Classification: Decision Trees

These slides were assembled by Byron Boots, with grateful acknowledgement to Eric Eaton and the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution.

Last Time

- Common decomposition of machine learning, based on differences in inputs and outputs
  - **Supervised Learning:** Learn a function mapping inputs to outputs using labeled training data (you get instances/examples with both inputs and ground truth output)
  - **Unsupervised Learning:** Summarize something about input data without any labels, for example clustering instances that are "similar"
  - **Reinforcement Learning:** Learn how to make decisions given a sparse reward

- ML is an interaction between:
  - data (features/attributes of data are important!)
  - the the function class (parameterized model) you choose, and
  - the optimization algorithm you use to explore space of functions to find the "best" one

Supervised Learning: Function Approximation

Problem Setting

- Set of instances $\mathcal{X}$
- Set of labels $\mathcal{Y}$

Based on slide by Tom Mitchell

Supervised Learning: Function Approximation

Problem Setting

- Set of instances (inputs, independent variables…) $\mathcal{X}$
- Set of labels (outputs, targets, dependent variables…) $\mathcal{Y}$

Based on slide by Tom Mitchell

Supervised Learning: Function Approximation

Problem Setting

- Set of instances $\mathcal{X}$
- Set of labels $\mathcal{Y}$
- Unknown target function $f : \mathcal{X} \rightarrow \mathcal{Y}$
- Set of function hypotheses $H = \{h \mid h : \mathcal{X} \rightarrow \mathcal{Y}\}$
- Performance metric

Input: Training examples of unknown target function f

$$\{\langle \boldsymbol{x}_i, y_i \rangle\}_{i=1}^{n} = \{\langle \boldsymbol{x}_1, y_1 \rangle, \ldots, \langle \boldsymbol{x}_n, y_n \rangle\}$$

Output: Hypothesis $h \in H$ that "best" approximates f according to the performance metric

Based on slide by Tom Mitchell

Sample Dataset (was Tennis Played?)

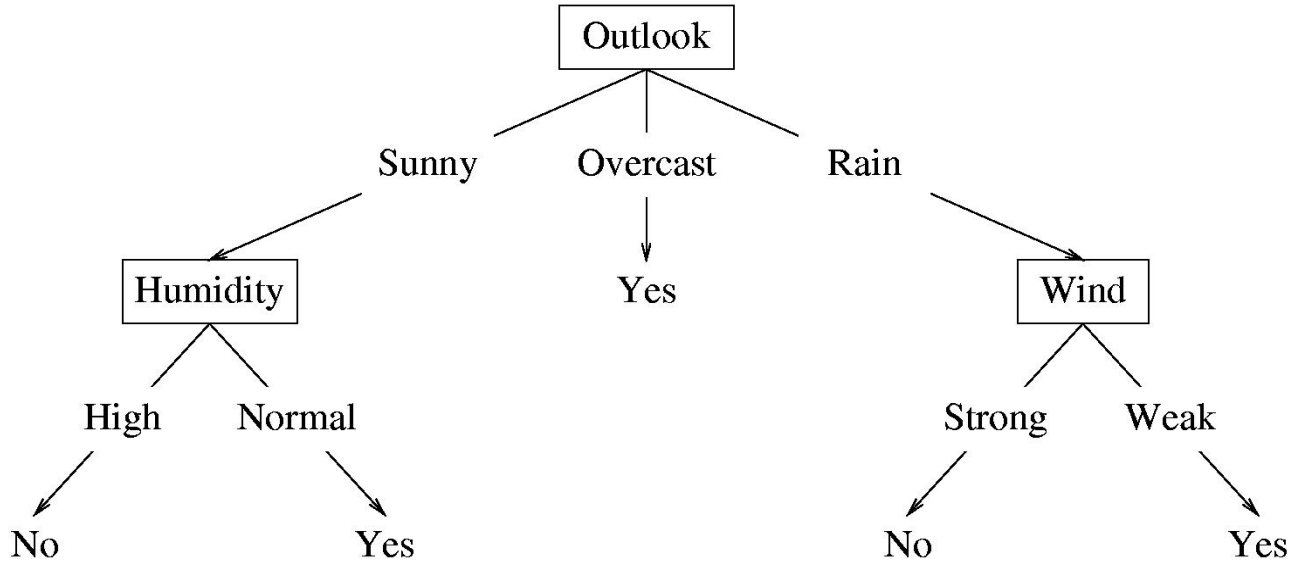
- Columns denote features X_i
- Rows denote labeled instances $\langle \boldsymbol{x}_i, y_i \rangle$
- Class label denotes whether a tennis game was played

$\langle \boldsymbol{x}_i, y_i \rangle$

Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Decision Tree

- A possible decision tree for the data:

