Machine Learning & Decision Trees

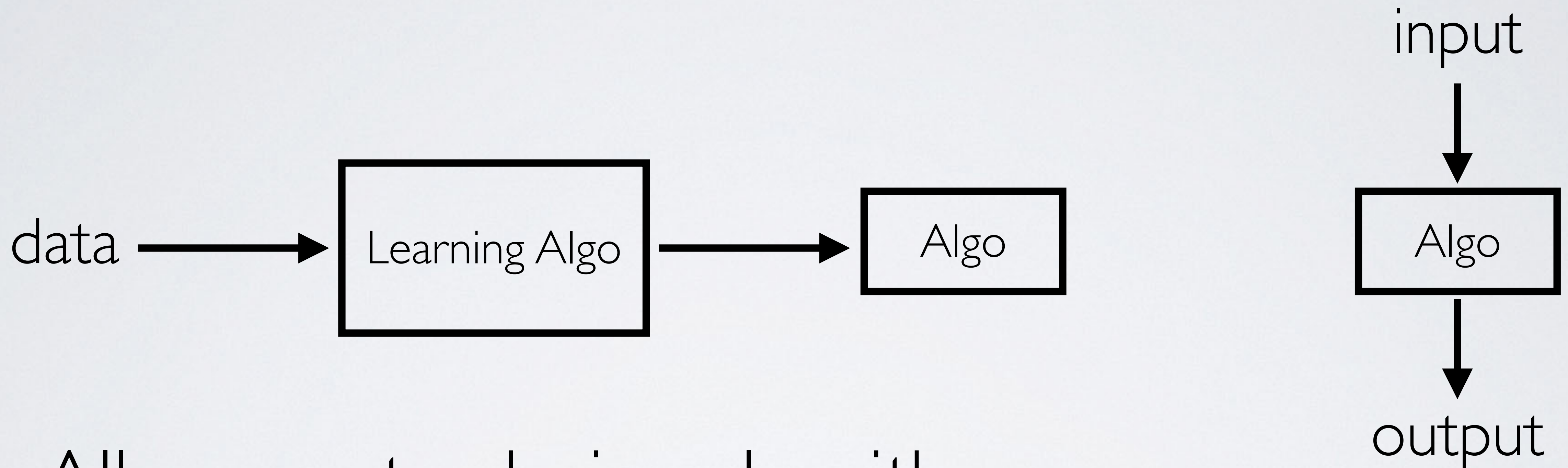
CS16: Introduction to Data Structures & Algorithms
Spring 2020

Outline

- Motivation
- Supervised learning
- Decision Trees
- ML Bias

Machine Learning

- Algorithms that use data to design algorithms



- Allows us to design algorithms
  - that predict the future (e.g., picking stocks)
  - even when we don't know how (e.g., facial recognition)

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

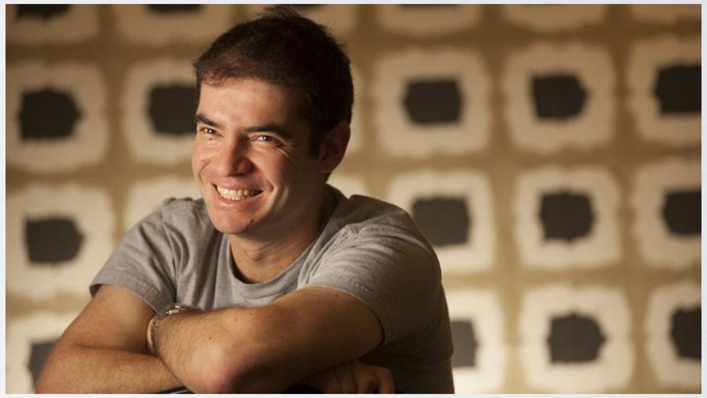
We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

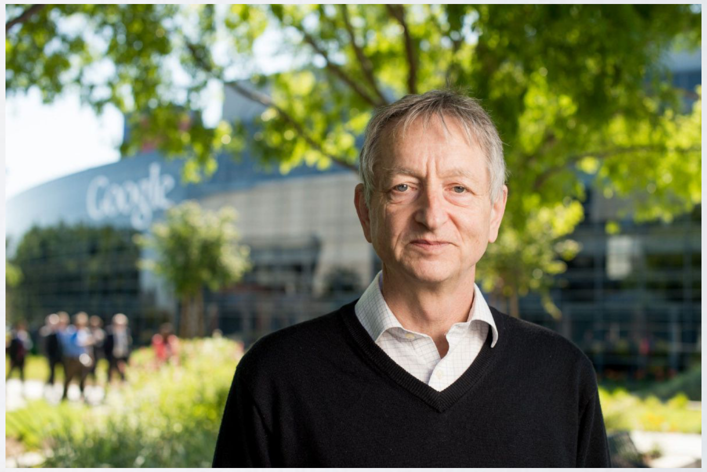
1 Introduction

Current approaches to object recognition make essential use of machine learning methods. To improve their performance, we can collect larger datasets, learn more powerful models, and use better techniques for preventing overfitting. Until recently, datasets of labeled images were relatively small — on the order of tens of thousands of images (e.g., NORB [16], Caltech-101/256 [8, 9], and CIFAR-10/100 [12]). Simple recognition tasks can be solved quite well with datasets of this size, especially if they are augmented with label-preserving transformations. For example, the current-best error rate on the MNIST digit-recognition task (<0.3%) approaches human performance [4]. But objects in realistic settings exhibit considerable variability, so to learn to recognize them it is necessary to use much larger training sets. And indeed, the shortcomings of small image datasets have been widely recognized (e.g., Pinto et al. [21]), but it has only recently become possible to collect labeled datasets with millions of images. The new larger datasets include LabelMe [23], which consists of hundreds of thousands of fully-segmented images, and ImageNet [6], which consists of over 15 million labeled high-resolution images in over 22,000 categories.

To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task means that this problem cannot be specified even by a dataset as large as ImageNet, so our model should also have lots of prior knowledge to compensate for all the data we don't have. Convolutional neural networks (CNNs) constitute one such class of models [16, 11, 13, 18, 15, 22, 26]. Their capacity can be controlled by varying their depth and breadth, and they also make strong and mostly correct assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies). Thus, compared to standard feedforward neural networks with similarly-sized layers, CNNs have much fewer connections and parameters and so they are easier to train, while their theoretically-best performance is likely to be only slightly worse.

UNIVERSITY OF TORONTO





CS 147

Applications of ML

- Agriculture
- Astronomy
- Bioinformatics
- Classifying DNA
- Computer Vision
- Finance
- Linguistics
- Medical diagnostics
- Insurance
- Economics
- Advertising
- Self-driving cars
- Recommendation systems (e.g., Netflix)
- Search engines
- Translations
- Robotics
- Risk assessment
- Drug discovery
- Fraud discovery
- Computational Anatomy

Classes of ML

- Supervised learning
  - learn to make accurate predictions from training data
- Unsupervised learning
  - find patterns in data without training data
- Reinforcement learning
  - improve performance with positive and negative feedback

Supervised Learning

- Make accurate predictions/classifications
  - Is this *email* spam?
  - Will the *snowstorm* cancel class?
  - Will this *flight* be delayed?
  - Will this *candidate* win the next election?
- How can our algorithm predict the future?
  - We train it using "training data" which are past examples
  - Examples of emails classified as spam and of emails classified as non-spam
  - Examples of snowstorms that have lead to cancelations and of snowstorms that have not
  - Examples of flights that have been delayed and of flights that have left on time
  - Examples of candidates that won and of candidates that have lost

Supervised Learning

- Training data is a collection of *examples*
  - An example includes an **input** and its **classification**
  - *inputs*: flights, snowstorms, candidates, …
  - *classifications*: delayed/non-delayed, canceled/not canceled, win/lose
- But how do we represent *inputs* for our algorithm?
  - What is a student? what is a flight? what is an email?
  - We have to choose **attributes** that describe the *inputs*
    - flight is represented by: source, destination, airline, number of passengers, …
    - snowstorm is represented by: duration, expected inches, winds, …
    - candidate is represented by: district, political affiliation, experience, …

Example: Waiting for a Table

- Design algorithm that predicts if patron will wait for a table
- What are the *inputs*?
  - the "context" of the patron's decision
- What are the *attributes* of this context?
  - is patron hungry? is the line long?

Example: Waiting for a Table?

- Input attributes
  - A_1: Alternatives = {Yes, No}
  - A_2: Bar = {Yes, No}
  - A_3: Fri/Sat = {Yes, No}
  - A_4: Hungry = {Yes, No}
  - A_5: Patrons = {None, Some, Full}
  - A_6: Price = {$, $$, $$$}
  - A_7: Raining = {Yes, No}
  - A_8: Reservation = {Yes, No}
  - A_9: Type = {French, Italian, Thai, Burger}
  - A_{10}: Wait = {10-30, 30-60, >60}
- Classification: {Yes, No}

Training Data

Ex.	Input Attributes										Classif.
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
4	Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30	Yes
5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No
6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No
10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
11	No	No	No	No	None	$	No	No	Thai	0-10	No
12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes

S. Russel & P. Norvig. Artificial Intelligence - A Modern Approac

Supervised Learning

- Classification
  - If classifications are from a finite set
  - ex: spam/not spam, delayed/not delayed
- Regression
  - If classifications are real numbers
  - ex: temperature

