Lecture 5: Decision Trees

COMP 411, Fall 2021

Victoria Manfredi

WESLEYAN UNIVERSITY

Acknowledgements: These slides are based primarily on content from the book "Machine Learning" by Tom Mitchell, and on slides created by Vivek Srikumar (Utah) and Dan Roth (Penn)

Today’s Topics

Decision trees (non-linear classifiers)

- **Motivation**
- **Representation**
  - What are decision trees?

Decision Trees

REPRESENTATION

Key issues in machine learning

Modeling

- How to formulate your problem as a machine learning problem? How to represent data? Which algorithms to use? What learning protocols?

Representation

- Good hypothesis spaces and good features

Algorithms

- What is a good learning algorithm?
- What is success?
- Generalization vs. overfitting
- The computational question: how long will learning take?

Coming up …

Different hypothesis spaces and learning algorithms

Linear regression

- Least mean squares regression

Linear classifiers

- Perceptron
- Logistic regression

Non-linear classifiers

- Decision trees and ID3 algorithm
- Multi-layer perceptron (neural networks)

Reinforcement learning

Provides a "gentle" introduction to machine learning concepts

Coming up …

Different hypothesis spaces and learning algorithms

Important issues to consider

1. What do these hypotheses represent?

2. Implicit assumptions and tradeoffs

3. Generalization?

4. How do we learn?

Decision Trees

REPRESENTATION

Representing data

Data can be represented as a big table, with columns denoting different attributes

Name	Label
Norman Danner	+
Karen Collins	-
Dan Licata	+
Danny Krizanc	+
Saray Shai	+
Wai Kiu Chan	-

Representing data

Data can be represented as a big table, with columns denoting different attributes

Name	**Faculty department**	**Name contains Dan**	**Second character of first name**	**Label**
Norman Danner	CS	Yes	o	+
Karen Collins	Math	No	a	-
Dan Licata	CS	Yes	a	+
Danny Krizanc	CS	Yes	a	+
Saray Shai	CS	No	a	+
Wai Kiu Chan	Math	No	a	-

Representing data

Data can be represented as a big table, with columns denoting different attributes

Name	Faculty department	Name contains Dan	Second character of first name	Label
Norman Danner	CS	Yes	o	+
Karen				
Dan L				
Danny				
Saray Shai	CS	No	a	+
Wai Kiu Chan	Math	No	a	-

With these three attributes, how many unique rows are possible?

Representing data

Data can be represented as a big table, with columns denoting different attributes

Name	Faculty department	Name contains Dan	Second character of first name	Label
Norman Danner	CS	Yes	o	+
Karen				
Dan L				
Danny				
Saray Shai	CS	No	a	+
Wai Kiu Chan	Math	No	a	-

With these three attributes, how many unique rows are possible?

$$2 \times 2 \times 26 = 104$$

If there are 100 attributes, all binary, how many unique rows are possible?

(100 times) $2 \times 2 \times 2 \times 2 \times \cdots \times 2 = 2^{100}$

Representing data

Data can be represented as a big table, with columns denoting different attributes

Name	Faculty department	Name contains Dan	Second character of first name	Label
Norman Danner	CS	Yes	o	+
Karen				
Dan L				
Danny				
Saray Shai	CS	No	a	
Wai K				

With these three attributes, how many unique rows are possible?

$$2 \times 2 \times 26 = 104$$

If there are 100 attributes, all binary, how many unique rows are possible?

(100 times) $2 \times 2 \times 2 \times 2 \times \cdots \times 2 = 2^{100}$

If we wanted to store all possible rows, this number is too large

We need to figure out how to represent data in a better, more efficient way

What are decision trees?

A hierarchical data structure that represents data using a divide-and-conquer strategy

Can be used as a hypothesis class for non-parametric classification or regression

General idea: given a collection of examples, learn a decision tree that represents it. Use this representation to classify new examples

What are decision trees?

