

# Decision Trees

*Learning If-Else-Then Rules from Data!*

Matthew Eicholtz

Assistant Professor of Computer Science

Florida Southern College

[meicholtz@flsouthern.edu](mailto:meicholtz@flsouthern.edu)

# Can you spot the pattern?

0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

# How about now?

1	2	5	0	0	1	1	0	5	-	A	1	-	\$7.25	-0.410	red	0
0	4	2	0	0	1	1	1	2	-	C	1	-	\$8.45	-0.248	green	1
0	1	1	0	1	0	0	0	8	-	B	0	yes	\$5.37	2.979	blue	0
1	3	2	1	0	0	0	1	1	6	A	0	-	\$18.34	2.818	orange	0
0	1	3	1	1	1	-	1	9	21	A	1	no	\$22.59	2.834	green	1
0	4	3	1	0	0	0	0	8	54	B	0	-	\$8.62	1.343	purple	1
0	2	5	0	1	1	1	0	7	32	C	0	no	\$3.19	-2.415	blue	1
1	1	5	0	0	1	1	1	3	1	C	1	-	\$13.08	1.435	green	0
1	4	1	0	1	0	-	0	2	-	B	0	-	\$35.78	3.261	orange	1
0	4	4	1	0	0	-	0	2	11	A	1	no	\$27.70	0.9778	purple	0
1	2	4	1	1	0	1	0	1	32	B	0	yes	\$13.50	2.069	red	0
1	3	4	1	0	1	0	1	9	10	B	1	yes	\$30.35	1.45	red	1
1	2	2	0	1	0	0	0	8	-	C	1	no	\$7.15	-0.607	blue	1

# Definitions

The task of **supervised learning** is this:

Given a training set of  $N$  example input-output pairs

$$(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)$$

where each  $y_j$  was generated by an unknown function  $y = f(x)$ ,

discover a function  $h$  that approximates the true function  $f$ .

A hypothesis is **consistent** if it agrees with the data.

A hypothesis **generalizes** well if it predicts the correct output on novel examples.

A learning problem is **realizable** if the hypothesis space contains the true function.

Learning problems: classification v. regression

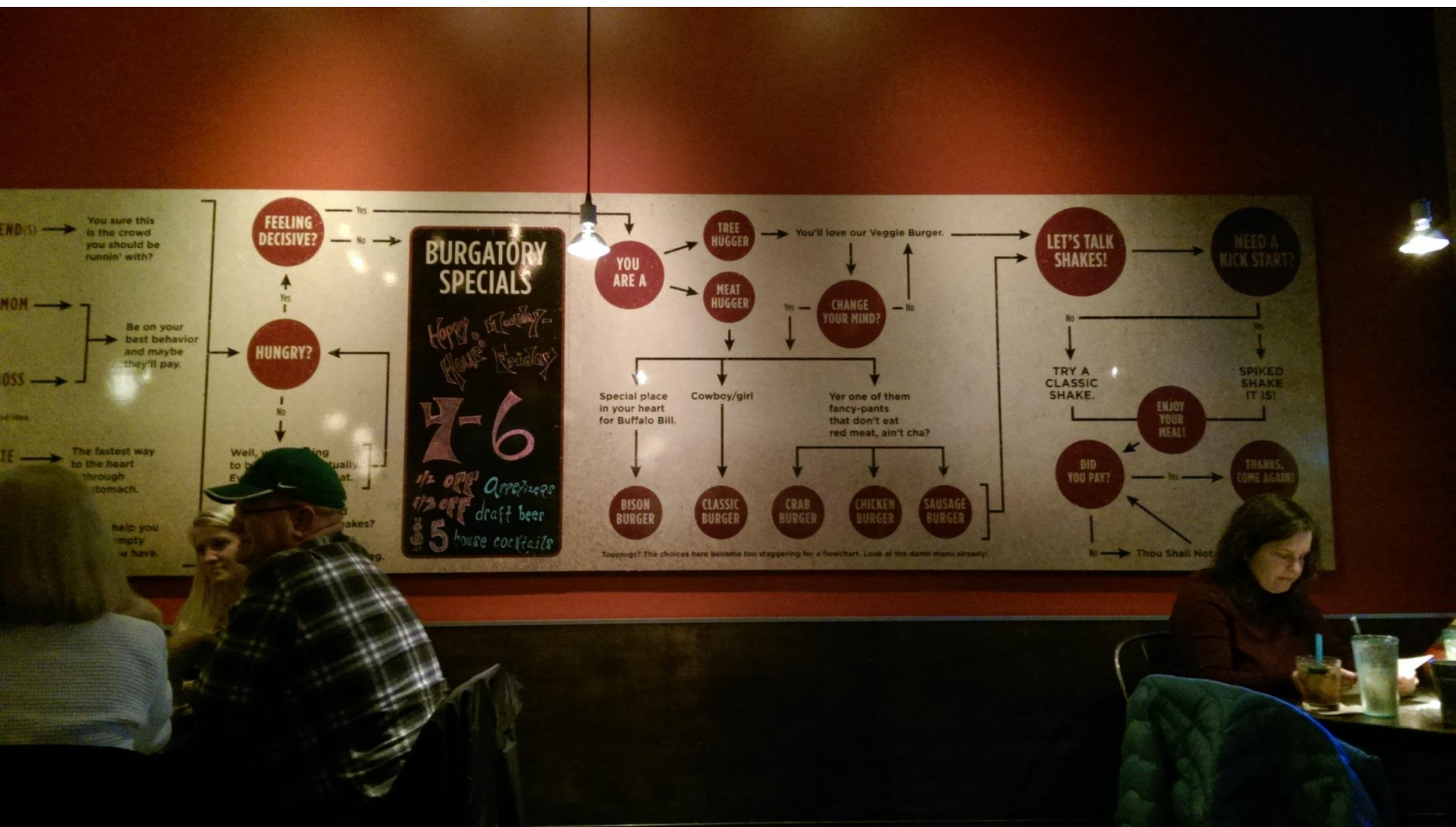
(Russell and Norvig pp. 695-697)

