Decision Trees

Wednesday, November 16, 2016 9:33 AM

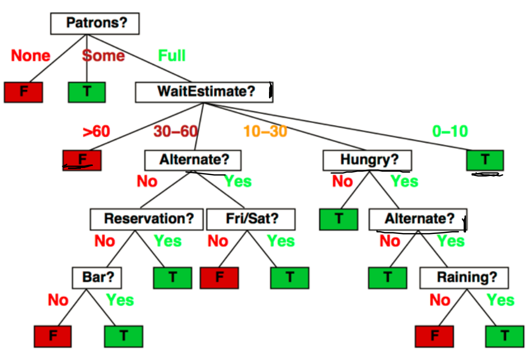
Attributes

Discrete data
Small data sets

- **Alternate**: suitable alternative restaurants nearby? (Y/N)
- **Bar**: A bar to wait in? (Y/N)
- **Fri/Sat** (Y/N)
- **Hungry** (Y/N)
- **Price**: ($, $$, $$$)
- **Raining**: (Y/N)
- **Reservation**: we made a reservation (Y/N)
- **Type**: Kind of restaurant (French, Italian, Thai, Burger)
- **WaitEstimate** (0-10, 10-30, 30-60, >60)
- **Patrons**: how busy? (none, some, full)

Decision trees

One possible representation for hypotheses
E.g., here is the "true" tree for deciding whether to wait:



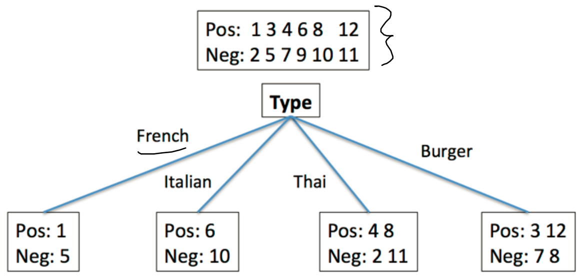
Chapter 18, Sections 1–3 14

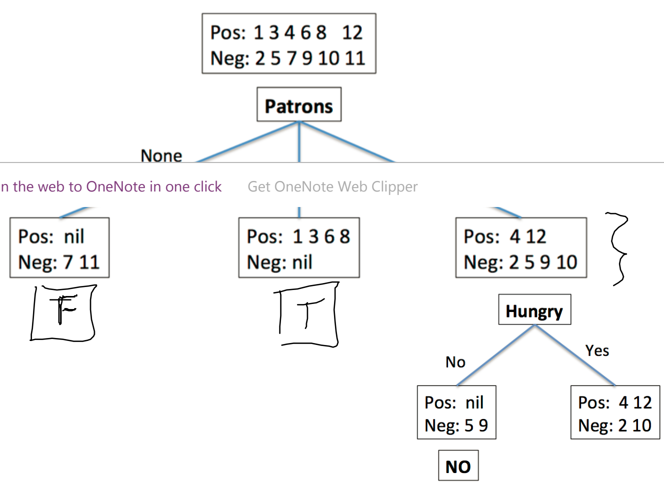
Examples described by attribute values (Boolean, discrete, continuous, etc.)
E.g., situations where I will/won't wait for a table:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
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

Classification of examples is positive (T) or negative (F)

Chapter 18, Sections 1–3 13

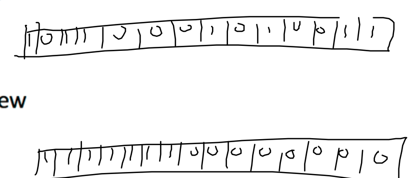




Which attribute to choose?

Information
= entropy

- The one that gives you the most information (aka the most diagnostic)
- Information theory
  - Answers the question: how much information does something contain?
  - Ask a question
  - Answer is information
  - Amount of information depends on how much you already knew
- Example: flipping a coin
  - If coin is random: 1 bit of information is gained
  - If you know the coin is weighted, there is less information gained because you could guess the outcome
  - Two-headed coin: 0 bits of information gained



