CS 3368 Introduction to Artificial Intelligence

Introduction to ML and Decision Tree

Department of Computer Science
Texas Tech University

- Instructor: Jingjing Yao
- Email: jingjing.yao@ttu.edu
- Office: EC 306F
- Office hours: 10 am – 11 am, Tuesday and Thursday

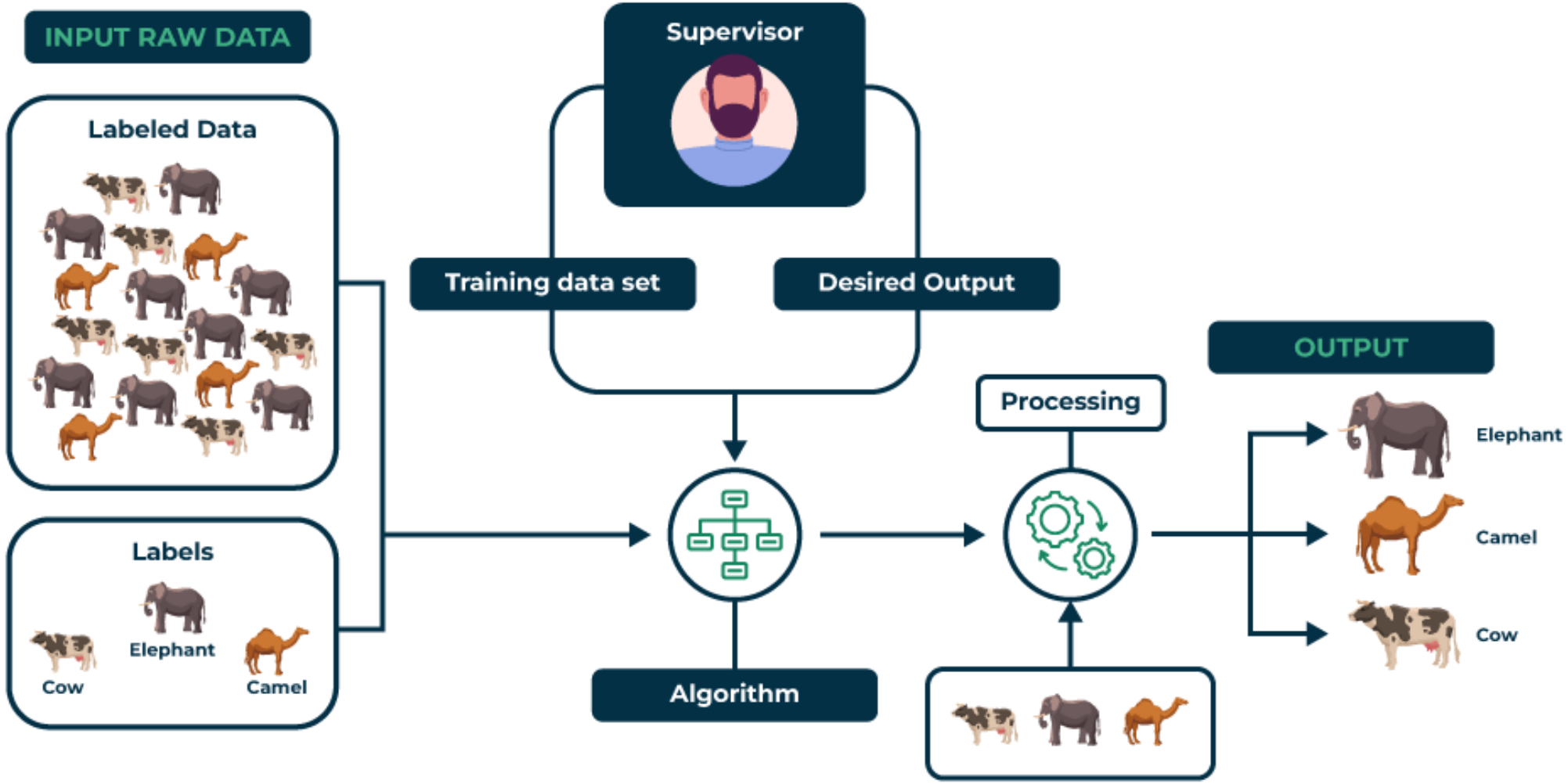
What is Machine Learning

- For many problems, it's difficult to program the correct behavior by hand
  - Recognizing people and objects
  - Understanding human speech
- Machine learning approach: program an algorithm to automatically learn from data or from experience
- ML is concerned with predictive performance, scalability, and autonomy
- Why might you want to use a learning algorithm?
  - Hard to code up a solution by hand, e.g., vision, speech
  - System needs to adapt to a changing environment, e.g., spam detection
  - Want the system to perform better than the human programmers

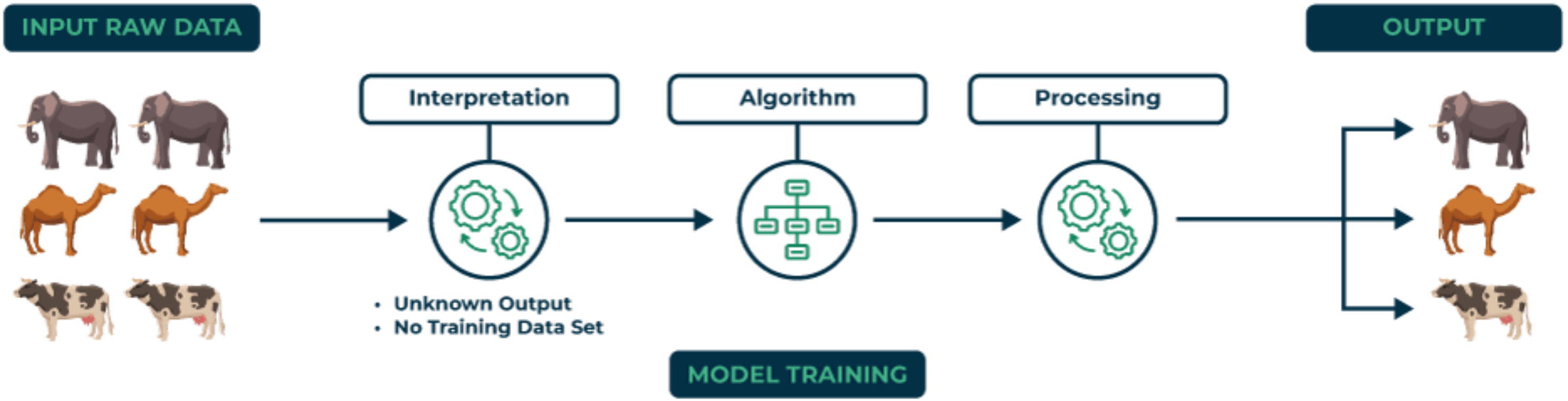
Types of Machine Learning

- Supervised learning: have labeled examples of the correct behavior
- Unsupervised learning: no labeled examples – instead, looking for "interesting" patterns in the data
- Reinforcement learning: learning system (agent) interacts with the world and learns to maximize the reward signal

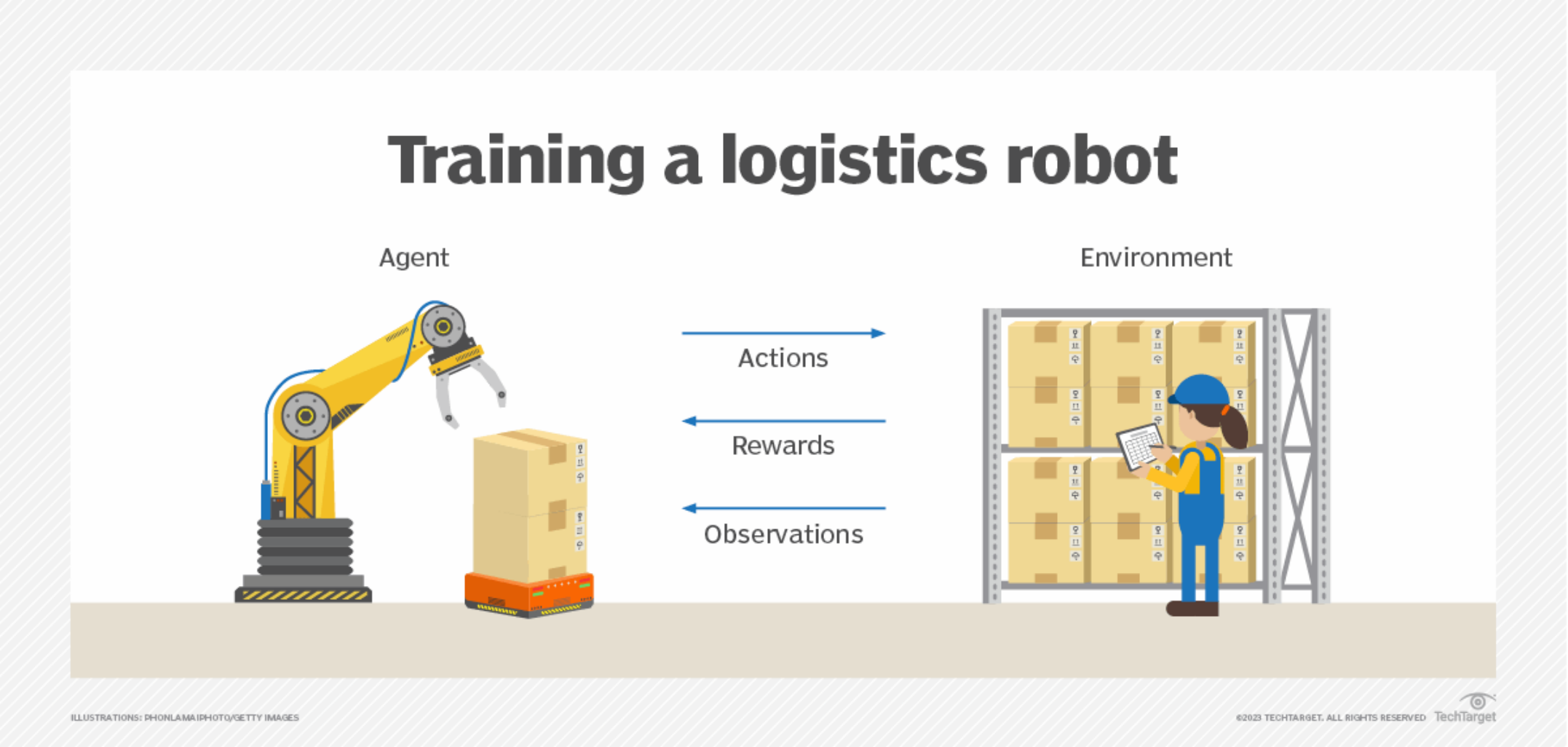
Supervised Learning



Unsupervised Learning



Reinforcement learning



Question

- We are given information on a user's credit card transactions. We would like to detect whether some of the transactions are fraudulent by finding transactions that are different from the other transactions. We have no information on whether any particular transaction is fraudulent or not.
- Is this a supervised or unsupervised learning problem?
- (A) Supervised learning
- (B) Unsupervised learning

Applications - Computer Vision

- Object detection, pose estimation, and almost every other task is done with ML

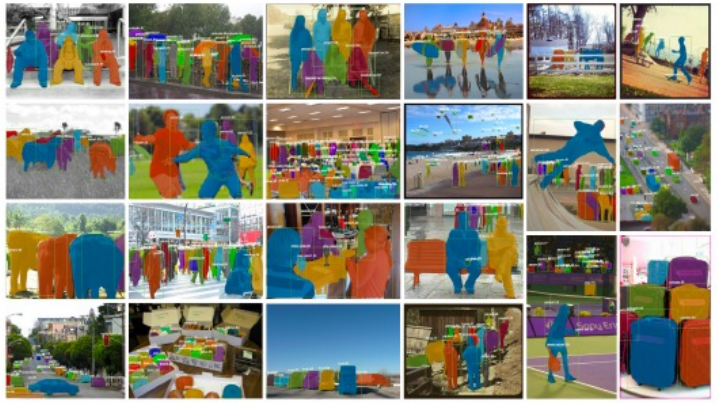


Figure 4. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).

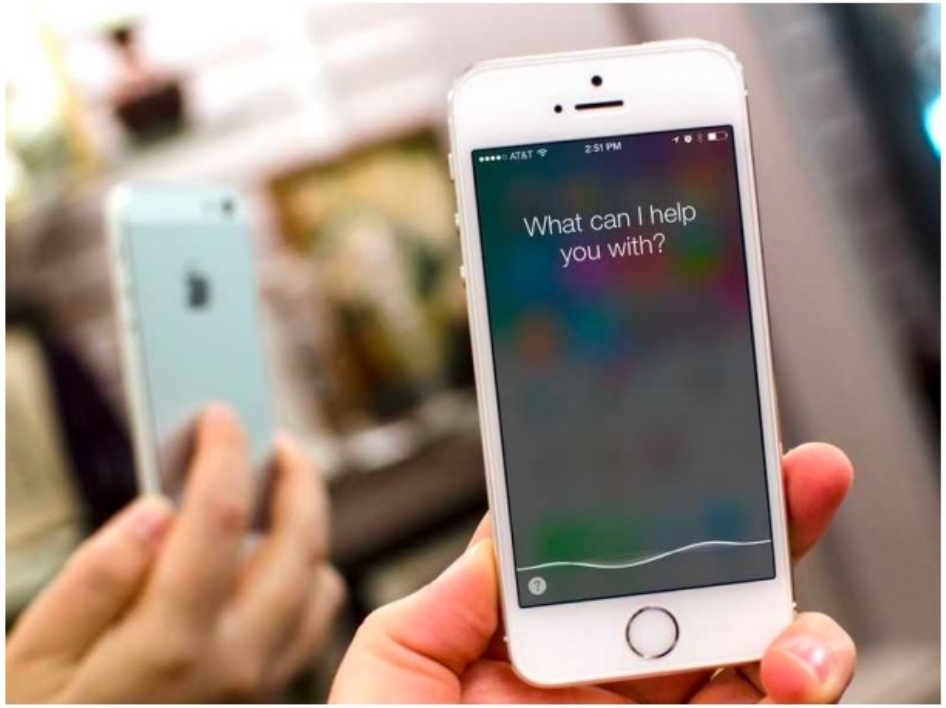
Instance segmentation - Link

DAQUAR 1553
What is there in front of the sofa?
Ground truth: table
IMG+BOW: table (0.74)
2-VIS+BLSTM: table (0.88)
LSTM: chair (0.47)