# Definitions

A **decision tree** is a tree-structured plan of a set of attributes to test in order to predict the output.

The tree contains **nodes** connected by **branches**.

The top node is the **root**, which is linked to **descendants**, until **leaf nodes** are reached.



# Questions to consider...

How do you learn a decision tree using example input-output pairs?

How do you test a decision tree on a new example?

What if some attributes have multiple values? What if they are continuous?

What if the output has multiple values? What if it is continuous?

What if data is missing?

How do you choose which attributes to split on?

When should you stop splitting the data?

How can you tell if a decision tree is good?

What if a decision tree performs well for the data it was trained on, but does not perform well on new data?



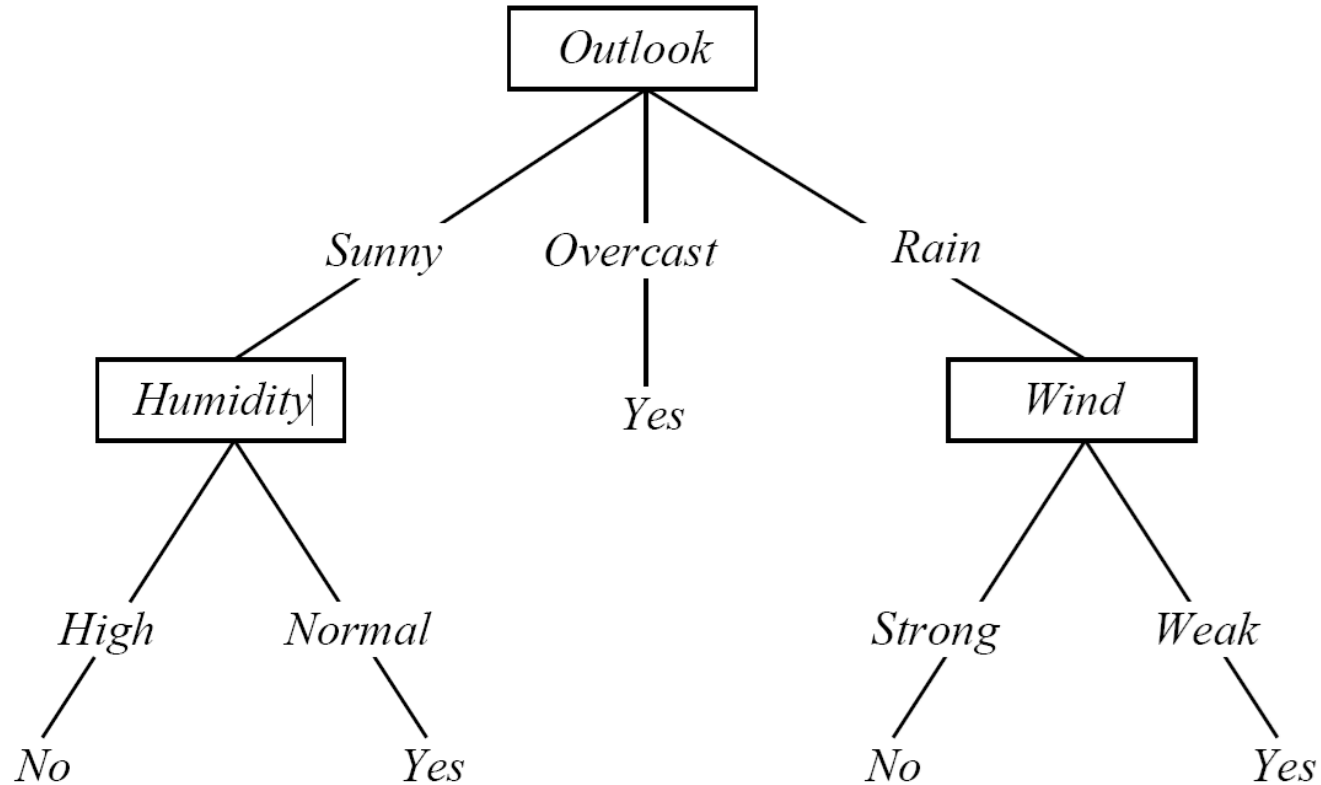
# Guess Who?