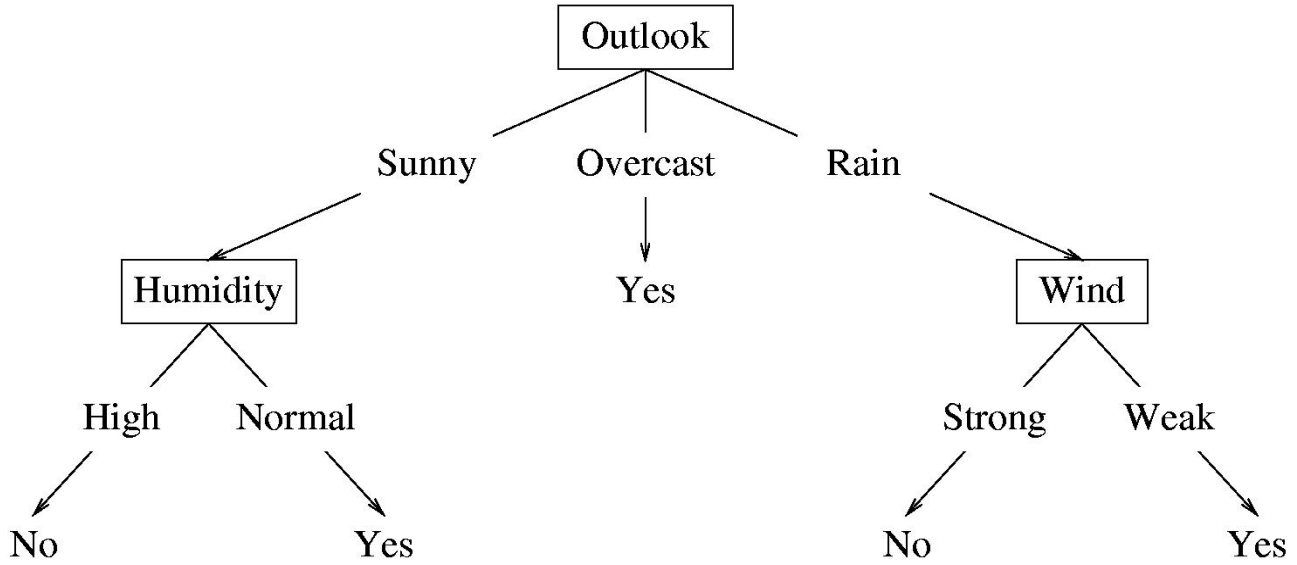


- Each internal node: test one attribute X_i
- Each branch from a node: selects one value for X_i
- Each leaf node: predict Y

Based on slide by Tom Mitchell

Decision Tree

- A possible decision tree for the data:



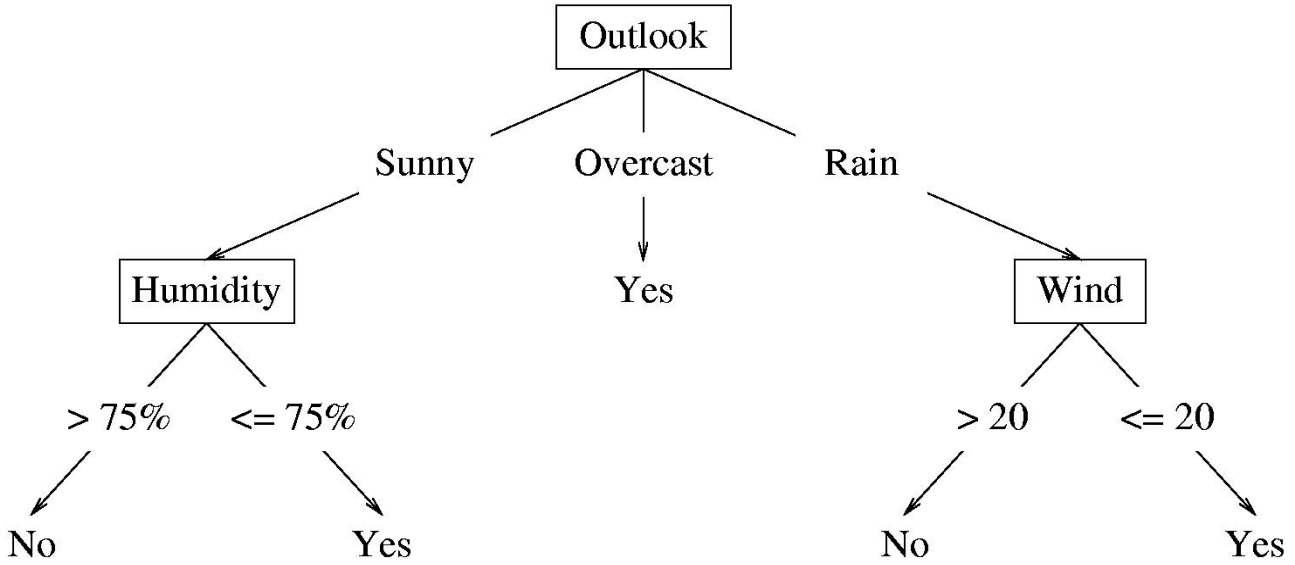
- What prediction would we make for

  <outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

Based on slide by Tom Mitchell

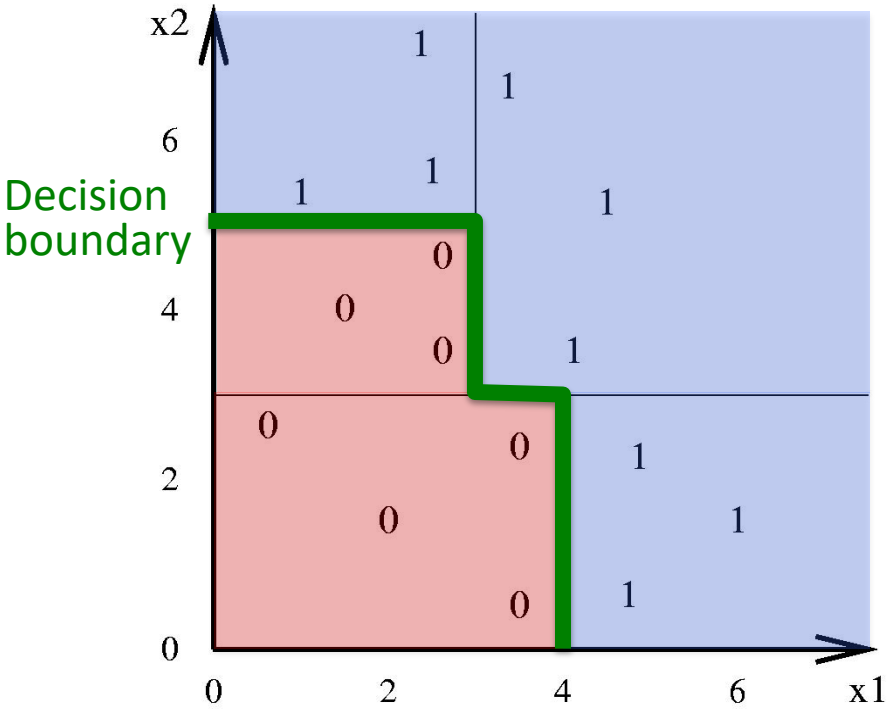
Decision Tree

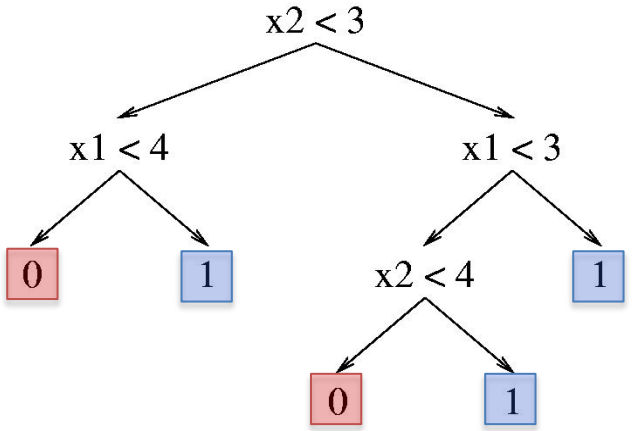
- If features are continuous, internal nodes can test the value of a feature against a threshold



