CMSC 422 Introduction to Machine Learning

Lecture 4 Decision Trees

Furong Huang / furongh@umd.edu

UNIVERSITY OF MARYLAND

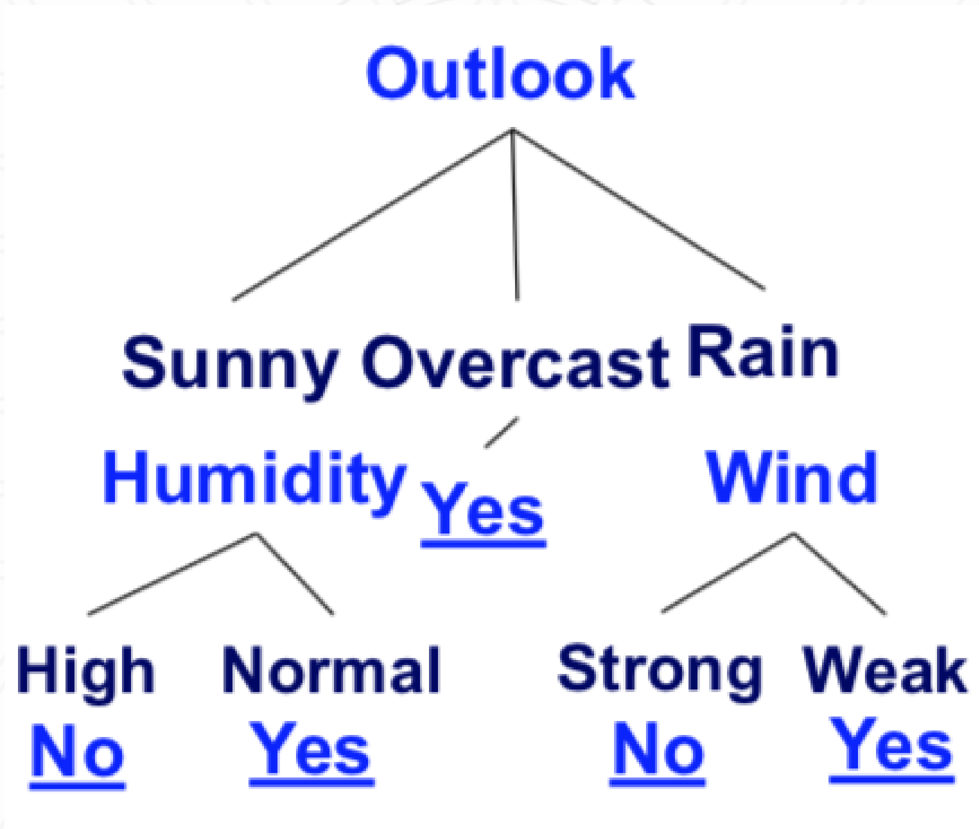
Decision Trees

- **What is a decision tree?**
- How to learn a decision tree from data?
- What is the inductive bias?
- Generalization?

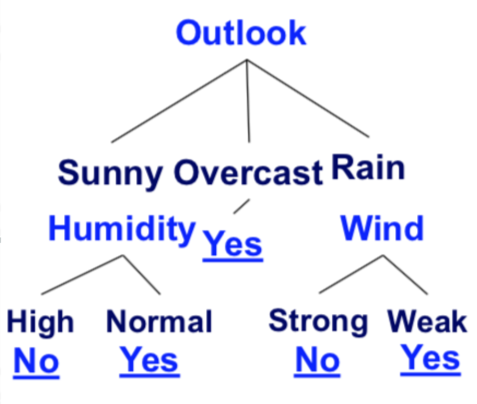
An example training set

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

A decision tree
to decide whether to play tennis



Today: Decision Trees



- Representation
  - Each internal node tests a feature
  - Each branch corresponds to a feature value
  - Each leaf node assigns a classification
    - or a probability distribution over classifications
- Decision trees represent functions that map examples in X to classes in Y

f: <Outlook, Temperature, Humidity, Wind> → PlayTennis?