<https://youtu.be/FRIbNOno5VA>



# Some examples...



1. There are **multiple trees** that approximate the **same function**.

2. **Any** decision tree has an **equivalent binary** decision tree.

*Branching factor ( $B$ ) of a node = number of discrete values node can take  
When  $B=2$  always  $\rightarrow$  binary decision tree*

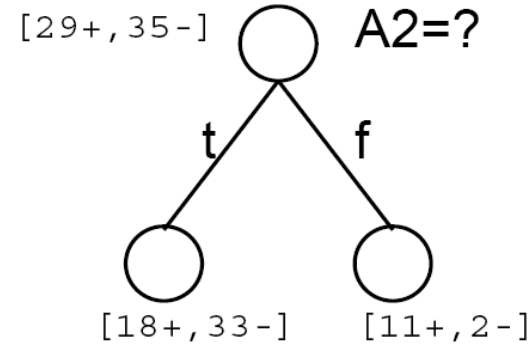
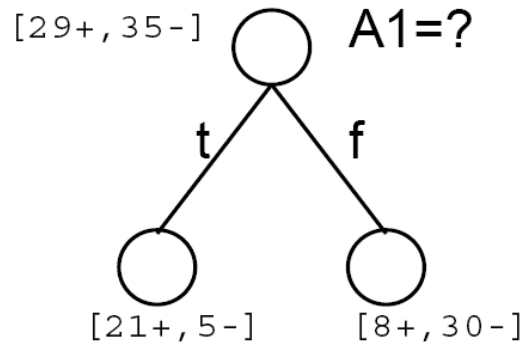
## Top-down induction

1. Determine the “*best*” *decision attribute* from those available
2. Assign that attribute to the next node
3. For each value of that attribute, create a descendant node
4. Sort training examples for each descendant
5. Recursively build tree from each descendant until *stopping criteria* is met
6. At each leaf node, assign the appropriate classification

## What are the stopping criteria?

- The subset of training examples have the same output
- The subset of training examples have the same values for all input attributes
- There are no training examples for a particular leaf node

## How to determine “best” decision attribute?



**Entropy** is a measure of the uncertainty associated with a random variable

- Suppose you are receiving a set of independent random samples of  $X$
- You see that  $X$  has four possible values with equal probabilities

$$P(X = A) = \frac{1}{4}, \quad P(X = B) = \frac{1}{4}, \quad P(X = C) = \frac{1}{4}, \quad P(X = D) = \frac{1}{4}$$

*BAACBADCDADDDA ...*

- You need to transmit the data over a binary serial link. You can encode each reading with two bits:

0100001001001110110011111100 ...