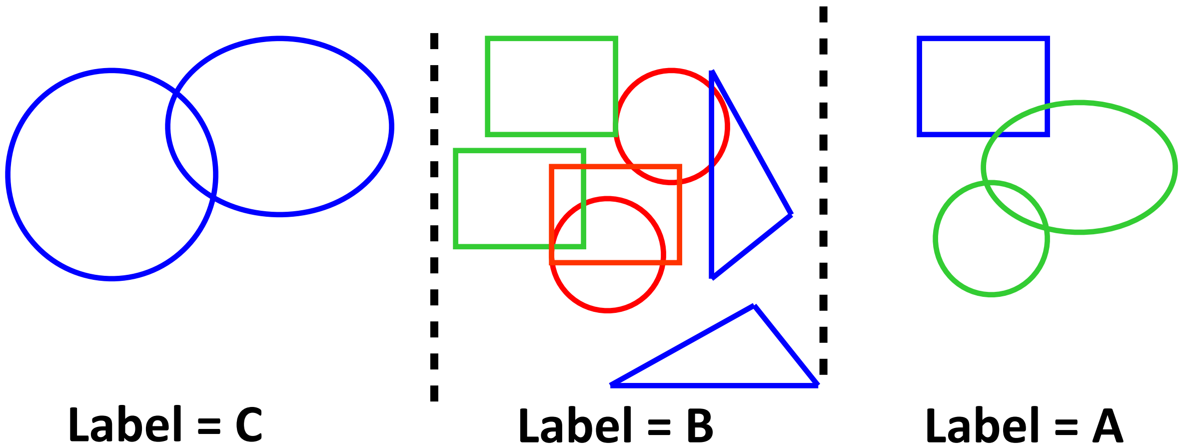
Decision trees are a family of classifiers for instance that are represented by collections of attributes (i.e., feature vectors: color=; shape=; label=)

Nodes are tests for feature values

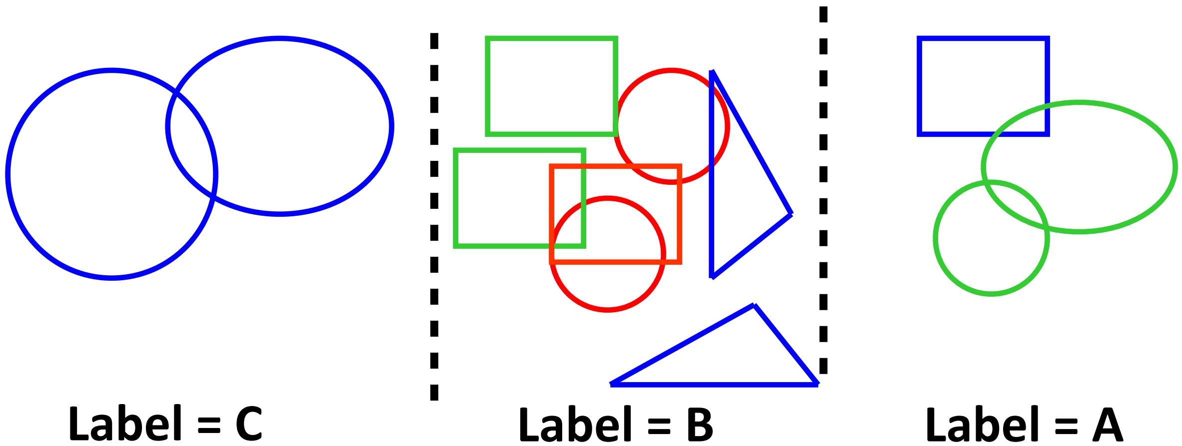
There is one branch for every value that the feature can take

Leaves of the tree specify the category (class labels)

Let’s build a decision tree for classifying shapes



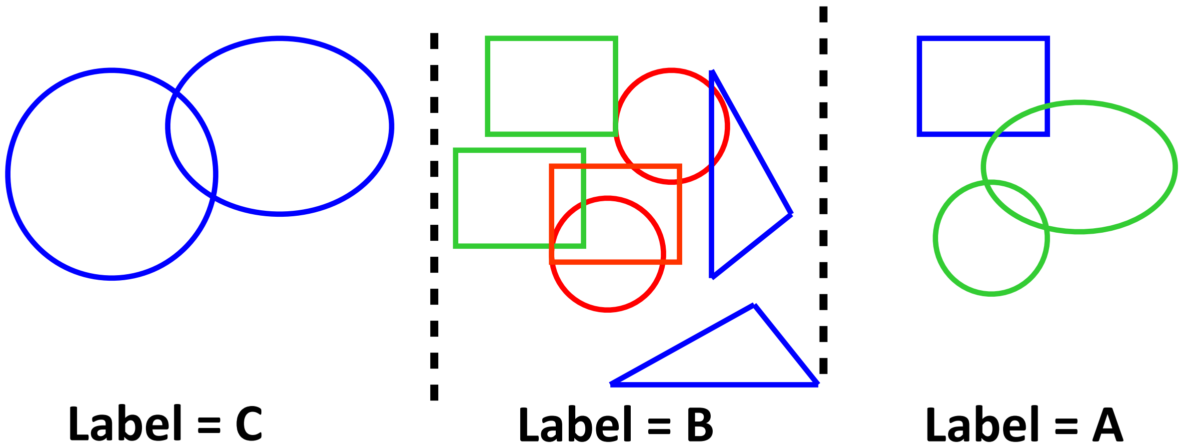
Let's build a decision tree for classifying shapes



Before building a decision tree:

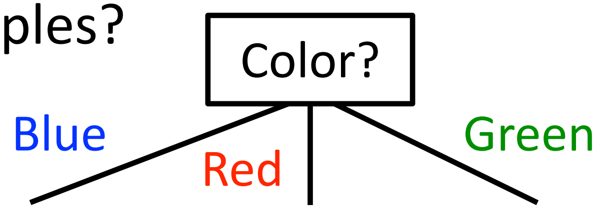
What is the label for a red triangle? And why?