Exercise

How would you represent the following Boolean functions with decision trees?

$$(A \cap B)$$

$$(A \cup B)$$

Take home exercise: $(A \cap B) \cup (C \cap \neg D)$

Today: Decision Trees

- What is a decision tree?
- **How to learn a decision tree from data?**
- What is the inductive bias?
- Generalization?

Function Approximation with Decision Trees

- Problem setting
  - Set of possible instances X
    - Each instance $x \in X$ is a feature vector $x = [x_1, \ldots, x_D]$
  - 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
- Input
  - Training examples $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots (\mathbf{x}^{(N)}, y^{(N)})\}$ of unknown target function f
- Output
  - Hypothesis $h \in H$ that best approximates target function f

Decision Trees Learning

- Finding the hypothesis $h \in H$
  - That minimizes training error
  - Or maximizes training accuracy

- How?
  - H is too large for exhaustive search!
  - We will use a heuristic search algorithm which
    - Picks questions to ask, in order
    - Such that classification accuracy is maximized

Top-down Induction of Decision Trees

```
CurrentNode = Root

DTtrain (examples for CurrentNode, features at
CurrentNode):
    1. Find F, the “best” decision feature for next
       node
    2. For each value of F, create new descendant of
       node
    3. Sort training examples to leaf nodes
    4. If training examples perfectly classified
        Stop
       Else
        Recursively apply DTtrain over new leaf nodes
```

How to select the “best” feature?

- A good feature is a feature that lets us make correct classification decision

- One way to do this:
  - select features based on their classification accuracy

- Let’s try it on the PlayTennis dataset

Let's build a decision tree using features O, T, H, W

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Partitioning examples according to Humidity feature

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

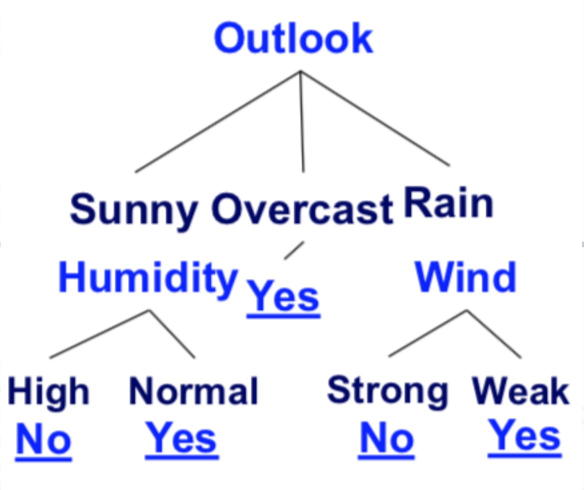
Partitioning examples: H = Normal

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Partitioning examples:
H = Normal and W = Strong

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision Trees



- Can represent any Boolean Function
- Can be viewed as a way to compactly represent a lot of data.
- The evaluation of the Classifier is easy
- Clearly, given data, there are many ways to represent it as a decision tree.
- Learning a good representation from data is the challenge.

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	-

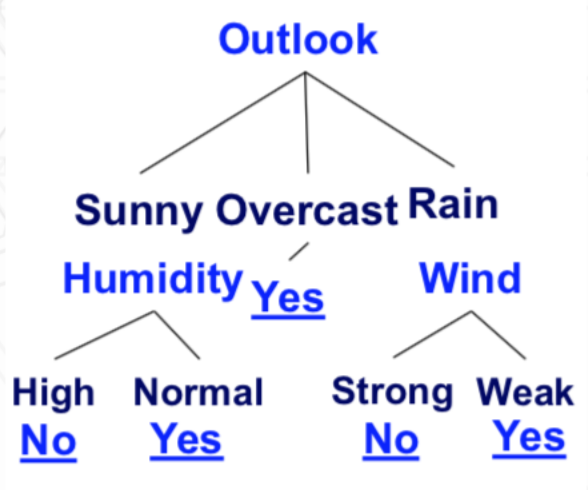
- Outlook: S(unny), O(vercast), R(ainy)
- Temperature: H(ot), M(ild), C(ool)
- Humidity: H(igh), N(ormal), L(ow)
- Wind: S(trong), W(eak)

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	-

- Data is processed in Batch (i.e. all the data available)
- Recursively build a decision tree top down.