# Entropy

- Now suppose someone tells you the probabilities are not equal

$$P(X = A) = \frac{1}{2}, \quad P(X = B) = \frac{1}{4}, \quad P(X = C) = \frac{1}{8}, \quad P(X = D) = \frac{1}{8}$$

- It is possible to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

# Entropy

- What happens if there are three equally likely values for  $X$ ?

$$P(X = A) = \frac{1}{3}, \quad P(X = B) = \frac{1}{3}, \quad P(X = C) = \frac{1}{3}$$

- A naïve coding (2 bits per symbol)

$A \rightarrow 00$

$B \rightarrow 01$

$C \rightarrow 10$

- Can you think of an encoding scheme that would need only 1.6 bits per symbol on average?

- In theory, it can actually be done with 1.58496 bits per symbol.

# Entropy

The **entropy** ( $H$ ) of a discrete random variable  $X$  with  $k$  possible values represents the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from the distribution of  $X$ .

$$H(X) = - \sum_{j=1}^k p_j \log_2 p_j$$

**High entropy** means  $X$  is from a uniform distribution

**Low entropy** means  $X$  is from a varied distribution (peaks and valleys)

# Entropy

$$H(X) = - \sum_{j=1}^k p_j \log_2 p_j$$

Back to our previous examples...

$$1. P(X = A) = \frac{1}{4}, \quad P(X = B) = \frac{1}{4}, \quad P(X = C) = \frac{1}{4}, \quad P(X = D) = \frac{1}{4}$$

$$2. P(X = A) = \frac{1}{2}, \quad P(X = B) = \frac{1}{4}, \quad P(X = C) = \frac{1}{8}, \quad P(X = D) = \frac{1}{8}$$

$$3. P(X = A) = \frac{1}{3}, \quad P(X = B) = \frac{1}{3}, \quad P(X = C) = \frac{1}{3}$$

# Specific Conditional Entropy

The **specific conditional entropy**,  $H(Y|X = x_i)$ , assumes you already know the value of  $X$  and want to measure the uncertainty of  $Y$  for that subset

$$H(Y|X = x_i) = - \sum_{j=1}^k P(Y = y_j|X = x_i) \log_2 P(Y = y_j|X = x_i)$$

# Conditional Entropy

The **conditional entropy**,  $H(Y|X)$ , is the average specific conditional entropy of  $Y$

$$H(Y|X) = \sum_{j=1}^k P(X = x_j) H(Y|X = x_j)$$

In other words, it is the entropy of  $Y$  given a randomly chosen example of  $X$

# Information Gain

The **information gain**,  $IG(Y|X)$ , represents the expected reduction in entropy for a test on attribute  $X$ :

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

How many bits would be saved by knowing  $X$ ?



# A simple example

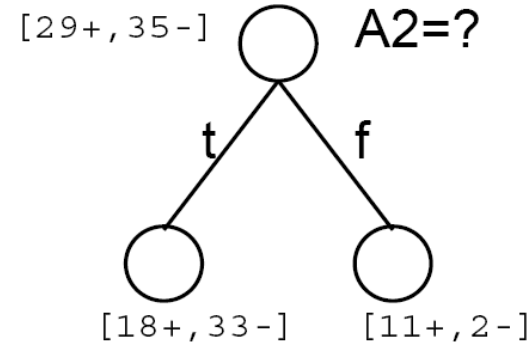
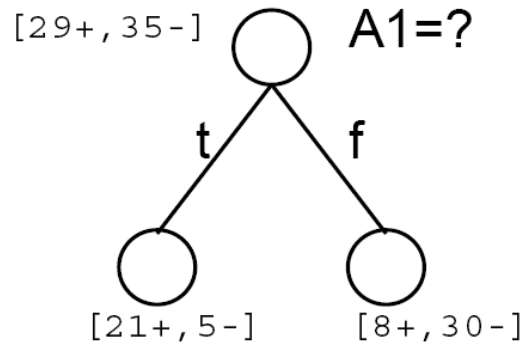
Suppose you want to predict  $Y$  and you know  $X$ :

$X$  = college major

$Y$  = likes Star Wars

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

## How to determine “best” decision attribute?



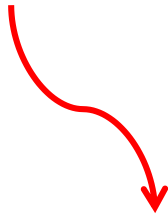
Use the one with the highest **information gain**!

## What are the stopping criteria?

- The subset of training examples have the same output
- The subset of training examples have the same values for all input attributes
- There are no training examples for a particular leaf node

## A new stopping criterion?

- The information gain is zero for all attributes???



Is this a good idea?



