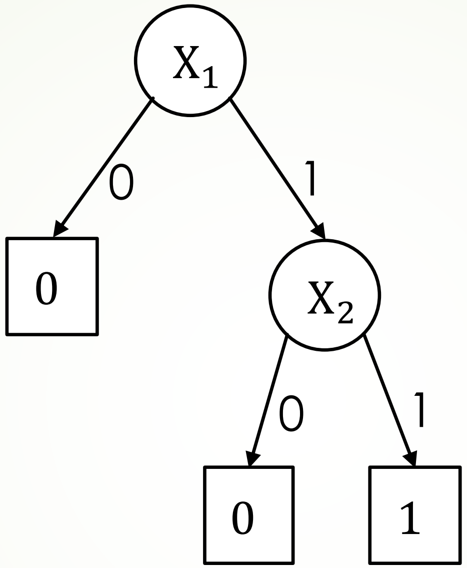
香港城市大學
City University of Hong Kong
專業 創新 胸懷全球
Professional · Creative
For The World

Classification:
Rule-Based Classification

C5483 Data Warehousing and Data Mining

Motivation



- When is the decision equal to 1?
  1. If ______________________, then $Y = 1$.
  2. Else $Y = 0$.

Rule-based classification

Knowledge representation

Antecedent

Conjunct $\mathrm{X_i}[=/\leq/>]v_i$

Consequence (class prediction)

Rule set

1. $\mathrm{X}_1 = 1, \mathrm{X}_2 = 1 \Rightarrow \mathrm{Y} = 1.$
2. $\mathrm{Y} = 0.$

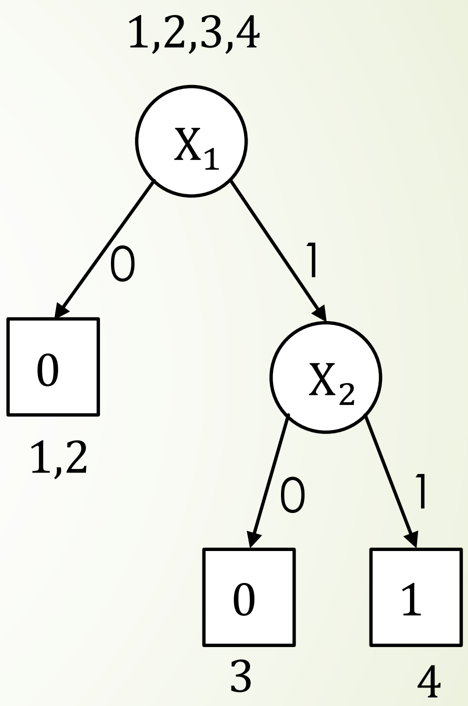
Default rule (no antecedent)

Decision list
(rule ordering)

- Benefits representing knowledge by rules: (c.f. decision tree or NN)
  - **M**___________________________
  - **I**____________________________
- How to generate rules?

Generate rules from a decision tree

	X_1	X_2	Y
1.	0	0	0
2.	0	1	0
3.	1	0	0
4.	1	1	1



- Each path from root to leaf corresponds to a rule:
  1. $X_1 = \_\_\_ \Rightarrow Y = 0$
  2. $X_1 = \_\_\_, X_2 = \_\_\_ \Rightarrow Y = 0$
  3. $X_1 = \_\_\_, X_2 = \_\_\_ \Rightarrow Y = 1$
- Does the ordering of these rules matter?
  Yes/No because____

5

Sequential covering

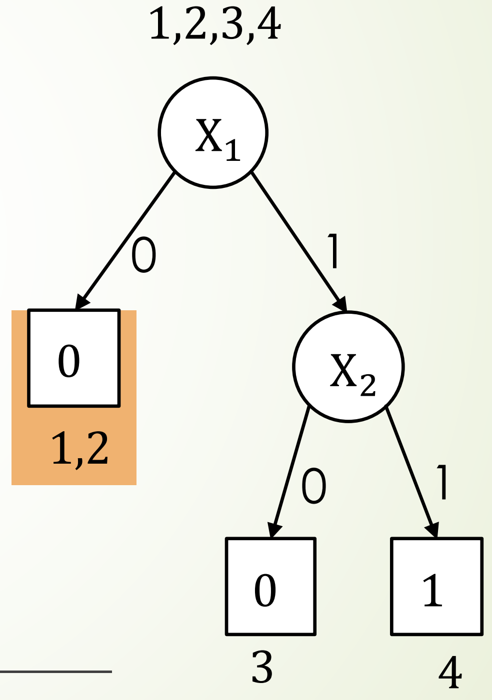
- **S_____________-and-c_________________** (c.f. divide-and-conquer)
  1. Learn a good rule.
  2. Remove covered instances and repeat 1 until all instances covered.
- How to learn a good rule?
- PART (partial tree) decision list
  1. Build a new decision tree (by C4.5) and extract the rule that maximizes **coverage:** fraction of instances satisfying the antecedent.
  2. Remove covered instances and repeat 1 until all instances are covered.