Algorithm?



Top-down Induction of Decision Trees

```
CurrentNode = Root

DTtrain (examples for CurrentNode, features at
CurrentNode):
    1. Find F, the "best" decision feature for next
       node
    2. For each value of F, create new descendant of
       node
    3. Sort training examples to leaf nodes
    4. If training examples perfectly classified
        Stop
       Else
        Recursively apply DTtrain over new leaf nodes
```

Picking the Root Attribute

- The goal is to have the resulting decision tree as small as possible (Occam's Razor)
  - However, finding the minimal decision tree consistent with the data is NP-hard
- The recursive algorithm is a greedy heuristic search for a simple tree, but cannot guarantee optimality.
- The main decision in the algorithm is the selection of the next attribute to condition on.

Picking the Root Attribute

Training Data with 2 Boolean attributes (A,B)

((A=0,B=0), -): 50 examples

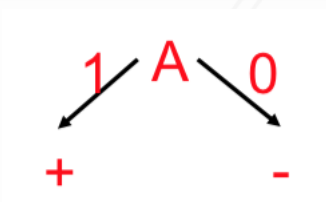
((A=0,B=1), -): 50 examples

((A=1,B=0), -): 0 examples

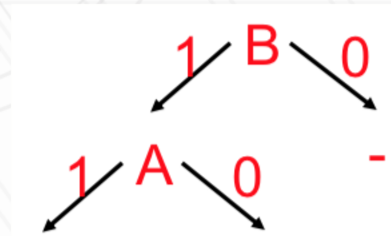
((A=1,B=1), +): 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes



Splitting on B: we don't get purely labeled nodes



What if we have ((A=1,B=0), -): 3 examples?

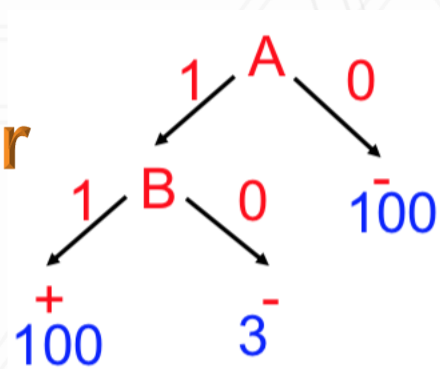
Picking the Root Attribute

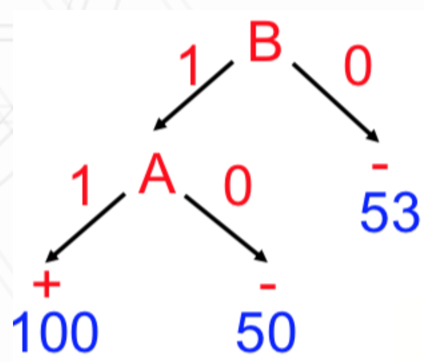
Training Data with 2 Boolean attributes (A,B)

((A=0,B=0), -): 50 examples
((A=0,B=1), -): 50 examples
((A=1,B=0), -): 3 examples
((A=1,B=1), +): 100 examples

Trees looks structurally similar; which attribute should we choose?