# How to be successful in this course?

- Come to class
- Take notes
- Be active participants
- Ask questions
- Solve suggested problems that we start in class on your own
- Come to office hours for 1-on-1 help
- Buy (and use!) the textbook
- Practice programming outside of class (e.g., Programming Team!)
- ...

# Let's build a decision tree

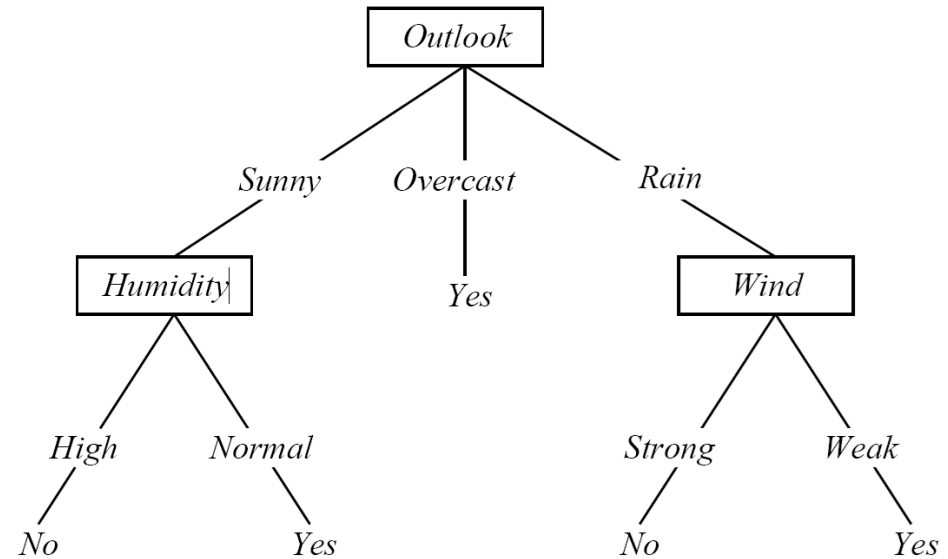
Should I play tennis today? Let's base our decision on the following factors: Outlook, Temperature, Humidity, and Wind.

Day	Outlook	Temp	Humidity	Wind	Play?
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

# Let's build a decision tree

Should I play tennis today? Let's base our decision on the following factors: Outlook, Temperature, Humidity, and Wind.

Day	Outlook	Temp	Humidity	Wind	Play?
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

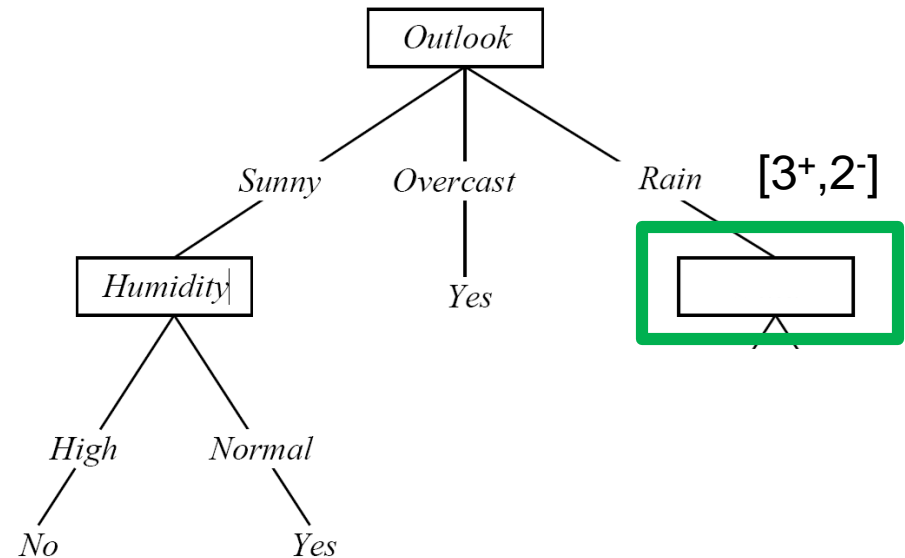




# Let's build a decision tree

Should I play tennis today? Let's base our decision on the following factors: Outlook, Temperature, Humidity, and Wind.

Day	Outlook	Temp	Humidity	Wind	Play?
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



# Let's build a decision tree

But wait...what if *Day* is also an attribute?

Day	Outlook	Temp	Humidity	Wind	Play?
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

# Information Gain Ratio

With the approach we've described so far, trees are biased toward using attributes that split into *many* branches.