COCOQA 5078
How many leftover donuts is the red bicycle holding?
Ground truth: three
IMG+BOW: two (0.51)
2-VIS+BLSTM: three (0.27)
BOW: one (0.29)

Speech

- Speech to text, personal assistants, speaker identification



Natural Language Processing

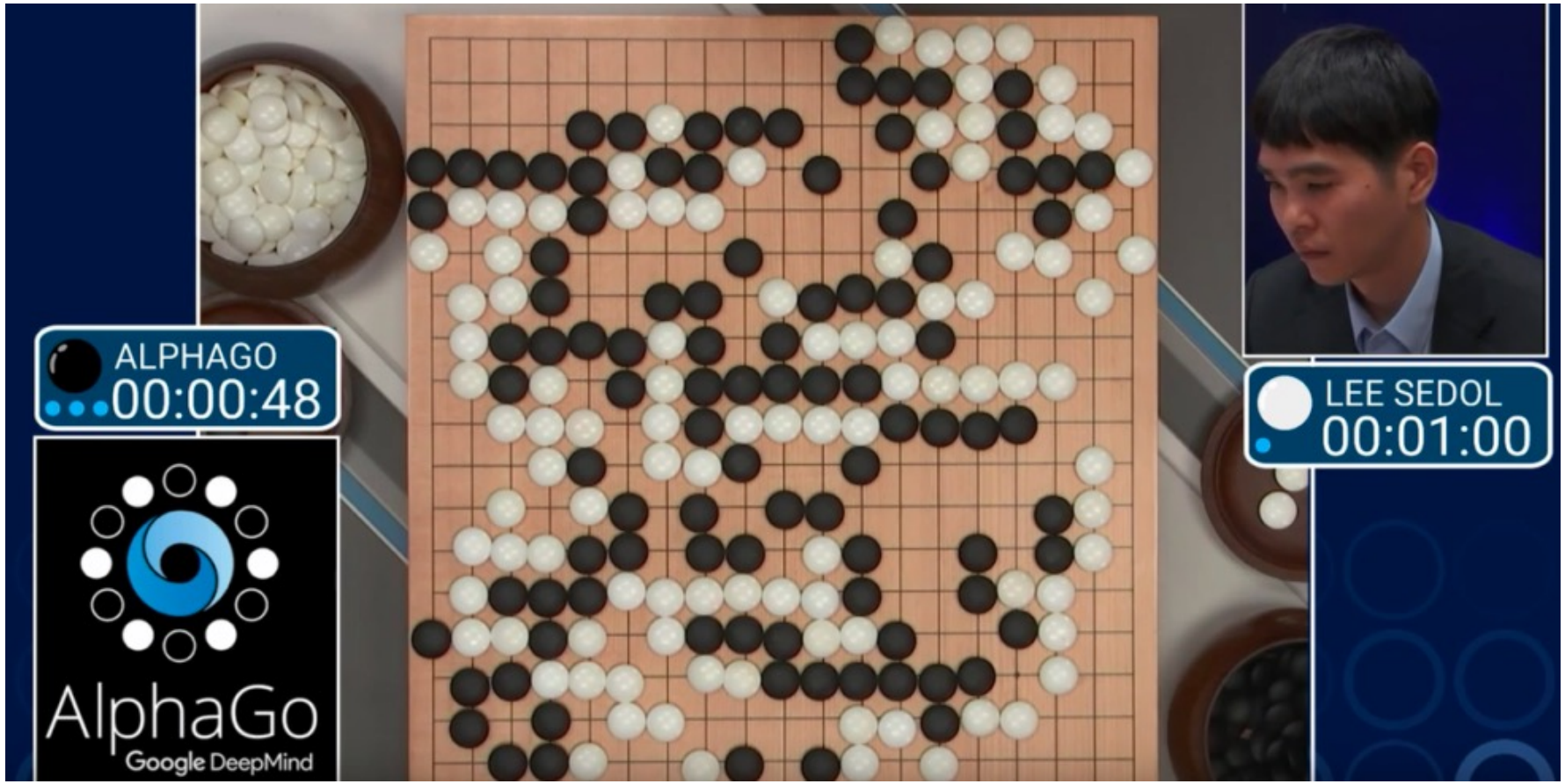
- Machine translation, topic modeling, spam filtering

Real world example: The New York Times

LDA analysis of 1.8M New York Times articles:

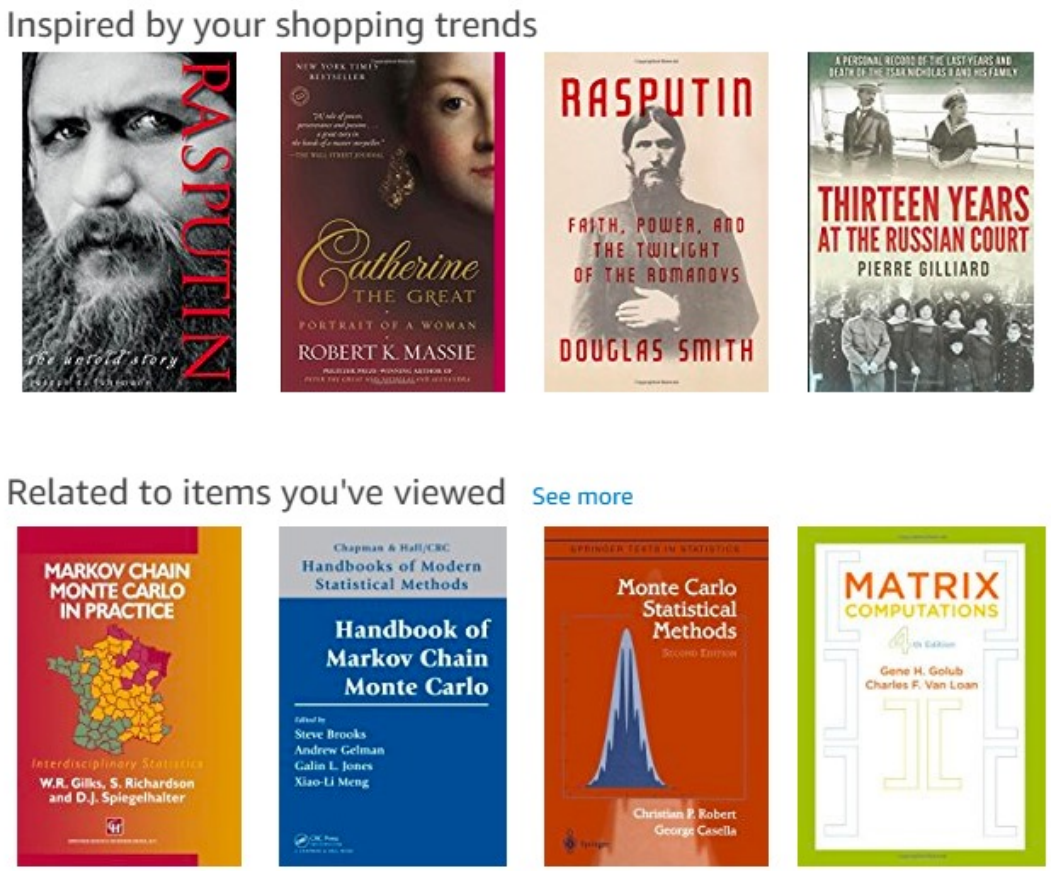
music band songs rock album jazz pop song singer night	book life novel story books man stories love children family	art museum show exhibition artist artists paintings painting century works	game knicks nets points team season play games night coach	show film television movie series says life man character know
theater play production show stage street broadway director musical directed	clinton bush campaign gore political republican dole presidential senator house	stock market percent fund investors funds companies stocks investment trading	restaurant sauce menu food dishes street dining dinner chicken served	budget tax governor county mayor billion taxes plan legislature fiscal

Playing Games

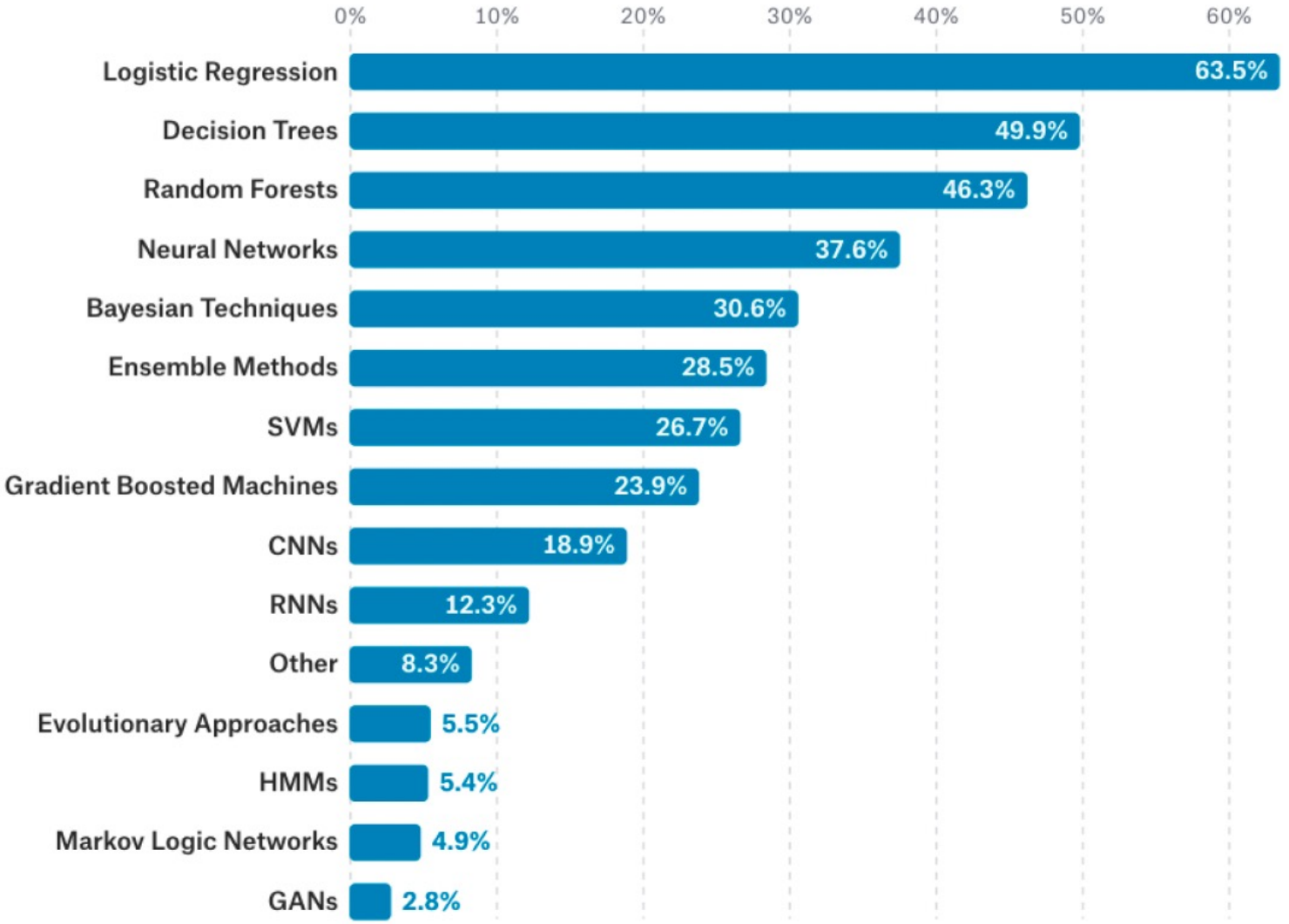


E-commerce & Recommender Systems

- Amazon, Netflix



2017 Kaggle survey of data science and ML practitioners: what data science methods do you use at work?



Implementing ML Systems

- Neural net frameworks: PyTorch, TensorFlow, etc…
  - Compiling computation graphs
  - Libraries of algorithms
  - Support for graphics processing units (GPUs)
- Why learn the details if these frameworks do so much for you
  - Debug when something wrong, which requires understanding what goes on beneath the hood

Introduction

- Our lecture will focus on supervised learning
- This means we are given a training set consisting of inputs and corresponding labels

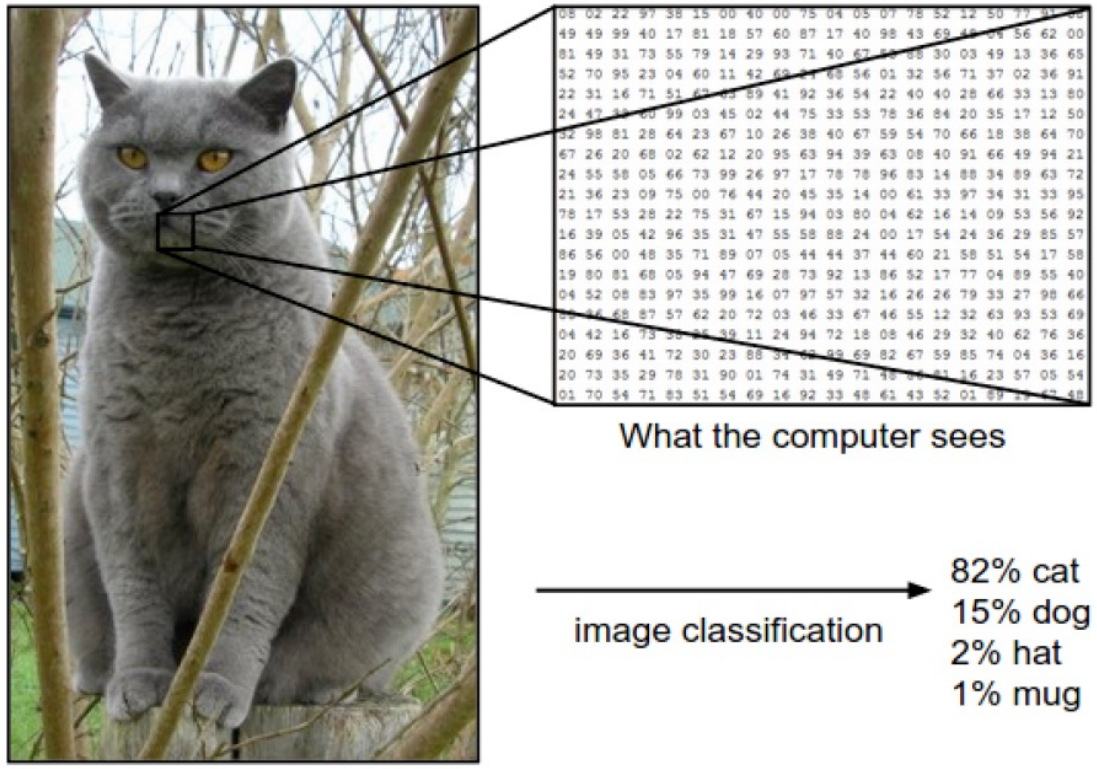
Task	Inputs	Labels
object recognition	image	object category
image captioning	image	caption
document classification	text	document category
speech-to-text	audio waveform	text
⋮	⋮	⋮

