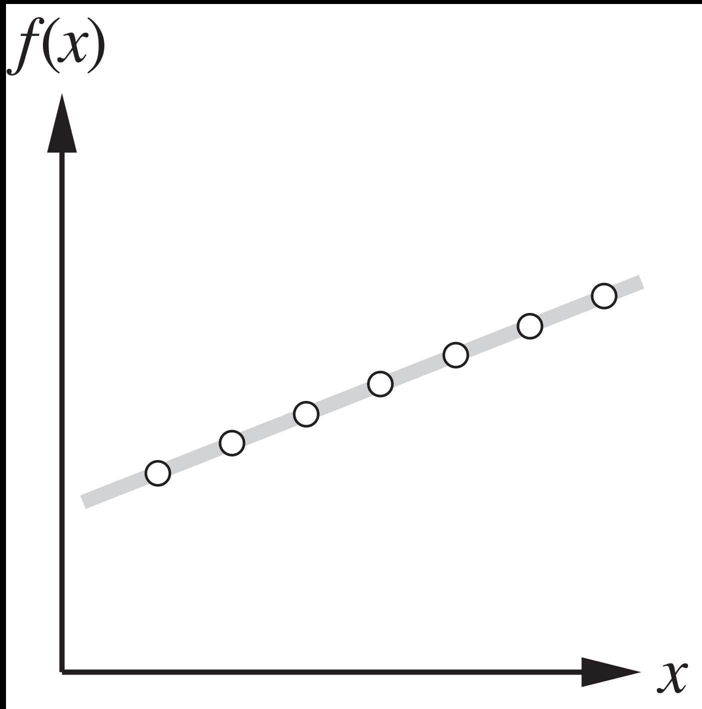
| $x$ | $y$ |
|---|---|
| 3 | 9 |
| 4 | 12 |
| 6 | 18 |

$f(x) = y$

$h(x) = y?$

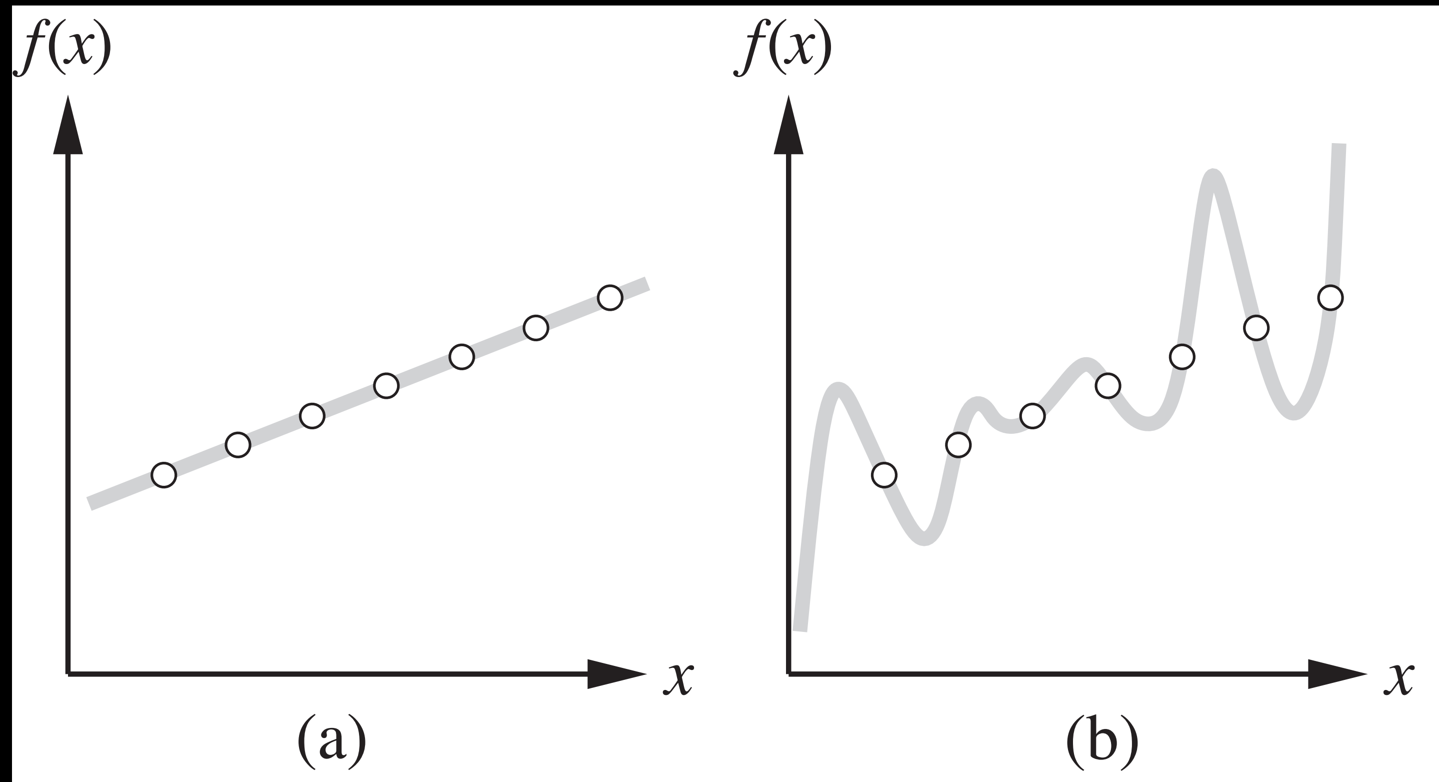
# Hypothesis Space

- The class of functions that are acceptable as solutions, e.g.
  - Linear functions $y = mx + b$
  - Polynomials (of some degree)
  - Decision trees
  - Neural networks
  - Turing machines



(a)