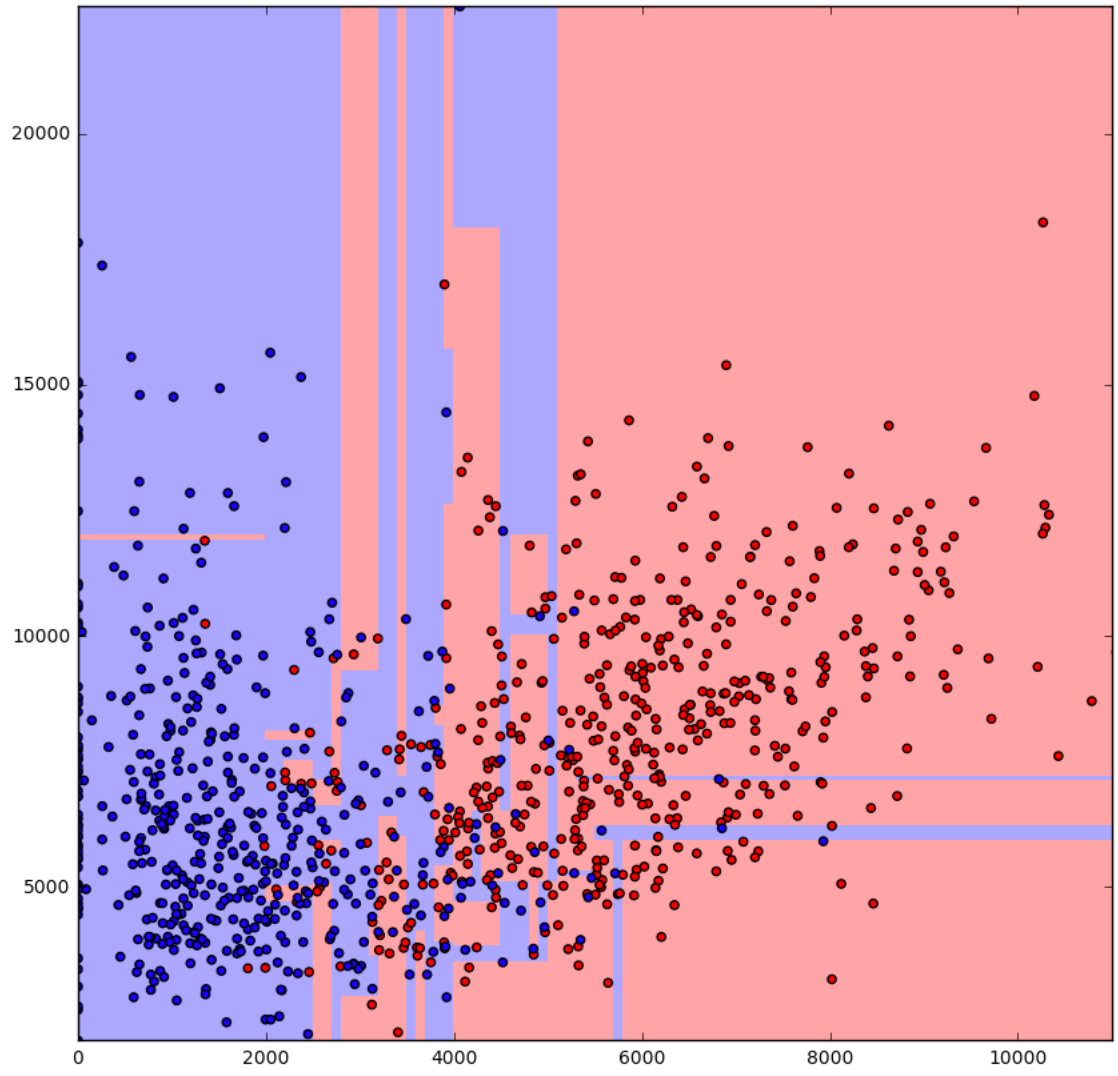
Decision Tree – Decision Boundary

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label
  - or a probability distribution over labels (will discuss next time)





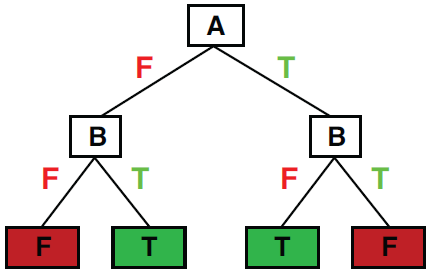
Decision Tree – Decision Boundary



Expressiveness

- Given a particular space of functions, you may not be able to represent everything
- What **functions** can decision trees represent?
- Decision trees can represent any function of the input attributes!
  - Boolean operations (and, or, xor, etc.)?
  - **Yes!**
  - All boolean functions?
  - **Yes!**

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

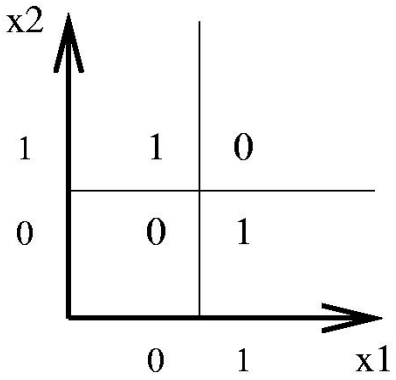


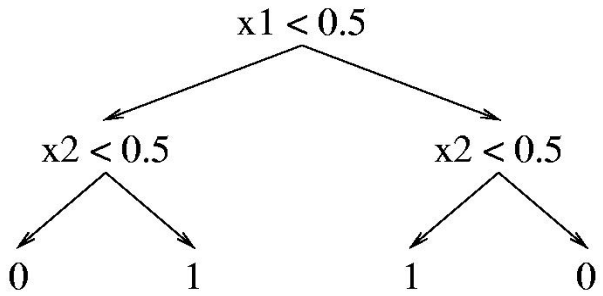
(Figure from Stuart Russell)

Expressiveness

Decision trees have a variable-sized hypothesis space

- As the #nodes (or depth) increases, the hypothesis space grows
  - Depth 1 ("decision stump"): can represent any boolean function of one feature
  - Depth 2: any boolean function of two features; some involving three features (e.g., $(x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_3)$)
  - etc.