Let's build a decision tree for classifying shapes

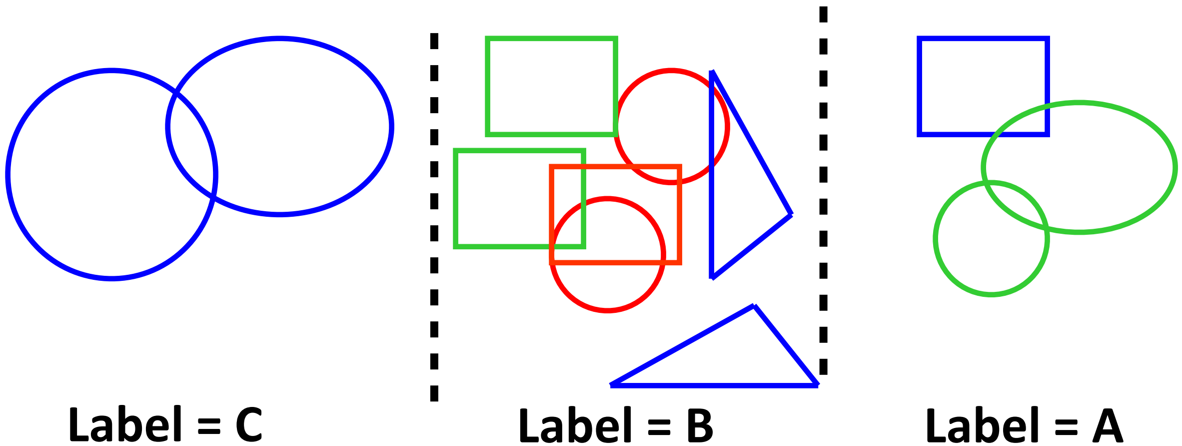


What are some attributes of the examples?

Color, shape

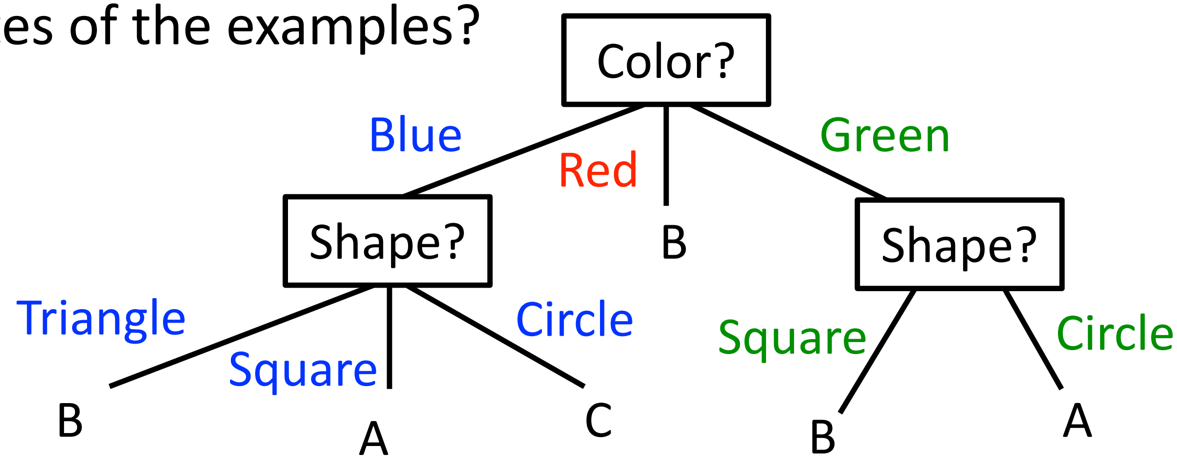


Let’s build a decision tree for classifying shapes

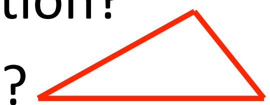


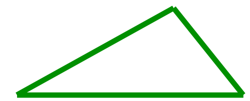
What are some attributes of the examples?