$$y = -0.4x+3$$

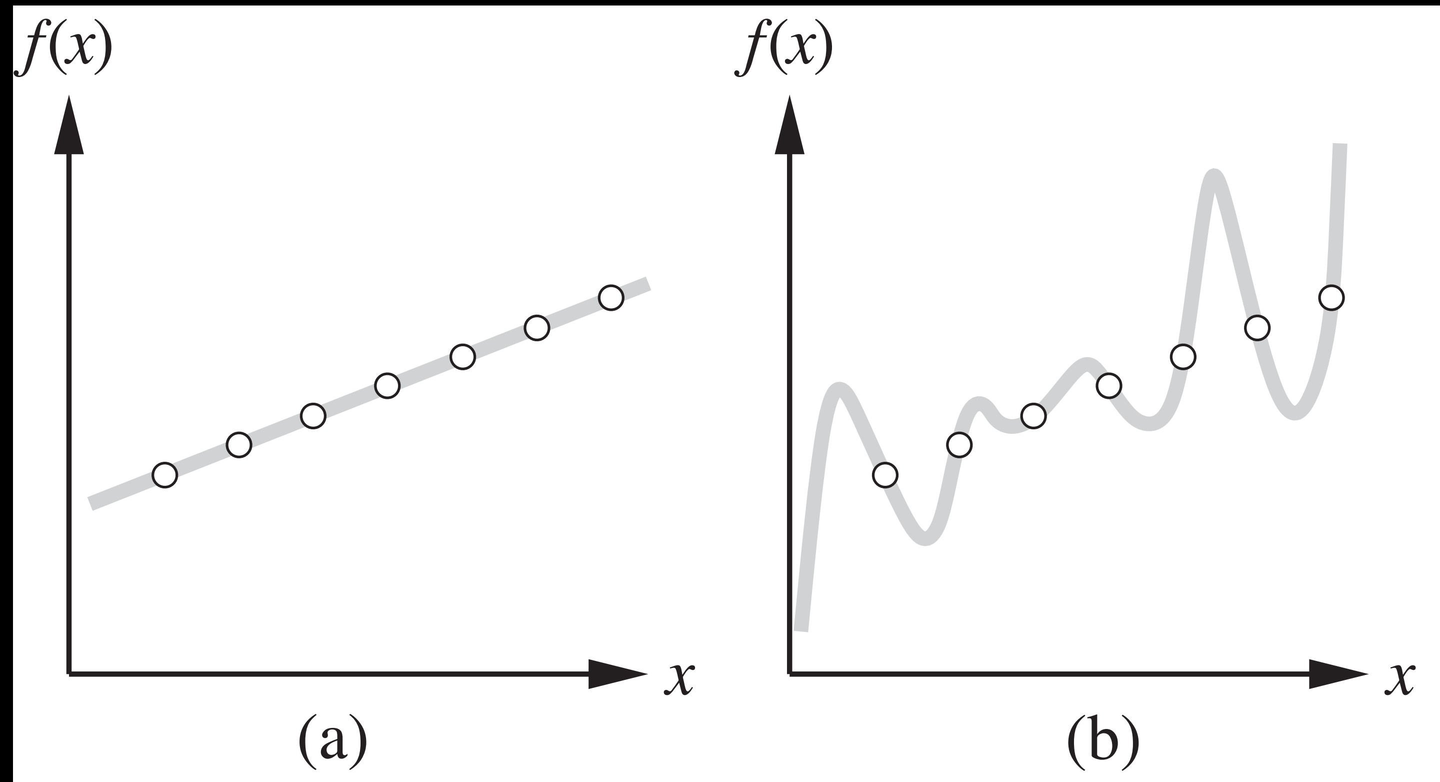



$$y = \text{-}0.4x + 3$$

$$y = c_7x^7 + c_6x^6 + \ldots + c_1x + c_0$$
$$= \sum_{i=0}^{7} c_i x^i$$

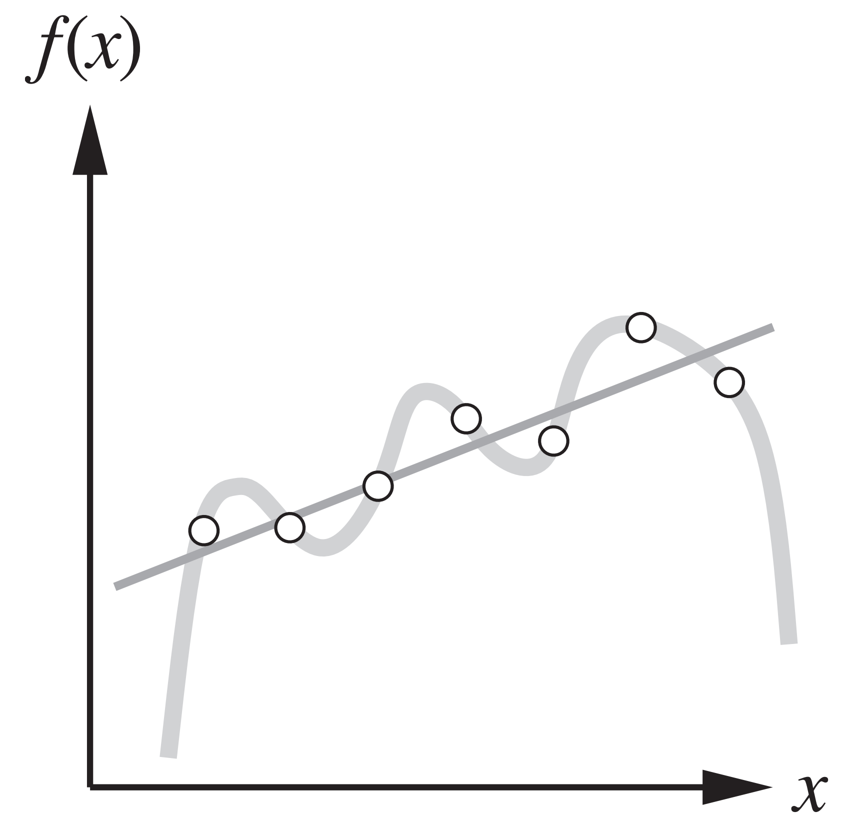
# Occam's Razor


William of Occam (or Ockham)
14th c.




$$y = \text{-}0.4x + 3$$

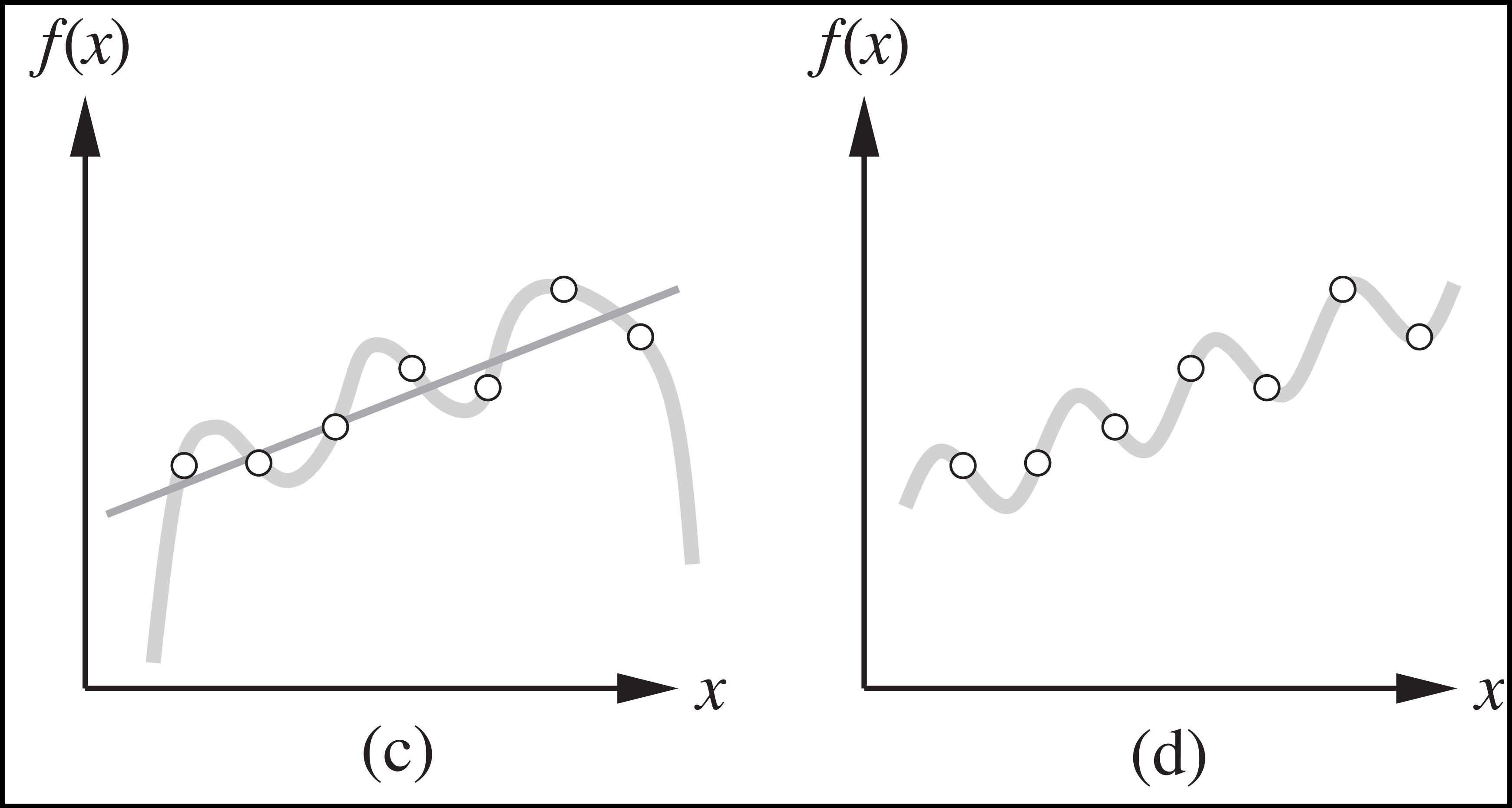
$$y = c_7 x^7 + c_6 x^6 + \ldots + c_1 x + c_0$$
$$= \sum_{i=0}^{7} c_i x^i$$



(c)

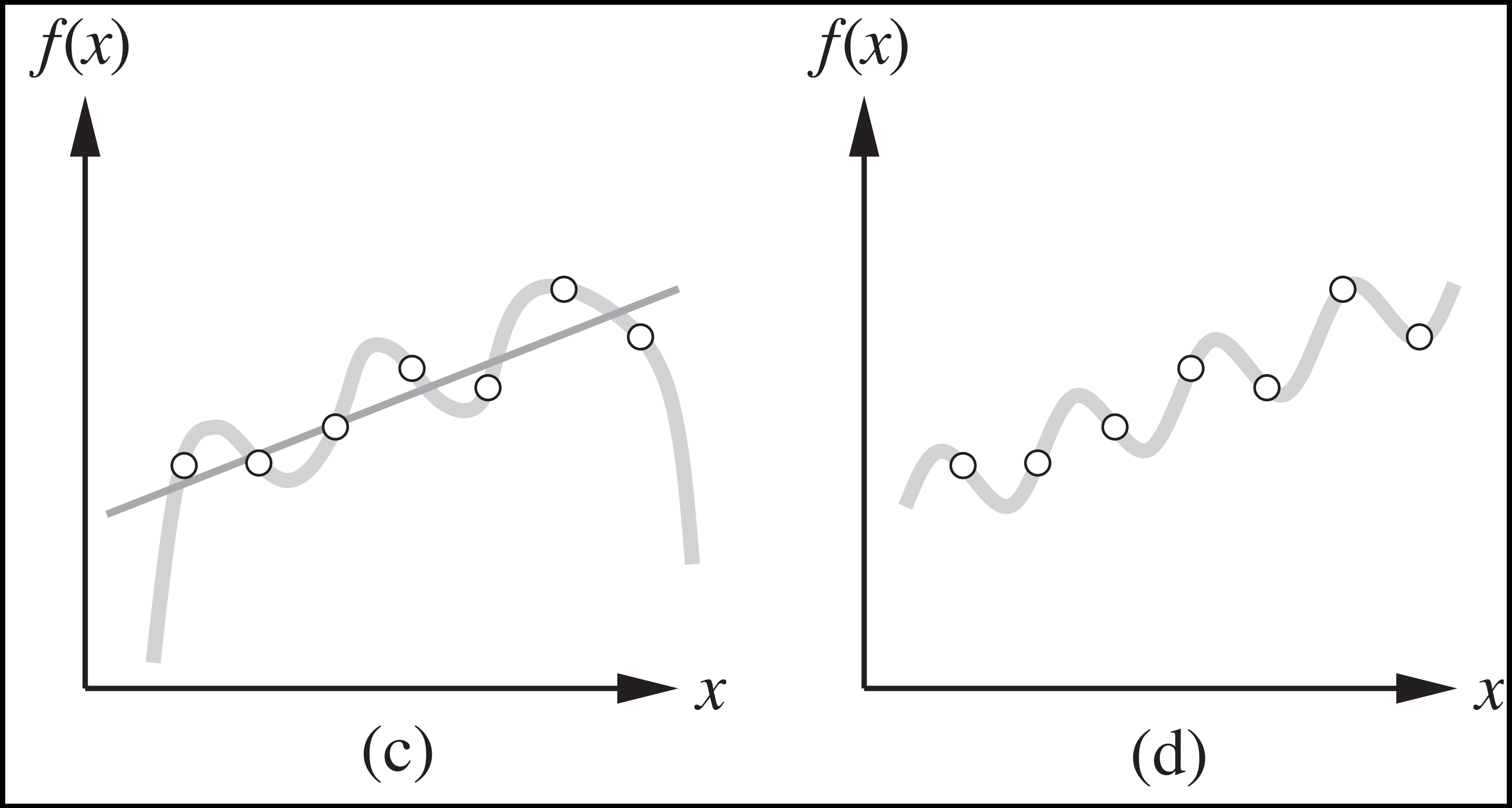
$$y = c_6 x^6 + c_5 x^5 \ldots + c_1 x + c_0$$