- If there are n possible answers, $v_1 \ldots v_n$ and v_i has probability $P(v_i)$ of being the right answer, then the amount of information is:

$$I(P(v_1),\ldots,P(v_n)) = \sum_{i=1}^{n} -P(v_i)\log_2 P(v_i)$$

- Example: coin toss

all answers

Coin toss

v_1 = heads $P(v_1) = 0.5$

v_2 = tails $P(v_2) = 0.5$

$I(0.5, 0.5)$

$$I(\tfrac{1}{2}, \tfrac{1}{2}) = \sum_{i=1}^{2} -P(v_i)\log_2(P(v_i))$$
$$= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}$$
$$= 1 \text{ bit}$$

$$I\left(\tfrac{1}{100}, \tfrac{99}{100}\right) = -\frac{1}{100}\log_2\frac{1}{100} - \frac{99}{100}\log_2\frac{99}{100}$$
$$= 0.08 \text{ bit}$$

$$I\left(\tfrac{2}{2}, \tfrac{0}{2}\right) = 0 \text{ bits}$$

- **For a training set:**
  - p = # of positive examples
  - n = # of negative examples

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Probability of a positive example

Probability of a negative example

- **For our restaurant behavior**
  - p = n = 6
  - I() = 1
  - Would not be 1 if training set weren't 50/50 yes/no, but the point is to arrange attributes to increase information gain

Pos:	1 3 4 6 8 12
Neg:	2 5 7 9 10 11

$$I\left(\frac{6}{12}, \frac{6}{12}\right) = -\frac{6}{12}\log_2\frac{6}{12} - \frac{6}{12}\log_2\frac{6}{12}$$
$$= 1 \text{ bit}$$

Measuring attributes

- Information gain is a function of how much more information you need after applying an attribute
  - If I use attribute A next, how much more information will I need to account for?

examples true w/ the attribute value

examples attribute and false

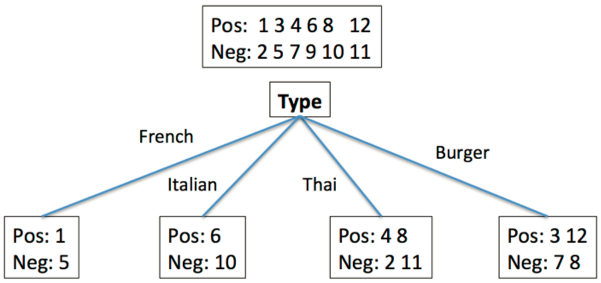
$$\text{Remainder(A)} = \sum_{i=1}^{|v|} \frac{p_i + n_i}{p+n} I\left(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\right)$$

Attribute

possible ~~constants~~ attribute values

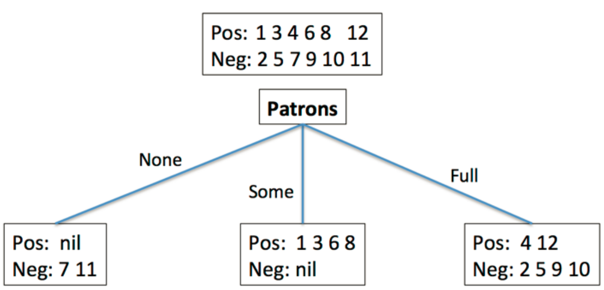
total examples

examples with this attribute value



$$\text{Remainder(type)} = \underbrace{\frac{2}{12}}_{\text{French}} I\left(\frac{1}{2},\frac{1}{2}\right) + \underbrace{\frac{2}{12}}_{\text{Italian}} I\left(\frac{1}{2},\frac{1}{2}\right) + \underbrace{\frac{4}{12}}_{\text{Thai}} I\left(\frac{2}{4},\frac{2}{4}\right) + \underbrace{\frac{4}{12}}_{\text{Burger}} I\left(\frac{2}{4},\frac{2}{4}\right) = 1 \text{ bit}$$

$$\frac{2}{12}(1\text{ bit}) + \frac{2}{12}(1\text{ bit}) + \frac{4}{12}(1\text{ bit}) + \frac{4}{12}(1\text{ bit}) = 1\text{ bit}$$



$$\text{Remainder(patrons)} = \underbrace{\frac{2}{12}}_{\text{none}} I\left(\frac{0}{2},\frac{2}{2}\right) + \underbrace{\frac{4}{12}}_{\text{some}} I\left(\frac{4}{4},\frac{0}{4}\right) + \underbrace{\frac{6}{12}}_{\text{full}} I\left(\frac{2}{6},\frac{4}{6}\right) \approx 0.459 \text{ bit}$$