Better

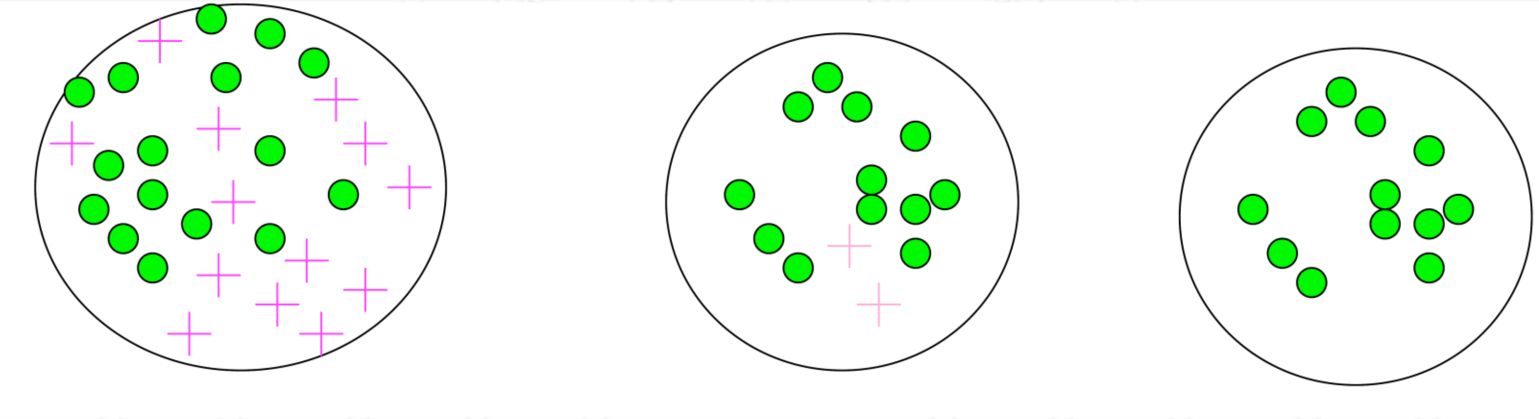




How to quantitatively choose?

Another feature selection criterion: Entropy

- Used in the ID3 algorithm [Quinlan, 1963]
  - pick feature with smallest entropy to split the examples at current iteration
- Entropy measures impurity of a sample of examples



Entropy

Entropy $H(X)$ of a random variable X

# of possible values for X

$$H(X) = -\sum_{i=1}^{n} P(X=i) \log_2 P(X=i)$$

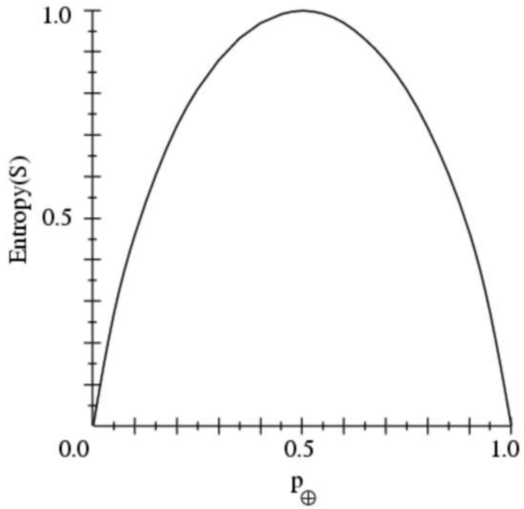
$H(X)$ is the expected number of bits needed to encode a randomly drawn value of X (under most efficient code)

Why? Information theory:

- Most efficient possible code assigns $-\log_2 P(X=i)$ bits to encode the message $X=i$
- So, expected number of bits to code one random X is:

$$\sum_{i=1}^{n} P(X=i)(-\log_2 P(X=i))$$

Sample Entropy



High Entropy –
High level of uncertainty

Low Entropy –
Low level of uncertainty

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

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

Information Gain

- The information gain of an attribute a is the expected reduction in entropy caused by partitioning on this attribute

$$Gain(S, a) = Entropy(S) - \sum_{v \in values(a)} \frac{|S_v|}{|S|} Entropy(S_v)$$

  - Original set is S.
  - S_v is the subset of S for which attribute a has value v.
  - The entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set.
    - Partitions of low entropy (imbalanced splits) lead to high gain
  - Take Home Exercise: go back and check which of the A, B splits is better.