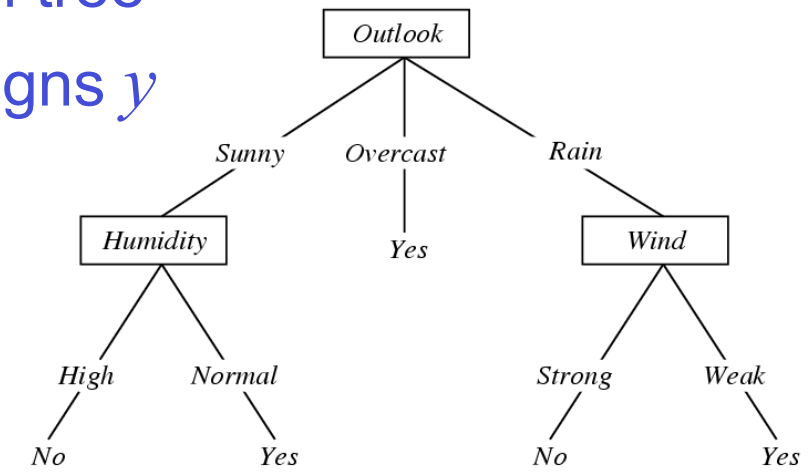




Decision Tree Learning

Problem Setting:

- Set of possible instances X
  - each instance x in X is a feature vector
  - e.g., <*Humidity=low, Wind=weak, Outlook=rain, Temp=hot*>
- Unknown target function $f : X \rightarrow Y$
  - Y is discrete valued
- Set of function hypotheses $H=\{ h \mid h : X \rightarrow Y \}$
  - each hypothesis h is a decision tree
  - trees sorts x to leaf, which assigns y



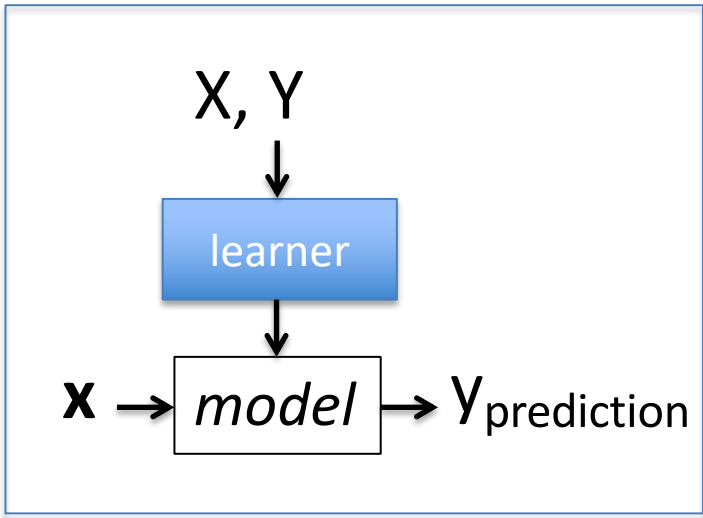
Stages of (Batch) Machine Learning

Given: labeled training data $X, Y = \{\langle \boldsymbol{x}_i, y_i \rangle\}_{i=1}^{n}$

- Assumes each $\boldsymbol{x}_i \sim \mathcal{D}(\mathcal{X})$ with $y_i = f_{target}(\boldsymbol{x}_i)$

Train the model:

model ← *classifier*.train(X, Y)



Apply the model to new data:

- Given: new unlabeled instance $\boldsymbol{x} \sim \mathcal{D}(\mathcal{X})$

$y_{prediction}$ ← *model*.predict(**x**)

Basic Algorithm for Top-Down Learning of Decision Trees

[ID3, C4.5 by Quinlan]

node = root of decision tree

Main loop:

1. $A \leftarrow$ the "best" decision attribute for the next node.
2. Assign *A* as decision attribute for *node*.
3. For each value of *A*, create a new descendant of *node*.
4. Sort training examples to leaf nodes.
5. If training examples are perfectly classified, stop. Else, recurse over new leaf nodes.

How do we choose which attribute is best?