$$y = mx + b$$




$$y = c_6 x^6 + c_5 x^5 \ldots + c_1 x + c_0 \quad ax + b + c\sin(x)$$

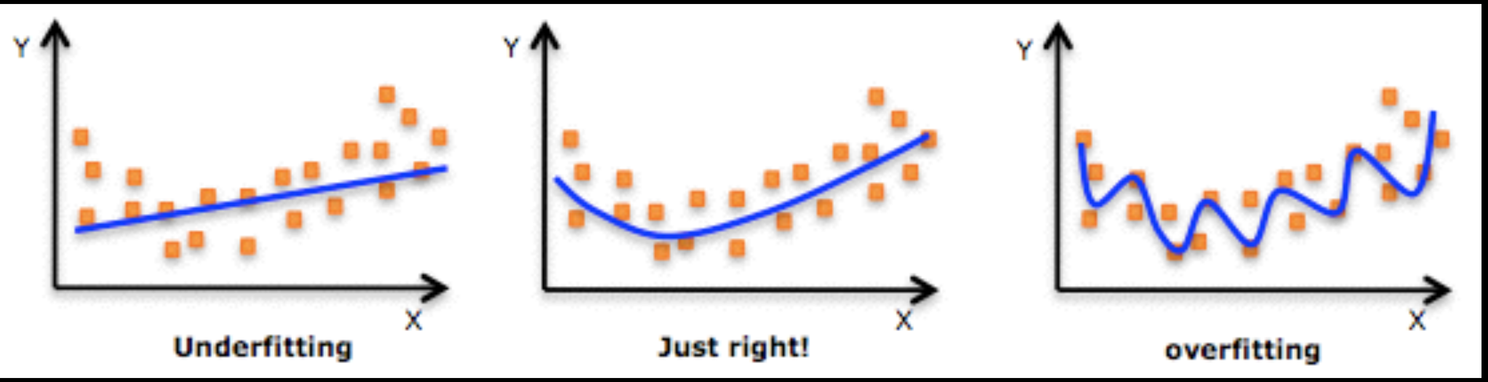
$$y = mx + b$$




$$y = c_6 x^6 + c_5 x^5 \ldots + c_1 x + c_0 \qquad ax + b + c\sin(x)$$

$$y = mx + b$$

# Error and Overfitting




- It is often preferable to allow some error in the fit of the hypothesis to the training data in order to improve generalization
- Allowing too little error - resulting in a complex hypothesis with poor generalization - is overfitting
- Using too simple a hypothesis that has very high error - resulting again in poor generalization - is underfitting

# Overfitting

- When a learned model adjusts to the noise in the input rather than the signal
- Becomes more likely as the hypothesis space and number of input attributes grows
- Becomes less likely as the number of training examples increases

# Learning Decision Trees

# Classification

- Output $y = f(x)$ is one of a finite set of values (classes, categories, ...)
  - Boolean classification: yes/no or true/false
- Input is vector $\mathbf{x}$ of values for attributes
  - Factored representation

# Example

- Going out to dinner with Stuart Russell
- Restaurants often busy in SF; sometimes have to wait for a table
- Decision: Do we wait or do something else?

# Attributes (Features)

$Alternate$ : is there a suitable alternative nearby

$Bar$: does it have a comfy bar

$FriSat$: is it a Friday or Saturday

$Hungry$: are we hungry

$Patrons$: $None,\ Some,\ Full$

$Price$: $\$,\$\$,\$\$\$$

$Raining$: is it raining outside

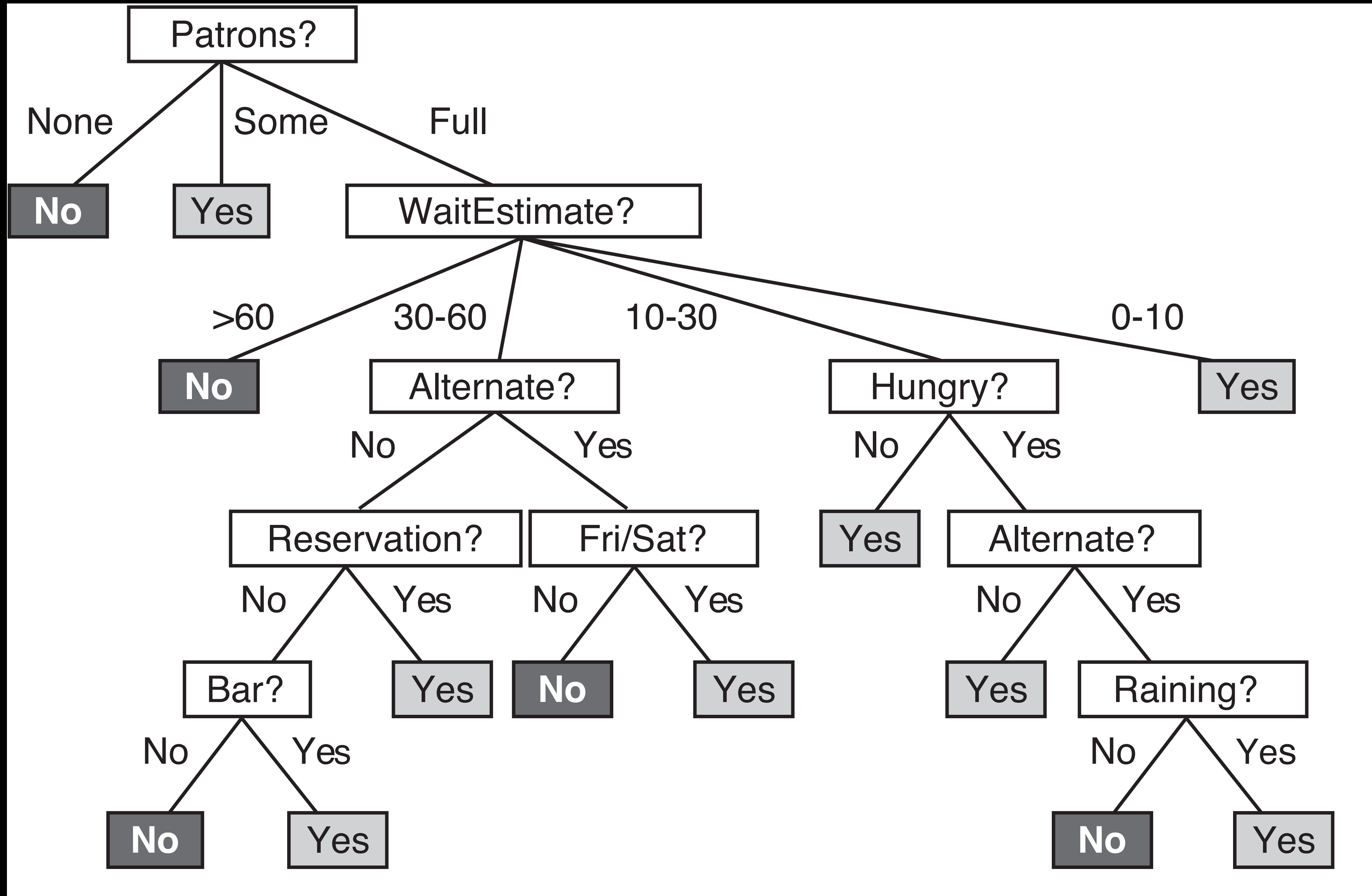
$Reservation$: do we have a reservation

$Type$: $French,\ Italian,\ Thai,\ burger,\ \ldots$

$WaitEstimate$: $0\text{-}10,\ 10\text{-}30,\ 30\text{-}60,\ >60$

# Decision Making

- If the host/hostess says you'll have to wait:
  - Then if there's no one in the restaurant you don't want to be there either;
  - But if there are a few people but it's not full, then you should wait
  - Otherwise you need to consider how long he/she told you the wait would be
    - ...


