CMU SCS

15-826: Multimedia Databases and Data Mining

Data Mining - AI reminders

C. Faloutsos

CMU SCS

Outline

Goal: 'Find similar / interesting things'

- Intro to DB
- Indexing - similarity search
- Data Mining

CMU SCS

Data Mining - Detailed outline

- Statistics
- AI - decision trees
  - Problem
  - Approach
  - Conclusions
- DB

CMU SCS

Decision Trees

- Problem: Classification - Ie.,
- given a training set (N tuples, with M attributes, plus a label attribute)
- find rules, to predict the label for newcomers

Pictorially:

CMU SCS

Decision trees

Age	Chol-level	Gender	…	CLASS-ID
30	150	M		+
				…
				-
				??

CMU SCS

Decision trees

- issues:
  - missing values
  - noise
  - 'rare' events

CMU SCS

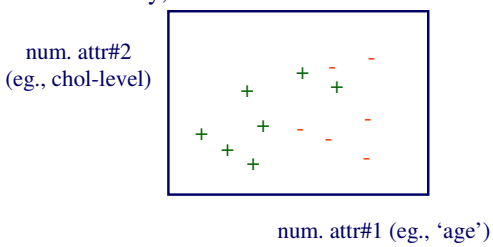
Decision trees

- types of attributes
  - numerical (= continuous) - eg: 'salary'
  - ordinal (= integer) - eg.: '# of children'
  - nominal (= categorical) - eg.: 'car-type'

CMU SCS

Decision trees

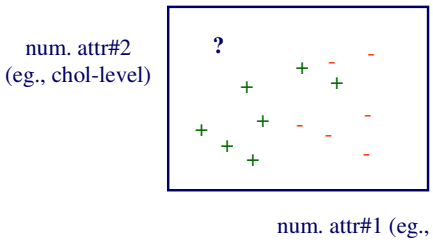
- Pictorially, we have



CMU SCS

Decision trees

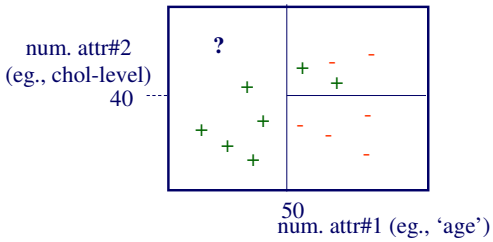
- and we want to label '?'



CMU SCS

Decision trees

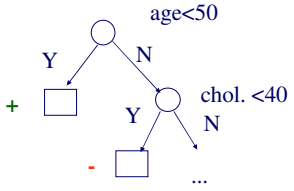
- so we build a decision tree:



CMU SCS

Decision trees

- so we build a decision tree:



CMU SCS

Data Mining - Detailed outline

- Statistics
- AI - decision trees
  - Problem
  - Approach
  - Conclusions
- DB

CMU SCS

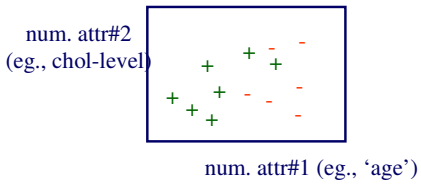
Decision trees

- Typically, two steps:
  - tree building
  - tree pruning (for over-training/over-fitting)

CMU SCS

Tree building

- How?



CMU SCS

Tree building

- How?
- A: Partition, recursively - pseudocode:

```
Partition ( Dataset S)
  if all points in S have same label
  then return
  evaluate splits along each attribute A
  pick best split, to divide S into S1 and S2
  Partition(S1); Partition(S2)
```


CMU SCS

Tree building

- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?

CMU SCS

Tree building

- Q1: how to introduce splits along attribute A_i
- A1:
  - for num. attributes:
    - binary split, or
    - multiple split
  - for categorical attributes:
    - compute all subsets (expensive!), or
    - use a greedy algo

CMU SCS

Tree building

- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?