Input Vectors

- ML algorithms need to handle lots of types of data
- Images, text, audio waveforms, credit card transactions, etc.
- Common strategy: represent the input as an input vector
- Vectors are a great representation since we can do linear algebra

Input Vectors

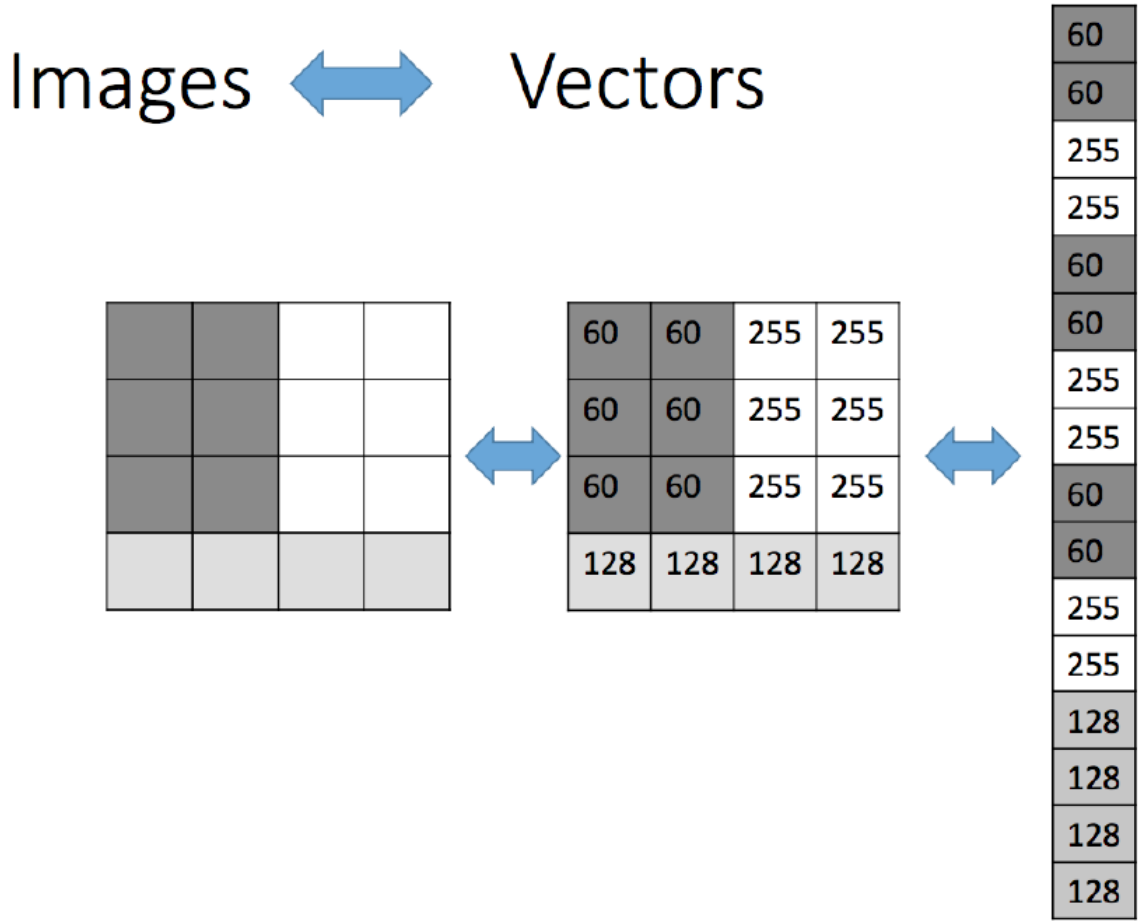
- What an image looks like to the computer



[Image credit: Andrej Karpathy]

Input Vectors

- Can use raw pixels



Feature Vectors

x $f(x)$ y

```
Hello,

Do you want free printr
cartriges?  Why pay more
when you can get them
ABSOLUTELY FREE!  Just
```

→

```
# free       : 2
YOUR_NAME    : 0
MISSPELLED   : 2
FROM_FRIEND  : 0
...
```

→ SPAM or +

→

```
PIXEL-7,12   : 1
PIXEL-7,13   : 0
...
NUM_LOOPS    : 1
...
```

→ “2”

Two Types of Supervised Learning

- Regression: target is continuous, e.g., temperature
- Classification: target is an element of a discrete set, e.g., {cloudy, windy, sunny}

Question

- Is the following problem classification or regression?

- You are given historical data on the weather condition (sunny, cloudy, rain, or snow) on a particular day of the year. You want to predict the weather condition of this day next year.
- (A) Classification
- (B) Regression
- (C) This is not supervised learning.

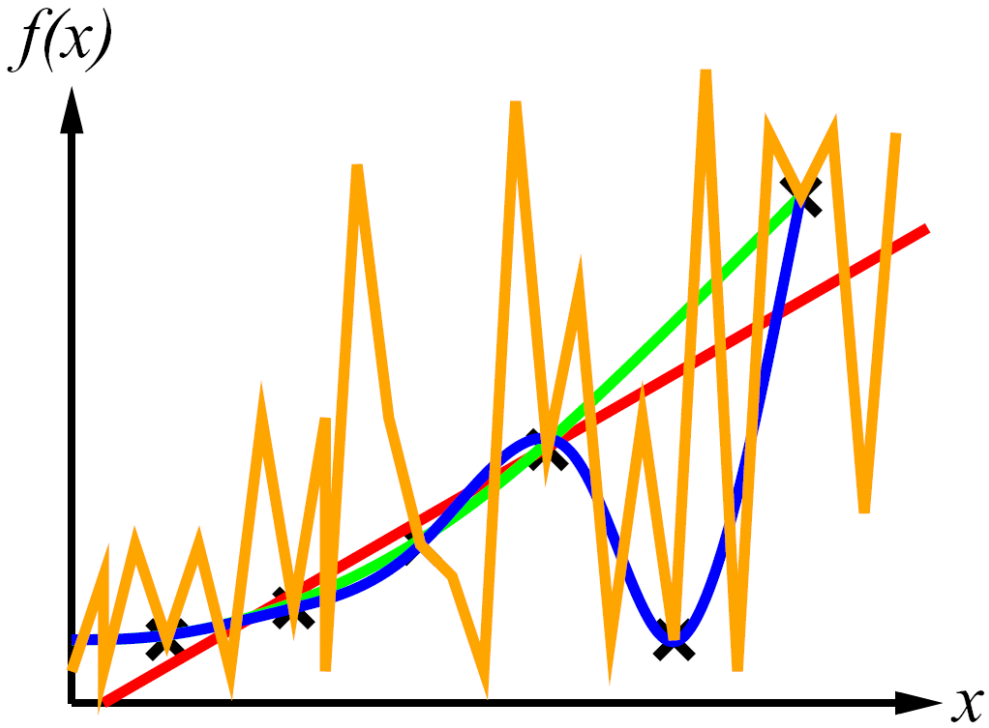
Question

- Is the following problem classification or regression?

- You are given historical data on the price of a house at several points in time. You want to predict the price of this house next month.
- (A) Classification
- (B) Regression
- (C) This is not supervised learning.

Example: A prediction task

- Curve fitting (regression, function approximation):



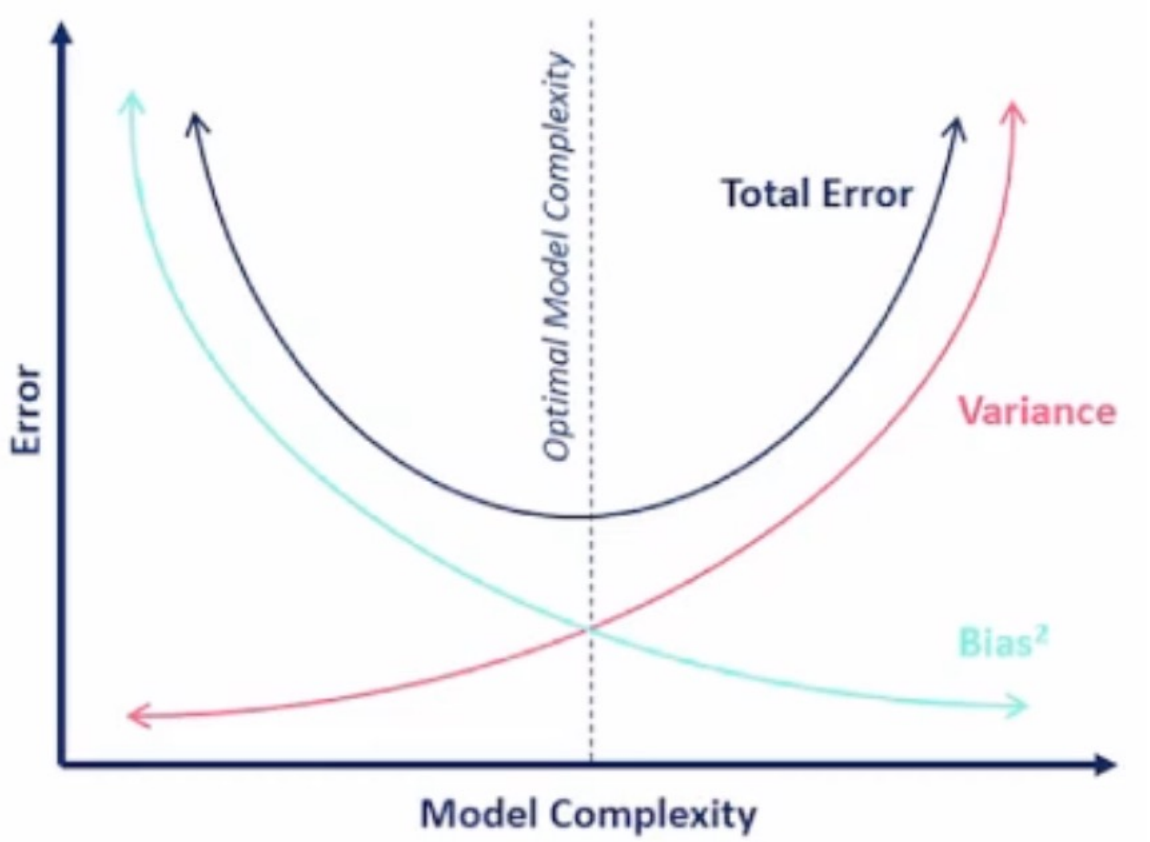
- Which function is correct ?
  - All curves can be justified as the correct one from some perspective

Generalization

- Goal of ML is to find a hypothesis that can predict unseen examples correctly
- How do we choose a hypothesis that generalizes well?
  - Ockham's razor (if you have two competing ideas to explain the same phenomenon, you should prefer the simpler one)
  - Cross validation
- A tradeoff between
  - Complex hypothesis that fit the training data well
  - Simpler hypothesis that may generalize better
  - Bias vs variance tradeoff

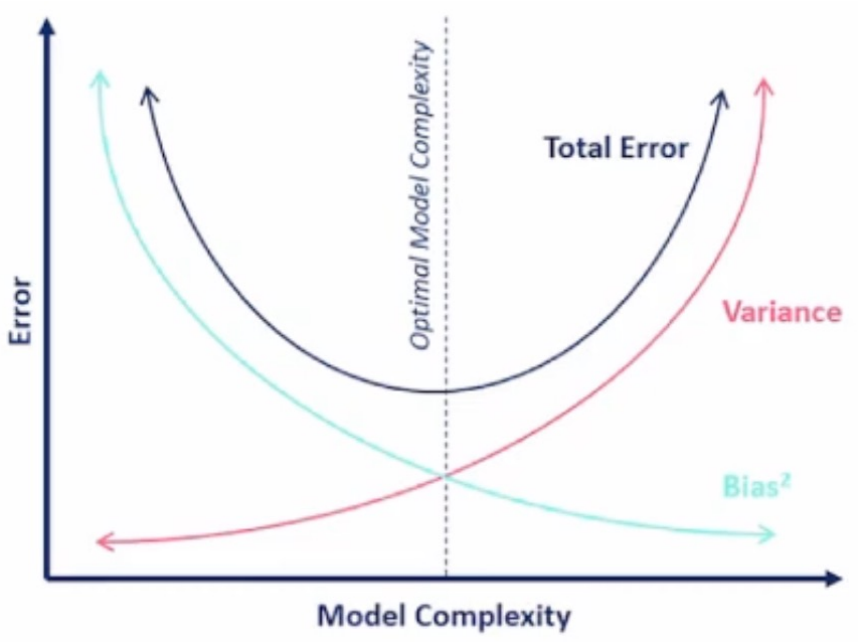
Bias-Variance Trade-off

- How well does the hypothesis fit the data as the hypothesis becomes more complex?