Outline

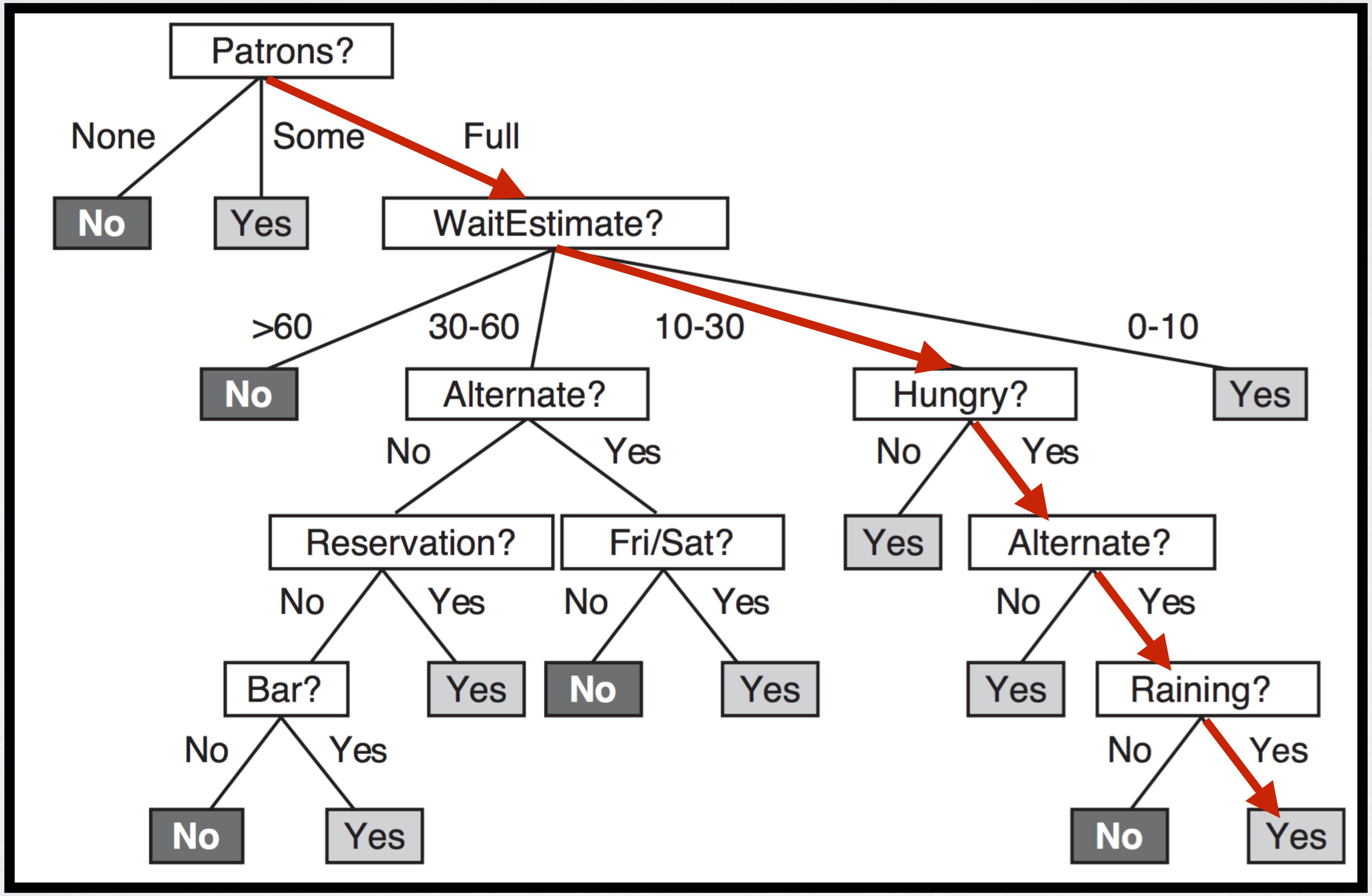
- Motivation
- Supervised learning
- Decision Trees
- Algorithmic Bias

Decision Trees

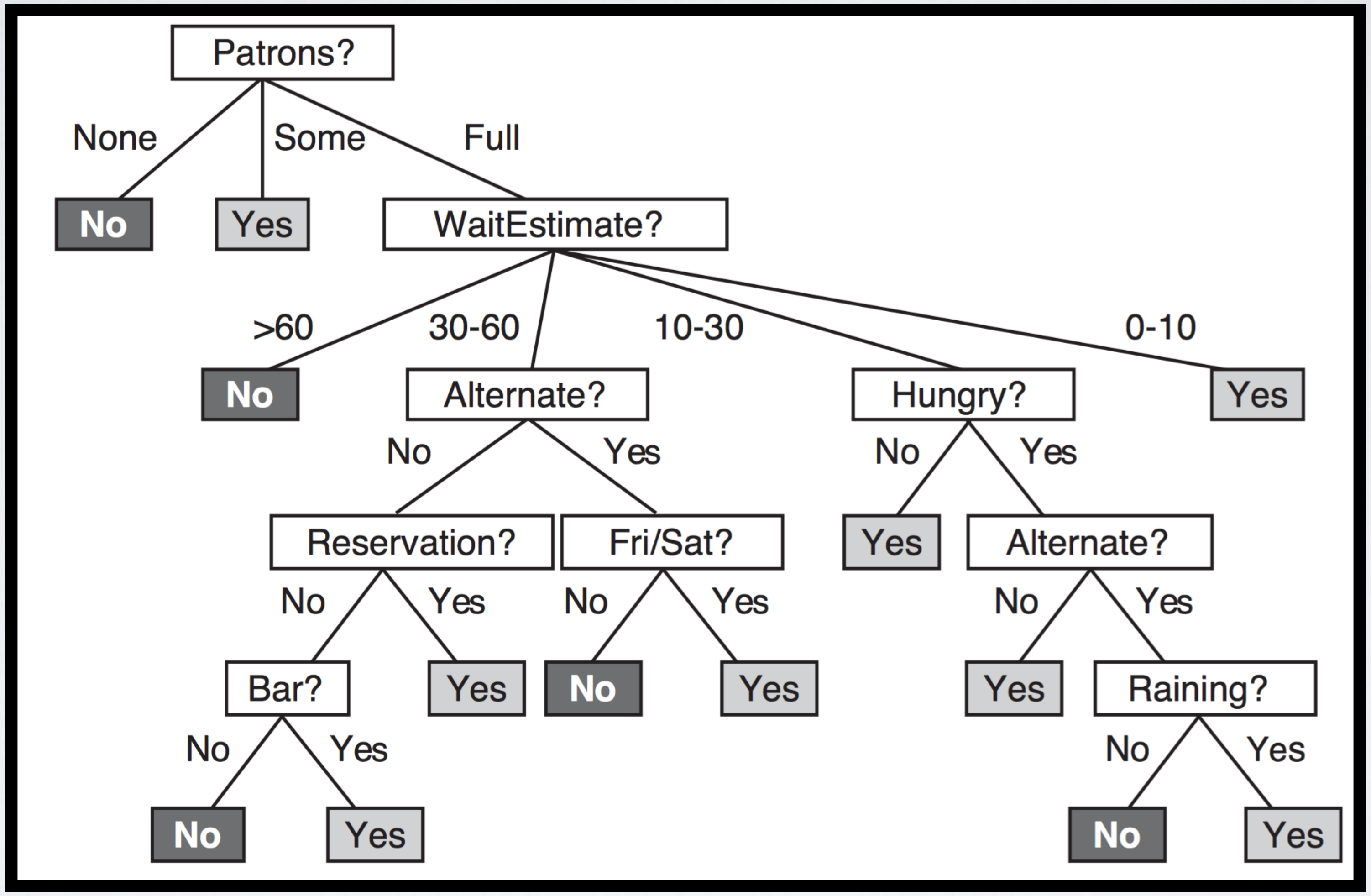
- A decision tree maps
  - *inputs* represented by *attributes*…
  - …to a *classification*
- Examples
  - `snowstorm_dt(12h,8”,strong winds)` returns `Yes`
  - `flight_dt(DL,PVD,Paris,night,no_storm,…)` returns `No`
  - `restaurant_dt(estimate,hungry,patrons,…)` returns `No`

Decision Tree Example

Input Attributes									
Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est
Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30