Choosing the Best Attribute

Key problem: choosing which attribute to split a given set of examples

- Some possibilities are:
  - **Random:** Select any attribute at random
  - **Least-Values:** Choose the attribute with the smallest number of possible values
  - **Most-Values:** Choose the attribute with the largest number of possible values
  - **Max-Gain:** Choose the attribute that has the largest expected *information gain*
    - i.e., attribute that results in smallest expected size of subtrees rooted at its children
- The ID3 algorithm uses the Max-Gain method of selecting the best attribute

Example: Restaurant Domain (Russell & Norvig)

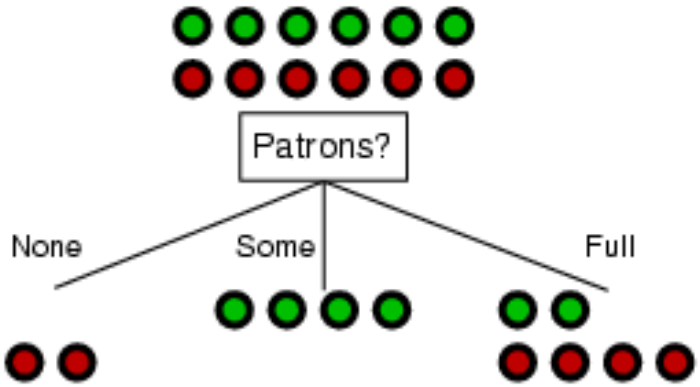
Model a patron's decision of whether to wait for a table at a restaurant

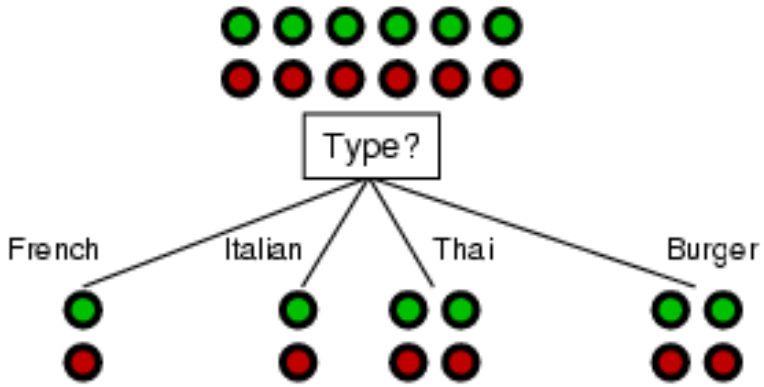
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

~7,000 possible cases

Choosing an Attribute

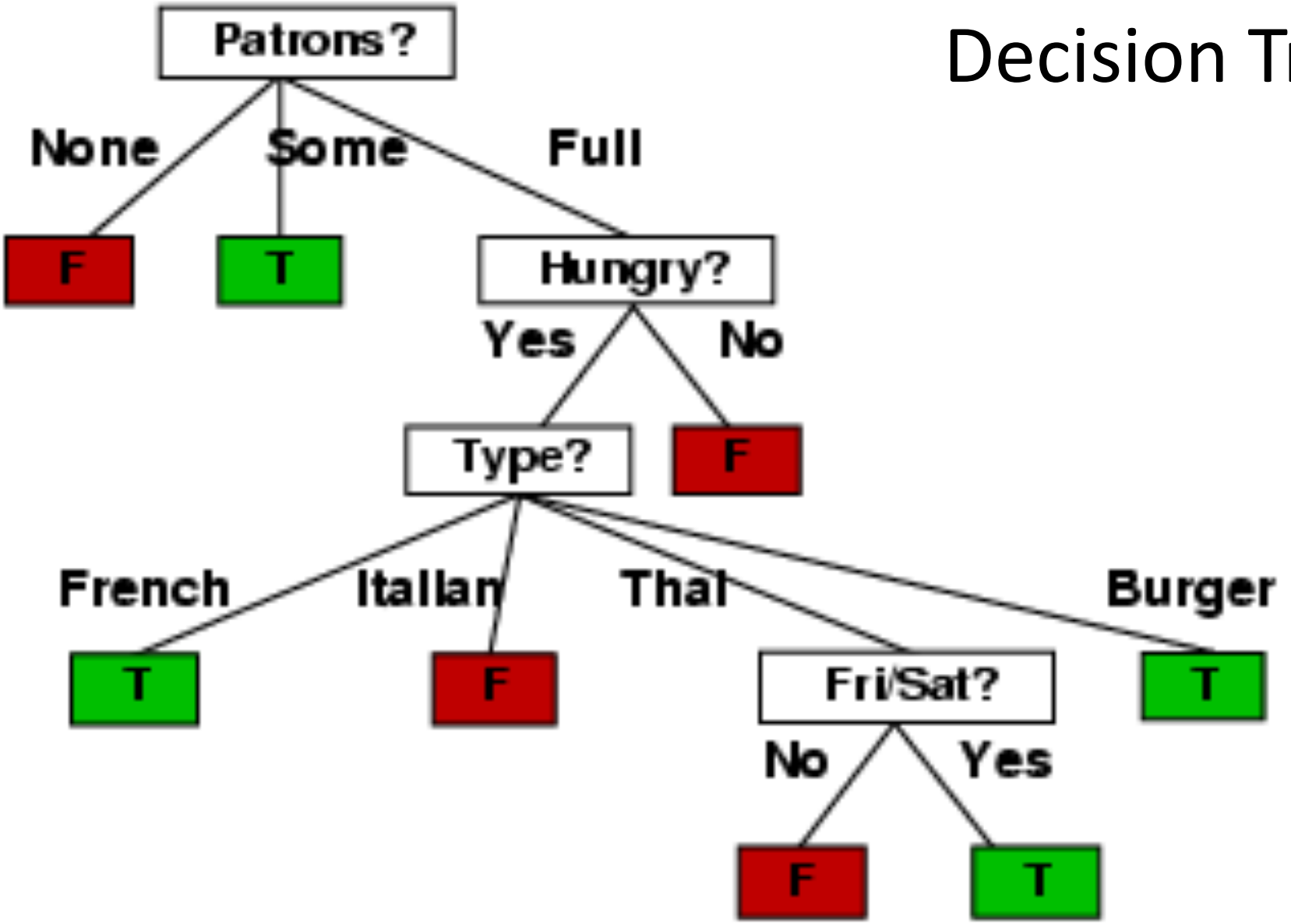
Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"