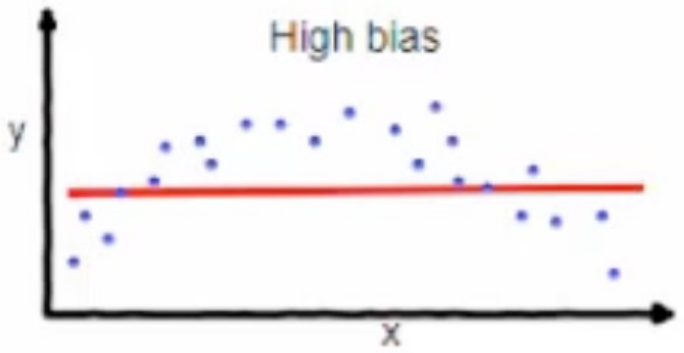


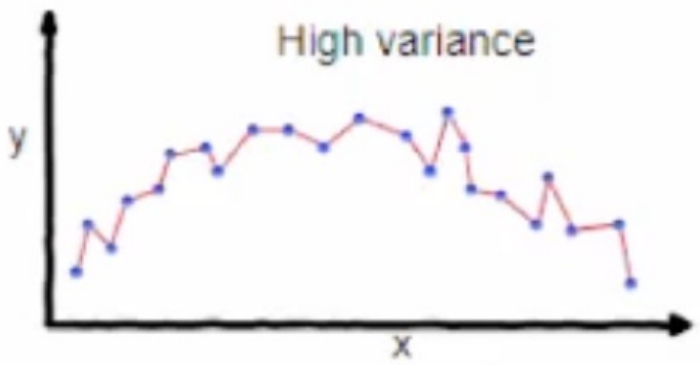
Bias-Variance Trade-off

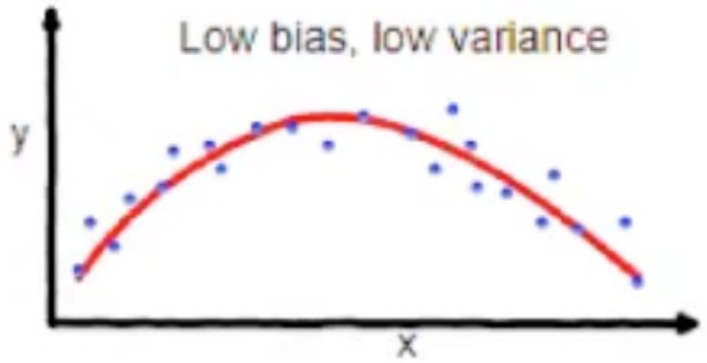
- Bias: If I have infinite data, how well can I fit the data with my learned hypothesis?
- A hypothesis with high bias: makes strong assumptions, too simplistic, has few degrees of freedom, does not fit the training data well.
- Variance: How much does the learned hypothesis vary given different training data?
- A hypothesis with high variance: has a lot of degrees of freedom, is very flexible. Whenever the training data changes, the hypothesis changes a lot. Fits the training data very well.

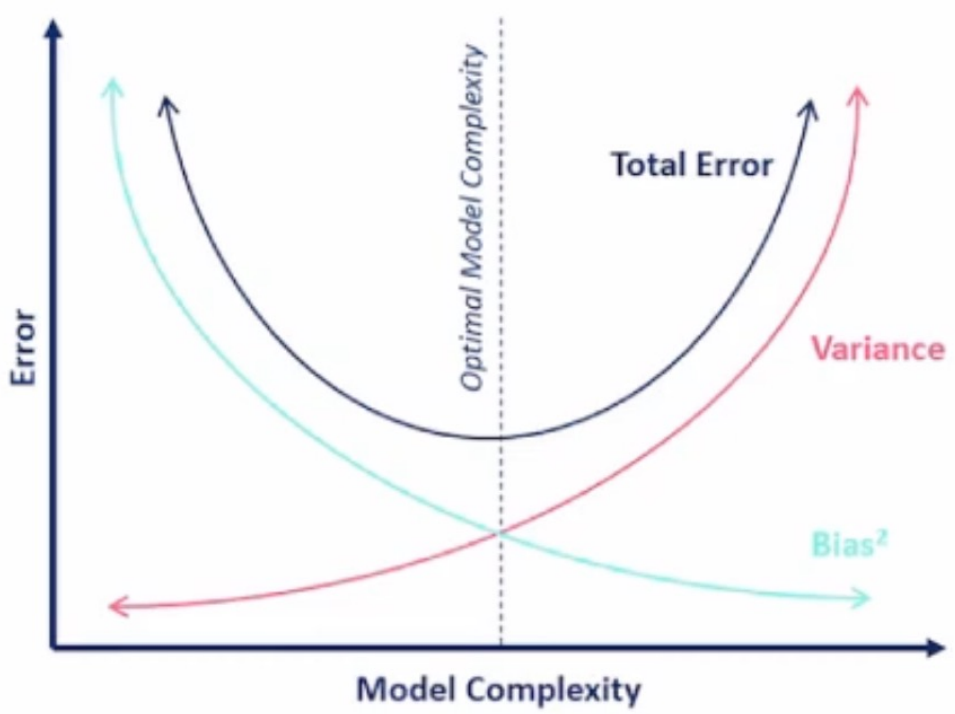


Bias-Variance Trade-off



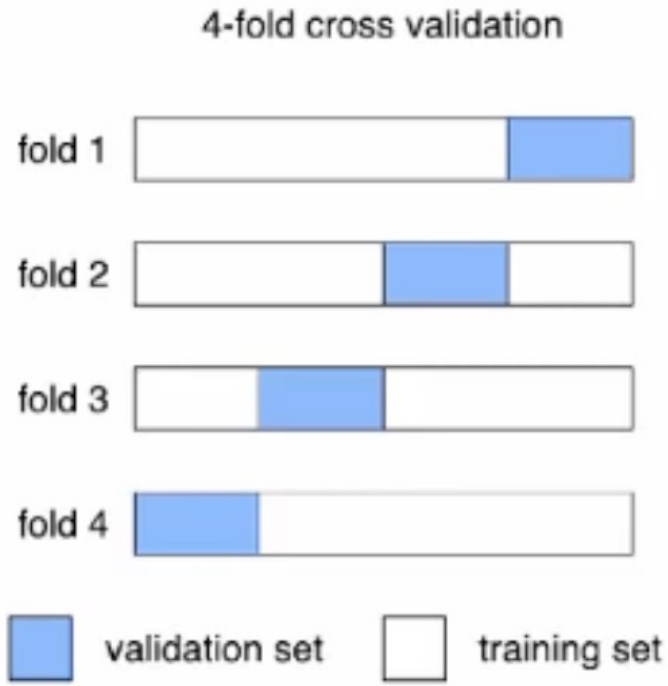




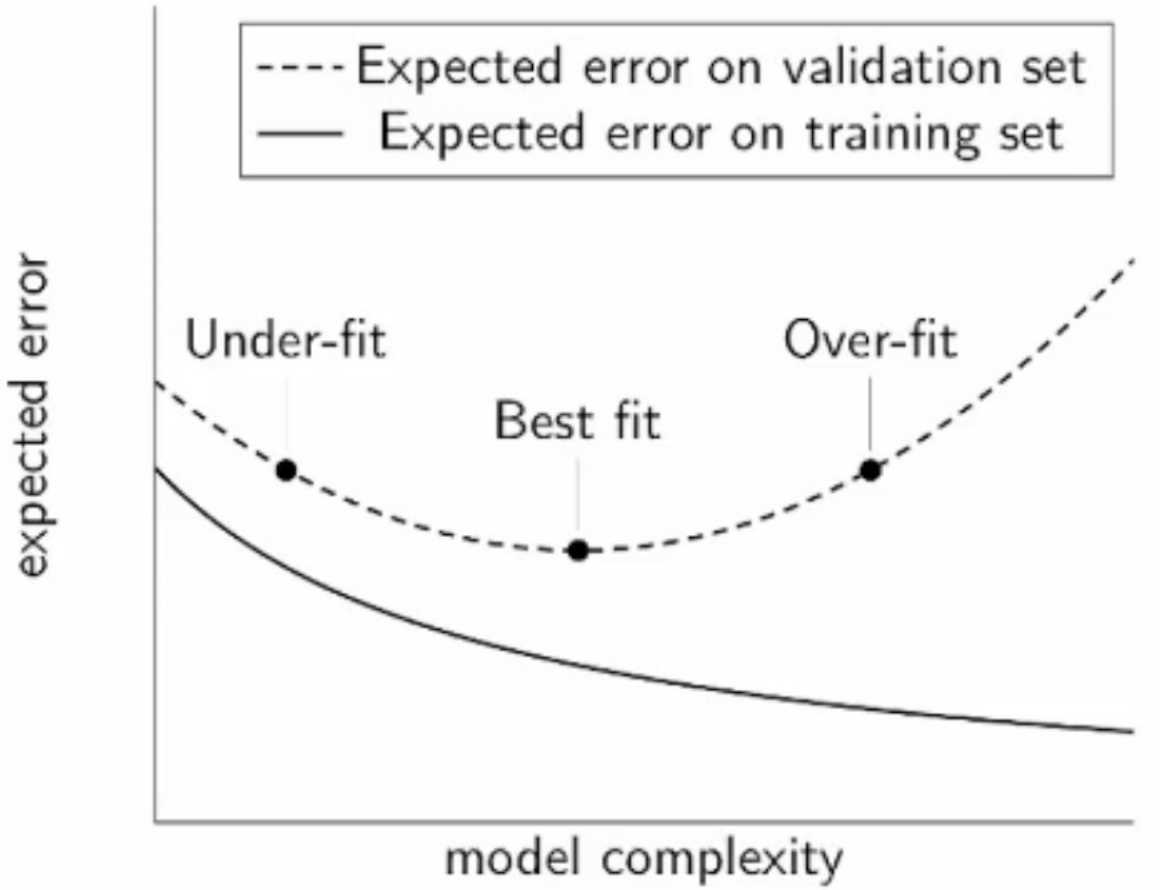


Cross Validation

- How do we find a hypothesis that has low bias and low variance?
- Use cross validation
- Select one of the K trained hypothesis as your final one

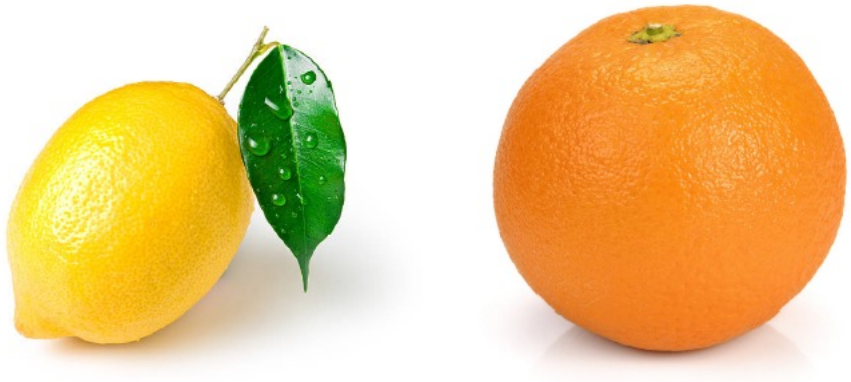


Overfitting



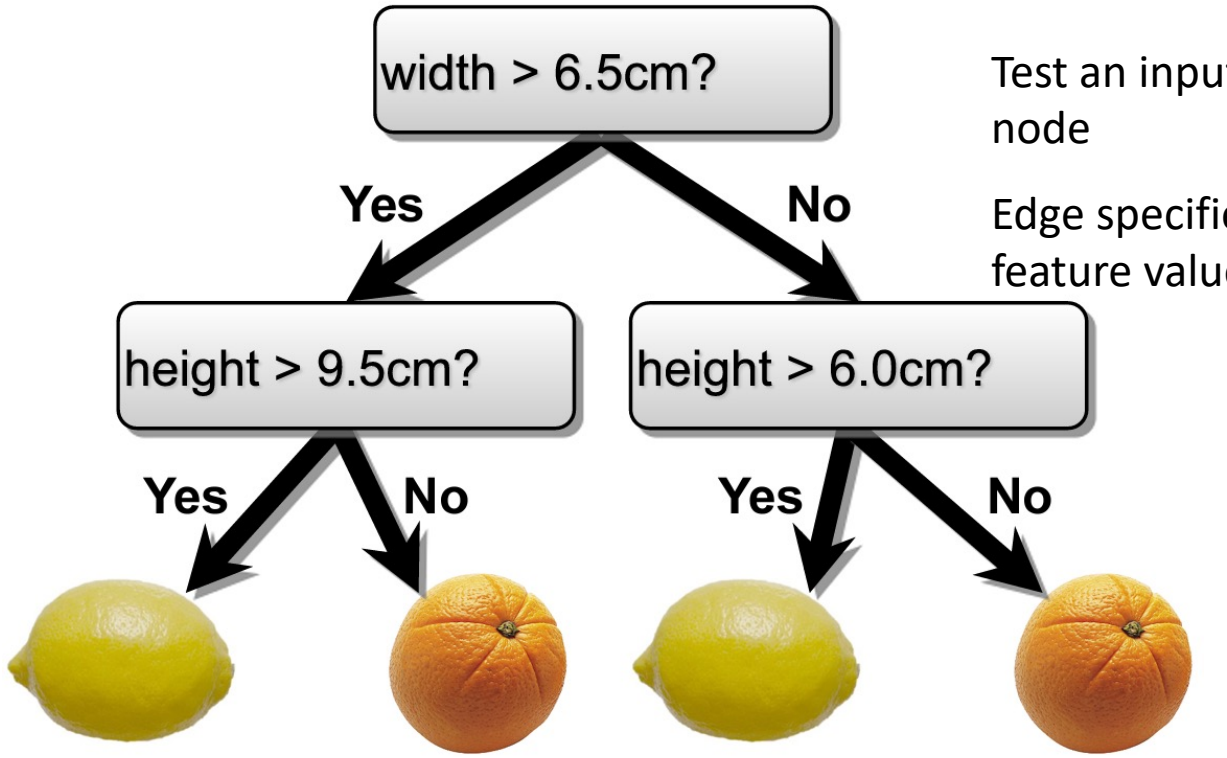
Lemons or Oranges

- Scenario: You run a sorting facility for citrus fruits
- Binary classification: lemons or oranges
- Features measured by sensor on conveyor belt: height and width