0 bits + 0 bits +

- Not done yet
- Need to measure information **gained** by an attribute

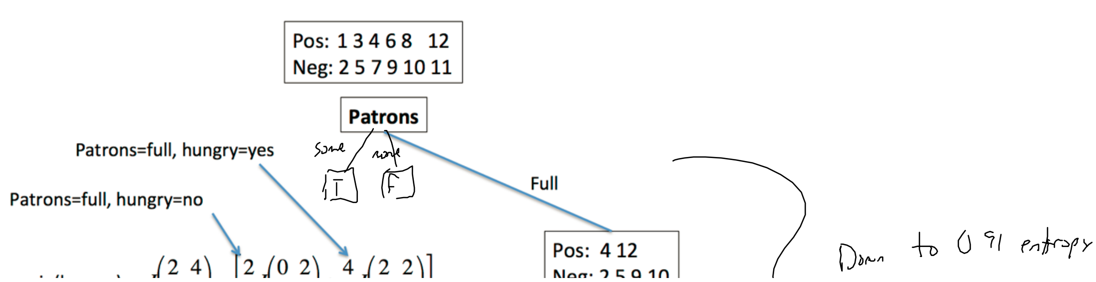
$$\text{Gain(A)} = \underbrace{I\left(\frac{p}{p+n},\frac{n}{p+n}\right)}_{\text{Info of the root of subtree}} - \text{remainder(A)}$$

- Pick the biggest

$$- \text{ Gain(type)} = \underbrace{I(6/12,6/12)}_{\text{total entropy}} - \underbrace{\left(\frac{2}{12}I\left(\frac{1}{2},\frac{1}{2}\right)+\frac{2}{12}I\left(\frac{1}{2},\frac{1}{2}\right)+\frac{4}{12}I\left(\frac{2}{4},\frac{2}{4}\right)+\frac{4}{12}I\left(\frac{2}{4},\frac{2}{4}\right)\right)}_{\text{Remainder (type)}}$$

$$- \text{ Gain(patrons)} = \underbrace{I(6/12,6/12)}_{\text{total entropy}} - \underbrace{\left(\frac{2}{12}I\left(\frac{0}{2},\frac{2}{2}\right)+\frac{4}{12}I\left(\frac{4}{4},\frac{0}{4}\right)+\frac{6}{12}I\left(\frac{2}{6},\frac{4}{6}\right)\right)}_{0.458 = \text{remainder(patrons)}}$$

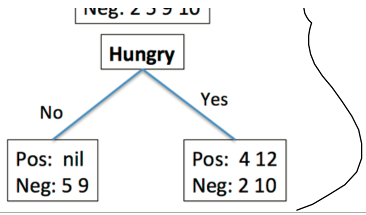
≈ 0.541 bits



gain(hungry) = $\underbrace{I\left(\frac{}{6},\frac{}{6}\right)}_{\text{6 remaining examples}} - \left[\underbrace{\frac{}{6} I\left(\frac{}{2},\frac{}{2}\right)}_{\text{no}} + \underbrace{\frac{}{6} I\left(\frac{}{4},\frac{}{4}\right)}_{\text{yes}}\right]$

= 0.9182958 – [0 + (4/6)(1)]

≈ 0.251 bits



Greedy search
Recursive

Decision-tree-learning ($\overbrace{\text{examples}}^{\text{data}}$, attributes, default)

OS

IF examples is empty THEN RETURN default

ELSE IF all examples have same classification THEN RETURN classification

ELSE IF attributes is empty RETURN majority-value(examples)

ELSE More examples, more attributes you haven't used

best = choose(attributes, example) ← Where info gain happens biggest info gain

tree = new decision tree with best as root

m = majority-value(examples)

FOREACH answer v_i of best DO

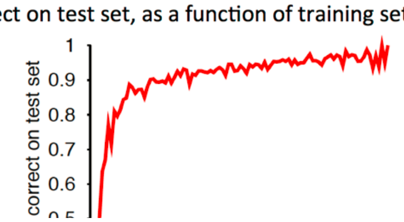
examples$_i$ = {elements of examples with best=v_i}

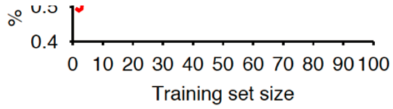
subtree$_i$ = **decision-tree-learning**(examples$_i$, attributes-{best}, m) recursive call

add a branch to tree based on v_i and subtree$_i$

RETURN tree

- Plot a learning curve
  - % correct on test set, as a function of training set size





- As training set grows, prediction quality should increase
  - Called a “happy graph”
  - There is a pattern in the data AND the algorithm is picking it up!