PART (partial tree) decision list

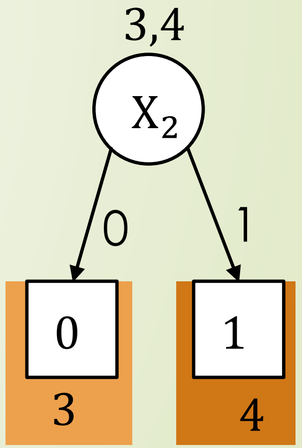
Example

1. Rule 1: ______________________
   - *i.* $X_1 = 0 \Rightarrow Y = 0$ (coverage: _____%)
   - *ii.* $X_1 = 1, X_2 = 0 \Rightarrow Y = 0$ (coverage: _____%)
   - *iii.* $X_1 = 1, X_2 = 1 \Rightarrow Y = 1$ (coverage: _____%)
2. Rule 2: ______________________
   - *i.* $X_2 = 0 \Rightarrow Y = 0$ (coverage: _____%)
   - *ii.* $X_2 = 1 \Rightarrow Y = 1$ (coverage: _____%)
3. Default rule: $Y =$ ______

- Issue: [Time complexity] ________________________________



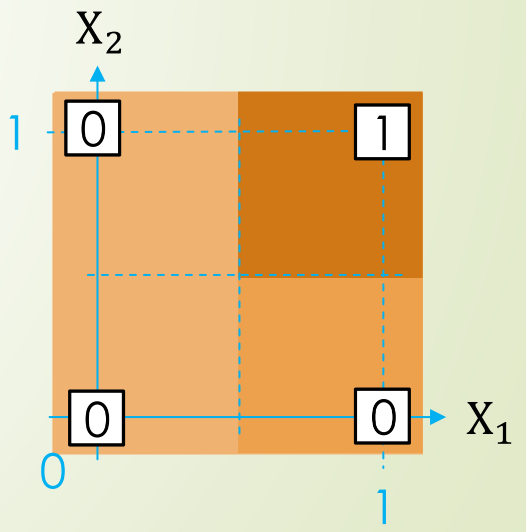
	X_1	X_2	Y
1.	0	0	0
2.	0	1	0
3.	1	0	0
4.	1	1	1



Generating rule directly

1. Start with ZeroR, add conjuncts to improve **confidence**: fraction of correctly classified instances.
   - Rule 1: $\mathrm{Y} = 0$
     - Confidence: ___________%
   - Rule 1 (refined): $\mathrm{X}_1 = 0 \Rightarrow \mathrm{Y} = 0$
     - Confidence: ____________%
2. Repeatedly add new rules to cover remaining tuples
   - Rule 2: $\mathrm{Y} = 0$
     - Confidence: ___________%
   - Rule 2 (refined): $\mathrm{X}_2 = 0 \Rightarrow \mathrm{Y} = 0$
     - Confidence: ___________%
   - Default rule: $\mathrm{Y} =$ ___________.

	X_1	X_2	Y
1.	0	0	0
2.	0	1	0
3.	1	0	0
4.	1	1	1



Generating rule directly

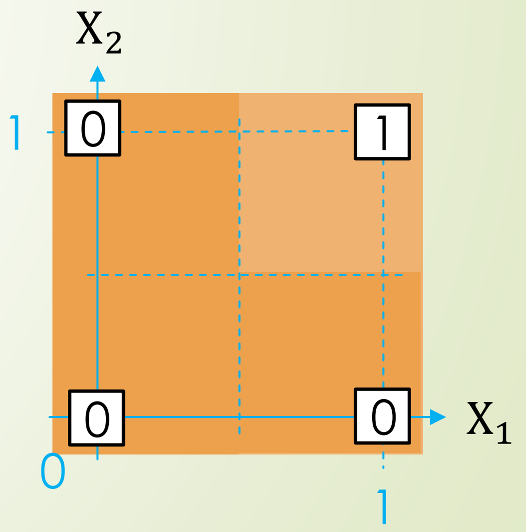
- Decision list
  1. Rule 1: $X_1 = 0 \Rightarrow Y = 0$
  2. Rule 2: $X_2 = 0 \Rightarrow Y = 0$
  3. Default rule: $Y = 1$.
- Is the list best possible? Y/N
  1. Time to detect positive class: ______________________
  2. Length of the list: _________________________________

