Master of Technology in Knowledge Engineering

KE4102: Intelligent Systems & Techniques for Business Analytics: Discovering Knowledge from Data using Direct Rule Induction

Charles Pang
Institute of Systems Science
National University of Singapore
E-mail: charlespang@nus.edu.sg

Problems with DT

- Over-specialization ("overfitting")
  - One branch for each value of the attribute
  - Maybe an irrelevant value
- Trees can be very large and complex
  - Confusing and hard to understand
  - Attributes may be fragmented across the tree structure
- Difficult to understand a part of the tree without reference to the whole
- Learning an Optimal DT is an NP-Complete problem
  - Local optimum achieved using heuristic greedy algorithm

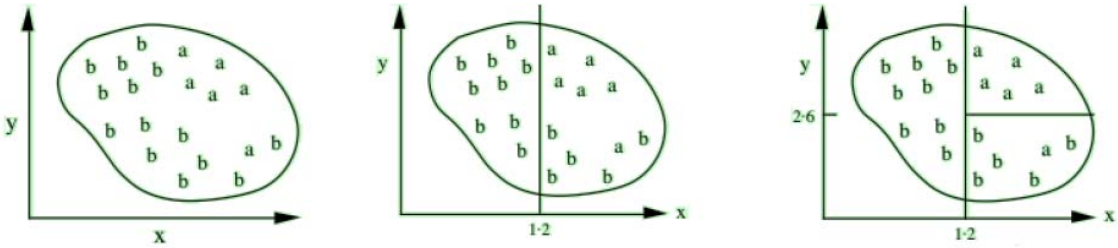
Extracting Rules from a Decision Tree

- A Decision tree can be converted into a set of decision-making rules
- Each path from the root to a leave corresponds to a rule
  - A conjunction of attributes along the path are the antecedents
  - Each leaf is the class
- Need to examine the whole tree
- Rule set can be overly complex in a straightforward conversion
- Can contain many unnecessary rules
- More effective conversions are not trivial

Direct Induction of Rules

- Is it possible to generate a set of rules without creating a Decision Tree?
- There are algorithms that can generate rules sequentially by finding the best rule that **covers** all instances in a class
- Called a ***Covering Approach***:
  - At each stage, a rule is identified that "covers" some of the instances
  - Once we have the best rule, we can eliminate the examples covered
  - This procedure can be iterated until we have rules that cover all the dataset
  - The result is a disjunction of rules that can be ordered by accuracy

Covering example

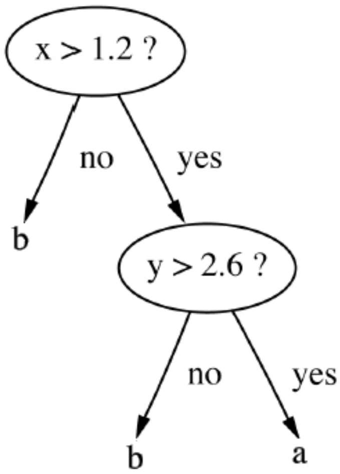


IF true THEN class=a | IF (x > 1.2) THEN class=a | IF (x > 1.2) and (y > 2.6) THEN class=a

- Possible rule set for class=a:
  - *If (x > 1.2) then class=a*
  - *If (x > 1.2) and (y > 2.6) then class=a*
- You could add more rules to get a “perfect” rule set

Rules vs Decision Trees

- The tree representation of our rules
- We face the same problem as DT - which attribute to select and split?
- Rules do not suffer from replicated subtrees – hence they tend to be more perspicuous
- Covering algorithms concentrate on one class at a time whereas trees take all classes into account

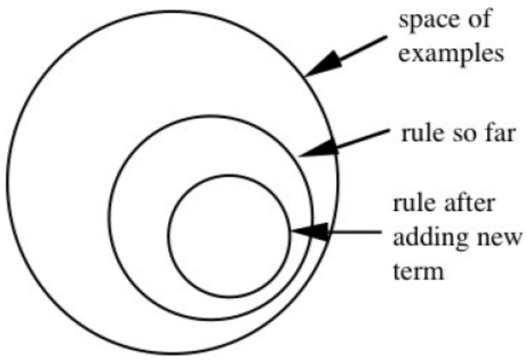


Basic Covering Algorithm

- Generate a rule by adding conditions (tests) to improve its accuracy:

  If (? and ? and …) then class=x

  – Add the condition that **maximizes** the rule's accuracy
- Each new test reduces the rules coverage



Accuracy Test Measures

p/t ratio:

t =total instances covered by rule

p =correct class instances covered by the rule

t - p =number of errors made by rule

Goal: maximize p/t ratio

Information Gain:

p (log(p/t) – log(P/T))