Which split is more informative: *Patrons?* or *Type?*

ID3-induced Decision Tree

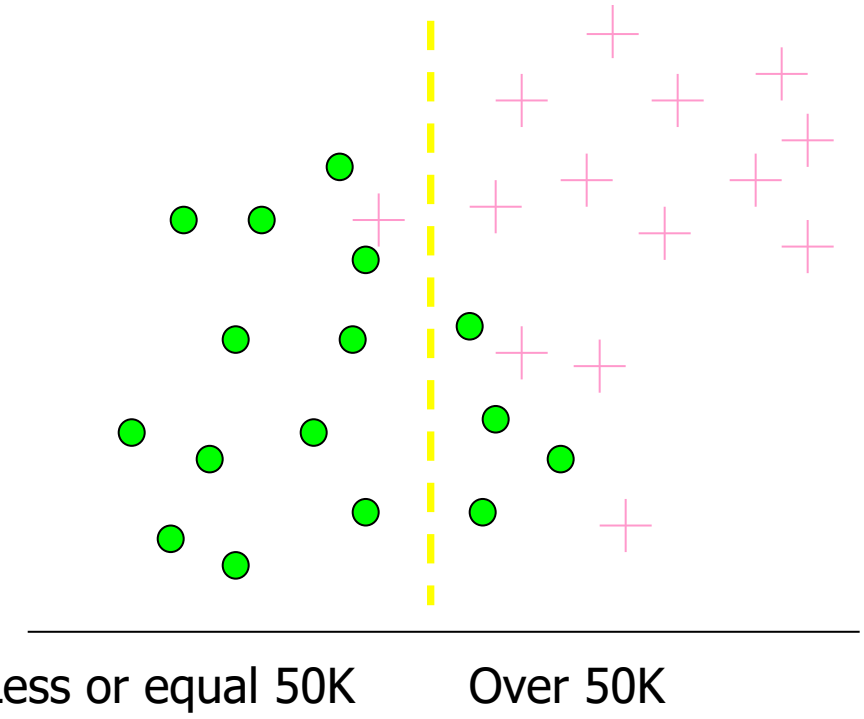


Based on Slide from M. desJardins & T. Finin

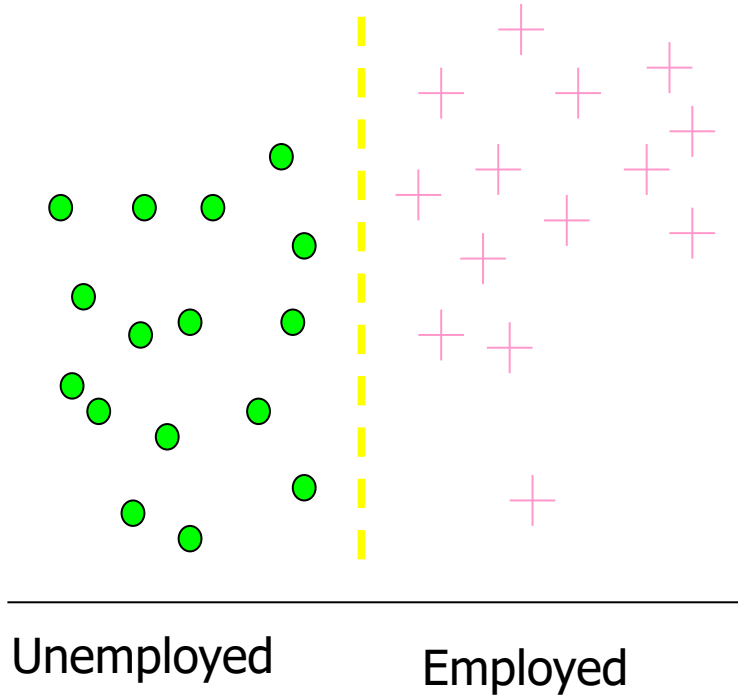
Information Gain

Which test is more informative?