Patrons?
None
Some
Full
No
Yes
WaitEstimate?
>60
30-60
10-30
0-10
No
Alternate?
Hungry?
Yes
No
Yes
No
Yes
Reservation?
Fri/Sat?
Yes
Alternate?
No
Yes
No
Yes
No
Yes
Bar?
Yes
No
Yes
Yes
Raining?
No
Yes
No
Yes
No
Yes
No
Yes

# Decision Tree

- Each node in the tree represents a test on a single attribute
- Children of the node are labelled with the possible values of the feature
- Each path represents a series of tests, and the leaf node gives the value of the function when the input passes those tests

# Inducing Decision Trees From Examples

- Examples: $(\mathrm{x}, y)$ where $\mathrm{x}$ is a vector of values for the input attributes and $y$ is a single Boolean value (yes/no, true/false)

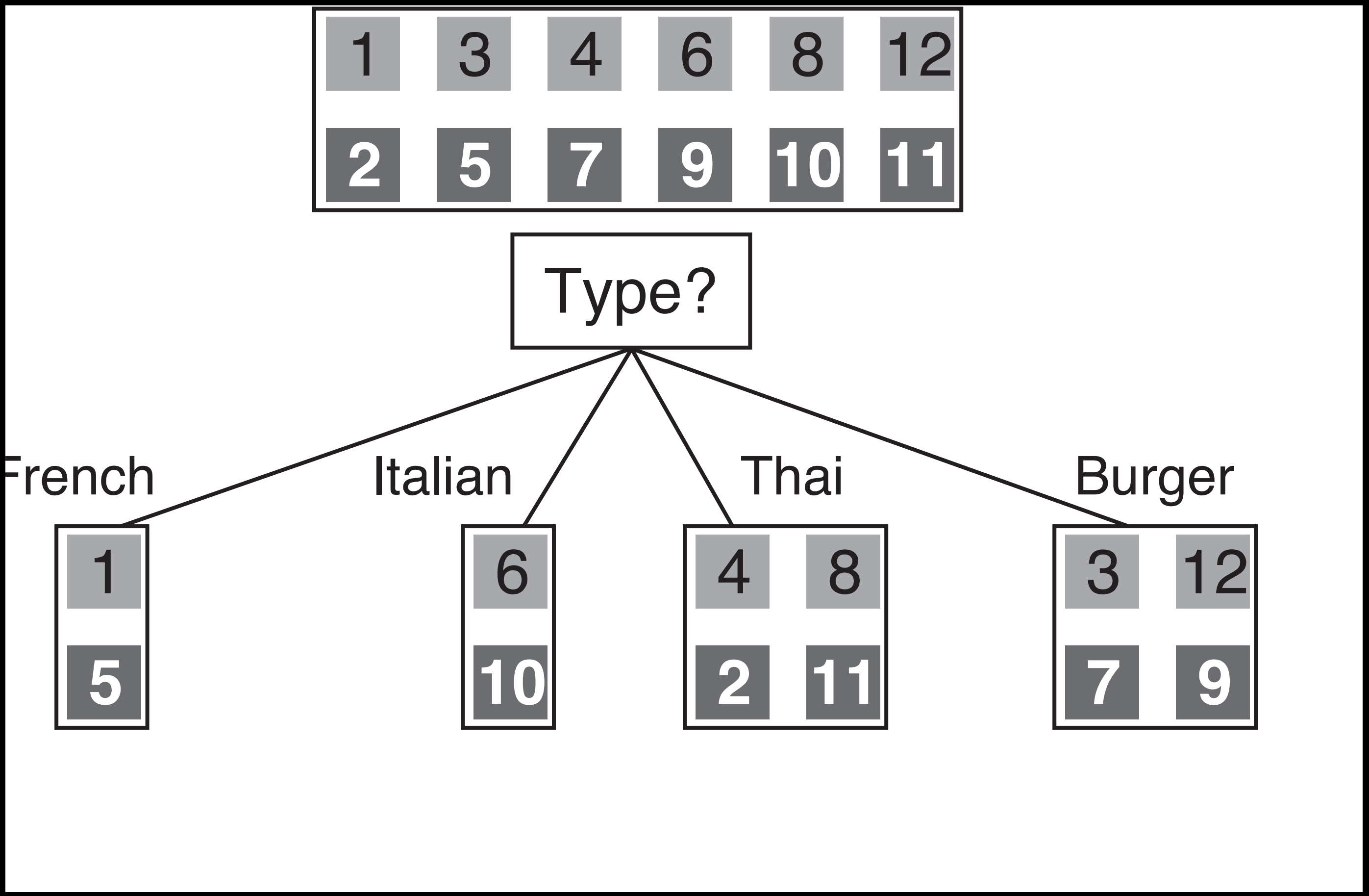
| | Input Attributes | | | | | | | | | | Will Wait |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | $y$ |

# Inducing Decision Trees From Examples

- Examples: $(\mathrm{x}, y)$
- Want a shallow tree (short paths, fewer tests)
- Greedy algorithm (AIMA Fig 18.5)
  - Always test the most important attribute first
  - Makes the most difference to classification of an example

| | Input Attributes | | | | | | | | | | *Will Wait* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | $y$ |

| | Input Attributes | | | | | | | | | | Will Wait |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | $y$ |