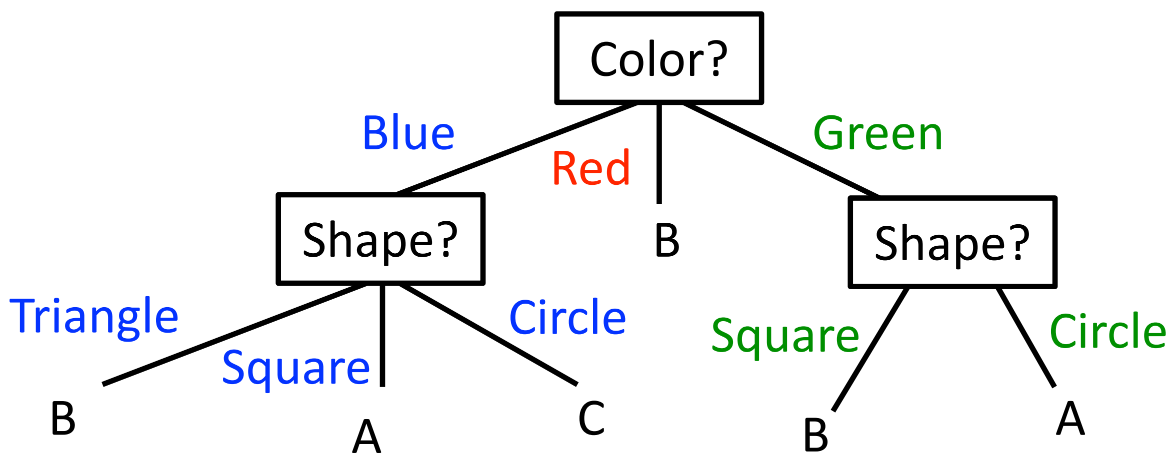
Color, shape



Let’s build a decision tree for classifying shapes

1. How do we learn a decision tree? Coming up soon
2. How to use a decision tree for prediction?
    - What is the label for a red triangle? 

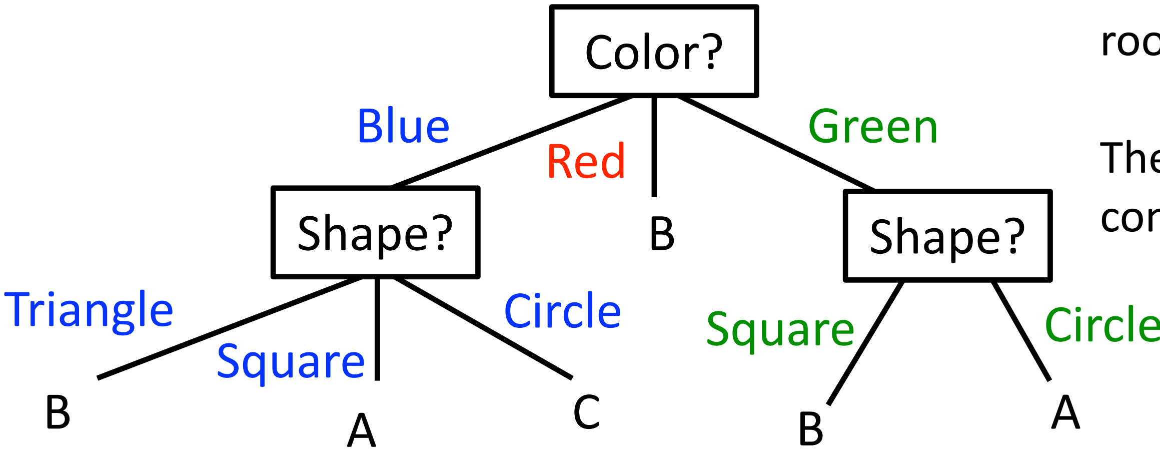
      $\Longrightarrow$ Just follow a path from the root to a leaf
    - What about a green triangle? 



Expressivity of decision trees

What Boolean function can decision trees represent?

