

DECISION TREES

Russell & Norvig AI:AMA 18.3

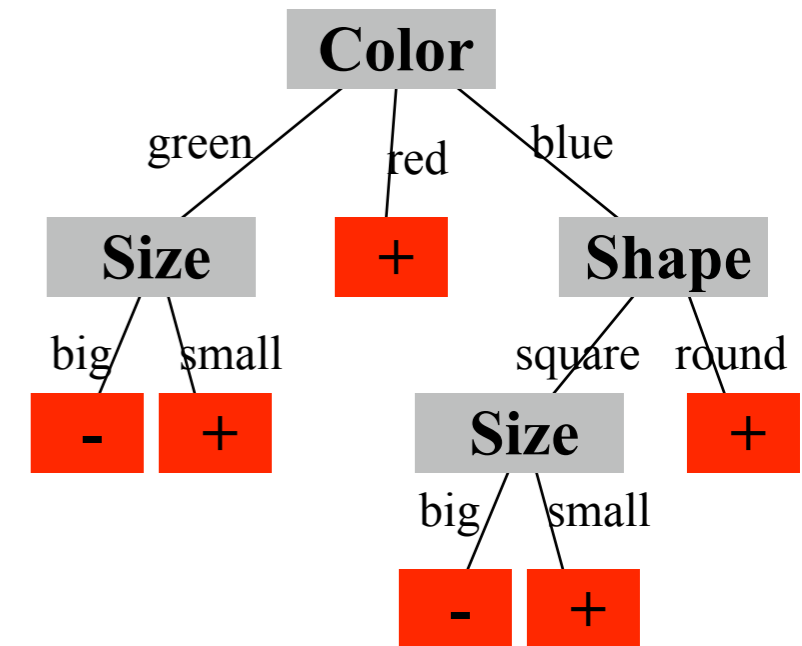
Slides by Don Miner

CMSC471

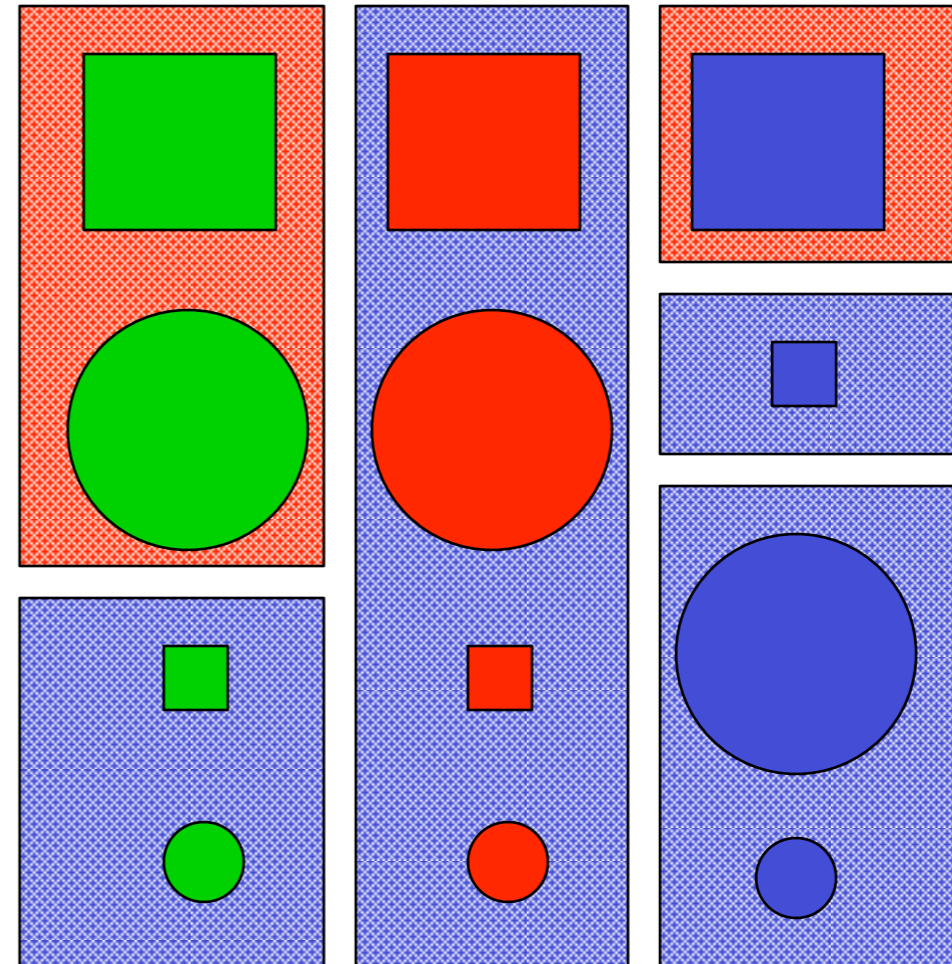
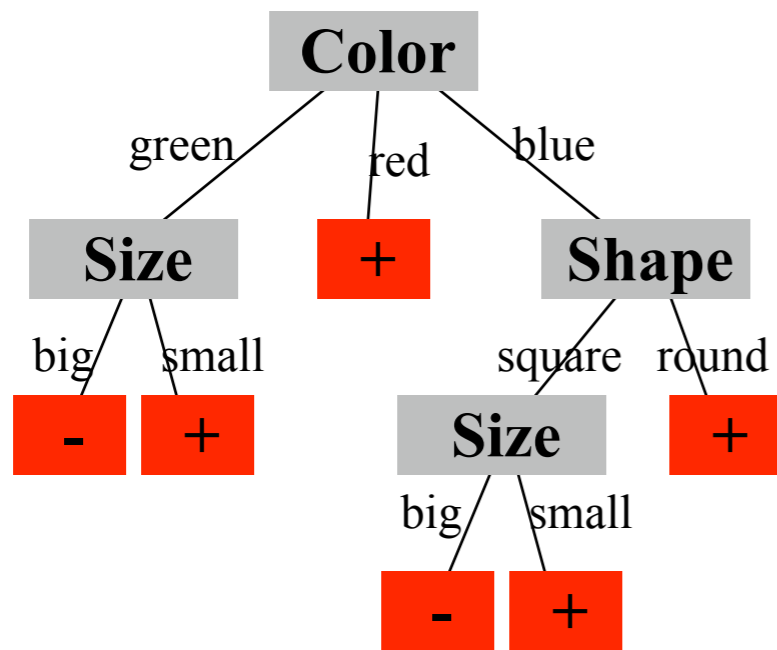
November 19, 2008

DECISION TREES

- Takes a set of **attributes** of a situation as input
- Returns a **decision** (the predicted value for the input)
- Boolean classification: decision is **positive** or **negative**
- Solution derived by running a series of tests on attributes; Leaf nodes contain the decision
- Any boolean function (propositional logic) can be written as a decision tree
 - Therefore, we are basically asking: what truth table will model our data well?



PARTITION EXAMPLE



INTERESTING PROPERTIES

- How many boolean functions are there for N attributes?
 - 2^N rows, each row can be true or false = $2^{(2^N)}$
 $2^{(2^6)} = 18,446,744,073,709,551,616$
- Can't represent tests that refer to two or more objects (e.g., there exists a cheaper restaurant nearby)
- Representing the parity function (half of inputs are true, half are false) require an exponentially large tree

INDUCING TREES FROM EXAMPLES

- An example: vector of input attributes and a single boolean output value
- We could create a decision tree with one leaf node per example
 - What do we do about new examples?
- We should instead find the smallest decision tree (Ockham's Razor)
 - Finding the smallest tree is intractable
 - Decision-Tree-Learning algorithm: test the most important attribute first (Figure 18.5, page 658)

DECISION TREE LEARNING ALGORITHM

- Pick an attribute which eliminates the most branches
 - For example:
Attribute splits 8 examples into 4 groups with 1 true and 1 false
vs.
Attribute splits 8 examples into 4 groups, with 4 groups with all true or all false values
 - Will people wait for a table at the restaurant? (restaurant ethnicity vs. fullness)
- Continue this process until all leaf nodes are reached