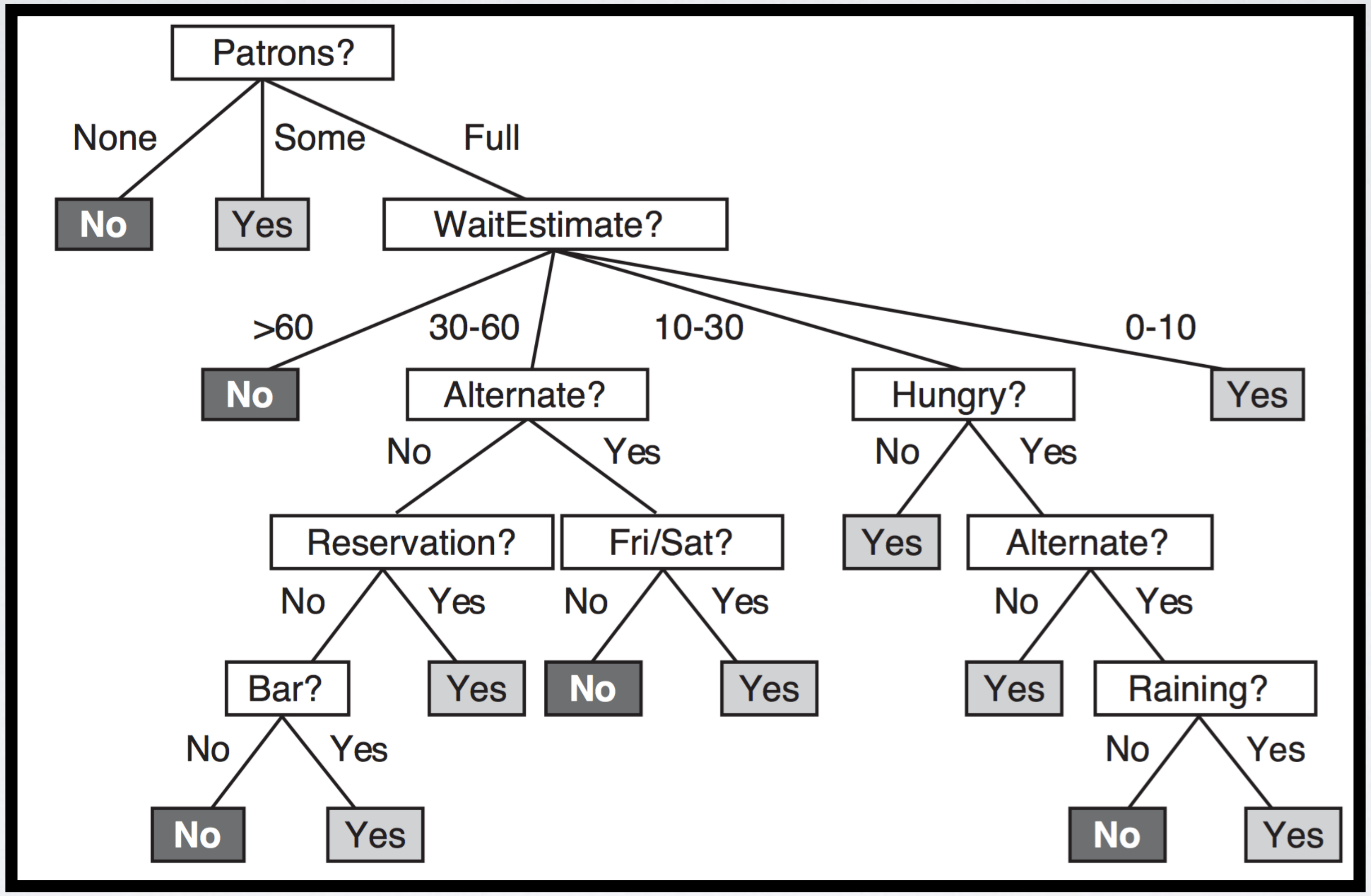
Decision Tree Example



Activity #1

2 min

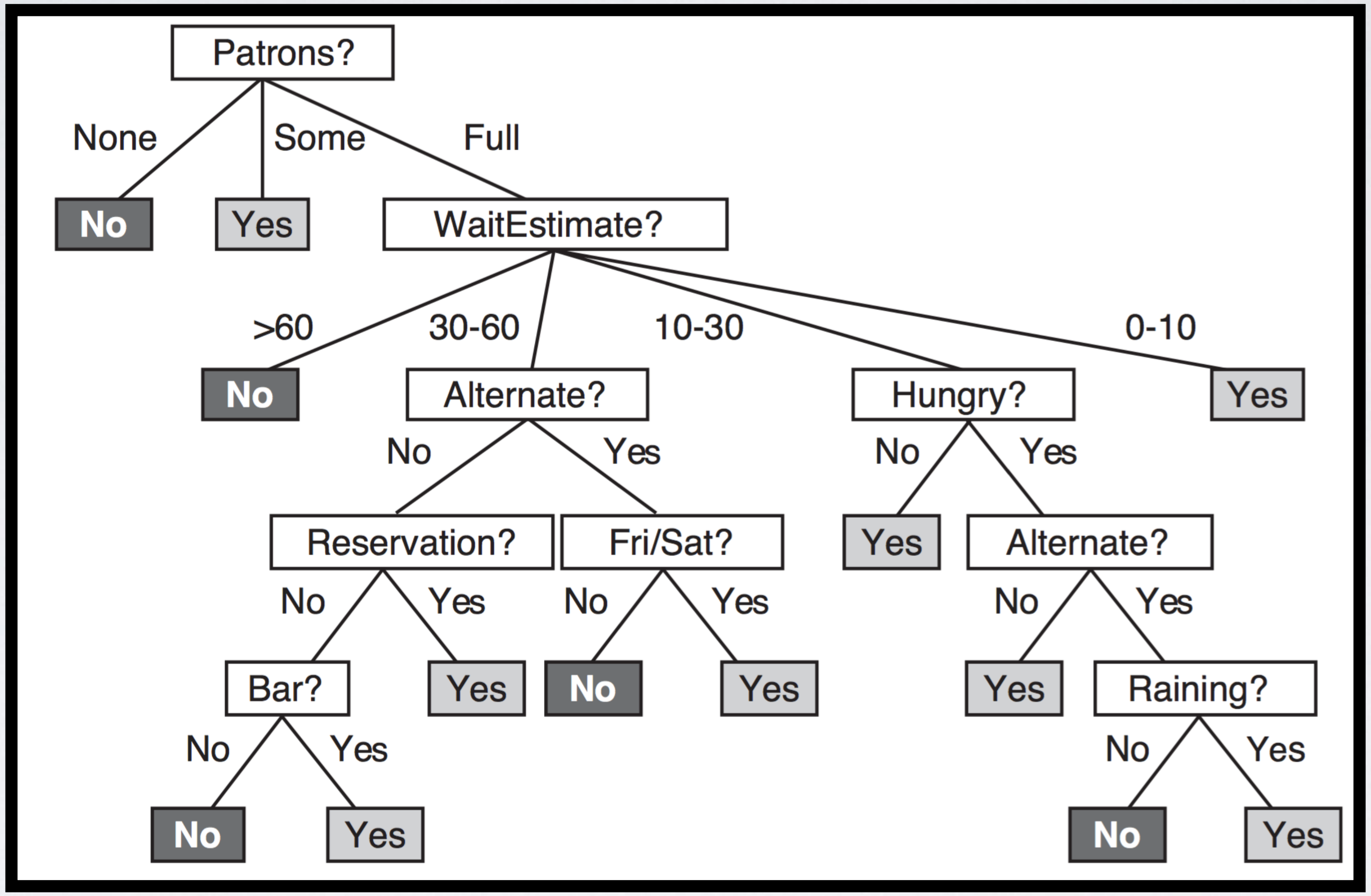
Decision Tree Example



Activity #1

1 min

Decision Tree Example

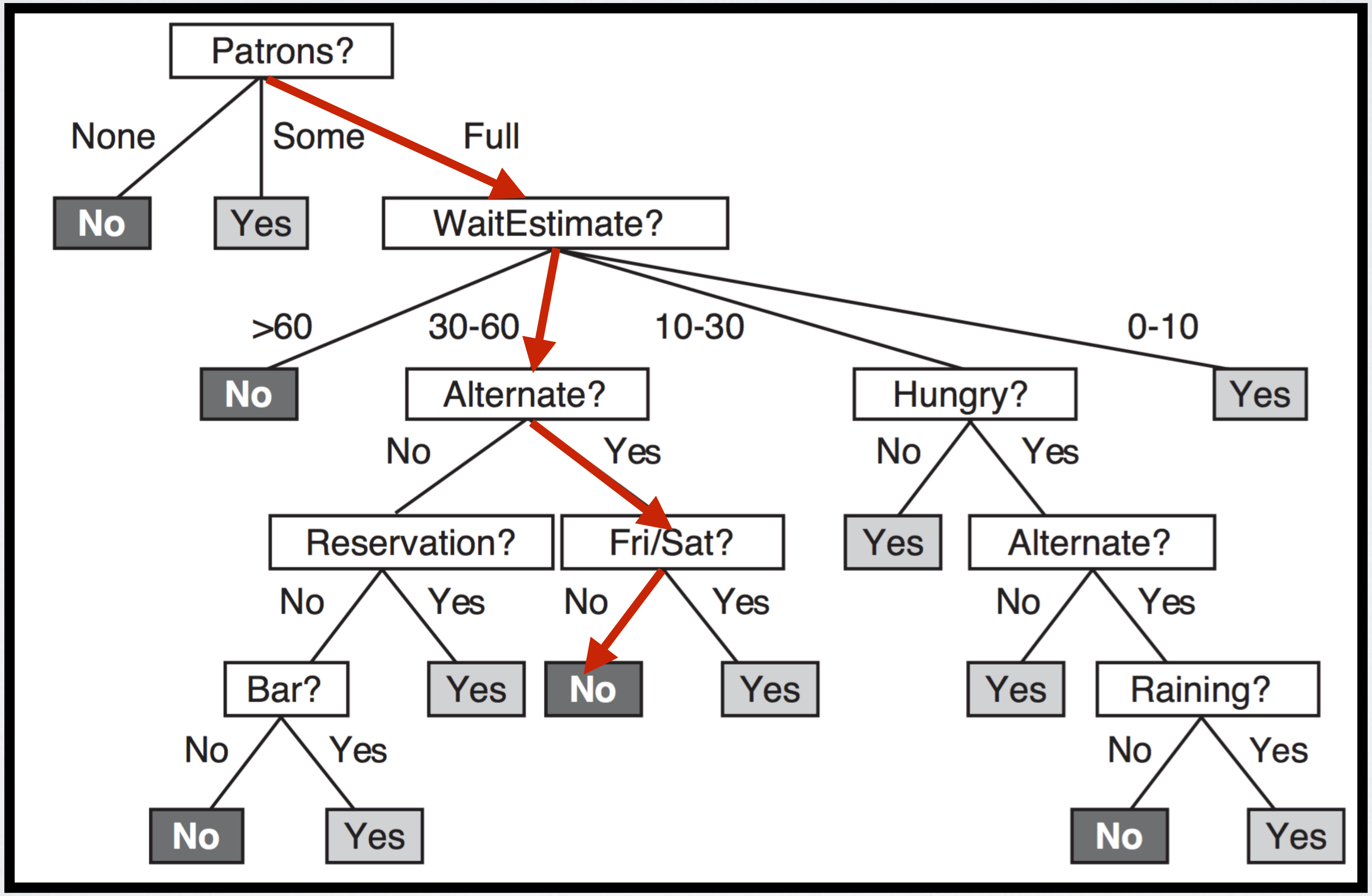


Activity #1

0 min

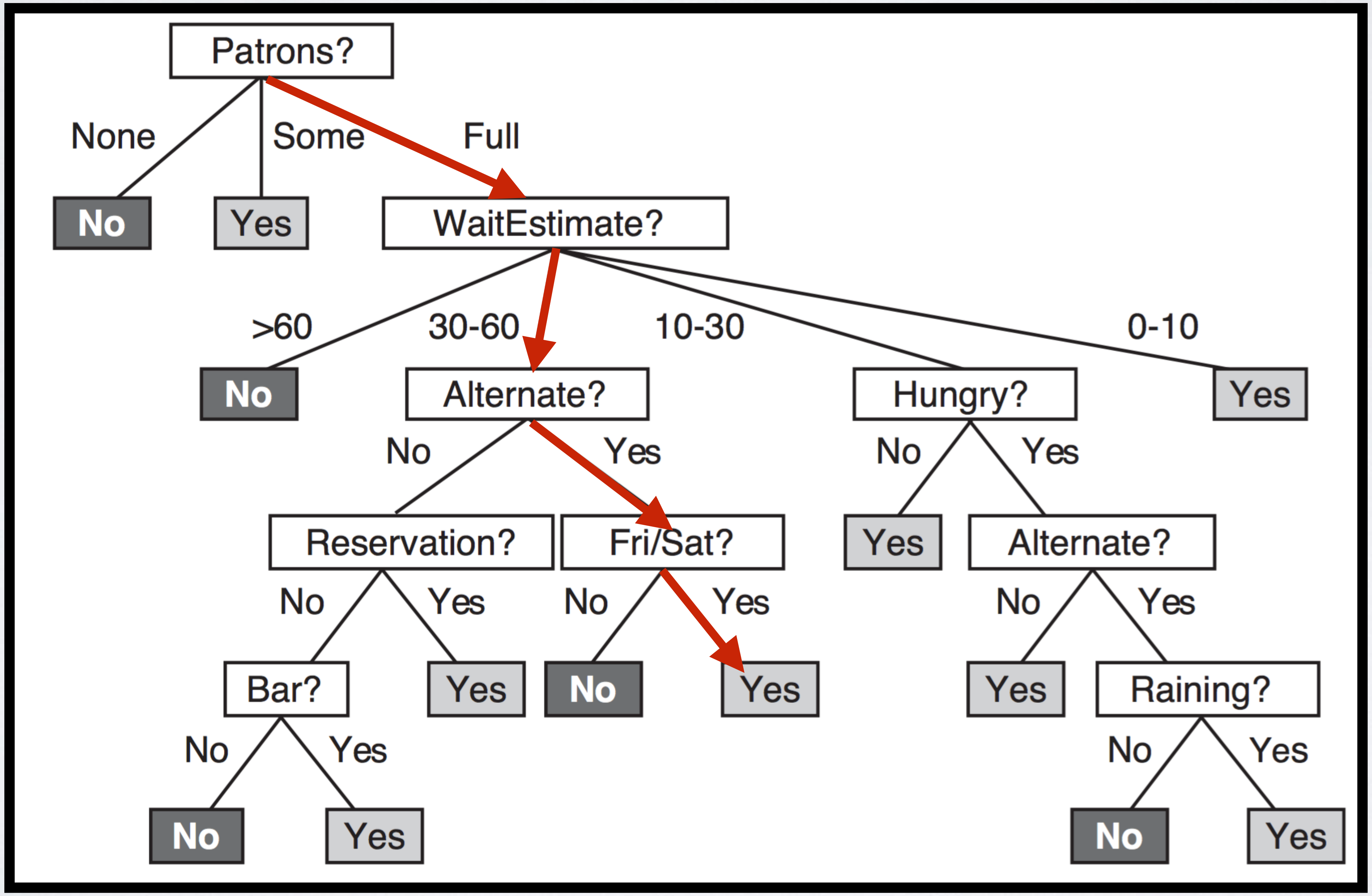
Decision Tree Example

Input Attributes									
Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est
Yes	No	No	Yes	Full	$	No	No	Thai	30-60



Decision Tree Example

Input Attributes									
Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est
Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60

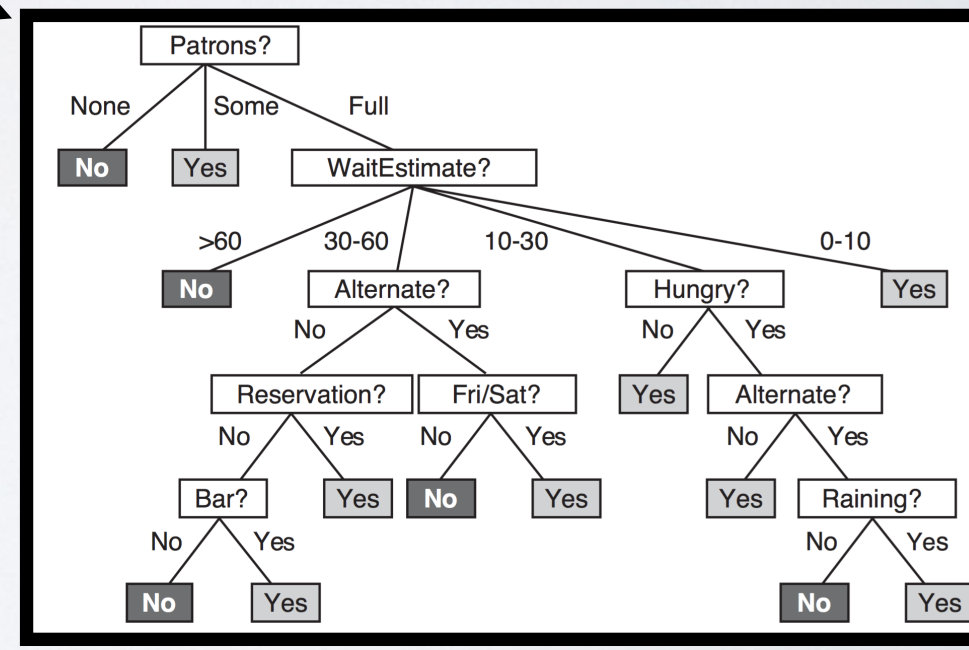


Our Goal: Learning a Decision Tree

Ex.	Input Attributes										Classif.
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