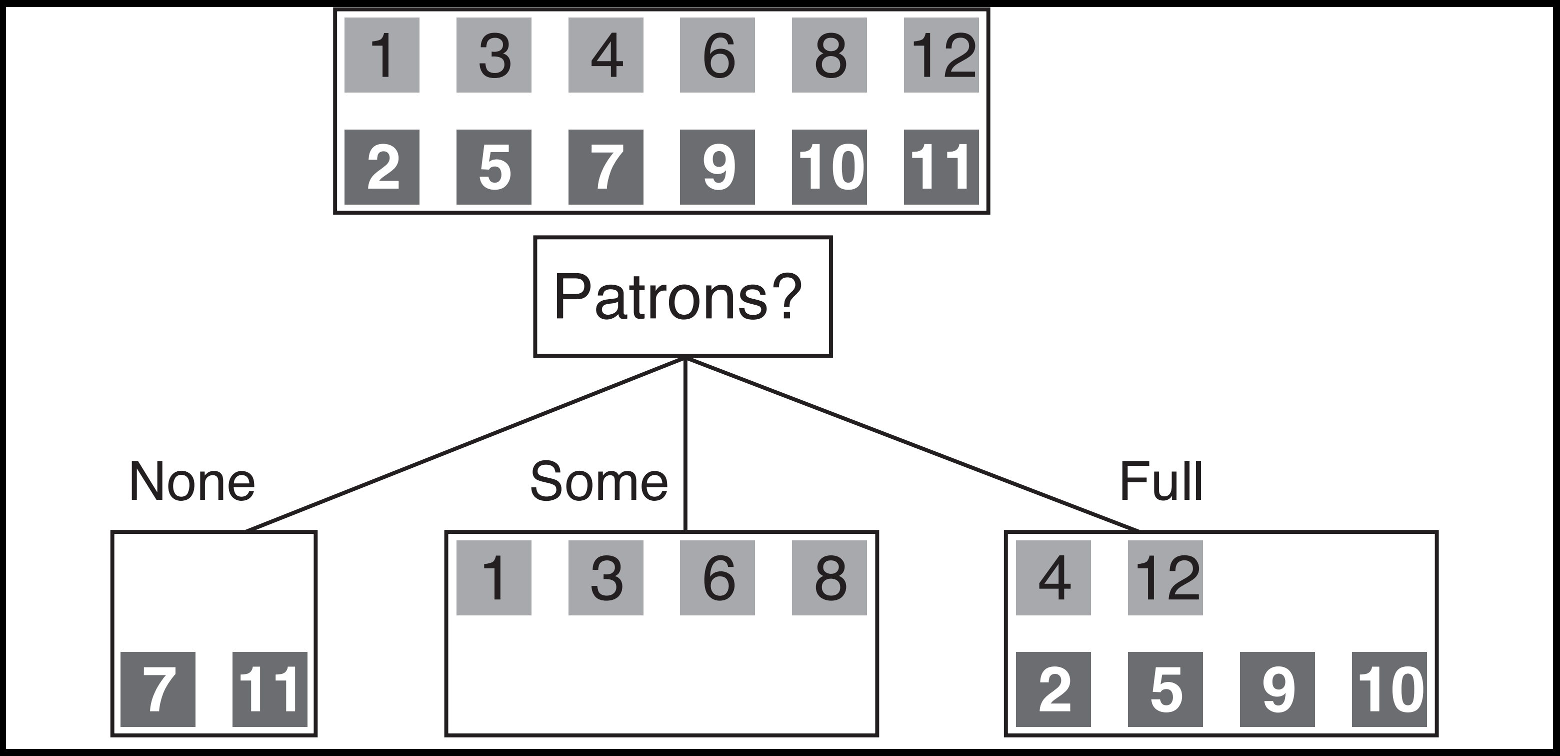


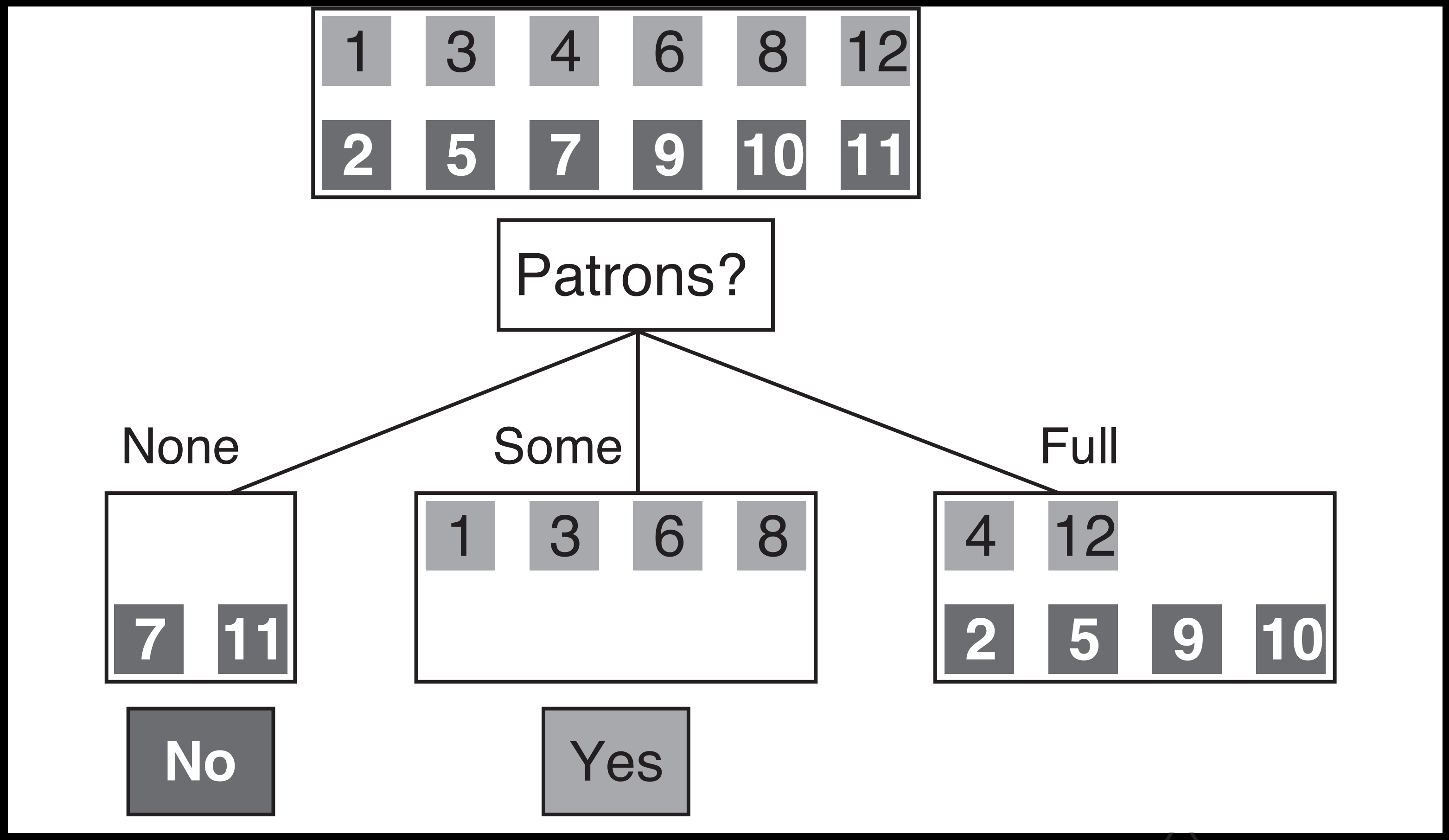

Poor split: children very mixed!

| | Input Attributes | | | | | | | | | | Will Wait |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | $y$ |
| $\mathbf{x}$ | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | $y$ |
| $\mathbf{x}$ | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | $y$ |
| $\mathbf{x}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | $y$ |

| | Input Attributes | | | | | | | | | | Will Wait |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| **x** | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | *y* |
| **x** | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | *y* |
| **x** | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | *y* |
| **x** | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | *y* |
| **x** | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | *y* |
| **x** | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | *y* |
| **x** | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | *y* |
| **x** | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | *y* |
| **x** | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | *y* |
| **x** | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | *y* |
| **x** | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | *y* |
| **x** | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | *y* |

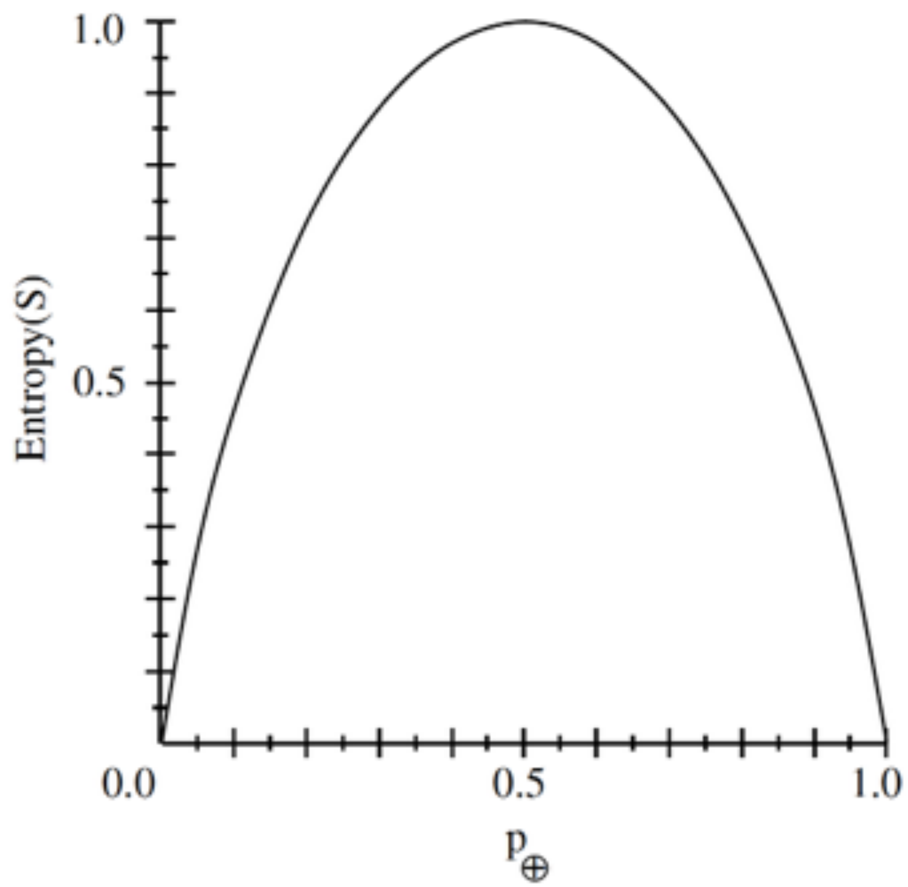
| | Input Attributes | | | | | | | | | | *Will Wait* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | |
| **x** | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0-10* | $y$ |
| **x** | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30-60* | $y$ |
| **x** | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0-10* | $y$ |
| **x** | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10-30* | $y$ |
| **x** | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *>60* | $y$ |
| **x** | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0-10* | $y$ |
| **x** | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0-10* | $y$ |
| **x** | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0-10* | $y$ |
| **x** | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *>60* | $y$ |
| **x** | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10-30* | $y$ |
| **x** | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0-10* | $y$ |
| **x** | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30-60* | $y$ |


1 3 4 6 8 12
2 5 7 9 10 11
Patrons?
None
Some
Full
7 11
1 3 6 8
4 12
2 5 9 10




Good split: children very unbalanced!