- *Any* Boolean function



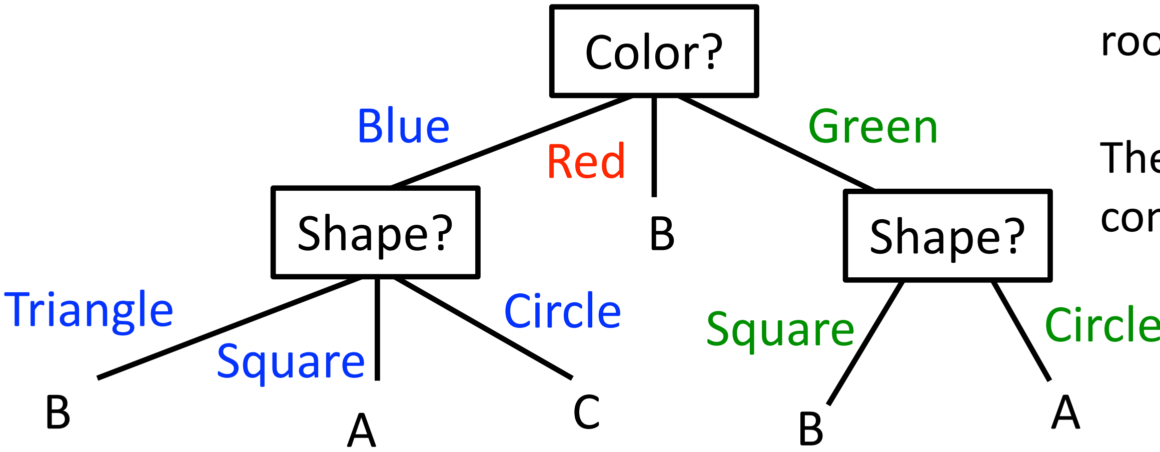
Every path from the tree to a root is a rule

The full tree is equivalent to the conjunction of all the rules

Expressivity of decision trees

What Boolean function can decision trees represent?

- *Any* Boolean function



Every path from the tree to a root is a rule

The full tree is equivalent to the conjunction of all the rules

(Color=Blue AND Shape=Triangle => Label=B) AND

(Color=Blue AND Shape=Square => Label=A) AND

(Color=Blue AND Shape=Circle => Label=C) AND ....

Any Boolean function can be represented as a decision tree

Decision Trees

Outputs are discrete categories

But real valued outputs are possible (regression trees)

There are methods for handling noisy data (noise in the label or in the features) and for handling missing attribute values. Pruning trees helps with noise. More on this later …

Numeric attributes and decision boundaries

We have seen instances represented as attribute-value pairs (color=Blue, shape=Square, second letter=a)

- Values have been categorical

How do we deal with numeric feature values (e.g., length = ?)

- discretize values or use thresholds on the values for splitting nodes
- This example divides the features space into axis-parallel rectangles

Numeric attributes and decision boundaries

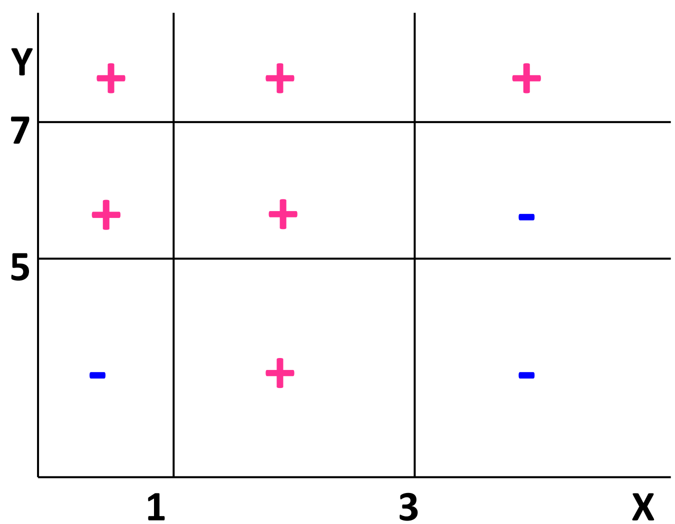
We have seen instances represented as attribute-value pairs (color=Blue, shape=Square, second letter=a)

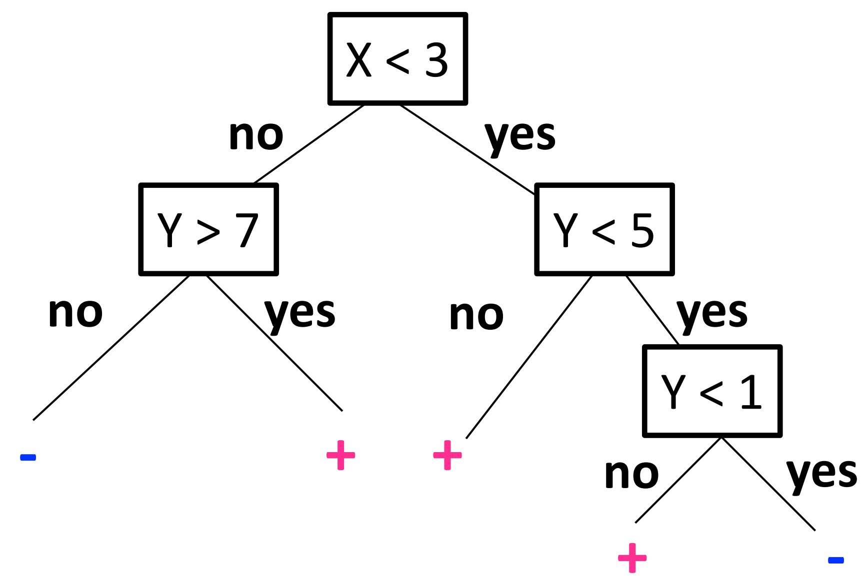
– Values have been categorical

How do we deal with numeric feature values (e.g., length = ?)