4	Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30	Yes
5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No
6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No
10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
11	No	No	No	No	None	$	No	No	Thai	0-10	No
12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes

Training Data

Learn



Decision Tree

What is a Good Decision Tree?

- Consistent with training data
  - classifies training examples correctly
- Performs well on future examples
  - classifies future inputs correctly
- As small as possible
  - Efficient classification
- How can we find a small decision tree?
  - there are $\Omega(2^{2^n})$ possible decision trees
  - so brute force is not possible

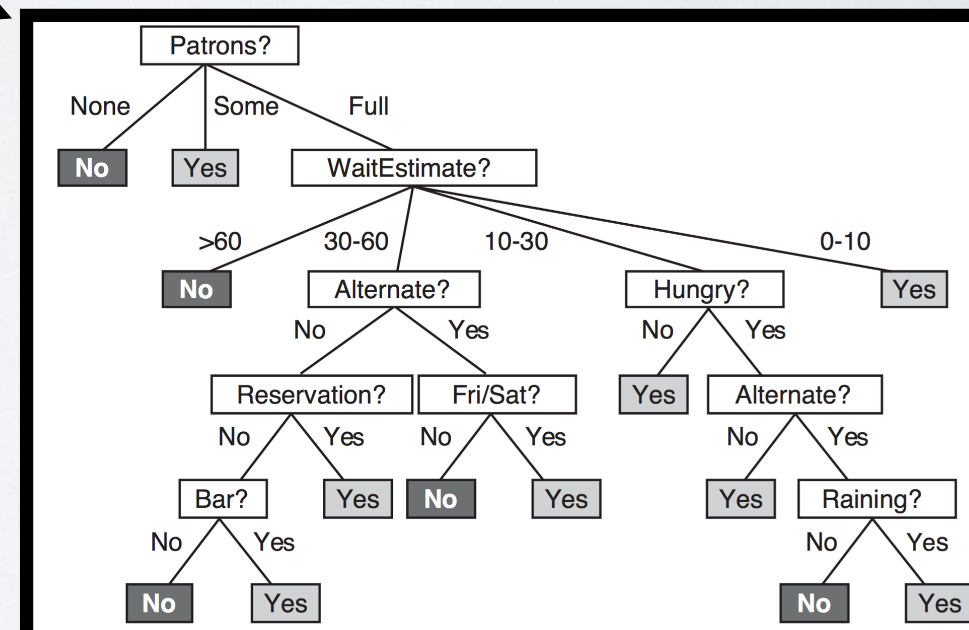
Iterative Dichotomizer 3 (**ID3**) Algorithm