Split over whether Balance exceeds 50K



Split over whether applicant is employed

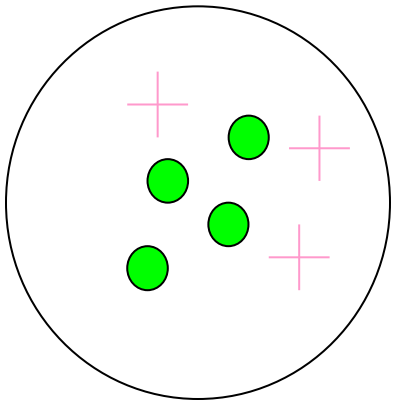


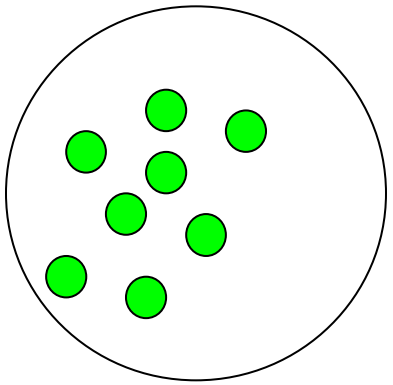
Based on slide by Pedro Domingos

Information Gain

Impurity/Entropy (informal)

– Measures the level of **impurity** in a group of examples

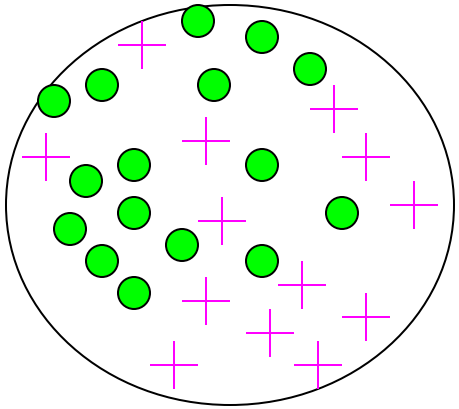




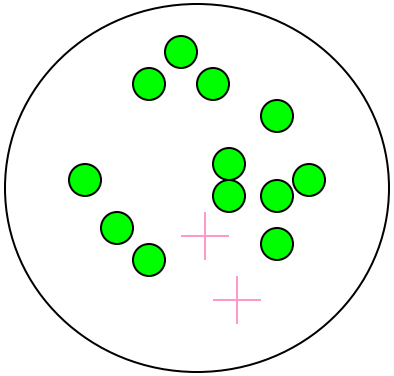
Based on slide by Pedro Domingos

Impurity

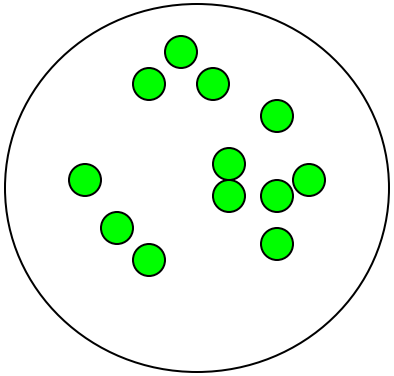
Very impure group



Less impure



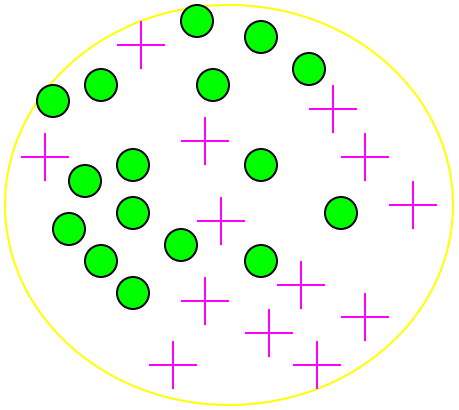
Minimum impurity



Based on slide by Pedro Domingos

Entropy: a common way to measure impurity

- Entropy = $\sum_i -p_i \log_2 p_i$



p_i is the probability of class i

Compute it as the proportion of class i in the set.

- Entropy comes from information theory. The higher the entropy the more the information content.

What does that mean for learning from examples?

Based on slide by Pedro Domingos