# Entropy



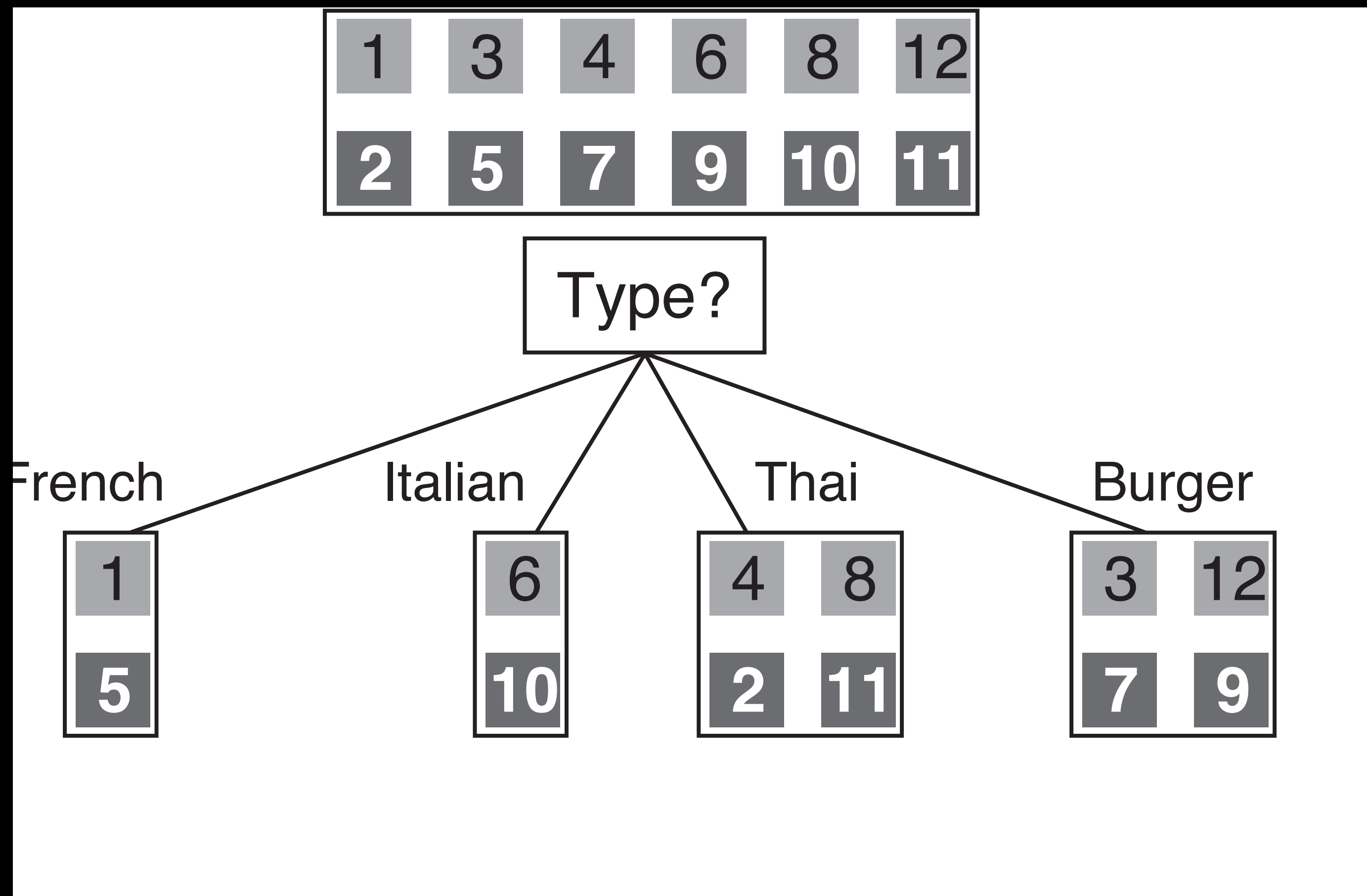

- $S$ is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in $S$
- $p_{\ominus}$ is the proportion of negative examples in $S$
- Entropy measures the impurity of $S$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

# Information Gain

$Gain(S, A) =$ expected reduction in entropy due to sorting on $A$

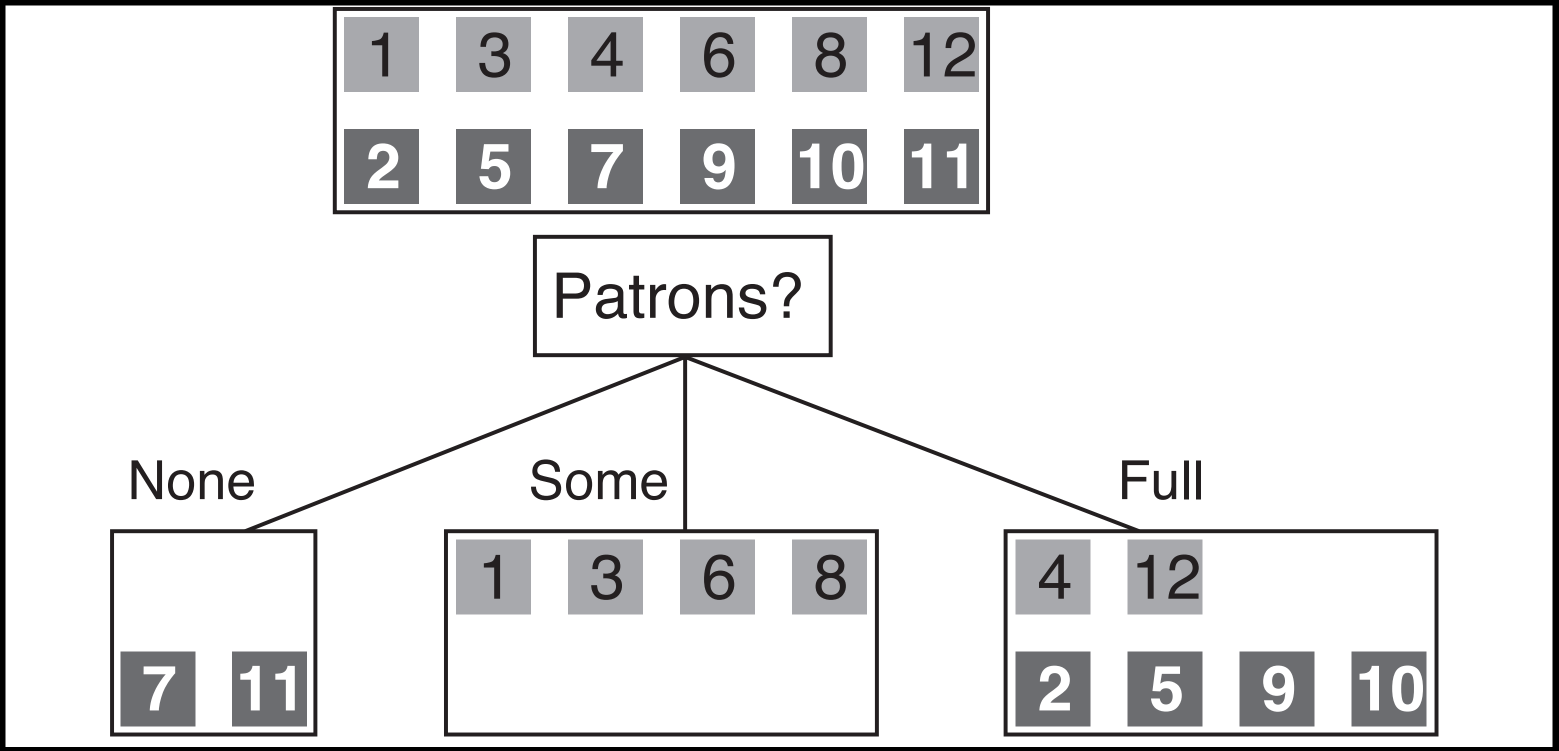
$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$




$$Entropy(S) = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1$$

$$Entropy(S_F) = Entropy(S_I) = Entropy(S_T) = Entropy(S_B) = 1$$

$$Gain(Type) = Entropy(S) - \sum_{v \in Type} \frac{|S_v|}{|S|} Entropy(S_v) = 1 - 1 = 0$$