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*	$y_{12} = Yes$

Data

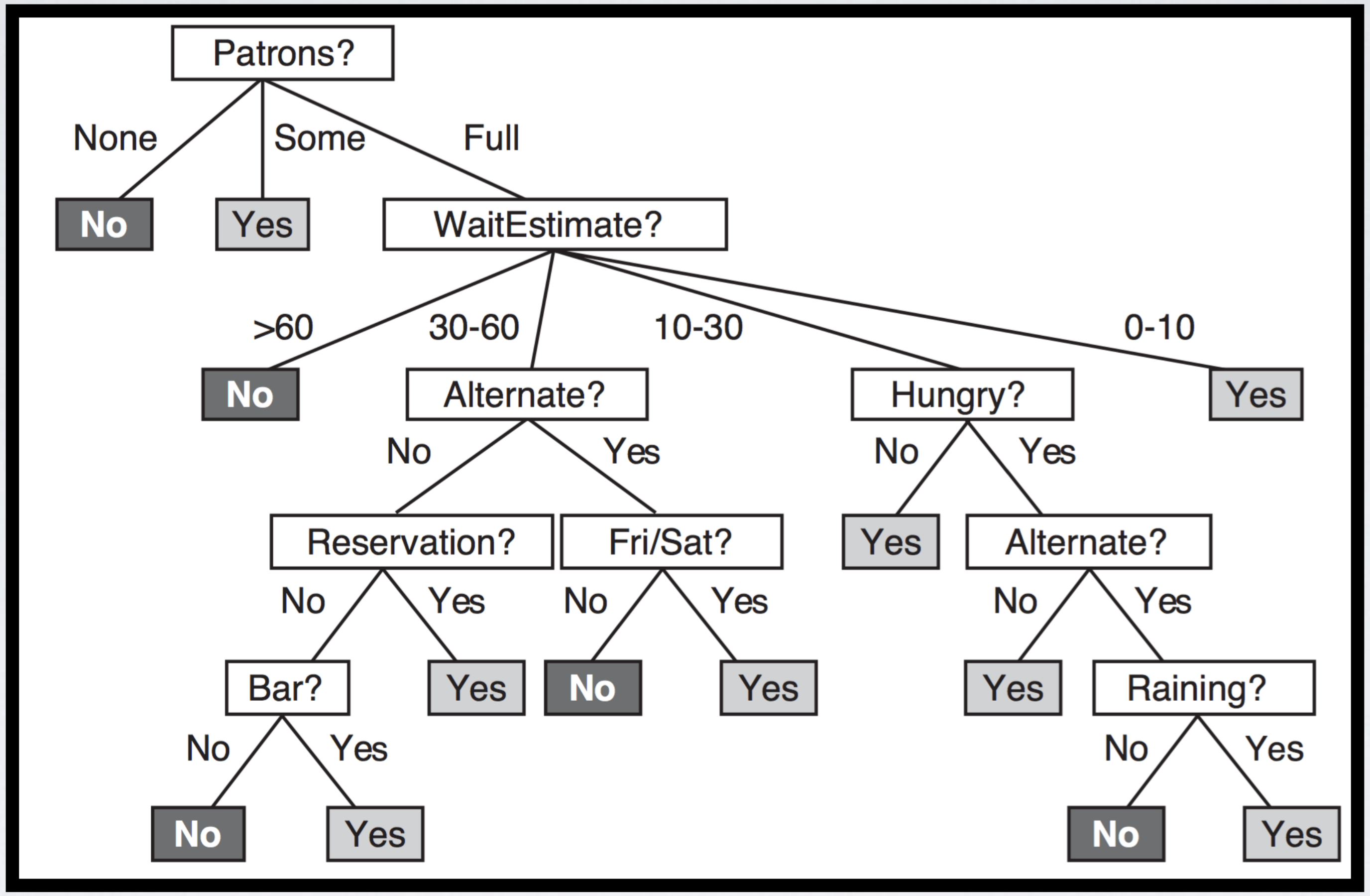
Ross Quinlan

ID3



(Learned) Decision Tree

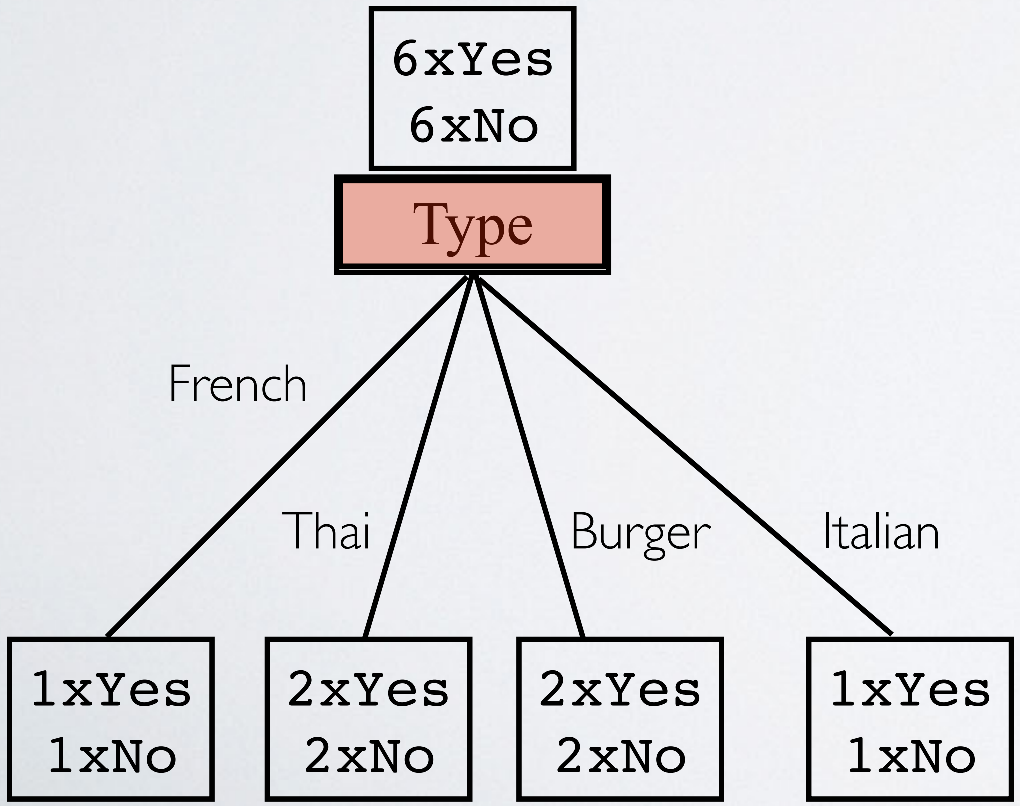
ID3



- Starting at root
  - node is either an *attribute* node or a *classification* node (leaf)
  - outgoing edges are labeled with attribute *values*
  - children are either a *classification* node or another *attribute* node
- Tree should be as small as possible

ID3

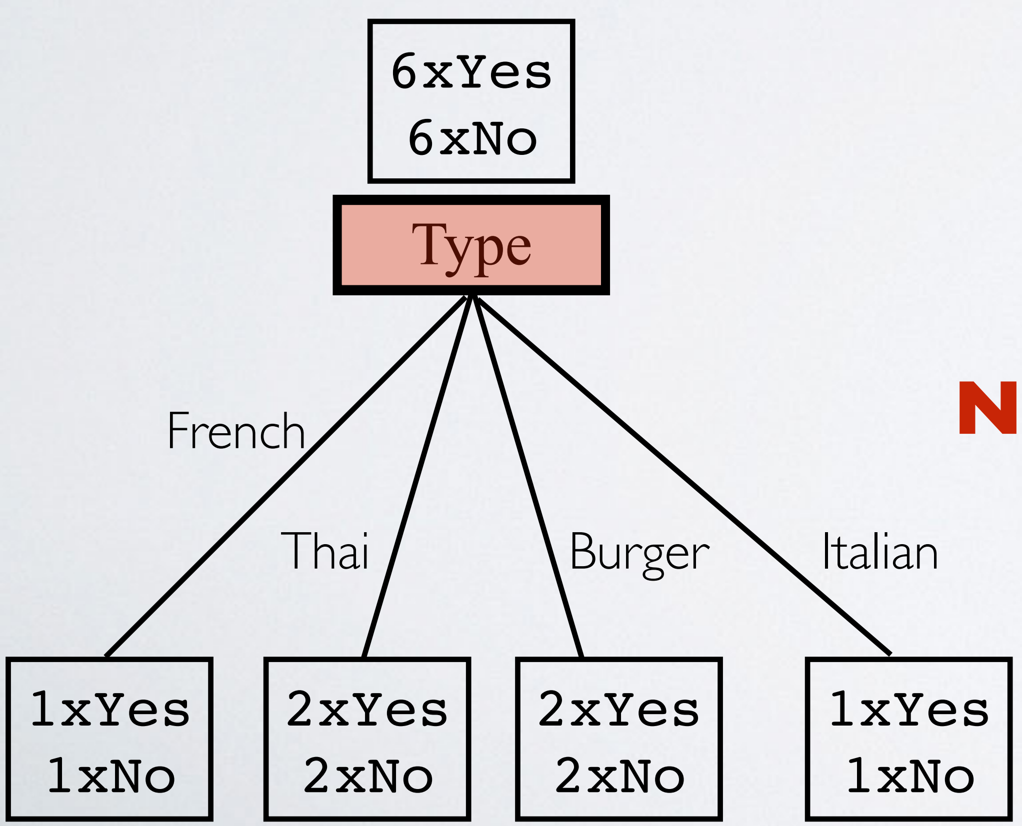
Ex.	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	Classif.
	Input Attributes										
1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
4	Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30	Yes
5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No
6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No
10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
11	No	No	No	No	None	$	No	No	Thai	0-10	No
12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes

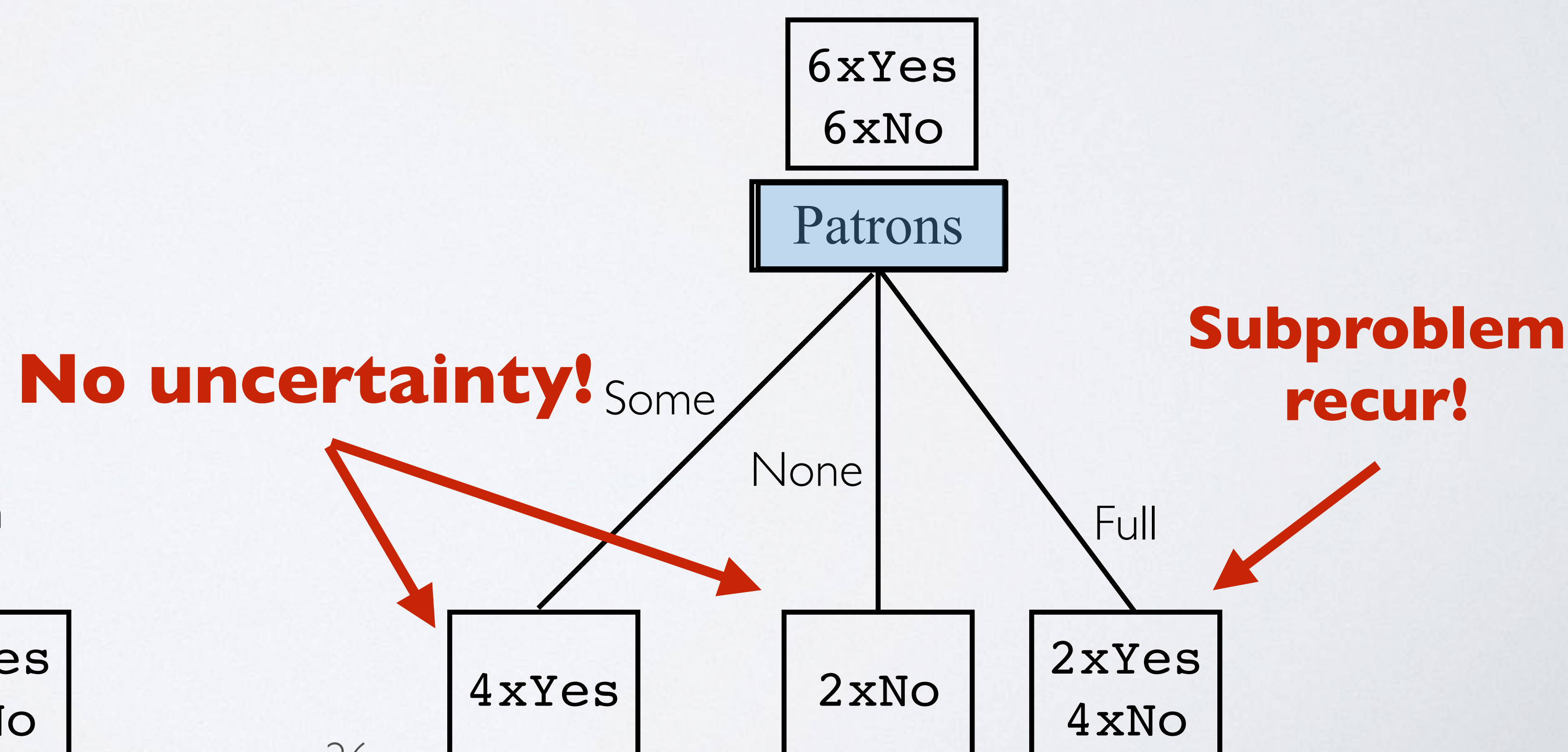


Uncertainty about whether we should wait or not

ID3

Ex.	Input Attributes										Classif.
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
4	Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30	Yes
5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No
6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No
10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
11	No	No	No	No	None	$	No	No	Thai	0-10	No
12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes