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(ild), C(ool)
- Humidity: H(igh), N(ormal), L(ow)
- Wind: S(trong), W(eak)

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	-

- Current entropy:
  P = 9/14
  N =5/14
- H(Y)=
  $-(9/14)\log_2(9/14)$
  $-(5/14)\log_2(5/14)$
  ≈ 0.94

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 = sunny:
  $p=2/5$ $n=3/5$ $H_S=0.971$
- Outlook =overcast:
  $p=4/4$ $n=0$ $H_O=0$
- Outlook = rainy:
  $p=3/5$ $n=2/5$ $H_R=0.971$
- Expected entropy
  $$\left(\frac{5}{14}\right) \times 0.971 + \left(\frac{4}{14}\right) \times 0 + \left(\frac{5}{14}\right) \times 0.971 = \mathbf{0.694}$$
- Information gain:
  0.940-0.694 = **0.246**

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	-

- Humidity = sunny:
  p=3/7 n =4/7 H_H=0.985
- Humidity =overcast:
  p=6/7 n =1/7 H_O=0.592
- Expected entropy
  $$\left(\frac{7}{14}\right) \times 0.985 + \left(\frac{7}{14}\right) \times 0.592$$
  $$= \mathbf{0.7785}$$
- Information gain:
  0.940-0.7785 = **0.1515**

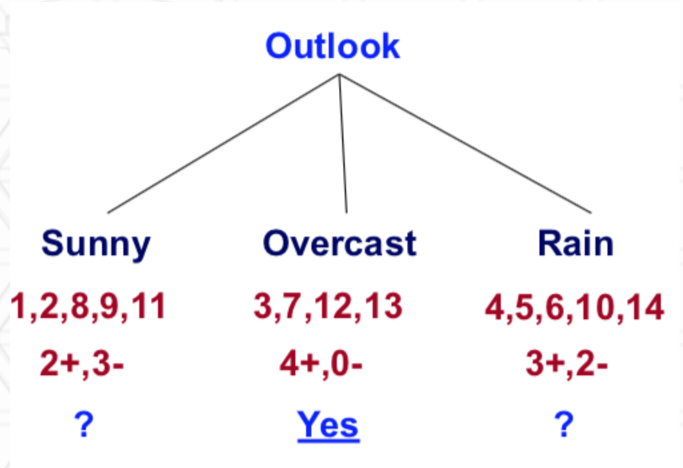
Which feature to split on?

	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	-

- Information gain:
  - Outlook: **0.246**
  - Humidity: 0.151
  - Wind: 0.048
  - Temperature: 0.029
- ✓ Split on Outlook

An Illustrative Example (III)

	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	-



Continue until:

- Every attribute is included in path, or,
- All examples in the leaf have same label

An Illustrative Example (IV)

$Gain(S_{sunny}, Humidity) = .97-(3/5)\ 0-(2/5)\ 0 = .97$

$Gain(S_{sunny}, Temp) = .97-\ 0-(2/5)\ 1 = .57$

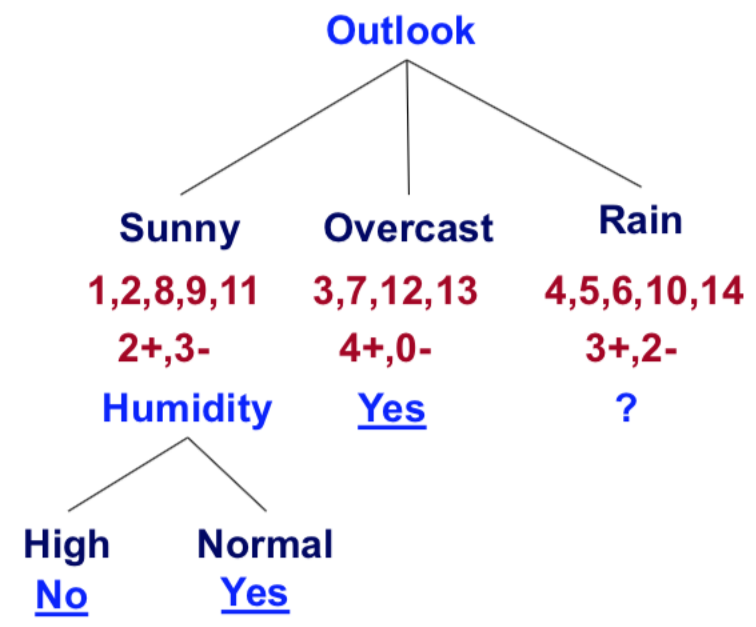
$Gain(S_{sunny}, Wind) = .97-(2/5)\ 1 - (3/5)\ .92 = .02$