	X_1	X_2	Y
1.	0	0	0
2.	0	1	0
3.	1	0	0
4.	1	1	1

Class-based ordering

- Learn rules for positive class first:
  1. Rule 1:
     i. $Y = 1$ (confidence: _____%)
     ii. $X_1 =$ ___ $\Rightarrow Y = 1$ (confidence: _____%)
     iii. $X_1 =$ ___, $X_2 =$ ____ $\Rightarrow Y = 1$ (confidence: _____%)
  2. Default rule: $Y =$ ______
- Will the above guarantee a short decision list in general? Y/N because ____

	X_1	X_2	Y
1.	0	0	0
2.	0	1	0
3.	1	0	0
4.	1	1	1



RIPPER

First Order Inductive Learner Gain

- Add conjunct that maximizes

$$\text{FOIL\_Gain} = p'\left(\log\frac{p'}{p'+n'} - \log\frac{p}{p+n}\right)$$

  - Change in # of positives: $p \rightarrow p'$
  - Change in # of negatives: $n \rightarrow n'$

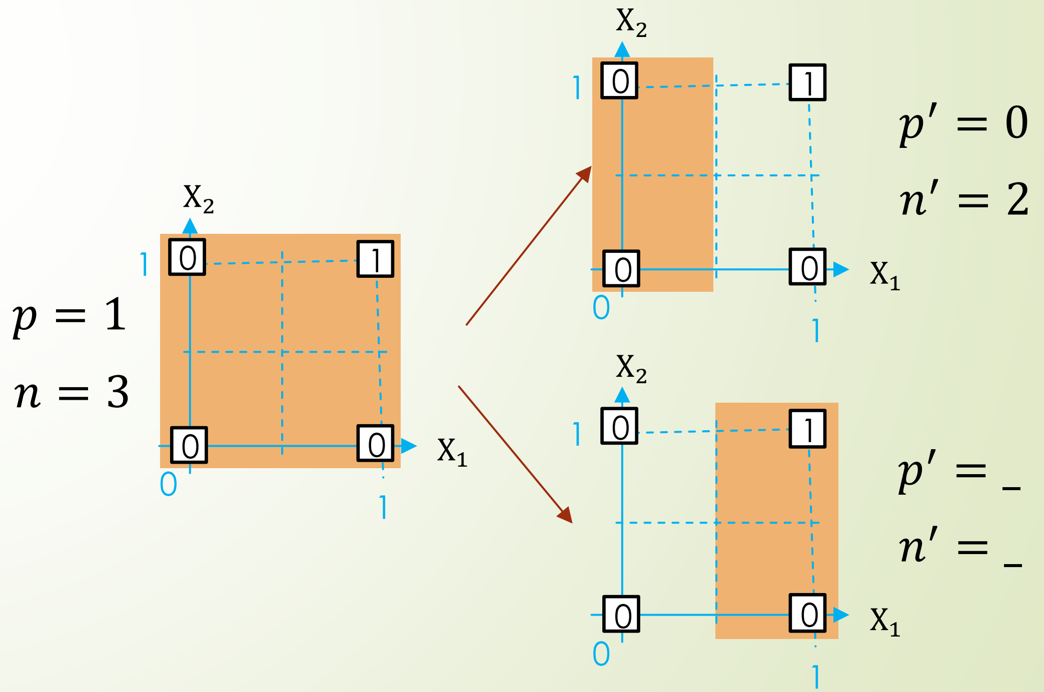
- $\mathrm{Y} = 1 \rightarrow \mathrm{X}_1 = 0 \Rightarrow \mathrm{Y} = 1$:

  FOILGain=____

- $\mathrm{Y} = 1 \rightarrow \mathrm{X}_1 = 1 \Rightarrow \mathrm{Y} = 1$:

  FOILGain=____

- First/Second is better.