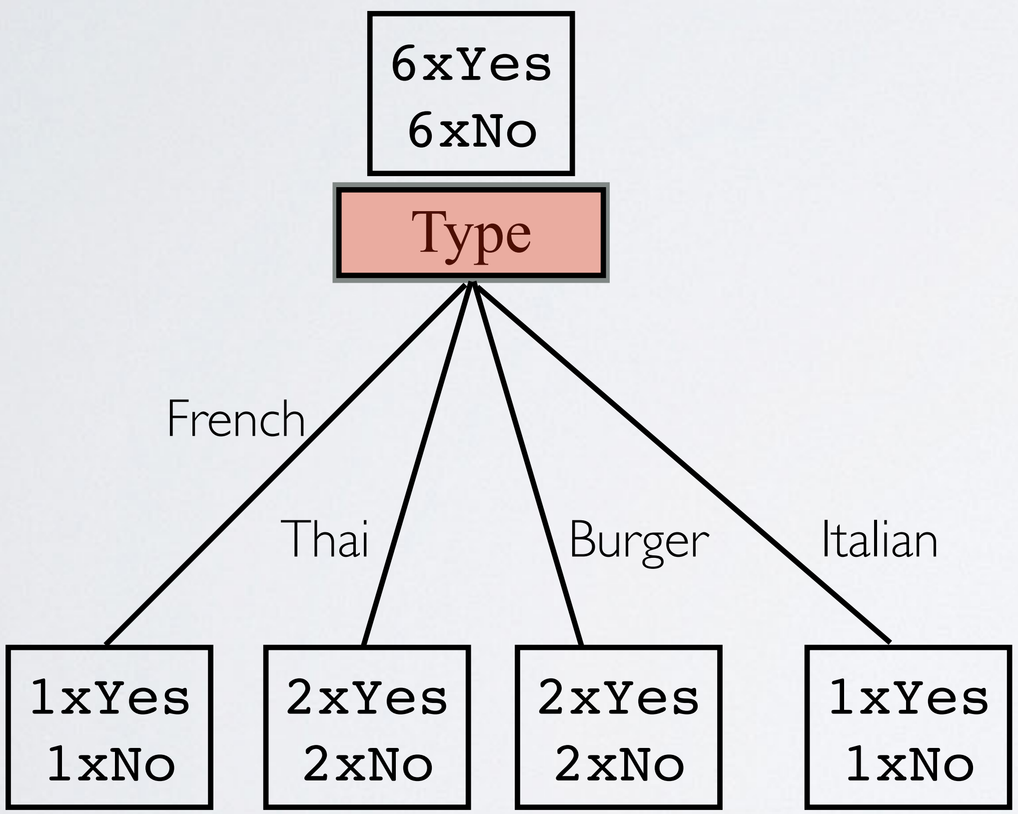
ID3

- Start at root with entire training data
- Choose attribute that creates a "good split"
  - Attribute "splits" data into subsets
  - good split: children with subsets that are unmixed (with same classification)
  - bad split: children with subsets that are mixed (with different classification)
- Children with unmixed subsets lead to a classification
- Children with mixed subsets handled with recursion

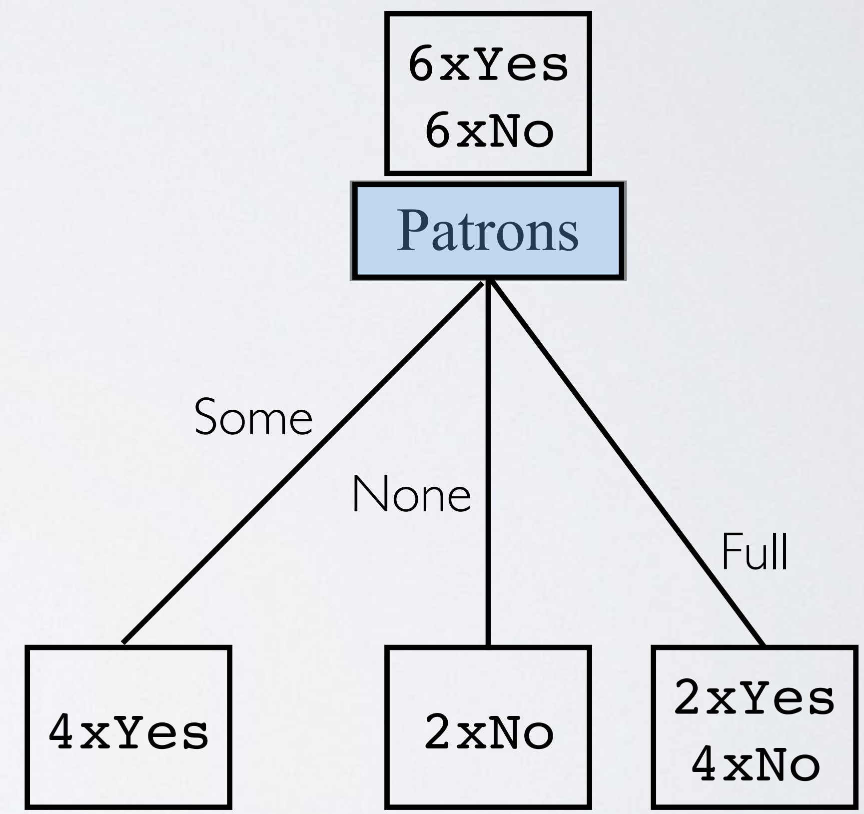
ID3

How do we distinguish

"bad" attributes from "good" attributes



many mixed subsets



many unmixed subsets

ID3

- How do we decide if attribute is good?
  - Compute **entropy** of each child
    - quantifies how mixed/alike it is
    - quantifies amount of certainty/uncertainty
  - Combine the entropies of all the children
  - Compare combined entropy of children to entropy of node
  - This is called the **information gain**

Entropy

- Entropy of a dataset of examples

$$H(\mathsf{data}) = -\left(\frac{\#\mathsf{Yes}}{\#\mathsf{Yes} + \#\mathsf{No}} \cdot \log_2\left(\frac{\#\mathsf{Yes}}{\#\mathsf{Yes} + \#\mathsf{No}}\right)\right. \qquad [\log(0) = 0]$$

$$\left. + \left(1 - \frac{\#\mathsf{Yes}}{\#\mathsf{Yes} + \#\mathsf{No}}\right) \cdot \log_2\left(1 - \frac{\#\mathsf{Yes}}{\#\mathsf{Yes} + \#\mathsf{No}}\right)\right)$$