Outlook

Sunny	Overcast	Rain
1,2,8,9,11	3,7,12,13	4,5,6,10,14
2+,3-	4+,0-	3+,2-
?	Yes	?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

An Illustrative Example (V)



induceDecisionTree(S)

1. Does S uniquely define a class?

   if all $s \in S$ have the same label y: **return** S;

2. Find the feature with the most information gain:

   $i = \text{argmax}_i Gain(S, X_i)$

3. Add children to S :

   for k **in** Values(X_i):

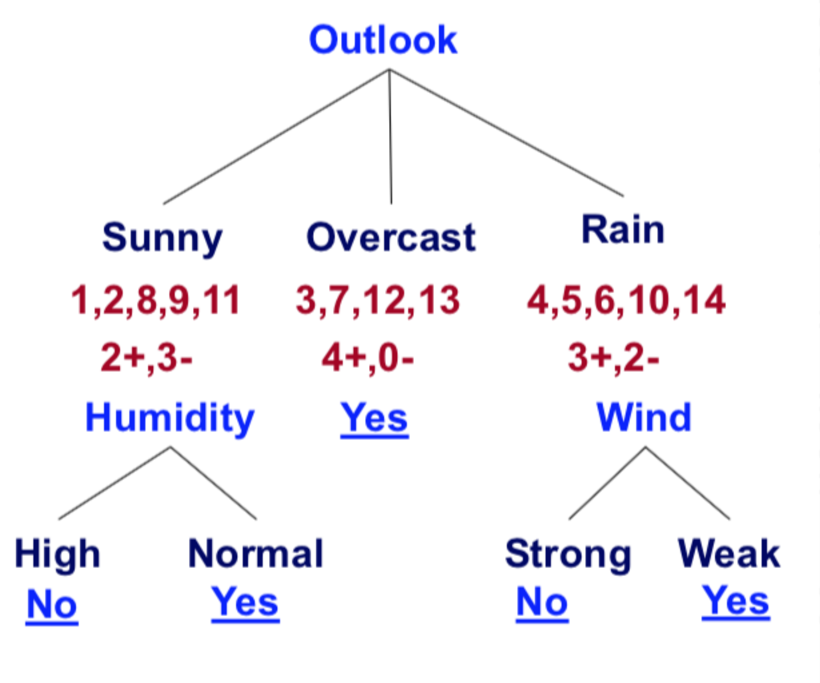
   $S_k = \{s \in S | X_i = k\}$

   addChild(S , S_k)

   induceDecisionTree(S_k)

   return S;

An Illustrative Example (VI)



Hypothesis Space in Decision Tree Induction

- Conduct a search of the space of decision trees which can represent all possible discrete functions. (pros and cons)
- Goal: to find the best decision tree
- Finding a minimal decision tree consistent with a set of data is NP-hard.
- Performs a greedy heuristic search: hill climbing without backtracking.
- Makes statistically based decisions using all data.

A decision tree to distinguish homes in New York from homes in San Francisco

Take a look at home:

http://www.r2d3.us/visual-intro-to-machine-learning-part-1/

- Check out course webpage, Canvas, Piazza
- Do the readings!

UNIVERSITY OF MARYLAND

Furong Huang
4124 IRB, College Park, MD 20740
301.405.8010 / furongh@umd.edu