Decision Trees

CS 540 section 2
Louis Oliphant
oliphant@cs.wisc.edu

Inductive learning

- Simplest form: learn a function from examples

f is the target function

An example is a pair (*x*, *f(x)*)

Problem: find a hypothesis *h*
 such that $h \approx f$
 given a training set of examples

(This is a highly simplified model of real learning:
 - Ignores prior knowledge
 - Assumes examples are given)

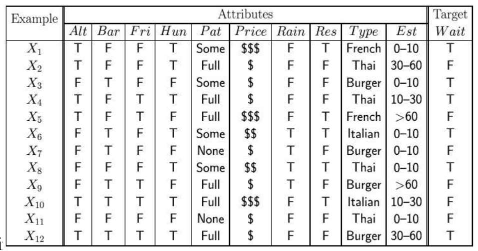
Learning decision trees

Problem: decide whether to wait for a table at a restaurant, based on the following attributes:

1. Alternate: is there an alternative restaurant nearby?
2. Bar: is there a comfortable bar area to wait in?
3. Fri/Sat: is today Friday or Saturday?
4. Hungry: are we hungry?
5. Patrons: number of people in the restaurant (None, Some, Full)
6. Price: price range ($, $$, $$$)
7. Raining: is it raining outside?
8. Reservation: have we made a reservation?
9. Type: kind of restaurant (French, Italian, Thai, Burger)
10. WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60)

Attribute-based representations

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

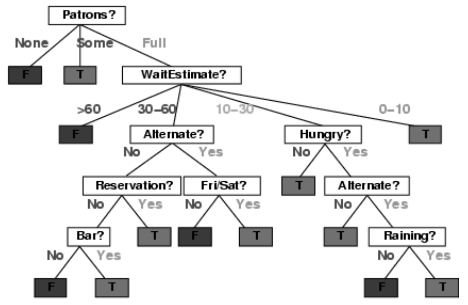


Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	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

- Classi

Decision trees

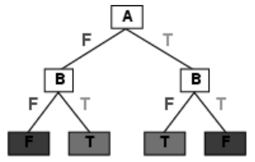
- One possible representation for hypotheses
- E.g., here is the "true" tree for deciding whether to wait:



Expressiveness

- Decision trees can express any function of the input attributes.
- E.g., for Boolean functions, truth table row → path to leaf:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- Trivially, there is a consistent decision tree for any training set with one path to leaf for each example (unless *f* nondeterministic in *x*) but it probably won't generalize to new examples
- Prefer to find more compact decision trees

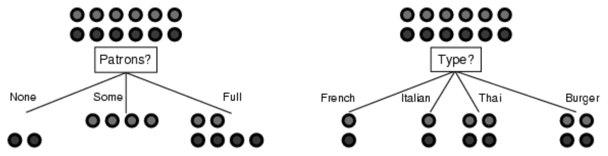
Decision tree learning

- Aim: find a small tree consistent with the training examples
- Idea: (recursively) choose "most significant" attribute as root of (sub)tree

```
function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
      best ← CHOOSE-ATTRIBUTE(attributes, examples)
      tree ← a new decision tree with root test best
      for each value $v_i$ of best do
          $examples_i$ ← {elements of examples with best = $v_i$}
          subtree ← DTL($examples_i$, attributes − best, MODE(examples))
          add a branch to tree with label $v_i$ and subtree subtree
      return tree
```

Choosing an attribute

- Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"



- *Patrons?* is a better choice

Using information theory

- To implement `Choose-Attribute` in the DTL algorithm
- Information Content (Entropy):

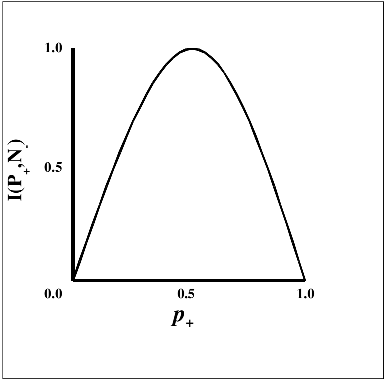
  $I(P(v_1), \ldots, P(v_n)) = \Sigma_{i=1} -P(v_i) \log_2 P(v_i)$
- For a training set containing *p* positive examples and *n* negative examples:

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

Information function with two categories

Entropy reflects the lack of "purity" of some particular set *S*

As the proportion of positives p_+ approaches 0.5 (very impure), the Entropy of *S* converges to 1.0



Information gain

- A chosen attribute *A* divides the training set *E* into subsets E_1, … , E_v according to their values for *A*, where *A* has *v* distinct values.

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

- Information Gain (IG) or reduction in entropy from the attribute test:

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

- Choose the attribute with the largest IG

Information gain

For the training set, $p = n = 6$, $I(6/12, 6/12) = 1$ bit

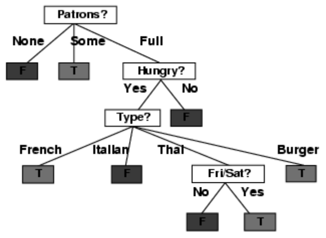
Consider the attributes *Patrons* and *Type* (and others too):

$$IG(Patrons) = 1 - \left[\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right)\right] = .541 \text{ bits}$$
$$IG(Type) = 1 - \left[\frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right)\right] = 0 \text{ bits}$$

Patrons has the highest IG of all attributes and so is chosen by the DTL algorithm as the root

Example contd.