We can reduce this bias by incorporating the **information gain ratio**, which is

$$\text{Information Gain Ratio} = \frac{IG(Y|X)}{-\sum_{j=1}^k p_j \log_2 p_j}$$

## How to handle continuous variables?

- Dynamically create attributes by choosing a threshold that maximizes information gain

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

**NOTE:** Continuous-valued attributes can be reused!

# Training Error

- For each example, follow the decision tree to see what it would predict
- For what number of examples does the decision tree's prediction disagree with the true value in the database?
- This quantity is called the **training error** (the smaller the better)

**But wait, why are we creating the decision tree in the first place?**

- Usually not to predict the output for data we have already seen.
- We want to predict the output for future data we have not yet seen.

# Testing Error

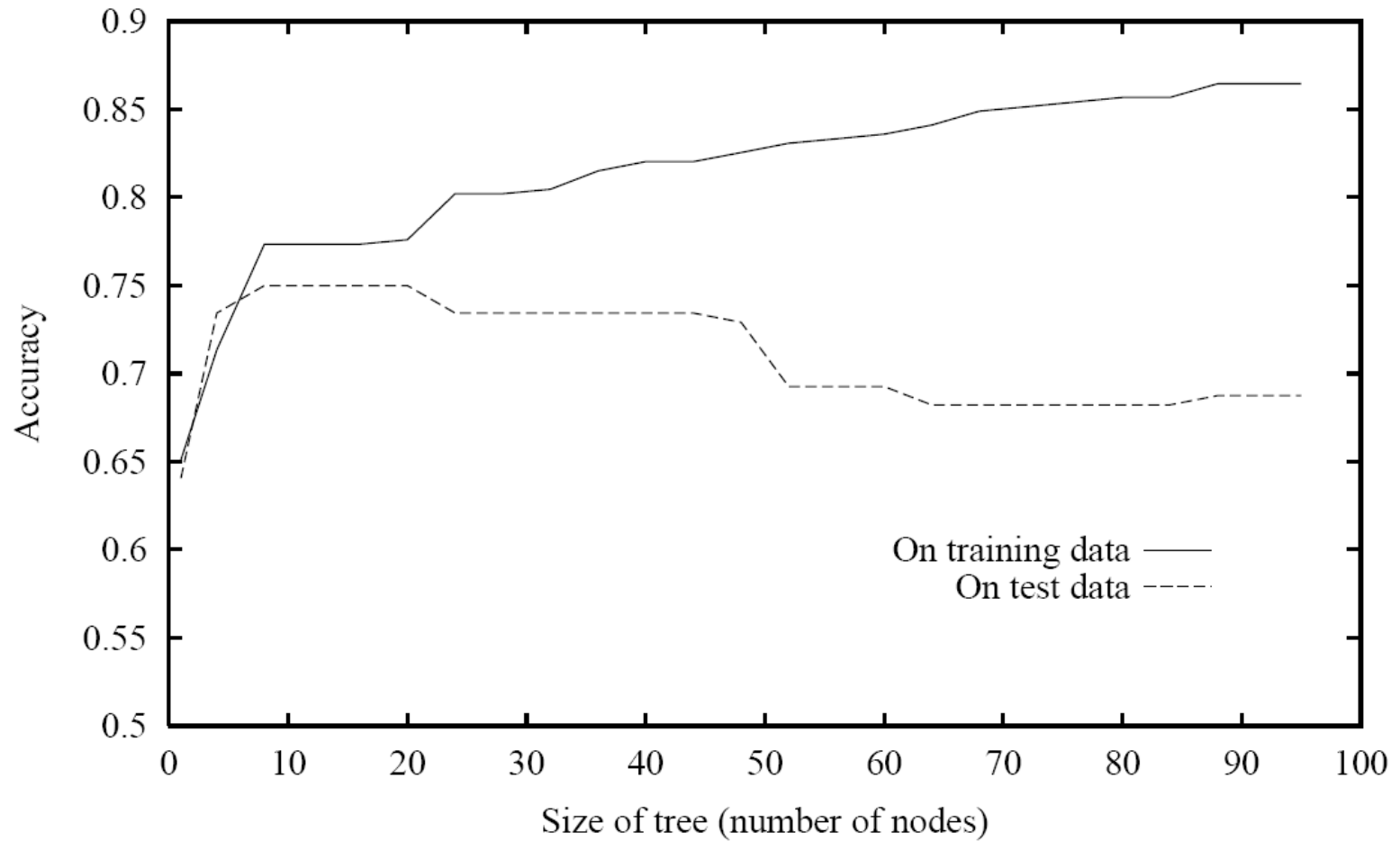
- For each example, follow the decision tree to see what it would predict
- For what number of examples does the decision tree's prediction disagree with the true value (if it is provided)?
- This quantity is called the **testing set error** (the smaller the better)

# Overfitting

- Perfect performance on training data does not mean the decision tree will work well for all cases.
- It may be **overfitting** the data!
- How do we measure/eliminate overfitting?