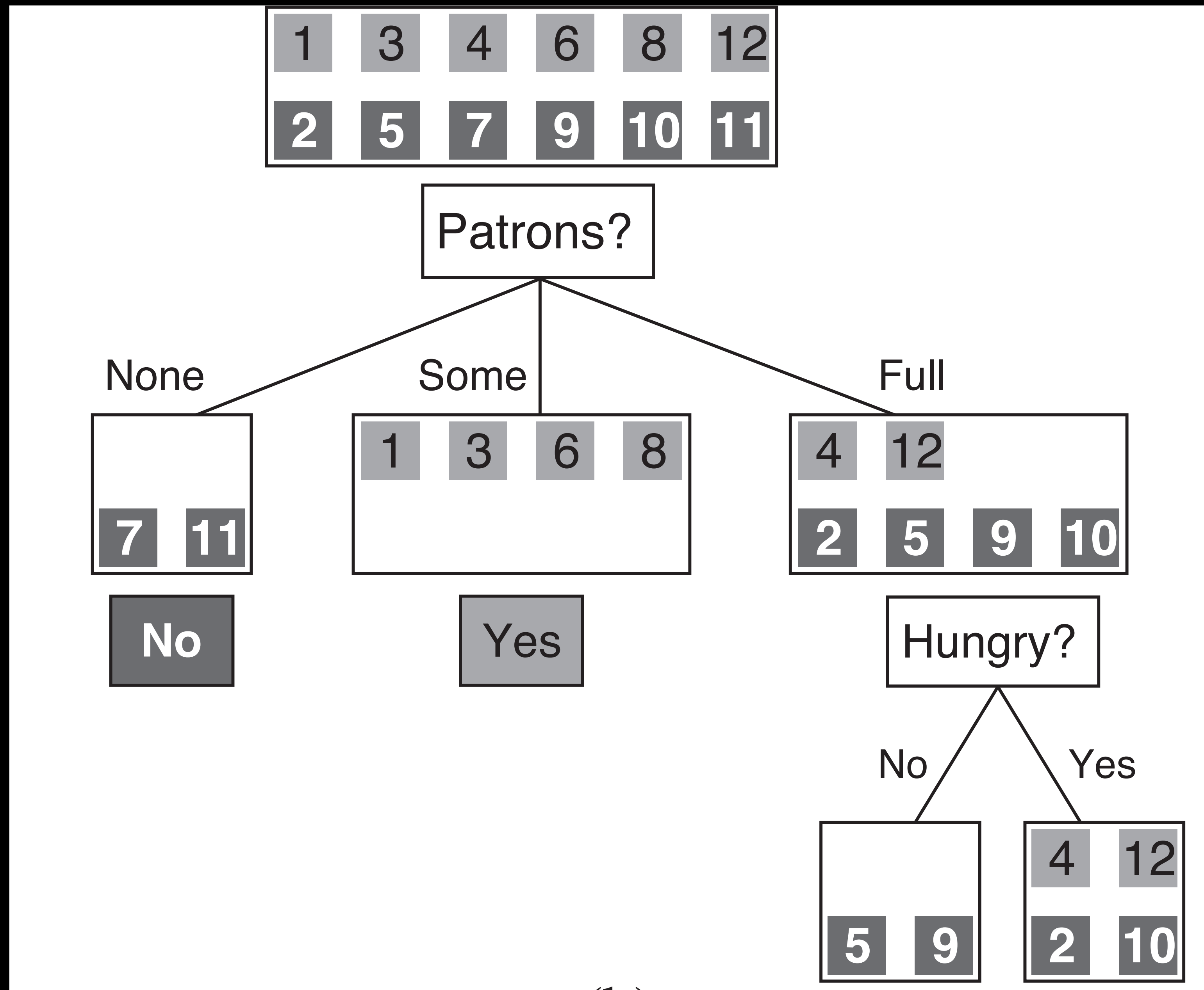

$$Entropy(S) = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1$$

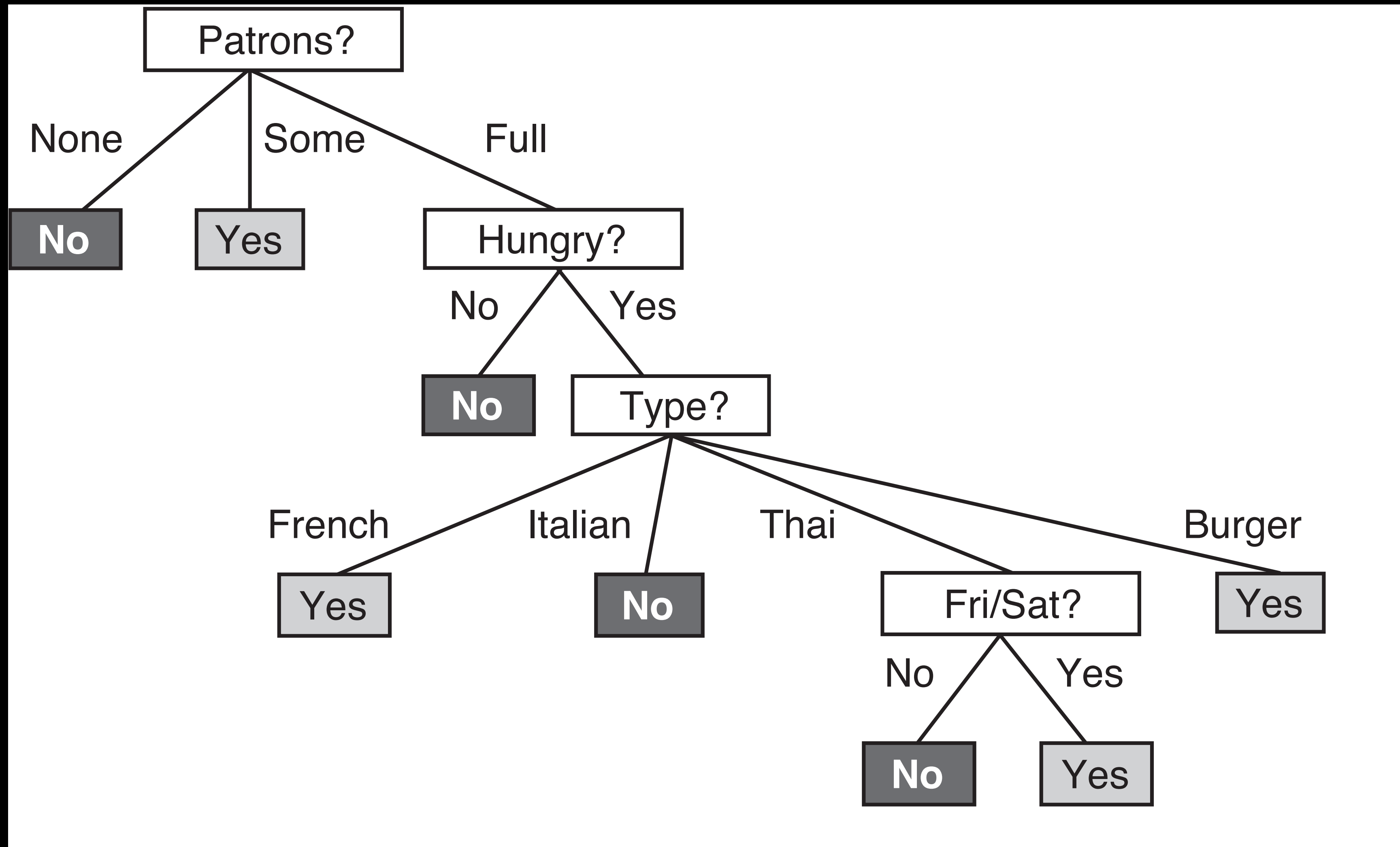
$$Entropy(S_N) = -0\log_2 0 - (1)\log_2 1 = 0$$

$$Entropy(S_S) = -(1)\log_2 1 - 0\log_2 0 = 0$$

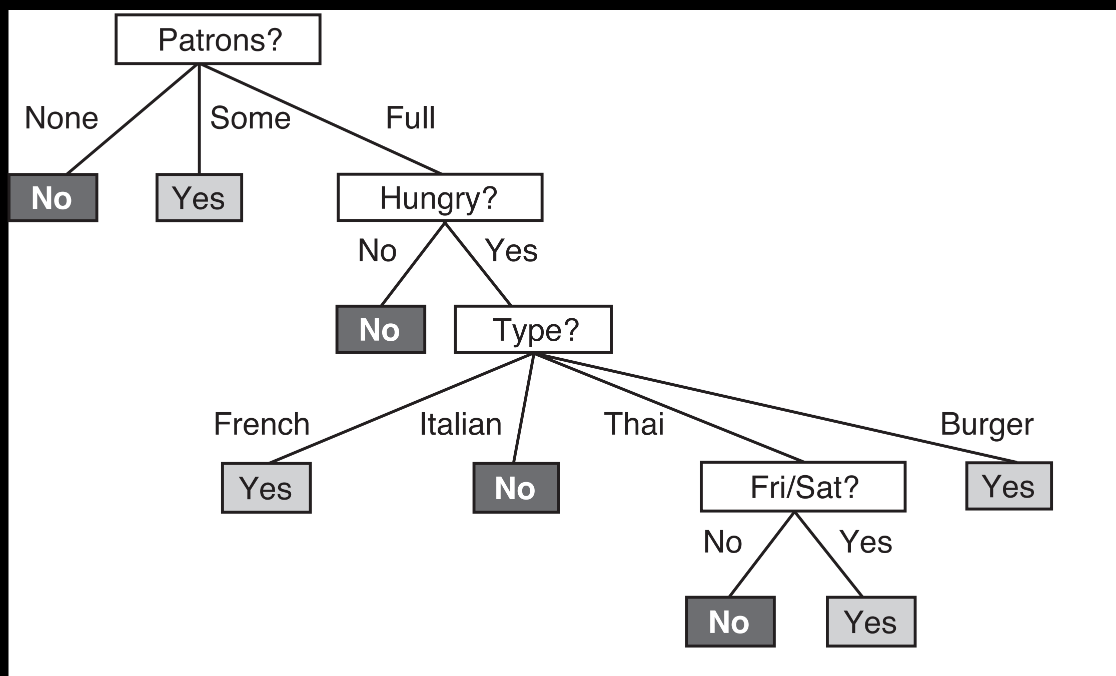
$$Entropy(S_F) = -(\tfrac{1}{3})\log_2 \tfrac{1}{3} - \tfrac{2}{3}\log_2 \tfrac{2}{3} = 0.92$$

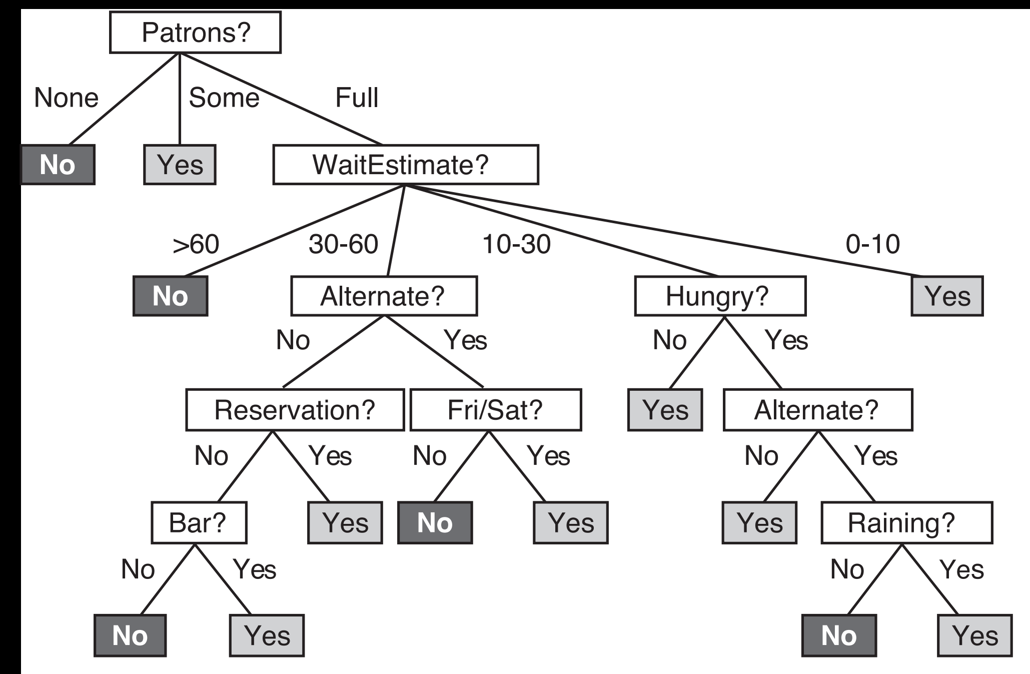
$$Gain(Patron) = 1 - \sum_{v \in Patron} \frac{|S_v|}{|S|} Entropy(S_v) = 1 - (\tfrac{1}{2})(0.92) = 0.54$$


1 3 4 6 8 12
2 5 7 9 10 11
Patrons?
None
Some
Full
7 11
1 3 6 8
4 12
2 5 9 10
No
Yes
Hungry?
No
Yes
5 9
4 12
2 10


Patrons?
None
Some
Full
No
Yes
Hungry?
No
Yes
No
Type?
French
Italian
Thai
Burger
Yes
No
Fri/Sat?
Yes
No
Yes
No
Yes

# Avoiding Overfitting




- Problem: How to determine when to stop growing the decision tree?