RIPPER

First Order Inductive Learner Gain

- Improve a rule by maximizing

$$\text{FOIL\_Gain} = p'\left(\log\frac{p'}{p'+n'} - \log\frac{p}{p+n}\right)$$

  - Change in # of positives: $p \rightarrow p'$
  - Change in # of negatives: $n \rightarrow n'$
- $X_1 = 1 \Rightarrow Y = 1 \rightarrow X_1 = 1, X_2 = 0 \Rightarrow Y = 1$:

  FOILGain=_________________________________

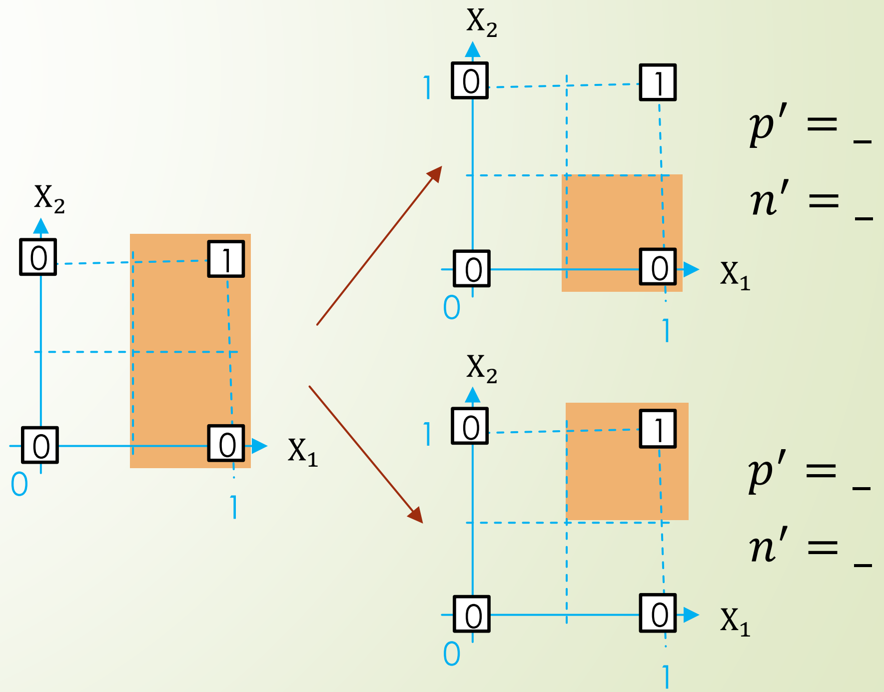
- $X_1 = 1 \Rightarrow Y = 1 \rightarrow X_1 = 1, X_2 = 1 \Rightarrow Y = 1$:

  FOILGain=_________________________________

- First/Second is better.

$p = 1$

$n = 1$



RIPPER

First Order Inductive Learner Gain

$$\begin{aligned}\text{FOIL\_Gain} &= p'\left(\log\frac{p'}{p'+n'} - \log\frac{p}{p+n}\right)\\ &= \underbrace{(p'+n')}_{(1)}\underbrace{\frac{p'}{p'+n'}}_{(2)}\underbrace{\left(\log\frac{p'}{p'+n'} - \log\frac{p}{p+n}\right)}_{(3)}\end{aligned}$$

- Heuristics:
  - (1) favors rules with large <u>coverage/confidence</u>.
  - (2)*(3) favors rules with large <u>coverage/confidence</u> given the same <u>coverage/confidence.</u>
  - (3) ensures FOIL_Gain is positive if <u>coverage/confidence</u> increases.
- [Challenge] Why not use information gain or gain ratio?

RIPPER
How to avoid overfitting?

- **R**epeated **I**ncremental **P**runing to **P**roduce **E**rror **R**eduction
- After each new rule, eliminate a conjunct (starting with the most recently added one) if it improves the following on a v__________________ set:

$$\text{FOIL\_Prune} = \frac{p-n}{p+n}$$

or equivalently reduces

$$\text{error} = \frac{n}{p+n}$$

References

- 8.4 Rule-Based Classification
- (Optional) Eibe Frank, Ian H. Witten. "Generating accurate rule sets without global optimization." Fifteenth International Conference on Machine Learning, 1998, p.144-151.
  - A partial tree is built with nodes (subsets of data) split (expanded) in the order of their entropy.
  - A node is considered for pruning by subtree replacement if all its children are leaf nodes.
- (Optional) Cohen, William W. "Fast effective rule induction." *Machine Learning Proceedings, 1995,* p.115-123. (See also WEKA JRIP or its source code.)
  - The algorithm stops adding rules to the rule-set if the description length of the new rule is 64 bits more than the minimum description length met.
  - After the algorithm stop adding rules, there is a rule optimization step that optimize each rule one-by-one.