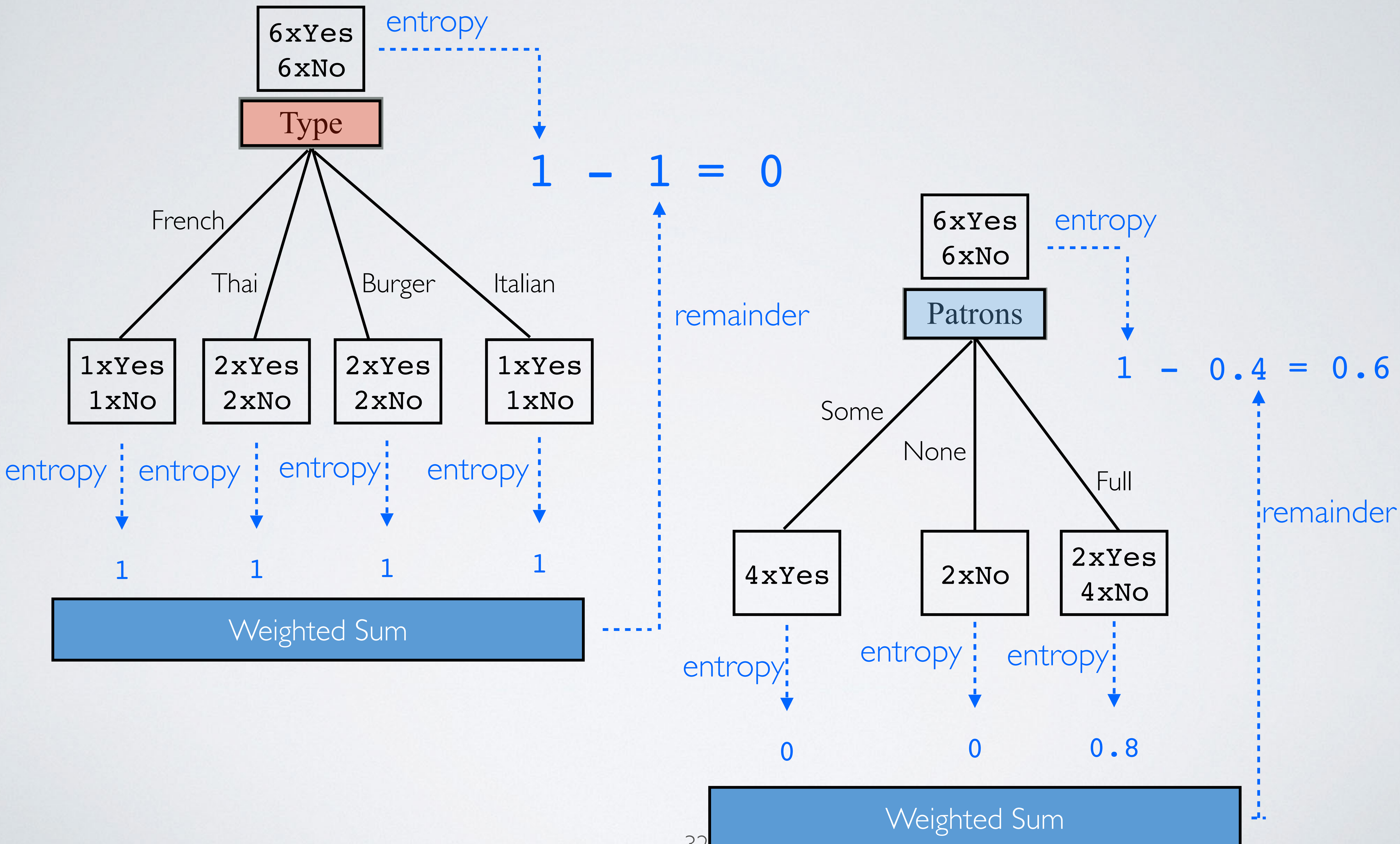
Entropy

- Entropy of a dataset of examples

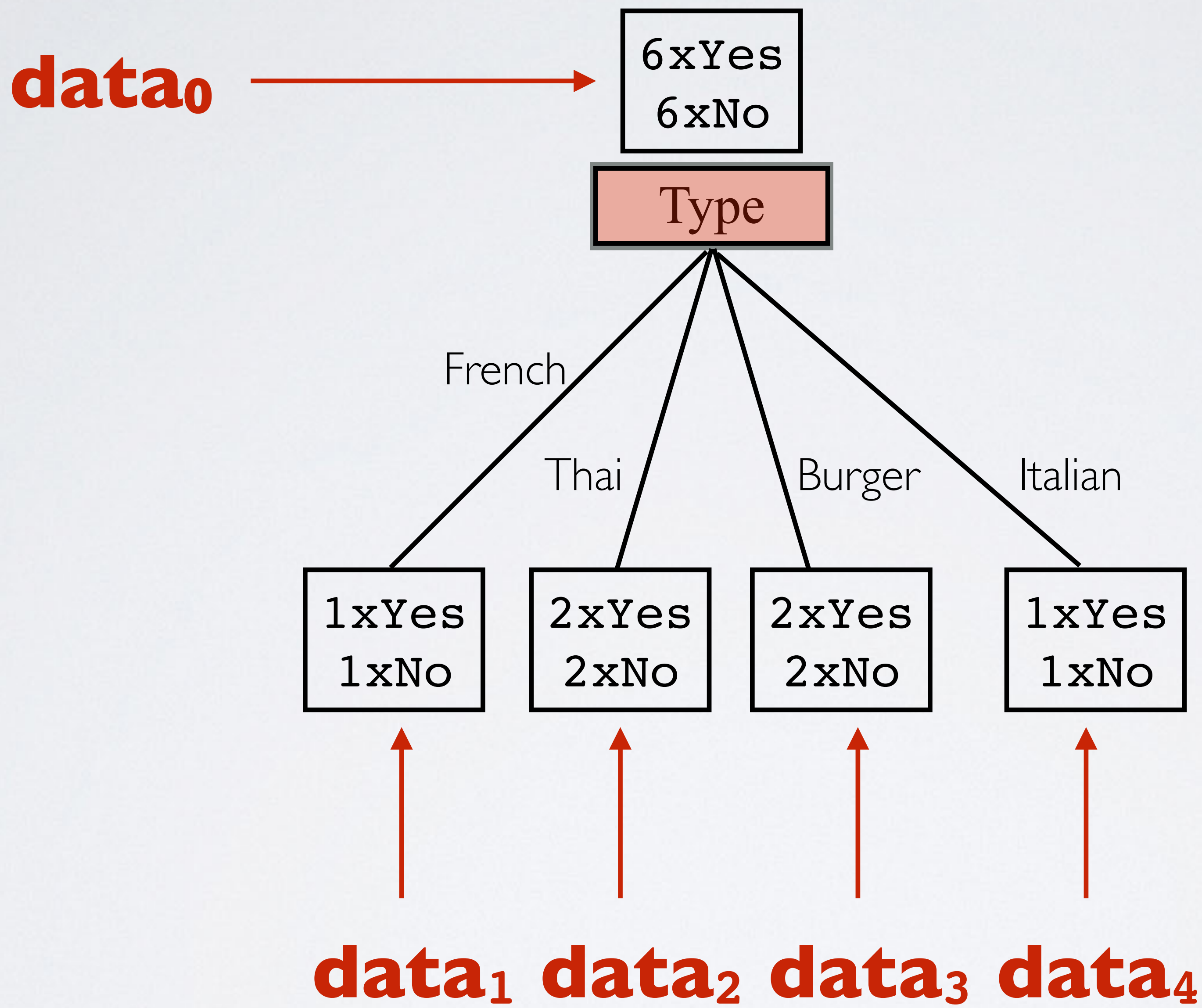
$$H(\textsf{data}) = -\left(\frac{\#\textsf{Yes}}{\#\textsf{Yes} + \#\textsf{No}} \cdot \log_2\left(\frac{\#\textsf{Yes}}{\#\textsf{Yes} + \#\textsf{No}}\right) \qquad [\log(0) = 0]\right.$$
$$\left. + \left(1 - \frac{\#\textsf{Yes}}{\#\textsf{Yes} + \#\textsf{No}}\right) \cdot \log_2\left(1 - \frac{\#\textsf{Yes}}{\#\textsf{Yes} + \#\textsf{No}}\right)\right)$$

- Intuition:
  - H(perfectly mixed) = 1
  - H(all the same) = 0

ID3 - Information Gain



ID3 - Notation



Information gain

- Entropy of a dataset of examples

$$H(\mathsf{data}) = -\left(\frac{\#\mathsf{Yes}}{\#\mathsf{Yes}+\#\mathsf{No}}\cdot\log_2\left(\frac{\#\mathsf{Yes}}{\#\mathsf{Yes}+\#\mathsf{No}}\right)\right.$$
$$\left.+\left(1-\frac{\#\mathsf{Yes}}{\#\mathsf{Yes}+\#\mathsf{No}}\right)\cdot\log_2\left(1-\frac{\#\mathsf{Yes}}{\#\mathsf{Yes}+\#\mathsf{No}}\right)\right)$$

- Remainder of an attribute

$$\mathsf{Rem}(\mathsf{Att}) = \sum_{i=1}^{d}\frac{\#\mathsf{data_i}}{\#\mathsf{data}_1+\cdots+\#\mathsf{data_d}}\cdot H(\mathsf{data_i})$$

- Information gain of an attribute

$$\mathsf{Gain}(\mathsf{Att}) = H(\mathsf{data}_0) - \mathsf{Rem}(\mathsf{Att})$$

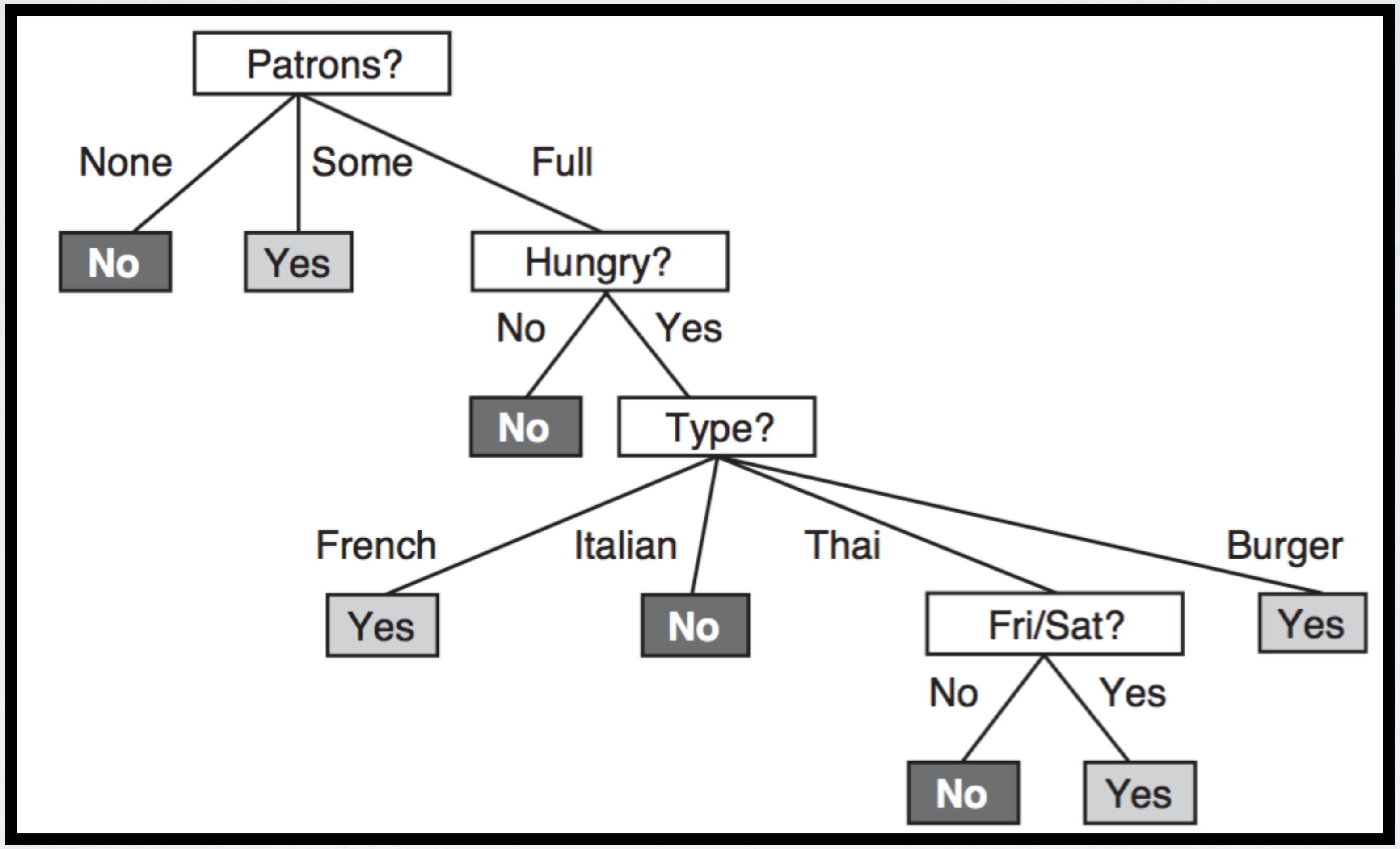
ID3 Pseudocode

No examples w/ this combination of attribute values

```
id3_algorithm(data, attributes, parent_data):
   if data is empty:
      return new node w/ most frequent classificat
   else if all examples in data have same classif
      return new node with that classification
   else if attributes is empty:
      return new node w/ most frequent classificat
   else:
      A = attribute with largest information gain
      tree = new decision tree with root A
      for each value a of attribute A
         new_data = all examples in data such that example.A = a
         subtree = id3_algorithm(new_data, attributes - A, data)
         attach subtree to tree with branch labeled "a"
      return tree
```