RESTAURANTS DOMAIN

- Some examples of situations:

French		Y	N
Italian		Y	N
Thai	N	Y	N Y
Burger	N	Y	N Y
	Empty	Some	Full

CHOOSING ATTRIBUTES

- Select attribute with highest “information gain”

- Number of information in **bits** is given by:

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

- Number of bits in a fair coin (what if it's not a fair coin?):

$$I\left(\frac{1}{2}, \frac{1}{2}\right) = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} = 1$$

- Estimated amount of information contained in a correct answer is (how much information do we need?) (e.g., 12 examples, 6 positive = 1):

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

CHOOSING ATTRIBUTES

- How much more information do we need to derive an answer with v subsets?

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

- So, how much information do we **gain** at a certain step in the decision tree by selecting attribute A ?

$$\textit{Gain}(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - \textit{Remainder}(A)$$

- Gain for choosing *Type* in the restaurant domain is 0; Gain for choosing *Patrons* is about .541

EXTENSIONS

- Missing attribute: how do you classify it? It has ALL of the values?
- When an attribute has many unique values, information gain becomes useless. What if we used *RestaurantName* as an attribute? Use a **gain ratio**: information value vs. gain?
- What if we have continuous values? Find the split point that gives the highest information gain.
- What if we want a continuous output value? Regression tree: each leaf is a linear function of its ancestors.

NAIVE BAYES

Russell & Norvig AI:AMA 20.2: Naive Bayes Models

http://en.wikipedia.org/wiki/Naive_Bayesian_classification

Slides by Don Miner

CMSC471 November 19, 2008

NAIVE BAYES

- We want to classify some instance F as a class C
- Naive Bayes (as with other Bayesian networks) returns the probability of C given F : $P(C|F)$ “the probability of C given F ”
- Naive: assume that the attributes are conditionally independent

- $$P(C|F) = \frac{1}{Z} P(C) \prod_{i=1}^n P(F_i|C)$$

NAIVE BAYES DERIVATION

(1) We want C: $P(C|F_1, \dots, F_n)$

(2) Bayes' Theorem: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

(3) Apply 2 to 1: $P(C|F_1, \dots, F_n) = P(C)P(F_1, \dots, F_n|C)$

(4) Expand 3:

$$P(C)P(F_1, \dots, F_n|C) = P(C)P(F_1|C)P(F_2|C, F_1)P(F_3|C, F_1, F_2)\dots P(F_n|C, F_1, \dots, F_{n-1})$$

(5) Apply naiveness to 4:

$$P(C, F_1, \dots, F_n) = P(C)P(F_1|C)P(F_2|C)P(F_3|C)\dots$$

(6) Express 5 as a product (Z normalizes probabilities): $\frac{1}{Z}P(C) \prod_{i=1}^n P(F_i|C)$

K-NEAREST NEIGHBOR

Russell & Norvig AI:AMA 20.4: Nearest-neighbor models

Slides by Don Miner

CMSC471 November 19, 2008

K-NEAREST NEIGHBOR

- Concept: properties of a input point \mathbf{x} are likely similar to points in its neighborhood
- k -nearest neighbor: define a neighborhood big enough to have k points
 - k too small: highly variable data
 - k too big: structure of data is lost
 - k just right: good reconstruction

KNN EXAMPLE: DENSITY

- We randomly sample data points in a density map $(x, y) \rightarrow$ density
- (x, y) data samples will be more frequent in denser areas
- Given (x, y) , what will the density be?
- Find the k -nearest neighbors
- In a sparse region, the size of the neighborhood is big

KNN SCALING & DISCRETE VALUES

- What if x, y weren't on the same scale?
- Standardize each dimension (e.g., make all values 0-1, proportionally)
- Mahalanobis distance - scales but takes into consideration the covariance
- Hamming distance - the number of differences (e.g., (A, C, F) vs. (A, F, D) is 2)

KNN EXAMPLE: FUNCTION APPROXIMATION

- Suppose we have a bunch of examples of $y = h(\mathbf{x})$
- Take the k nearest points to \mathbf{x} and average the values of y
- If y is discrete we can take a majority vote