Decision Trees

- Make predictions by splitting on features according to a tree structure



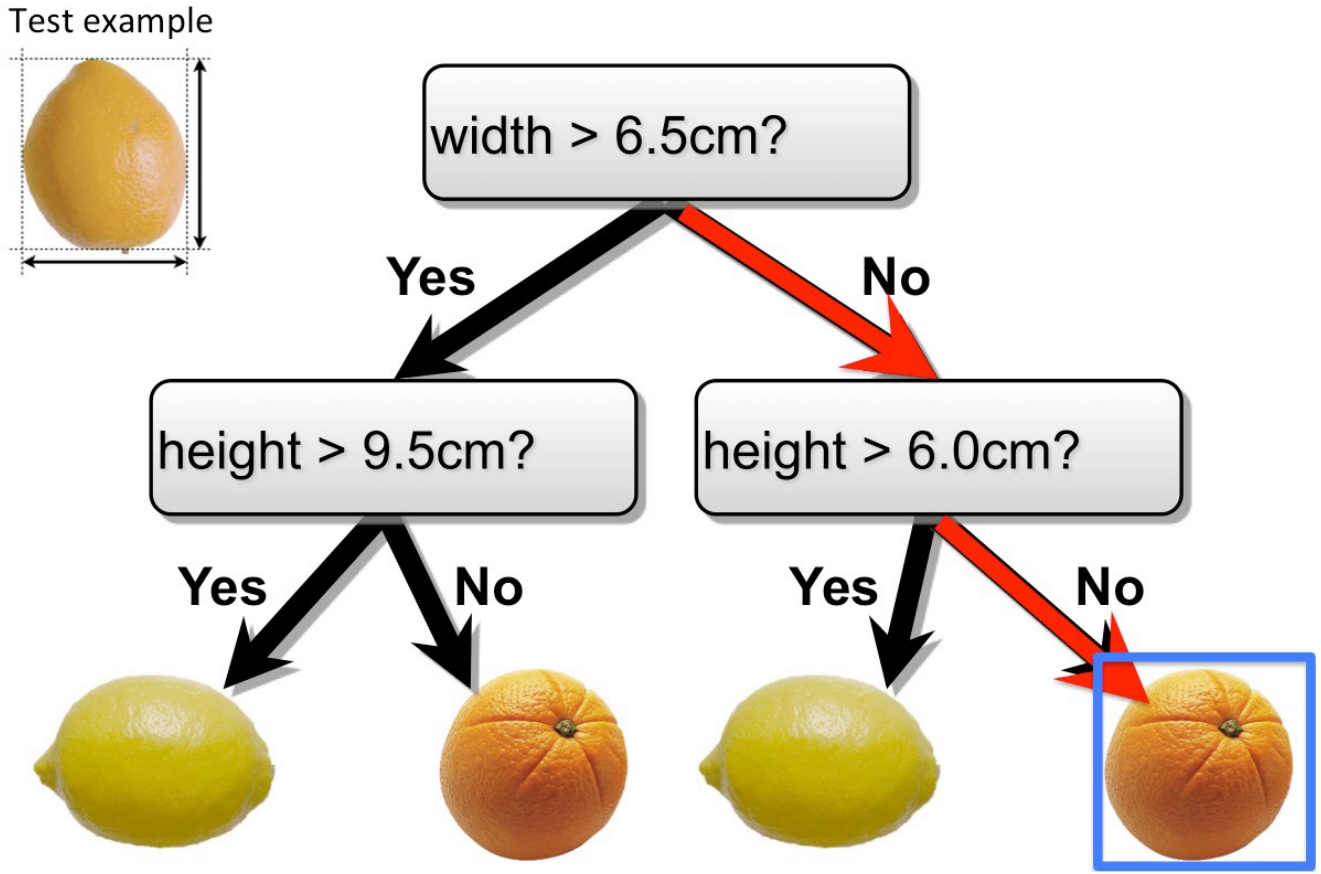
Test an input feature at each node

Edge specifies feature value

Leaf specifies class label

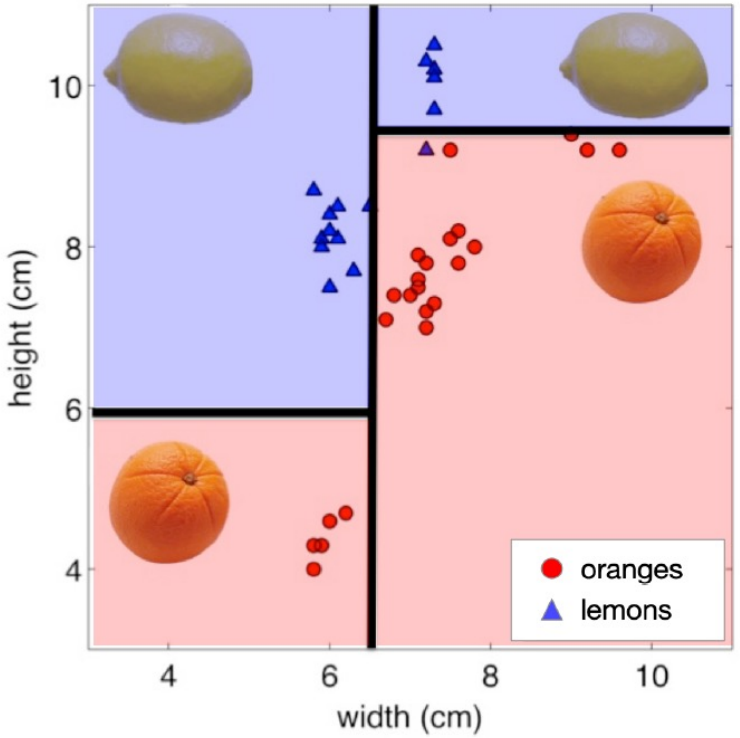
Decision Trees

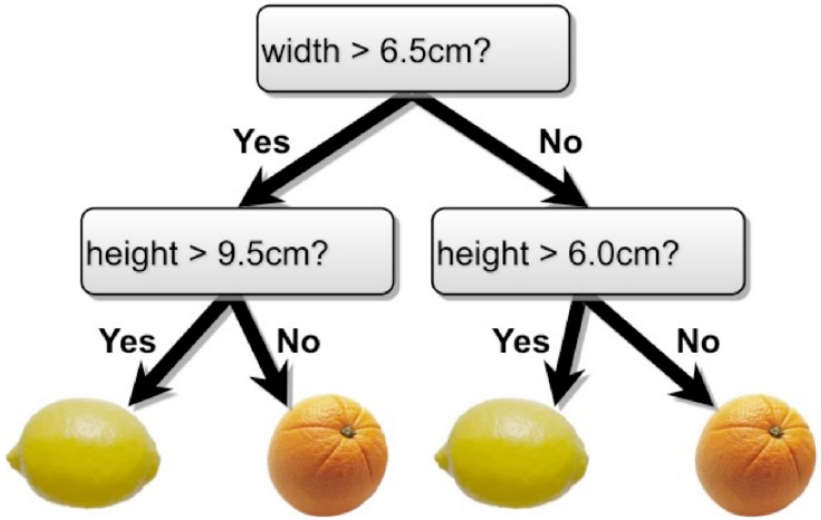
- Make predictions by splitting on features according to a tree structure



Decision Trees – Continuous Features

- Split continuous features by checking whether that feature is greater than or less than some threshold
- Decision boundary is made up of axis-aligned planes

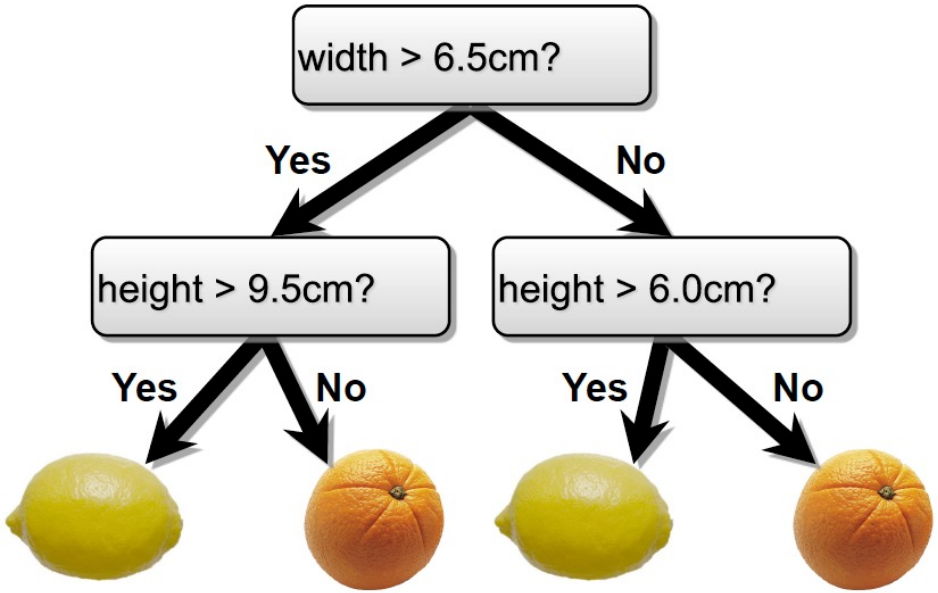




Decision trees

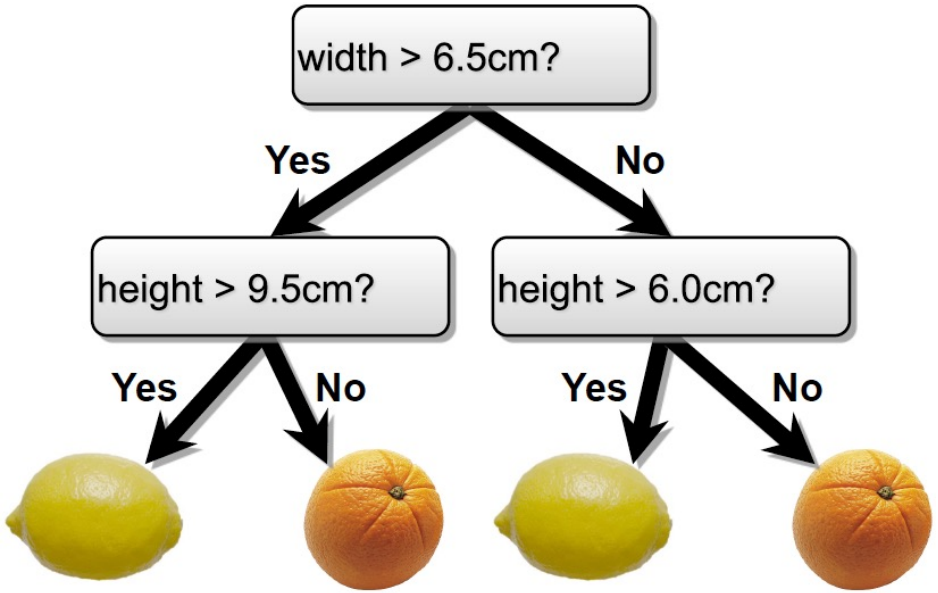
- Internal nodes test a feature
- Branching is determined by the feature value
- Leaf nodes are outputs (predictions)

Question: What are the hyperparameters of this model?



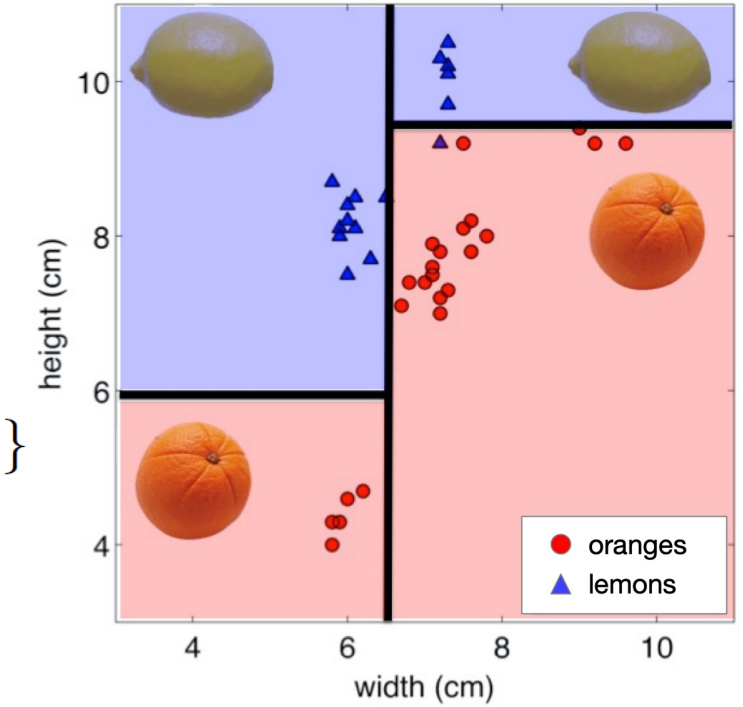
Hyper-parameters of decision tree

- # of nodes in the tree
- Max depth of tree
- # of branches at split
- Min # of examples at a node
- Max # of features to consider



Decision trees

- Each path from root to a leaf defines a region R_m of input space
- Let $\{(x^{(m_1)}, t^{(m_1)}), \ldots, (x^{(m_k)}, t^{(m_k)})\}$ be the training examples that fall into R_m
- Regression tree:
  - Continuous output
  - Leaf value is typically set to the mean value in $\{t^{(m_1)}, \ldots, t^{(m_k)}\}$
- Classification tree (we will focus on this):
  - Discrete output
  - Leaf value is typically set to the most common value in $\{t^{(m_1)}, \ldots, t^{(m_k)}\}$



Decision trees – discrete features

- Will I eat at this restaurant?