# Overfitting



# How to avoid overfitting?

- Generate several trees with different number of nodes
  - Full trees vs trees stopped early.
  - Say you create 5 such trees ( $T_1, T_2, \dots, T_5$ )
- Randomly choose 2/3 of the available data for training, use 1/3 of for testing (validation set).
  - Evaluate classification performance of each  $T_i$

$$\frac{\text{num\_of\_correctly\_classified\_instances\_in\_validation\_set}}{\text{size(validation set)}}$$

- Change the training set, re-evaluate
- Choose the  $T^*$  that gives the best overall performance

# Decision Tree Learning Algorithms

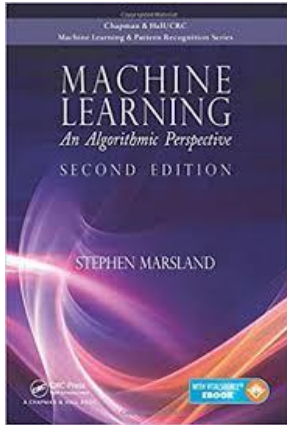
**ID3:** Iterative Dichotomiser 3; Ross Quinlan

**CART:** Classification and Regression Tree

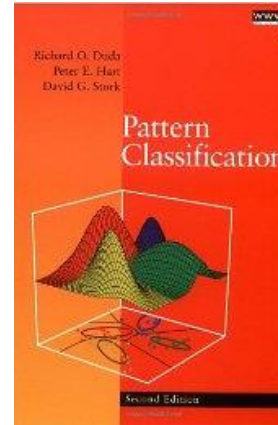
**C4.5:** also Ross Quinlan; improves upon ID3

**J48:** Java implementation of C4.5

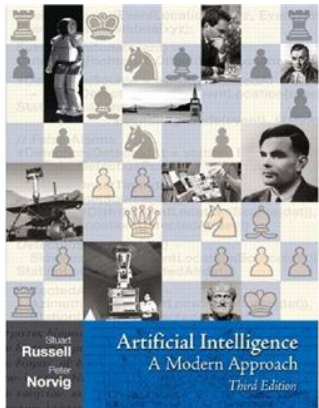
# References



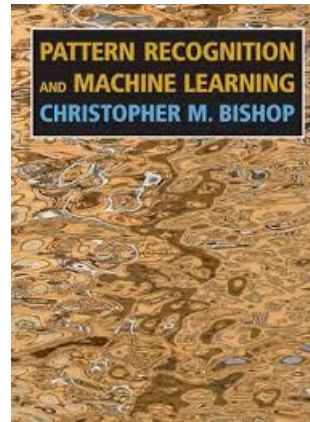
Ch. 12



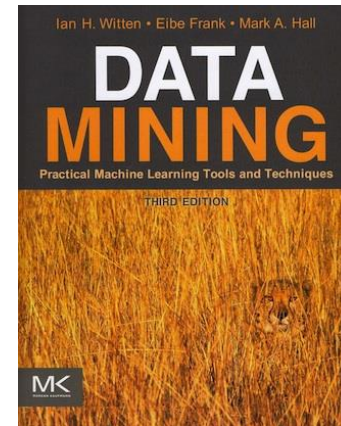
8.1 – 8.4, Appendix A.7



18.1 – 18.3



14.4



3.3, 4.3, 6.1

\*Some slides have been adapted from Andrew Moore (<http://www.autonlab.org/tutorials/>) and Levent Burak Kara

# Project Idea



[http://espn.go.com/mens-college-basketball/story/\\_/id/10328805/1-billion-offered-perfect-tournament-bracket&ex\\_cid=sportscenter](http://espn.go.com/mens-college-basketball/story/_/id/10328805/1-billion-offered-perfect-tournament-bracket&ex_cid=sportscenter)