- Decision tree learned from the 12 examples:



- Substantially simpler than “true” tree---a more complex hypothesis isn’t justified by small amount of data

Make The Decision Tree
Pick Features that maximize IG

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

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

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

Size	Shape	Color	Class
small	square	red	+
large	circle	blue	+
small	triangle	blue	-
large	circle	red	-
small	circle	blue	+
small	circle	red	-

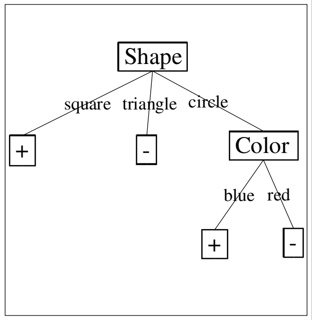
Make The Decision Tree
Pick Features that maximize IG

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

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

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

Size	Shape	Color	Class
small	square	red	+
large	circle	blue	+
small	triangle	blue	-
large	circle	red	-
small	circle	blue	+
small	circle	red	-



Handling Continuous Features

- One way of dealing with a continuous feature *F* is to treat them like Boolean features, partitioned on a dynamically chosen threshold *t*:
  - Sort the examples in *S* according to *F*
  - Identify adjacent examples with differing class labels
  - Compute *InfoGain* with *t* equal to the average of the values of at these boundaries
  - Can also be generalized to multiple thresholds
    - U. Fayyad and K. Irani, “Multi-interval descretization of continuous-valued attributes for classification learning,” *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, 1993

Handling Continuous Features

- There are two candidates for threshold *t* in this example:

Temperature	40	48	60	72	80	90
>1,000?	No	No	Yes	Yes	Yes	No

$t = (48+60)/2 = 54$ $t = (80+90)/2 = 85$

- The dynamically-created Boolean features $Temp_{>54}$ and $Temp_{>85}$ can now compete with the other Boolean and discrete features in the dataset

Noisy Data

- Noisy data could be in the examples:
  - examples have the same attribute values, but different classifications (rare case: `if(empty(atts))`)
  - classification is wrong
  - attributes values are incorrect because of errors getting or preprocessing the data
  - irrelevant attributes used in the decision-making process
- Use Pruning to improve performance when working with noisy data

Missing Data

- Missing data:
  - while learning: replace with most likely value
  - while learning: use *NotKnown* as a value
  - while classifying: follow arc for all values and weight each by the frequency of exs. crossing that arc

Tree Induction as Search

- We can think of inducing the "best tree" as an optimization search problem:
  - **States:** possible (sub-)trees
  - **Actions:** add a feature as a node of the tree
  - **Objective Function:** increase the overall information gain of the tree
- Essentially, Decision Tree Learner is a hill-climbing search through the hypothesis space, where the heuristic picks features that are likely to lead to small trees

Pruning

- Overfitting
  meaningless regularity is found in the data
  - irrelevant attributes confound the true, important, distinguishing features
  - fix by pruning lower nodes in the decision tree
  - if gain of best attribute is below a threshold, make this node a leaf rather than generating child nodes

Pruning

```
randomly partition training examples into:
  TRAIN set (~80% of training exs.)
  TUNE  set (~10% of training exs.)
  TEST  set (~10% of training exs.)
build decision tree as usual using TRAIN set
bestTree     = decision tree produced on the TRAIN set;
bestAccuracy = accuracy of bestTree on the TUNE set;
progressMade = true;
while (progressMade) { //while accuracy on TUNE improves
  find better tree;
  //starting at root, consider various pruned versions
  //of the current tree and see if any are better than
  //the best tree found so far
}
return bestTree;
use TEST to determine performance accuracy;
```

Pruning

```
//find better tree
progressMade = false;
currentTree  = bestTree;
for (each interiorNode N in currentTree) { //start at root
  prunedTree  = pruned copy of currentTree;
  newAccuracy = accuracy of prunedTree on TUNE set;
  if (newAccuracy >= bestAccuracy) {
    bestAccuracy = newAccuracy;
    bestTree     = prunedTree;
    progressMade = true;
  }
}
```

Pruning

```
//pruned copy of currentTree
replace interiorNode N in currentTree by a leaf node
label leaf node with the majorityClass among TRAIN set
  examples that reached node N
break ties in favor of '-'
```

Case Studies

- Decision trees have been shown to be at least as accurate as human experts.
- Diagnosing breast cancer
  - humans correct 65% of the time
  - decision tree classified 72% correct
- BP designed a decision tree for gas-oil separation for offshore oil platforms
- Cessna designed a flight controller using 90,000 exs. and 20 attributes per ex.

Conclusions

- Information (Entropy) of a set of data
- Information Gain using some feature on a set of data
- Handling
  - continuous features
  - noisy data
  - missing values
- Pruning
- Constructing decision trees as Search