P and T the positive and total numbers before the new condition was added

Information gain emphasizes positive rather than negative instances

Example: Contact Lens Data

age	spectacle-prescrip	astigmatism	tear-prod-rate	contact-lenses
young	myope	no	reduced	none
young	myope	no	normal	soft
young	myope	yes	reduced	none
young	myope	yes	normal	hard
young	hypermetrope	no	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	no	reduced	none
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	hypermetrope	no	normal	soft
pre-presbyopic	hypermetrope	yes	reduced	none
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	no	reduced	none
presbyopic	myope	no	normal	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	no	reduced	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none

Possible Tests

- Rule:

  If ?
  then contact-lenses = hard

- Possible tests:

```
Age = Young                      2/8
Age = Pre-Presbyopic             1/8
Age = Presbyopic                 1/8
Spectacle prescrip = Myope       3/12
Spectacle prescrip = Hypermetrope 1/12
Astigmatism = no                 0/12
Astigmatism = yes                4/12
Tear-prod-rate = Reduced         0/12
Tear-prod-rate = Normal          4/12
```

1^{st} Test added to rule

- Rule:

  If astigmatism = yes
  then contact-lenses = hard

- Examples covered by rule:

age	spectacle-prescrip	astigmatism	tear-prod-rate	contact-lenses
young	myope	yes	reduced	none
young	myope	yes	normal	hard
young	hypermetrope	yes	reduced	none
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	yes	reduced	none
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none

Possible Tests

- Rule:

  If astigmatism = yes and ?
  then contact-lenses = hard

- Possible Tests:

```
Age = Young                        2/4
Age = Pre-presbyopic               1/4
Age = Presbyopic                   1/4
Spectacle prescrip = Myope         3/6
Spectacle prescrip = Hypermetrope 1/6
Tear-prod-rate = Reduced           0/6
Tear-prod-rate = Normal            4/6
```

2nd Test added to rule

- Rule:

  If astigmatism = yes and tear-prod-rate = Normal
  then contact-lenses = hard

- Examples covered by rule:

age	spectacle-prescrip	astigmatism	tear-prod-rate	contact-lenses
young	myope	yes	normal	hard
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	normal	none

Possible Tests

- Rule:

  If astigmatism = yes and tear-prod-rate = Normal
  and ?
  then contact-lenses = hard

- Possible Tests:

```
Age = Young                      2/2
Age = Pre-Presbyopic             1/2
Age = Presbyopic                 1/2
Spectacle prescrip = Myope       3/3
Spectacle prescrip = Hypermetrope 1/3
```

3^{rd} Test added to rule

- Rule:

  If astigmatism = yes and tear-prod-rate = Normal
  and spectacle prescrip = Myope
  then contact-lenses = hard

- Examples covered by rule:

age	spectacle-prescrip	astigmatism	tear-prod-rate	contact-lenses
young	myope	yes	normal	hard
pre-presbyopic	myope	yes	normal	hard
presbyopic	myope	yes	normal	hard

Second rule to cover the "stray"

- Example not covered by the first rule:

age	spectacle-prescrip	astigmatism	tear-prod-rate	contact-lenses
young	hypermetrope	yes	normal	hard

- Create a second rule:

  If age = young and astigmatism = yes
  and tear-prod-rate = normal
  then contact-lenses = hard

Result of rule induction

- Two rules to cover contact-lenses = hard:

If astigmatism = yes and tear-prod-rate = Normal
and spectacle prescrip = Myope
then contact-lenses = hard

If age = young and astigmatism = yes
and tear-prod-rate = normal
then contact-lenses = hard

- Repeat the process with the other two classes

Sequential Covering Algorithm (PRISM)

```
A= Attribute
v= Attribute Value

For each class C
    E = instance set
    While E contains instances of C
        Create a rule R: [If ? then class = C]
        Repeat
            For each A and v not in R, do
                Select (A = v) with maximum p/t
                (break ties by choosing the the largest p)
            Add (A = v) to R
        Until R is perfect (or there are no more A to use)
        Remove the instances covered by R from E
```

Missing Values & Numeric Attributes

- Missing values will fail any test
  - Algorithms must either
    - Use other tests to separate out the positive instances
    - Leave them uncovered until later in the process
    - Missing values can be treated as a category
- Numeric attributes are treated the same as in decision trees (refer to notes on decision trees)

Advantage of Rules

- If the primary objective is for the results be used and interpreted by people
  - Intuitive and easy to understand
  - Represents chunks of knowledge
  - Can be used as knowledge acquisition
  - Supplement/Verify/Validate knowledge acquired from other sources (Interviews, Documents, anecdotes, etc.)
- Rules can be used to build intelligent systems
  - Easily deployed and integrated to existing IS
  - Used for decision-support
  - Can be updated to include new conditions
  - Does not require re-training