– discretize values or use thresholds on the values for splitting nodes

– This example divides the features space into axis-parallel rectangles





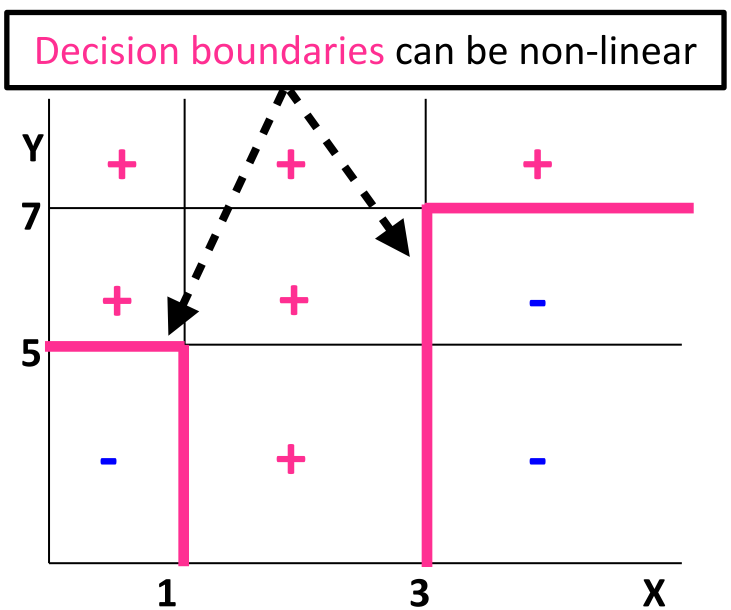
Numeric attributes and decision boundaries

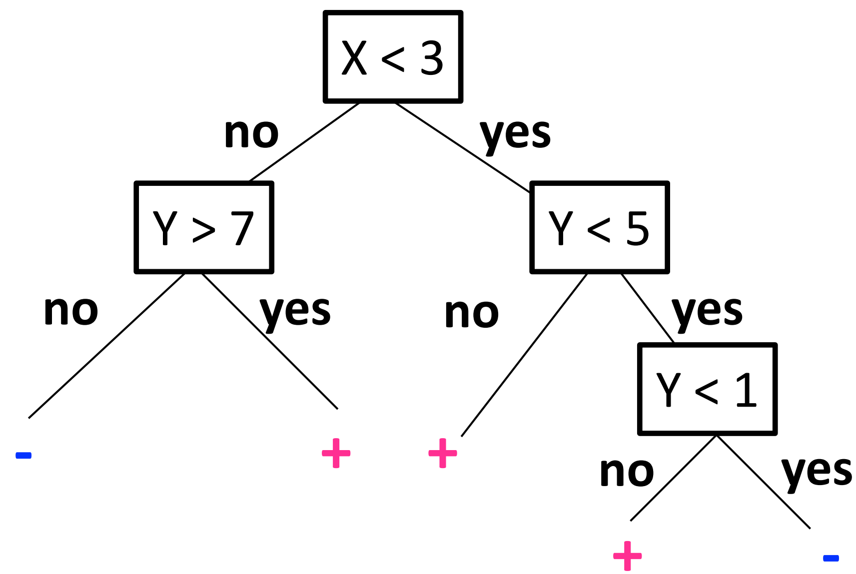
We have seen instances represented as attribute-value pairs (color=Blue, shape=Square, second letter=a)

- Values have been categorical

How do we deal with numeric feature values (e.g., length = ?)

- discretize values or use thresholds on the values for splitting nodes
- This example divides the features space into axis-parallel rectangles





Summary: decision trees

Decision trees can represent any Boolean function

A way to represent a lot of data

A natural representation (think playing 20 questions)

Predicting with a decision tree classifier is easy

Clearly, given a dataset, there are many decision trees that can represent it. [Exercise: Why?]

Learning a good representation from data is the challenge

Decision Trees

LEARNING

History of decision tree research

Full search decision tree methods to model human concept learning: Hunt et al 60s, psychology

Quinlan developed the ID3 (*Iterative Dichotomiser 3*) algorithm with the information gain heuristic to learn expert systems from examples (late 70s)

Breiman, Freidman and colleagues in statistics developed CART (*Classification And Regression Trees*)

A variety of improvements in the 80s: coping with noise, continuous attributes, missing data, non-axis parallel, etc.