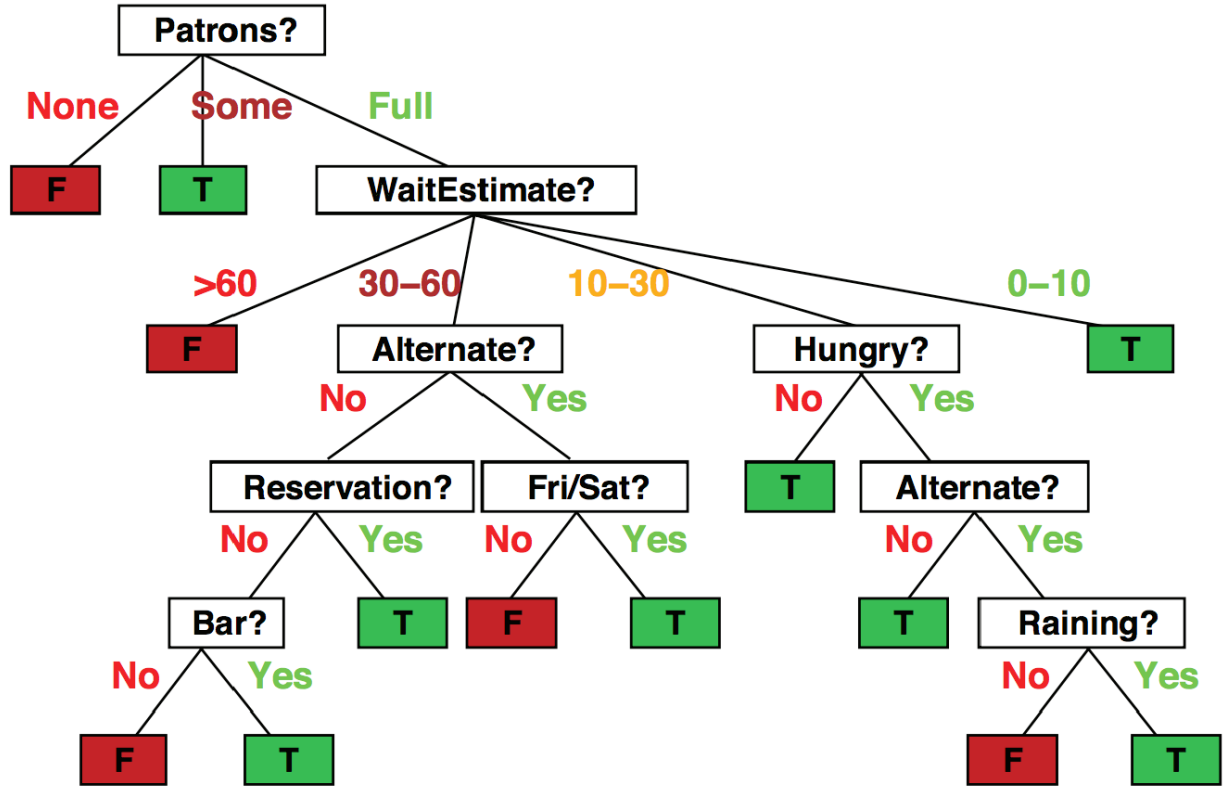
Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 =$ *Yes*
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 =$ *No*
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 =$ *Yes*
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 =$ *Yes*
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 =$ *No*
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 =$ *Yes*
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 =$ *No*
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 =$ *Yes*
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 =$ *No*
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} =$ *No*
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} =$ *No*
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} =$ *Yes*

1.	Alternate: whether there is a suitable alternative restaurant nearby.
2.	Bar: whether the restaurant has a comfortable bar area to wait in.
3.	Fri/Sat: true on Fridays and Saturdays.
4.	Hungry: whether we are hungry.
5.	Patrons: how many people are in the restaurant (values are None, Some, and Full).
6.	Price: the restaurant's price range ($, $$, $$$).
7.	Raining: whether it is raining outside.
8.	Reservation: whether we made a reservation.
9.	Type: the kind of restaurant (French, Italian, Thai or Burger).
10.	WaitEstimate: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).

Decision trees – discrete features

- Will I eat at this restaurant?

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$



Learning Decision Trees

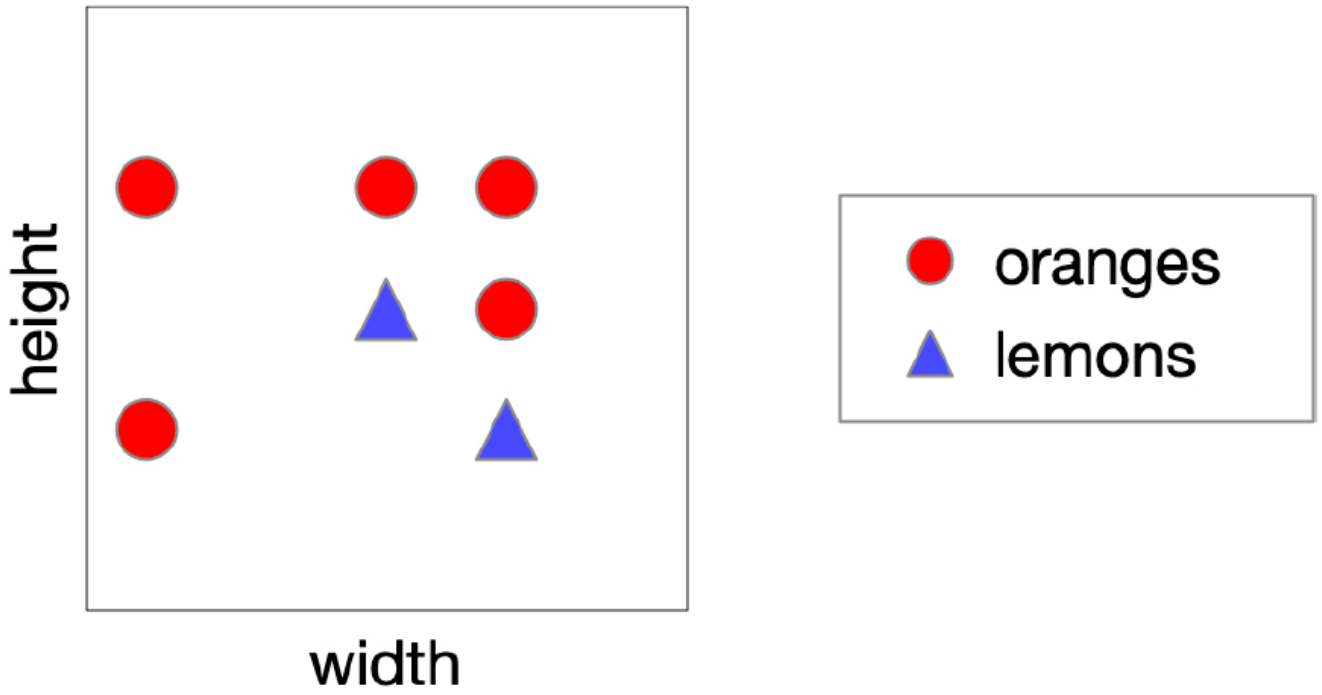
- Decision trees are universal function approximators
  - For any training set we can construct a decision tree that has exactly the one leaf for every training point, but it probably won't generalize
  - Example- if all D features were binary, and we had 2^D unique training examples, a full binary tree would have one leaf per example
- Finding the smallest decision tree that correctly classifies a training set is NP complete
- How do we construct a useful decision tree?

Learning decision trees

- Resort to a greedy heuristic:
  - Start with the whole training set and an empty decision tree
  - Pick a feature and candidate split that would most reduce a loss, i.e., the most informative feature
  - Split on that feature and recurse on subpartitions
- What is a loss?
  - When learning a model, we use a scalar number to assess whether we're on track
  - Scalar value: low is good, high is bad
- Which loss should we use?
  - Optimal choice needs to think about the future

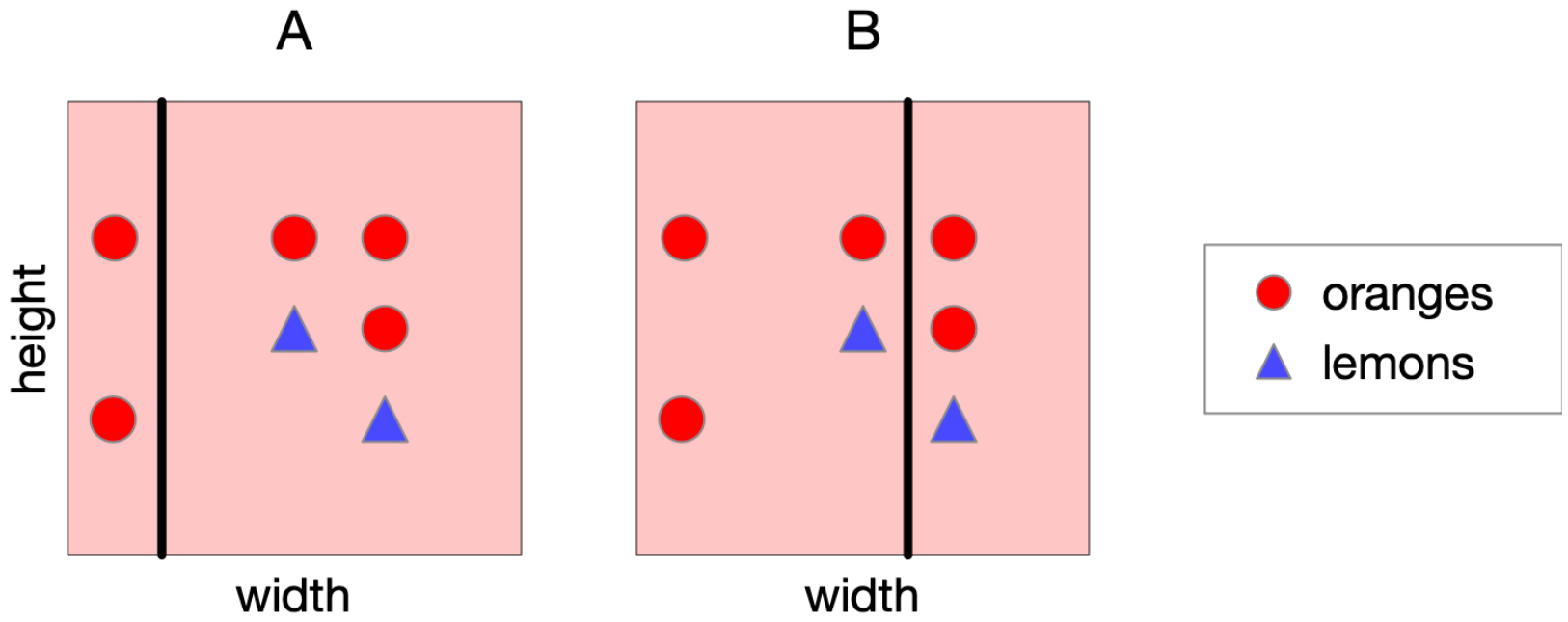
Choosing a Good Split

- Consider the following data. Let's split on width.
- Classify by majority.



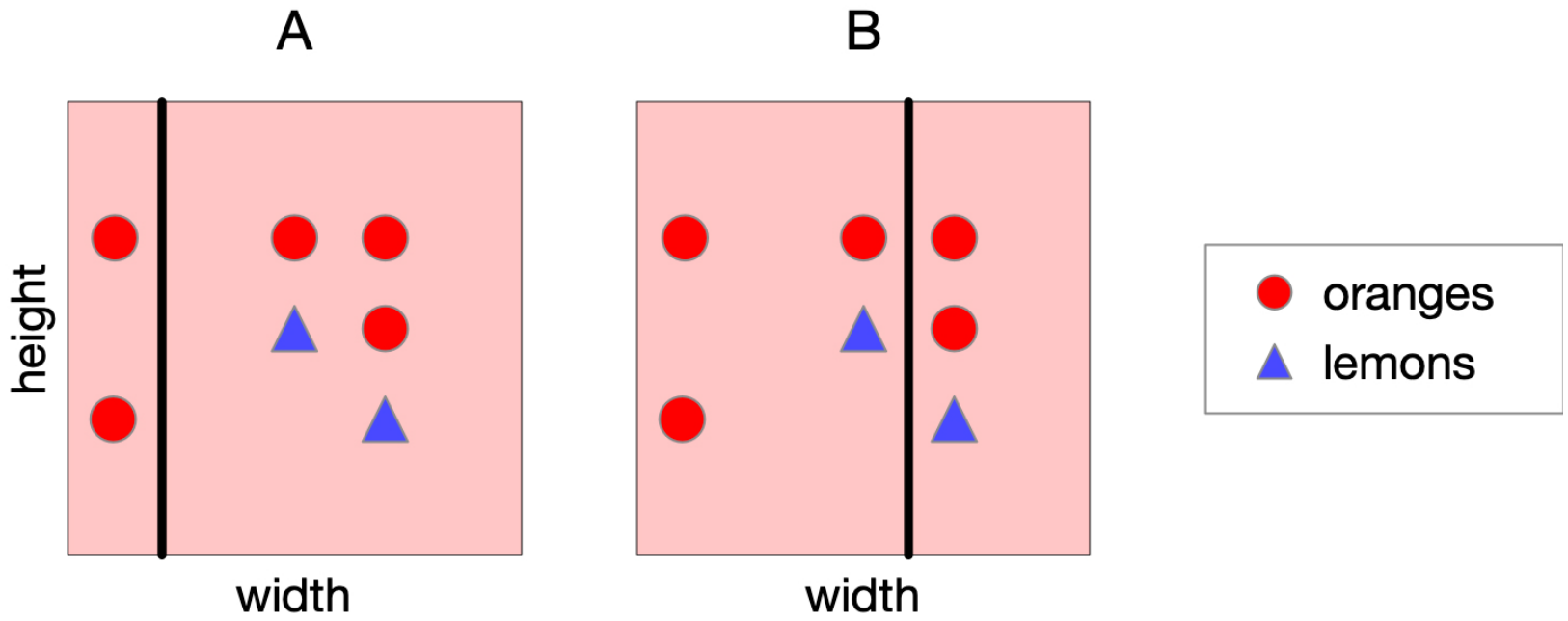
Choosing a Good Split

- Which is the best split?



Choosing a Good Split

- A feels like a better split, because the left-hand region is very certain about whether the fruit is an orange
- The faster we can assign a class label or reach a leaf node, the better
- Can we quantify this?



Choosing a Good Split

- How can we quantify uncertainty in prediction for a given leaf node?
  - If all examples in leaf have same class: good, low uncertainty
  - If each class has same amount of examples in leaf: bad, high uncertainty (half lemon, half orange)
- Idea: use counts at leaves to define probability distributions; use a probabilistic notion of uncertainty to decide splits
- A brief detour through information theory…

Entropy-Quantifying uncertainty

- You may have encountered the term entropy quantifying the state of chaos in chemical and physical systems
- In statistics, it is a property of a random variable
- The entropy of a discrete random variable is a number that quantifies the uncertainty inherent in its possible outcomes

We flip two different coins

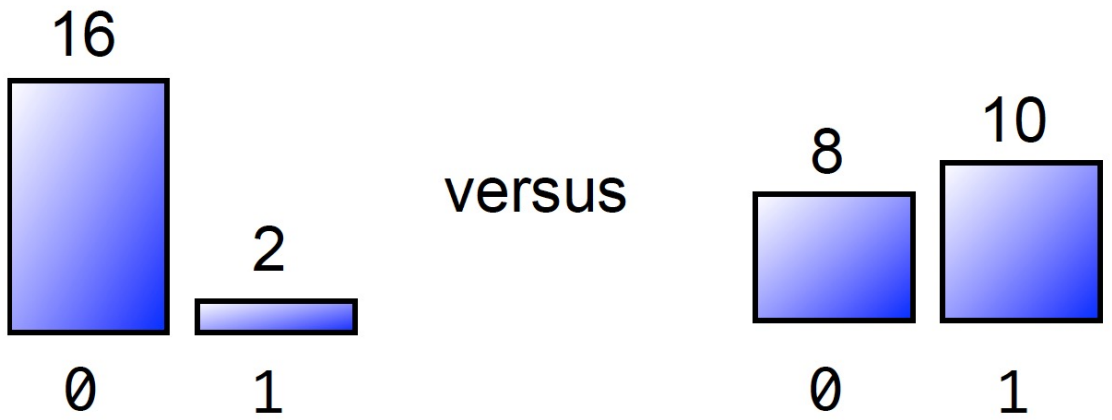
- Two biased coins. How biased are they?
- Each coin is a binary random variable with outcomes Heads (0) or Tails (1)

Sequence 1:

0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 ... ?