Examples w/ this combination of attribute values but they have different classifications

ID3



S. Russel & P. Norvig. Artificial Intelligence - A Modern Approach

Testing Decision Trees

- ID3 learns a decision tree from training data
- How good is this decision tree?
  - How accurately does it classify inputs it hasn't seen?
  - New flights, new snowstorms, new patrons, …
- Split examples into two non-overlapping sets
  - training data & testing data
  - by assigning each example to one set uniformly at random
- Accuracy of decision tree
  - # of correct classifications on testing data / # of testing examples

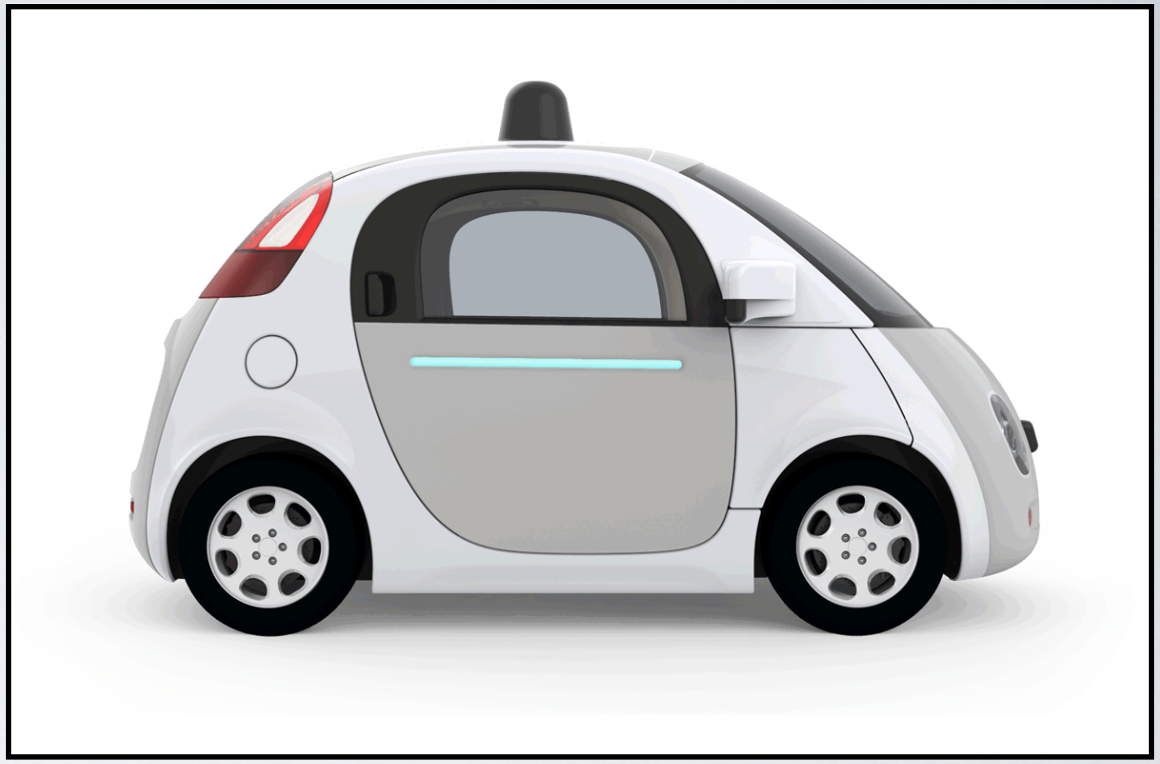
Outline

- Motivation
- Supervised learning
- Decision Trees
- ML Bias

ML Applications

- Learned algorithms are different than classical algorithms
  - A learned algorithm depends on training data
  - A learned algorithm depends on features
- Learned algorithms are being embedded everywhere
- Traditional applications
  - spell checking
  - ad targeting
  - recommendations
  - speech & handwriting recognition
  - computer vision

ML Applications



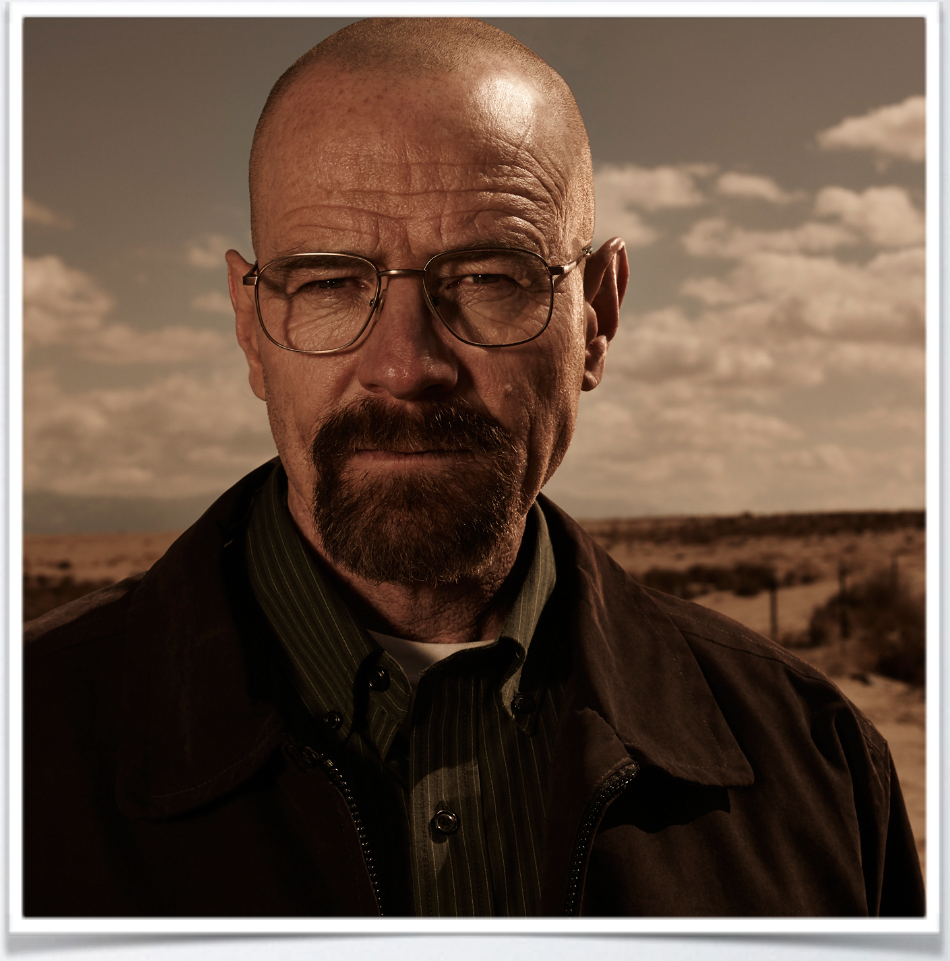
- Newer applications
  - News feeds, self-driving cars
  - insurance companies, medical systems, K-12 education
  - Risk assessment
- These have serious impact on society
  - should news be tailored to your political beliefs?
  - should car save driver or pedestrian?
  - should we deny freedoms based on risk assessments?

Risk Assessment

- Learned algorithms that predict risk
  - Insurance premiums
  - Credit
  - Crime
    - Recidivism
    - Bonds
    - Sentencing

Criminal Risk Assessment

- ML in criminal justice system
  - better predict who will commit new crimes
- Purported goals
  - a fairer system
  - less people in jail
  - for less time



Criminal Risk Assessment Tools

- COMPAS by Northpointe predicts
  - Risk of new violent crime
  - Risk of general recidivism
  - Pretrial risk (failure to appear)
- Public Safety Assessment by Arnold Foundation
  - helps Judges make release/detention decisions

Criminal Risk Assessment Tools

- Used in
  - Arizona, Colorado, Delaware, Kentucky, Louisiana, Maryland, New York, Oklahoma, Virginia, Washington, Wisconsin, …
  - Often adopted without independent study of effectiveness
- Example
  - New York started using tool in `2001`
  - Studied it only in `2012`

ProPublica Study

- In 2016 ProPublica conducted a study of COMPAS
  - 7000 arrests in Broward County, FL
  - between 2013 and 2014
- OK predictions for *all* crimes (misdemeanors included)
  - 61% of people labeled high risk committed new crimes
- But unreliable for *violent* crimes
  - 20% of people labeled high risk committed new *violent* crimes

ProPublica Study

- Found significant *racial* disparities
- Out of people labeled high risk but didn't re-offend
  - `44.9%` were African American
  - `23.5%` were White
- Out of people labeled low risk but did re-offend
  - `28%` were African American
  - `47.7%` were White
- Study accounted for
  - Criminal history, age and gender

Algorithms

- Algorithms are impacting our lives
- You are learning how to
  - design algorithms
  - analyze algorithms
  - implement algorithms
- But don’t forget that…
  - …your algorithms will impact **people**

Algorithms

- Algorithms can do a lot amazing things
- But they can also
  - deny people their freedom
  - addict people
  - sway elections
  - expose people's private lives
  - …

References

- *Artificial Intelligence - A Modern Approach* by S. Russel & P. Norvig
- *Machine Bias* by ProPublica
  - https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing
- *New York State COMPAS-Probation Risk and Need Assessment Study*
  - http://www.northpointeinc.com/downloads/research/DCJS_OPCA_COMPAS_Probation_Validity.pdf
- *Evaluating the Predictive Validity of the COMPAS Risk and Needs Assessment System* by Northpointe
  - http://www.northpointeinc.com/files/publications/Criminal-Justice-Behavior-COMPAS.pdf

References

- Fairness, Accountability and Transparency in ML
  - http://fatml.mysociety.org
- Center for Humane Technology
  - http://humanetech.com/

References

- Slide #2
  - Toothless from "*How to* ***Train*** *Your Dragon*"