Quinlan's updated algorithms, C4.5 (1993) and C5 are more commonly used

Boosting (or Bagging) over decision trees is a very good general purpose algorithm

Will I play tennis today?

Features

– Outlook:	{Sun, Overcast, Rain}
– Temperature:	{Hot, Mild, Cool}
– Humidity:	{High, Normal, Low}
– Wind:	{Strong, Weak}

Labels

– Binary classification task: Y = {+, -}

Will I play tennis today?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook: **S**unny, **O**vercast, **R**ainy

Temperature: **H**ot, **M**edium, **C**ool

Humidity: **H**igh, **N**ormal, **L**ow

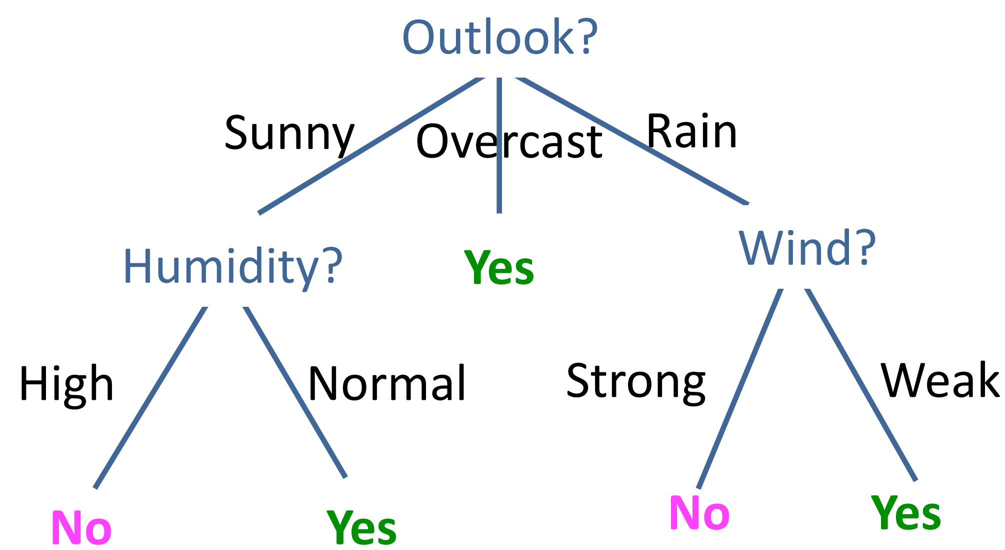
Wind: **S**trong, **W**eak

Basic decision tree learning algorithm

Data is processed in batch (i.e., all of the data available)

Recursively build a decision tree top down

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

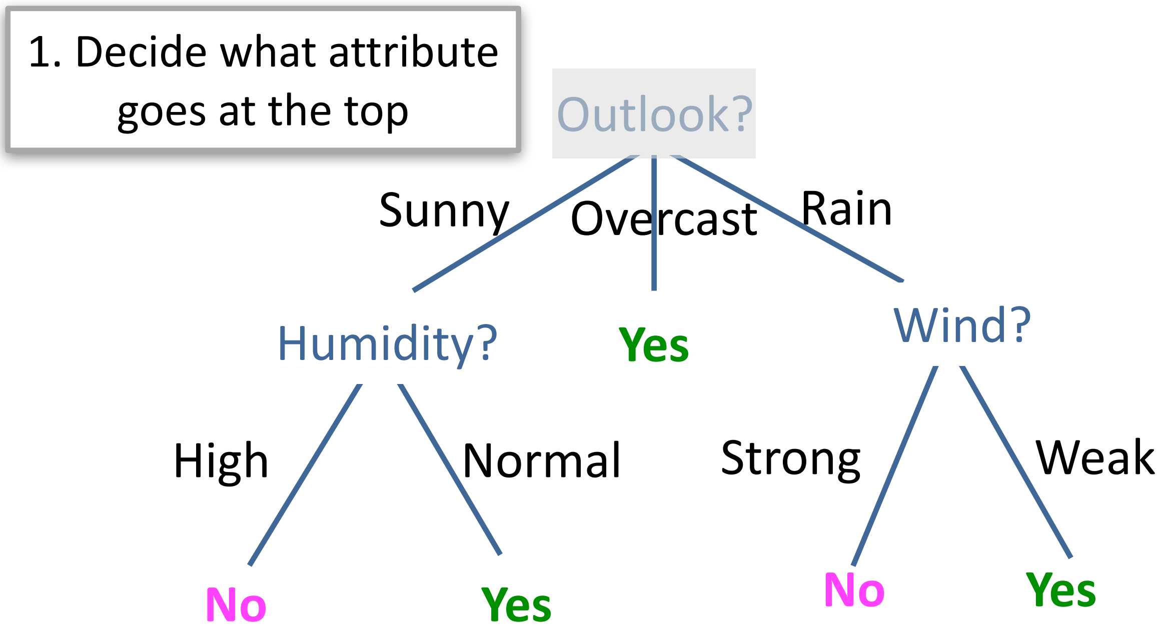


Basic decision tree learning algorithm

Data is processed in batch (i.e., all of the data available)