Sequence 2:

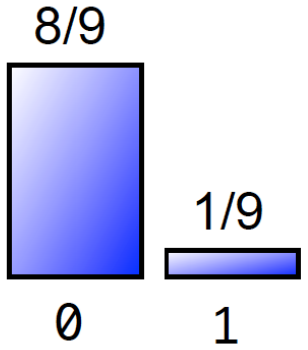
0 1 0 1 0 1 1 1 0 1 0 0 1 1 0 1 0 1 ... ?

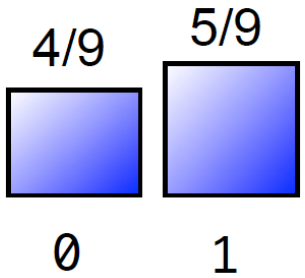


Quantifying Uncertainty

- The entropy of a loaded coin with probability p of heads is given by

$$-p\log_2(p) - (1-p)\log_2(1-p)$$





$$-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9} \approx \frac{1}{2}$$

$$-\frac{4}{9}\log_2\frac{4}{9} - \frac{5}{9}\log_2\frac{5}{9} \approx 0.99$$

- The coin whose outcomes are more certain has a lower entropy
- In the extreme case p = 0 or 1, we are certain of the outcome. So the uncertainty and the entropy is 0

Entropy

- The entropy of a discrete random variable Y is given by

$$H(Y) = -\sum_{y \in Y} p(y) \log_2 p(y)$$

- High entropy
  - Variable has a uniform like distribution over many outcomes
  - Flat histogram
  - Values sampled from it are less predictable
- Low entropy
  - Distribution is concentrated on only a few outcomes
  - Histogram is concentrated in a few areas
  - Values sampled from it are more predictable

Question

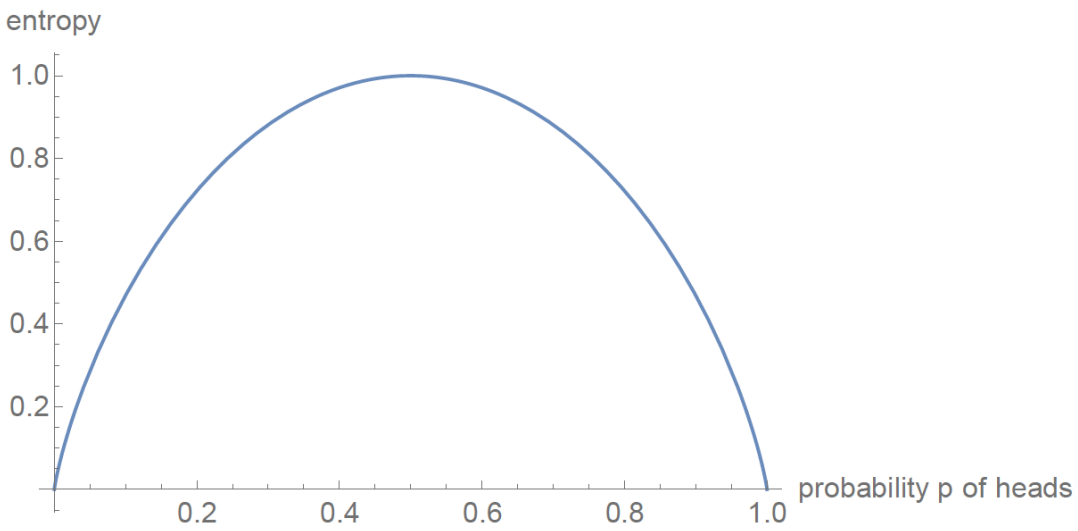
- What is the entropy of the distribution (0.5, 0.5)?
- (A) 0.2
- (B) 0.4
- (C) 0.6
- (D) 0.8
- (E) 1

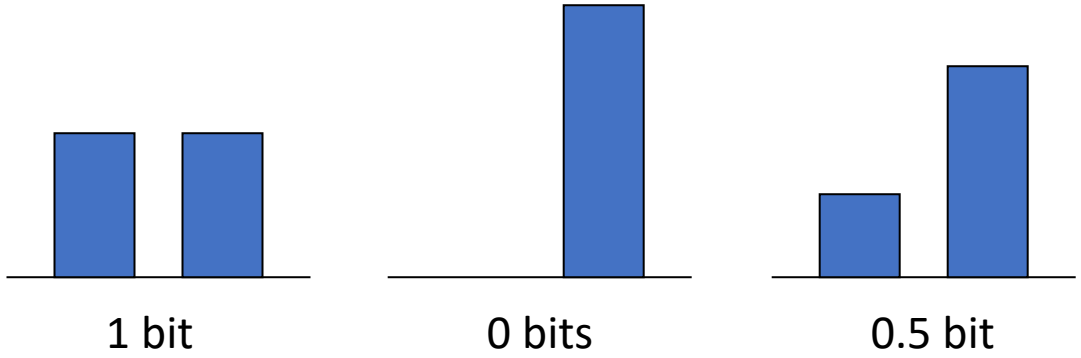
Question

- What is the entropy of the distribution (0.01, 0.99)?
- (A) 0.2
- (B) 0.4
- (C) 0.6
- (D) 0.8
- (E) 1

Quantifying Uncertainty

- Think of entropy as the expected information content of a random draw from a probability distribution
- Unit of entropy are bits
- A fair coin flip has 1 bit of entropy





Entropy of a Joint Distribution

- Example: X = {Rain, Not raining}, Y = {Cloudy, Not cloudy}

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

$$
\begin{aligned}
H(X,Y) &= -\sum_{x \in X} \sum_{y \in Y} p(x,y) \log_2 p(x,y) \\
&= -\frac{24}{100} \log_2 \frac{24}{100} - \frac{1}{100} \log_2 \frac{1}{100} - \frac{25}{100} \log_2 \frac{25}{100} - \frac{50}{100} \log_2 \frac{50}{100} \\
&\approx 1.56 \text{bits}
\end{aligned}
$$

Conditional Entropy

- Example: X = {Rain, Not raining}, Y = {Cloudy, Not cloudy}

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness Y, given that it is raining?

$$
\begin{aligned}
H(Y|X=x) &= -\sum_{y \in Y} p(y|x) \log_2 p(y|x) \\
&= -\frac{24}{25} \log_2 \frac{24}{25} - \frac{1}{25} \log_2 \frac{1}{25} \\
&\approx 0.24 \text{bits}
\end{aligned}
$$

Conditional Entropy

- Example: X = {Rain, Not raining}, Y = {Cloudy, Not cloudy}

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the expected entropy

$$
\begin{aligned}
H(Y|X) &= \mathbb{E}_x[H[Y|x]] \\
&= \sum_{x \in X} p(x) H(Y|X = x) \\
&= -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(y|x)
\end{aligned}
$$

Conditional Entropy

- H is always non-negative
- Chain rule: $H(X,Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$
- If X and Y independent, then X does not affect our uncertainty about Y: $H(Y|X) = H(Y)$
- Knowing Y makes our knowledge of Y certain: $H(Y|Y) = 0$
- Knowing X, we can only decrease uncertainty about Y: $H(Y|X) \leq H(Y)$

Information Gain

- How much more certain am I about whether it's cloudy if I'm told whether it is raining?
- My uncertainty in Y minus my expected uncertainty that would remain in Y after seeing X
- Amount of info obtained about one random variable by observing the other random variable
- Information gain IG(Y | X) in Y due to X, or the mutual information of Y and X

$$IG(Y|X) = H(Y) - H(Y|X)$$

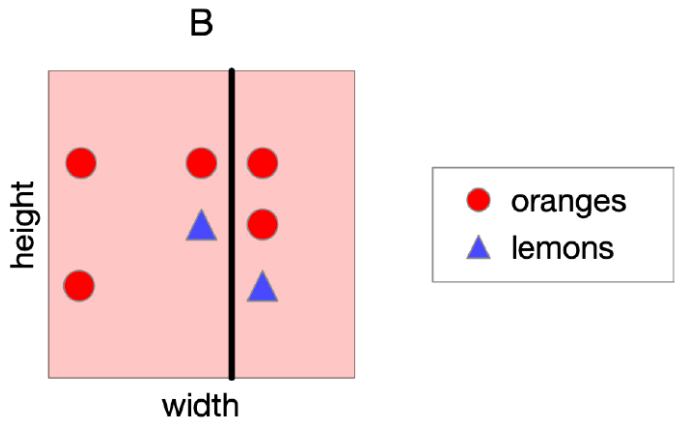
- If X is completely uninformative about Y $IG(Y|X) = 0$
- If X is completely informative about Y $IG(Y|X) = H(Y)$

Revisiting Our Original Example

- Information gain measures the informativeness of a variable, which is exactly what we desire in a decision tree split
- The information gain of a split: how much information about the class label Y is gained by knowing which side of a split you're on

Information Gain of Split B

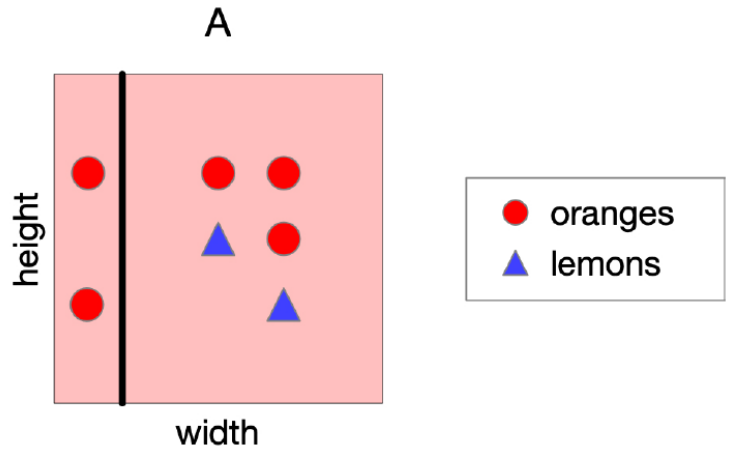
- What is the information gain of split B?



- Entropy of class outcome before split $H(Y) = -\frac{2}{7}\log_2(\frac{2}{7}) - \frac{5}{7}\log_2(\frac{5}{7}) \approx 0.86$
- Conditional entropy of class outcome after split $H(Y|left) \approx 0.81,\ H(Y|right) \approx 0.92$
- Information gain $IG(split) \approx 0.86 - (\frac{4}{7} \cdot 0.81 + \frac{3}{7} \cdot 0.92) \approx 0.006$

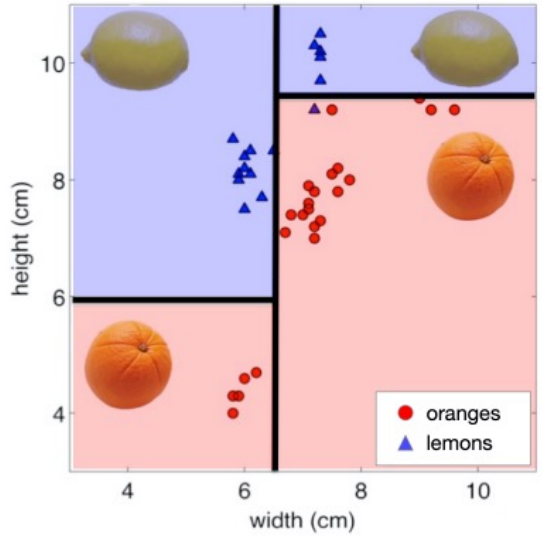
Information Gain of Split A

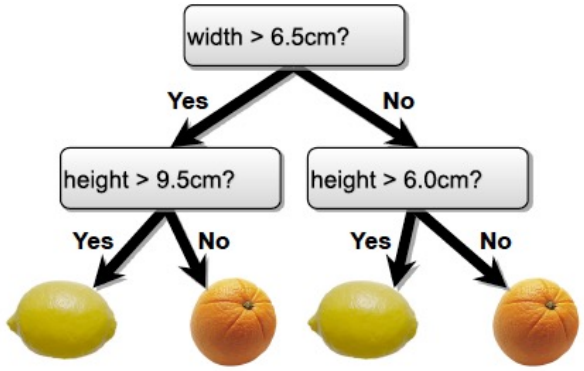
- What is the information gain of split A?