# Reduced-Error Pruning

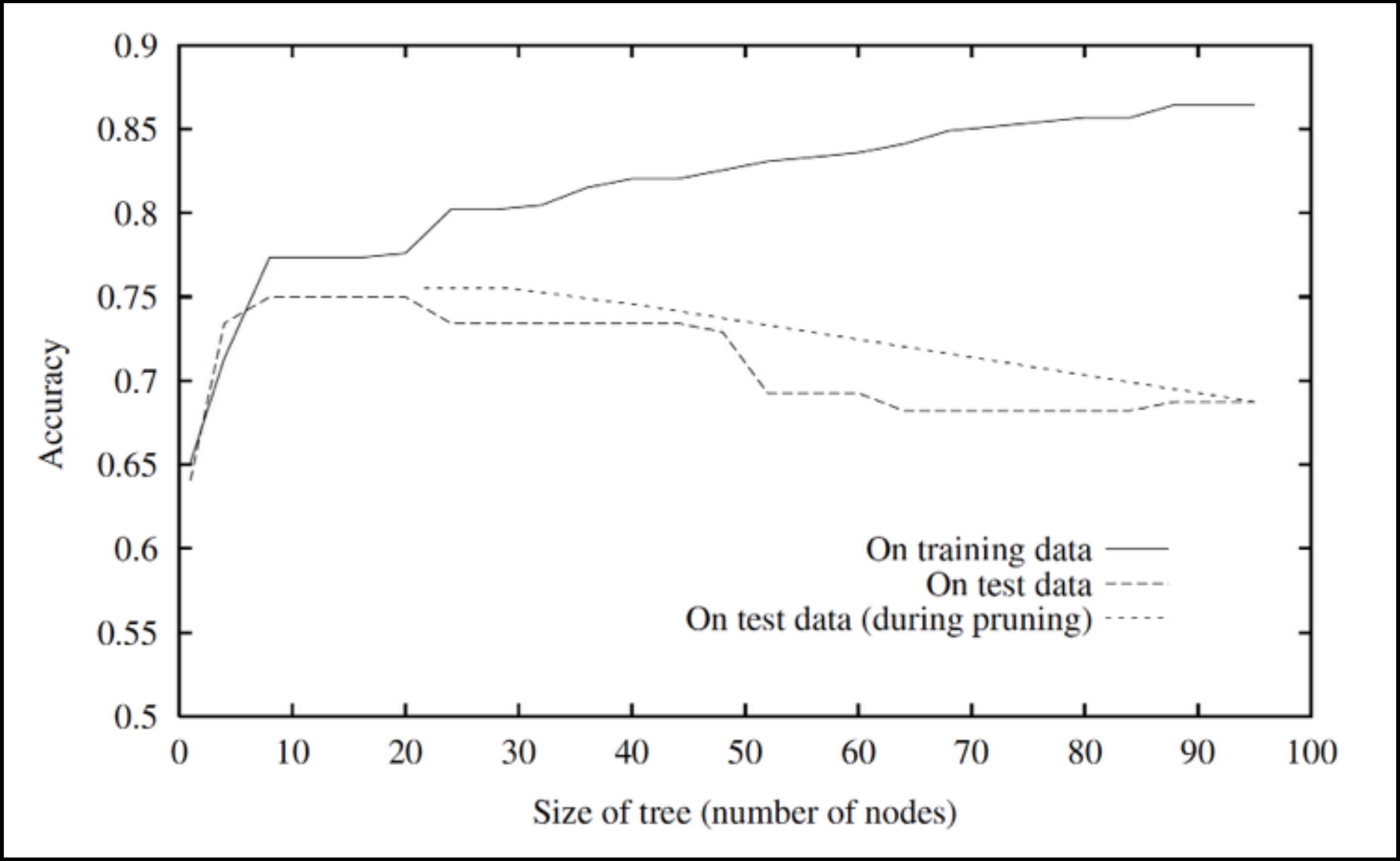
Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree

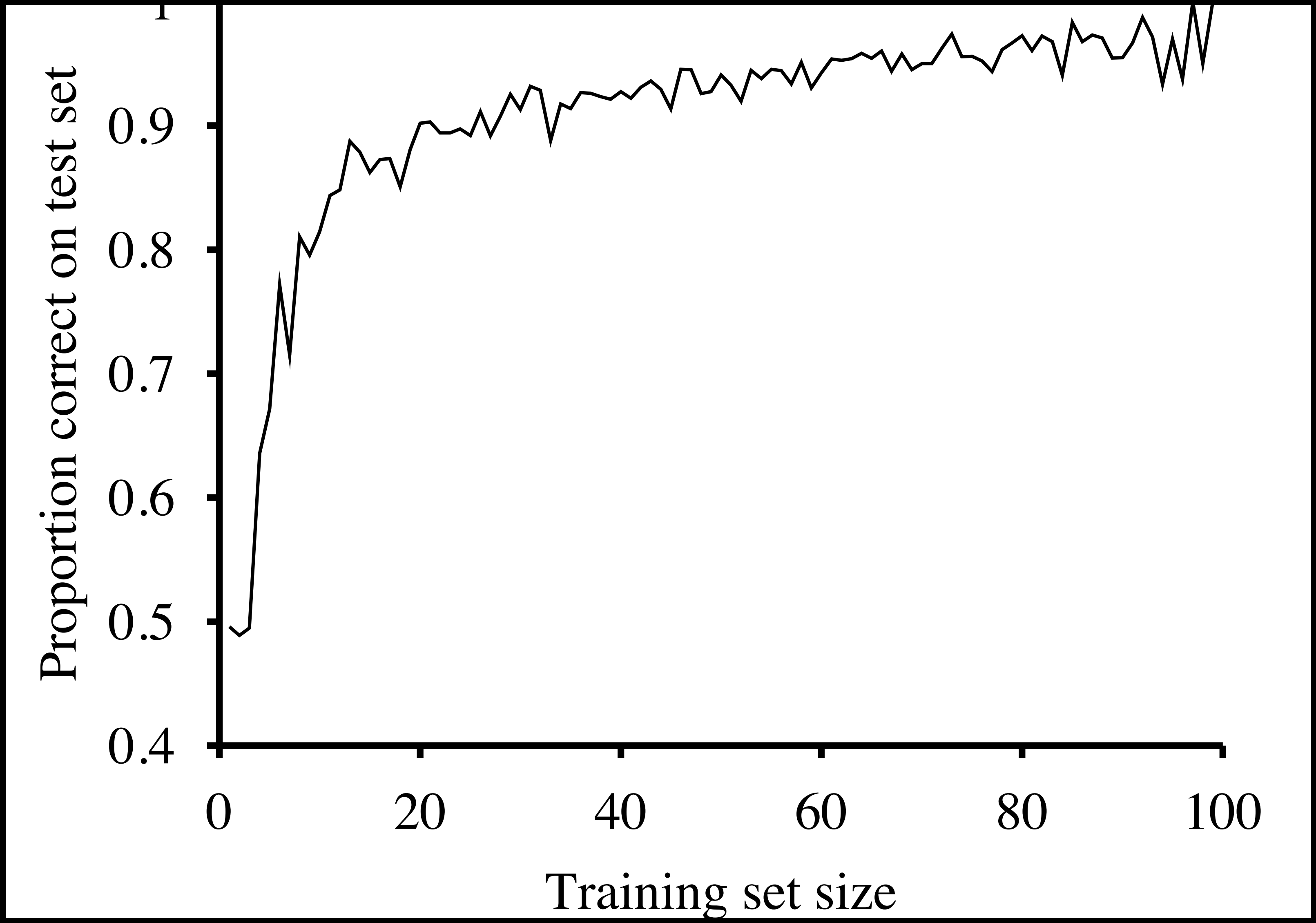
# Effect of Reduced-Error Pruning



# Evaluating Learning Mechanisms

# Evaluating Learning

- Split data into training set and testing set
- Learn a hypothesis $h$ using the training set and evaluate it on the testing set
- Start with training set of size $1$ up to size $N$-$1$




Learning Curve

# Error Rate

- Error rate: proportion of times $h(x) \neq y$ for an $(x, y)$ example
    - Inverse of proportion correct (accuracy)
- Need to evaluate error rate on examples not used in training

# Cross-Validation

- Randomly split data into training and testing (in some proportion)
  - Hold out test data during training
- Doesn't use all data for training

# k-Fold Cross-Validation

- Divide data into $k$ equal subsets
- Perform $k$ rounds of learning
  - Leave out $1$ subset ($1/k$ of the data) each round; use for testing that round
- Average test scores over $k$ rounds

# Learning
# (from Examples)
# Summary

# Learning

- Kinds and dimensions of learning
- General framework for supervised, passive, immediate feedback learning
- Classification and Regression
- Data: training, testing, (pruning)
- Generalization, error, overfitting
- Hypothesis space: lines, curves, decision trees, …

# Coming Up

- April 8: Exam 3, Probability & Introduction to Learning
- Project 3: Learning to Recognize Faces using Neural Networks
  - Assigned: April 10th
  - Due on April 29
- April 11: Neural Networks Part I