Recursively build a decision tree top down

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

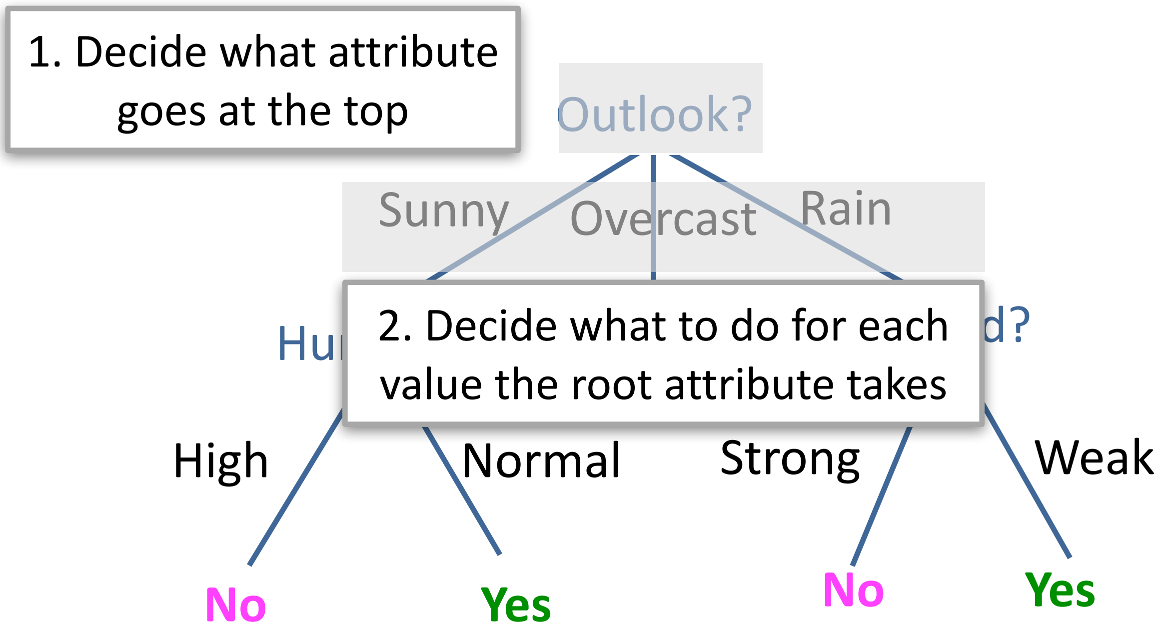


Basic decision tree learning algorithm

Data is processed in batch (i.e., all of the data available)

Recursively build a decision tree top down

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-



Basic decision tree algorithm: ID3

Input:
S is the set of examples
A is the set of measured attributes

ID3(S, A):

1. **If** all examples have same label
   Return a single node tree with the label

2. **Otherwise**
   1. Create a root node, R, for tree

   Decide what attribute goes at the top

   4. Return root node R

Basic decision tree algorithm: ID3

Input:
S is the set of examples
A is the set of measured attributes

ID3(S, A):

1. **If** all examples have same label
   Return a single node tree with the label

2. **Otherwise**
   1. Create a root node, R, for tree
   2. $A_b \in A$ is the attribute that <u>*best*</u> classifies S
   3. For each possible value v that A_b can take on

   Decide what attribute goes at the top

   Decide what to do for each value root attribute takes

   4. Return root node R

Basic decision tree algorithm: ID3

Input:
S is the set of examples
A is the set of measured attributes

ID3(S, A):

1. **If** all examples have same label
   Return a single node tree with the label

2. **Otherwise**
   1. Create a root node, R, for tree
   2. $A_b \in A$ is the attribute that ***best*** classifies S
   3. For each possible value v that A_b can take on
      - Add a new tree branch for attribute A_b taking value v
      - Let $S_v \subseteq S$ be the subset of examples with $A_b = v$
      - If $S_v = \emptyset$: add leaf node with the common value of label in S

        Else: below this branch add the subtree ID3($S_v, A - \{A_b\}$)
   4. Return root node R

Decide what attribute goes at the top

Decide what to do for each value root attribute takes

Why?

For generalization at test time

Recursive call to the Id3 algorithm with all the remaining attributes