- Entropy of class outcome before split $H(Y) = -\frac{2}{7}\log_2(\frac{2}{7}) - \frac{5}{7}\log_2(\frac{5}{7}) \approx 0.86$
- Conditional entropy of class outcome after split $H(Y|left) = 0,\ H(Y|right) \approx 0.97$
- Information gain $IG(split) \approx 0.86 - (\frac{2}{7} \cdot 0 + \frac{5}{7} \cdot 0.97) \approx 0.17!!$

Constructing Decision Trees





- At each level, one must choose:
  - 1. Which feature to split
  - 2. Possibly where to split it
- Choose them based on how much information we would gain from the decision (choose feature that gives the highest gain)

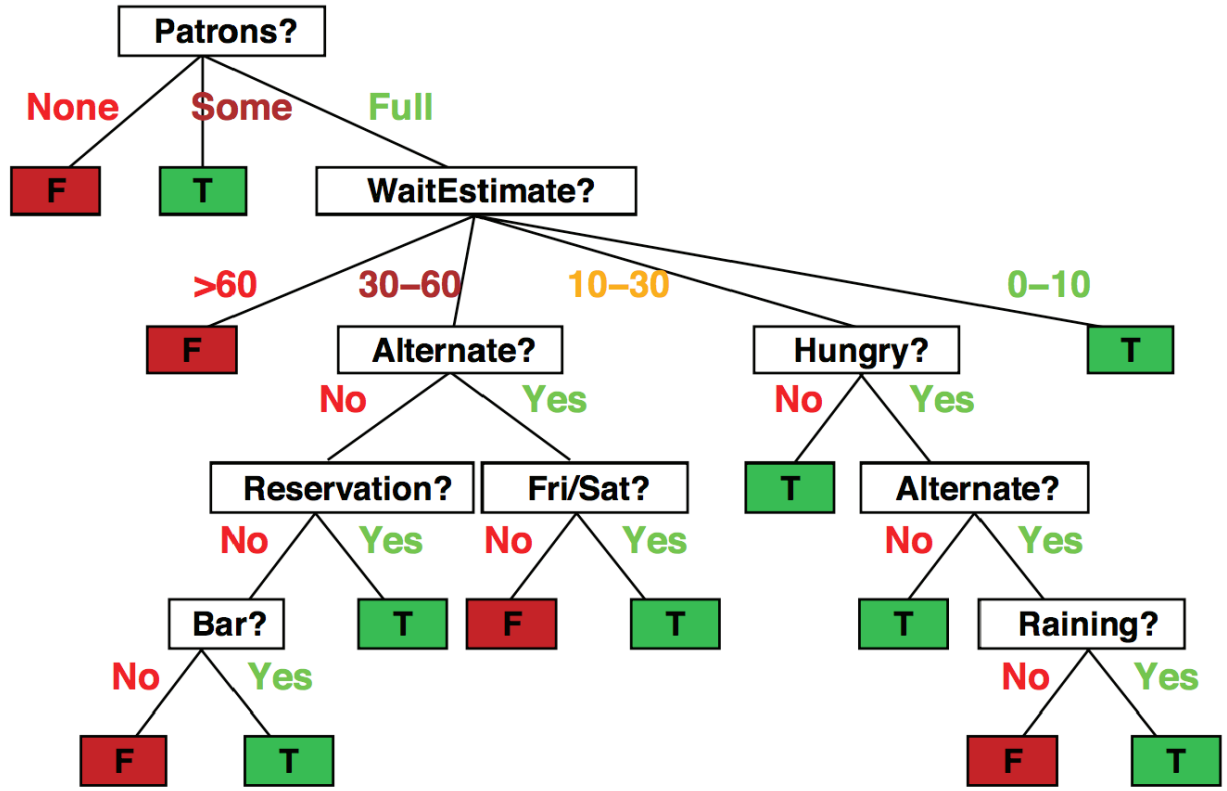
Information Gain of testing a feature

- The feature has k values v_1, v_2, …, v_k
- Before testing the feature, we have p positive and n negative examples
  - I(p/p+n, n/p+n)
- After testing the feature, for each value v_i, we have p_i positive and n_i negative examples
  - Sum up (p_i+n_i)/(p+n) I(p_i, n_i)
- What is the information gain?

Question

- What is the entropy of the examples before we select a feature for the root node of the tree

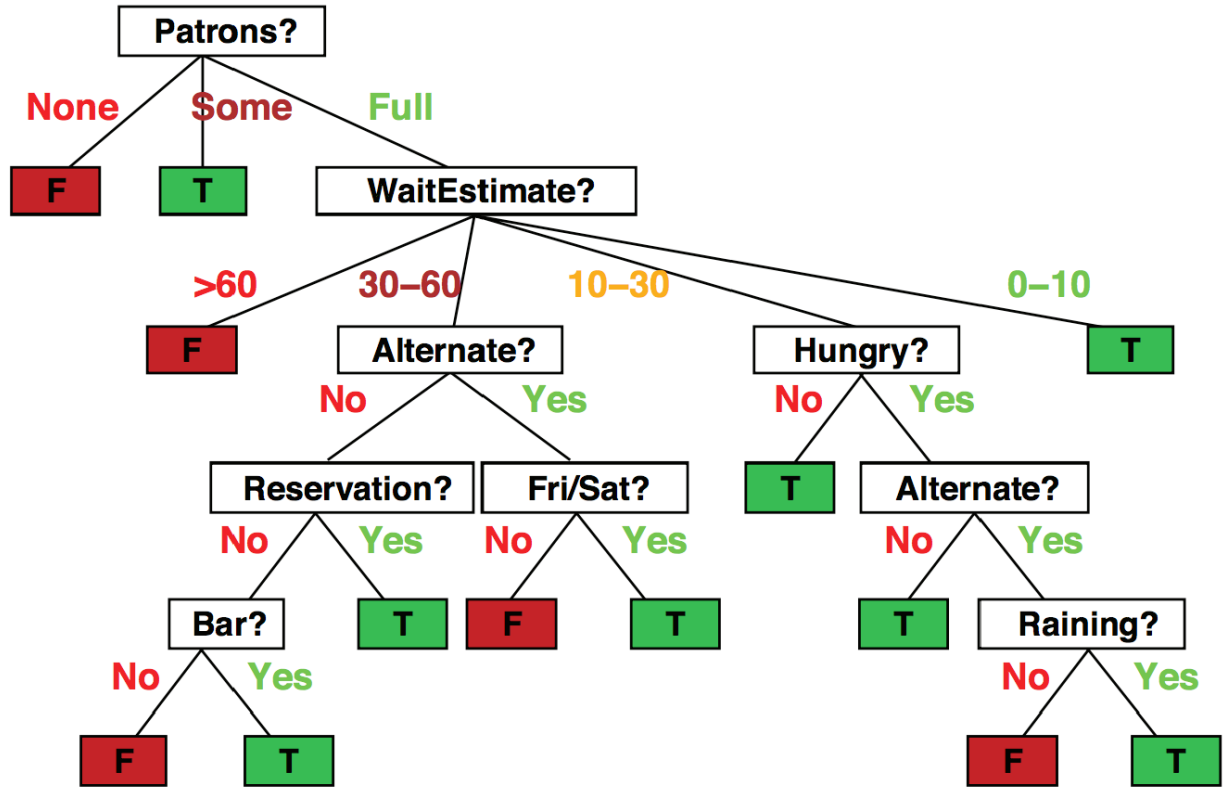
Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 =$ *Yes*
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 =$ *No*
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 =$ *Yes*
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 =$ *Yes*
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 =$ *No*
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 =$ *Yes*
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 =$ *No*
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 =$ *Yes*
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 =$ *No*
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} =$ *No*
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} =$ *No*
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} =$ *Yes*



Question

- What is the information gain if we select Patrons for the root node of the tree

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 =$ *Yes*
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 =$ *No*
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 =$ *Yes*
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 =$ *Yes*
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 =$ *No*
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 =$ *Yes*
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 =$ *No*
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 =$ *Yes*
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 =$ *No*
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} =$ *No*
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} =$ *No*
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} =$ *Yes*



Decision Tree Construction Algorithm

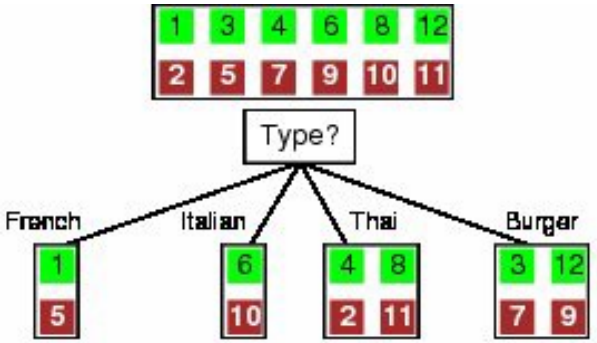
- Simple, greedy, recursive approach, builds up tree node-by-node
  - 1. pick a feature to split at a non-terminal node
  - 2. split examples into groups based on feature value
  - 3. for each group:
    - if no example – return majority from parent
    - Else if all example in same class – return class
    - Else loop to step 1
- Terminates when all leaves contain only examples in the same class or are empty

Back to our example

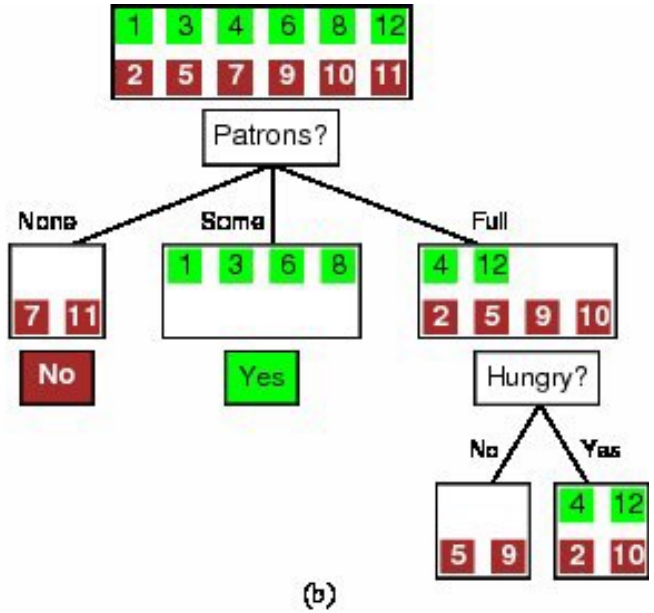
Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 =$ *Yes*
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 =$ *No*
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 =$ *Yes*
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 =$ *Yes*
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 =$ *No*
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 =$ *Yes*
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 =$ *No*
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 =$ *Yes*
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 =$ *No*
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} =$ *No*
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} =$ *No*
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} =$ *Yes*

Feature Selection

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$



(a)



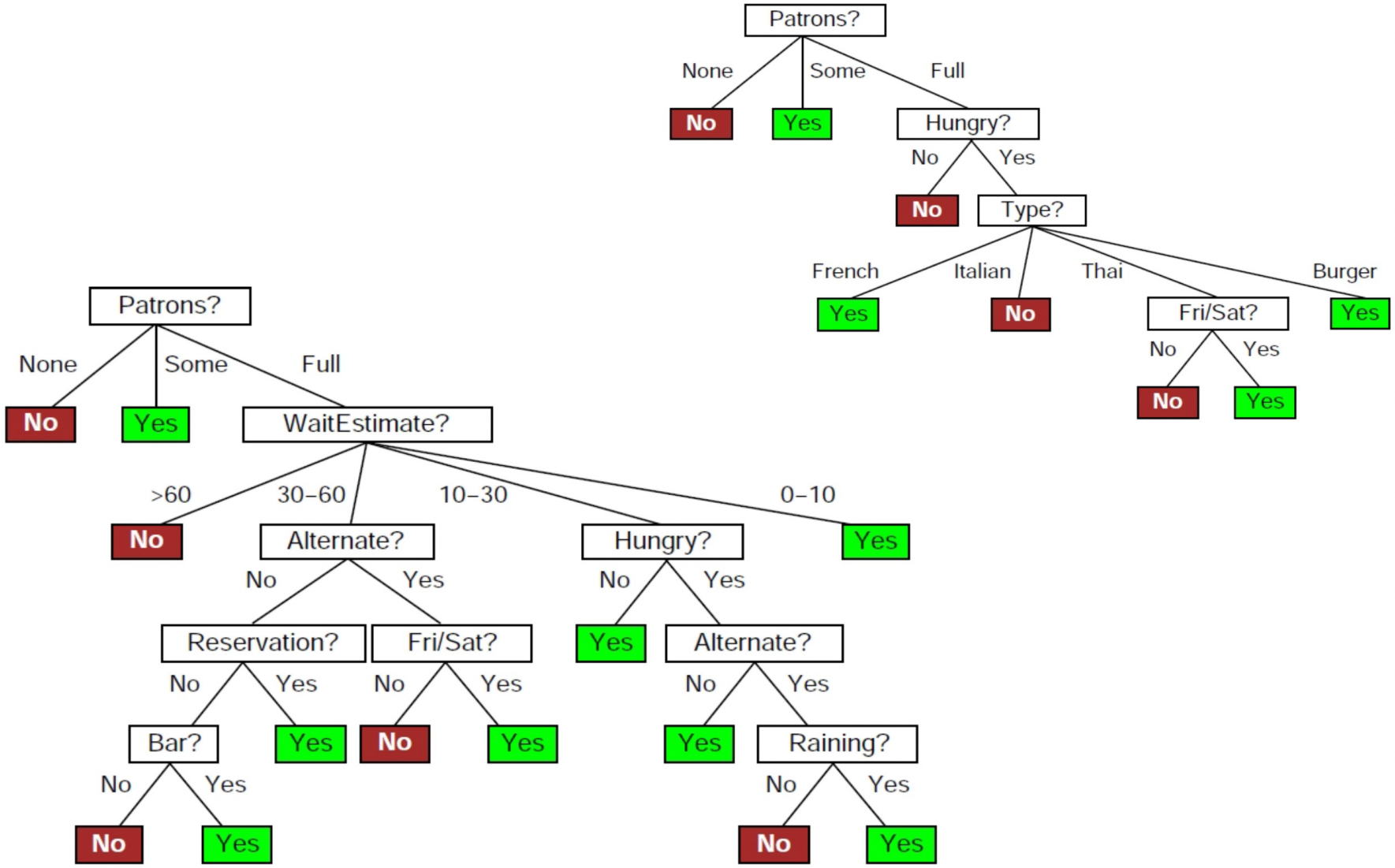
(b)

$$IG(Y) = H(Y) - H(Y|X)$$

$$IG(type) = 1 - \left[\frac{2}{12}H(Y|\text{Fr.}) + \frac{2}{12}H(Y|\text{It.}) + \frac{4}{12}H(Y|\text{Thai}) + \frac{4}{12}H(Y|\text{Bur.})\right] = 0$$

$$IG(Patrons) = 1 - \left[\frac{2}{12}H(0,1) + \frac{4}{12}H(1,0) + \frac{6}{12}H(\frac{2}{6},\frac{4}{6})\right] \approx 0.541$$

Which tree is better?



What makes a good tree?

- Not too small: need to handle important but possibly subtle distinctions in data
- Not too big:
  - Computational efficiency (avoid redundant, spurious attributes)
  - Avoid over-fitting training examples
  - Human interpretability
- Occam's Razor: find the simplest hypothesis that fits the observations
- We desire small trees with informative nodes near the root

Decision Tree Miscellany

- Problems:
  - Too big of a tree can overfit the data
  - Greedy algorithms don't necessarily yield the global optimum
- Handling continuous attributes
  - Split based on a threshold, chosen to maximize information gain

Example - Tennis

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Example

- What is the entropy of the examples before we select a feature for the root node of the tree
- (A) 0.54
- (B) 0.64
- (C) 0.94

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Example

- What is the information gain if we select Outlook as the root node?
- (A) 0.237
- (B) 0.247
- (C) 0.257

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Example

- What is the information gain if we select Humidity as the root node?
- (A) 0.151
- (B) 0.251
- (C) 0.351

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Example

- Root node
  - IG(Outlook) = 0.247, IG(Humidity) = 0.151, IG(Temp) = 0.029, IG(Wind) = 0.048
- Other nodes

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No