CMU SCS

Tree building

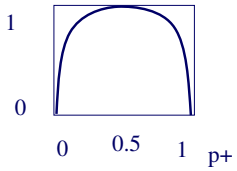
- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?
- A: by how close to uniform each subset is - ie., we need a measure of uniformity:

CMU SCS

Tree building

entropy: H(p+, p-)

Any other measure?

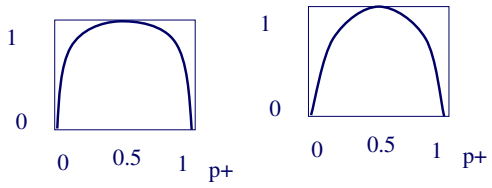


CMU SCS

Tree building

entropy: $H(p_+, p_-)$

'gini' index: $1-p_+^2 - p_-^2$



CMU SCS

Tree building

entropy: $H(p_+, p_-)$

'gini' index: $1-p_+^2 - p_-^2$

(How about multiple labels?)

CMU SCS

Tree building

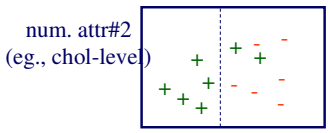
Intuition:

- entropy: #bits to encode the class label
- gini: classification error, if we randomly guess '+' with prob. p_+

CMU SCS

Tree building

Thus, we choose the split that reduces entropy/classification-error the most: Eg.:



CMU SCS

Tree building

- Before split: we need
  $(n_+ + n_-) * H(p_+, p_-) = (7+6) * H(7/13, 6/13)$
  bits total, to encode all the class labels
- After the split we need:
  0 bits for the first half and
  (2+6) * H(2/8, 6/8) bits for the second half

CMU SCS

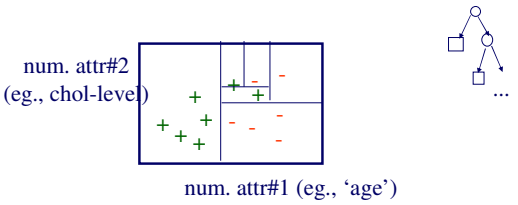
Data Mining - Detailed outline

- Statistics
- AI - decision trees
  - Problem
  - Approach
    - tree building
    - tree pruning
  - Conclusions
- DB

CMU SCS

Tree pruning

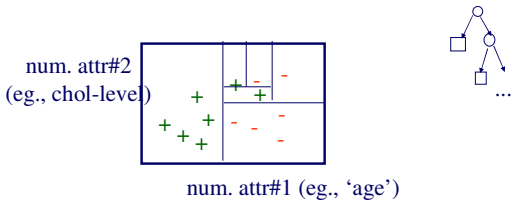
- What for?



CMU SCS

Tree pruning

- Q: How to do it?



CMU SCS

Tree pruning

- Q: How to do it?
- A1: use a 'training' and a 'testing' set - prune nodes that improve classification in the 'testing' set. (Drawbacks?)

CMU SCS

Tree pruning

- Q: How to do it?
- A1: use a 'training' and a 'testing' set - prune nodes that improve classification in the 'testing' set. (Drawbacks?)
- A2: or, rely on MDL (= Minimum Description Language) - in detail:

CMU SCS

Tree pruning

- envision the problem as compression (of what?)

CMU SCS

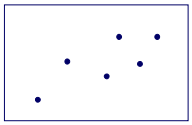
Tree pruning

- envision the problem as compression (of what?)
- and try to min. the # bits to compress
  (a) the class labels AND
  (b) the representation of the decision tree

CMU SCS

(MDL)

- a brilliant idea - eg.: best *n*-degree polynomial to compress these points:
- the one that minimizes (sum of errors + *n*)



CMU SCS

Conclusions

- Classification through trees
- Building phase - splitting policies
- Pruning phase (to avoid over-fitting)
- Observation: classification is subtly related to compression

CMU SCS

Reference

- M. Mehta, R. Agrawal and J. Rissanen, `*SLIQ: A Fast Scalable Classifier for Data Mining'*, Proc. of the Fifth Int'l Conference on Extending Database Technology, Avignon, France, March 1996