Rule-Based Learning

What Is Rule-Based Learning?

- Learns **classification rules directly** rather than building a tree
- Each rule predicts a **single class**
- Algorithm builds rules **one at a time**
- Uses **separate-and-conquer** strategy
- Produces **interpretable, modular rules**

Rules vs. Decision Trees

- Decision Trees
  - Produce **one large tree**
  - Splits consider **all classes**
  - Recursively divide dataset
- Rule Learners
  - Produce **multiple independent rules**
  - Learn **one class at a time**
  - Focus on **covering** instances of the target class

Separate-and-Conquer Process

1. Pick a target class
2. Create a rule that covers **many examples** of that class
3. Remove covered examples
4. Repeat until no examples remain
5. Move on to next class

The PRISM Algorithm

- Characteristics:
  - Creates **rules with 100% training accuracy**
  - Adds conditions one at a time
  - Picks condition maximizing accuracy **p/t**
  - p = correct examples covered
  - t = total examples covered

PRISM Pseudocode

```
For each class C:
    Let E = all instances
    While E contains instances of class C:
        Start rule R predicting C
        Repeat:
            For each attribute-value pair (A=v):
                Compute accuracy p/t if (A=v) added to R
            Select condition maximizing p/t
            Add condition to R
        Until R is perfect or no conditions left
        Output R
        Remove examples covered by R from E
```

Key Idea Behind PRISM

- At each step, choose condition giving **highest accuracy**
- Rules expand one condition at a time
- Rule ends when it becomes **perfect**
- Results in small, precise rule sets

Example Dataset: Contact Lens Problem

- Attributes:
  - Age
  - Spectacle prescription
  - Astigmatism
  - Tear production
- Class: lens type (hard, soft, none)

Example Rule Construction (Hard Lenses)

- Evaluate initial tests:

```
age = young            -> 2/8
astigmatism = yes      -> 4/12  (best)
tear production normal -> 4/12
spectacle = myope      -> 3/12
```

- Choose:

```
astigmatism = yes
```

Adding Next Condition

- Now restrict to examples with astigmatism=yes.

```
tear production normal -> 4/6 (best)
age = young             -> 2/4
spectacle = myope       -> 3/6
```

- Choose:

```
AND tear production = normal
```

Final Rule (Hard Lenses)

```
IF astigmatism = yes
AND tear production = normal
AND spectacle = myope
THEN lens = hard
```

- This rule is **perfect** (no errors).

After Building a Rule

- Remove covered hard-lens examples
- Build next rule for same class
- When no examples remain -> switch classes

Strengths of Rule-Based Learning

- Highly interpretable
- Rules are modular
- Easy to add/remove knowledge
- Captures **local** patterns well
- Often effective in expert-system domains

Limitations

- Greedy -> may miss global optimum
- Sensitive to noise $\rightarrow$ many rules
- Perfect rules overfit
- Conflicts:
  - Multiple rules may apply
  - No rules may apply

Rules vs. Decision Lists

- PRISM rules do **not require ordering**
- Order affects classification only in **decision lists**
- Decision lists:

```
IF cond1 THEN class1
ELSE IF cond2 THEN class2
ELSE default
```

- Execution stops at first match.

Covering Algorithms

- Train by removing covered examples
- Efficient for large datasets
- Natural rule generators
- Basis for modern algorithms:
  - RIPPER
  - CN2
  - FOIL

Summary

- Rule learners generate **direct classification rules**
- PRISM maximizes **accuracy p/t** while building rules
- Produces precise, interpretable rule sets
- Ideal for domains needing transparency