Data Mining

Practical Machine Learning Tools and Techniques

Slides for Chapter 4 of *Data Mining* by I. H. Witten and E. Frank
Learning Rules

Simplicity first

- Simple algorithms often work very well!
- There are many kinds of simple structure, eg:
  - One attribute does all the work
  - All attributes contribute equally & independently
  - A weighted linear combination might do
  - Instance-based: use a few prototypes
  - Use simple logical rules
- Success of method depends on the domain

Inferring rudimentary rules

- 1R: learns a 1-level decision tree
  - I.e., rules that all test one particular attribute
- Basic version
  - One branch for each value
  - Each branch assigns most frequent class
  - Error rate: proportion of instances that don't belong to the majority class of their corresponding branch
  - Choose attribute with lowest error rate

(*assumes nominal attributes*)

Pseudo-code for 1R

```
For each attribute,
   For each value of the attribute, make a rule as follows:
      count how often each class appears
      find the most frequent class
      make the rule assign that class to this attribute-value
   Calculate the error rate of the rules
Choose the rules with the smallest error rate
```

- Note: "missing" is treated as a separate attribute value

Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Attribute	Rules	Errors	Total errors
Outlook	Sunny → No	2/5	4/14
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	
Temp	Hot → No*	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	High → No	3/7	4/14
	Normal → Yes	1/7	
Windy	False → Yes	2/8	5/14
	True → No*	3/6	

* indicates a tie

Dealing with numeric attributes

- Discretize numeric attributes
- Divide each attribute's range into intervals
  - Sort instances according to attribute's values
  - Place breakpoints where class changes (majority class)
  - This minimizes the total error
- Example: *temperature* from weather data

64 65 68 69 70 71 72 72 75 75 80 81 83 85
Yes | No | Yes Yes Yes | No No Yes | Yes Yes | No | Yes Yes | No

Outlook	Temperature	Humidity	Windy	Play
Sunny	85	85	False	No
Sunny	80	90	True	No
Overcast	83	86	False	Yes
Rainy	75	80	False	Yes
...	...	...	...	...

The problem of overfitting

- This procedure is very sensitive to noise
  - One instance with an incorrect class label will probably produce a separate interval
- Also: *time stamp* attribute will have zero errors
- Simple solution:
  enforce minimum number of instances in majority class per interval
- Example (with min = 3):

64 65 68 69 70 71 72 72 75 75 80 81 83 85
Yes No Yes Yes Yes | No No Yes | Yes Yes | No Yes Yes No

64 65 68 69 70 71 72 72 75 75 80 81 83 85
Yes No Yes Yes Yes | No No Yes Yes Yes | No Yes Yes No

With overfitting avoidance

- Resulting rule set:

Attribute	Rules	Errors	Total errors
Outlook	Sunny → No	2/5	4/14
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	
Temperature	≤ 77.5 → Yes	3/10	5/14
	> 77.5 → No*	2/4	
Humidity	≤ 82.5 → Yes	1/7	3/14
	> 82.5 and ≤ 95.5 → No	2/6	
	> 95.5 → Yes	0/1	
Windy	False → Yes	2/8	5/14
	True → No*	3/6	

Discussion of 1R

- 1R was described in a paper by Holte (1993)
  - Contains an experimental evaluation on 16 datasets (using *cross-validation* so that results were representative of performance on future data)
  - Minimum number of instances was set to 6 after some experimentation
  - 1R's simple rules performed not much worse than much more complex decision trees
- Simplicity first pays off!

Very Simple Classification Rules Perform Well on Most Commonly Used Datasets
Robert C. Holte, Computer Science Department, University of Ottawa

Discussion of 1R: Hyperpipes

- Another simple technique: build one rule for each class
  - Each rule is a conjunction of tests, one for each attribute
  - For numeric attributes: test checks whether instance's value is inside an interval
    - Interval given by minimum and maximum observed in training data
  - For nominal attributes: test checks whether value is one of a subset of attribute values
    - Subset given by all possible values observed in training data
  - Class with most matching tests is predicted