Rudimentary Rules

What Are Rudimentary Rules?

- The **simplest possible** classification approach.
- Uses **one attribute only** to make predictions.
- Produces a **one-level decision tree** (a rule for each value of the attribute).
- Surprisingly effective despite its simplicity.

Why Use Simple Rules?

- Many real-world datasets contain **simple underlying structure**.
- Complex models can **overfit** and hide patterns.
- 1R often performs **near state-of-the-art** with tiny computation.
- Ideal **baseline model** before trying complex algorithms.

The 1R Idea

1. For each attribute:
   - Create **rules** for every value of that attribute.
2. Assign the **majority class** for that value.
3. Compute **error rate** for all rules of that attribute.
4. Choose the attribute with the **lowest total error**.

1R Algorithm (Pseudocode)

```
For each attribute A:
    For each value v of A:
        Count class frequencies for instances with A = v
        Assign the majority class to rule (A = v → majority
    Compute total error for attribute A
Choose attribute whose rule set has the minimum total error
Return rule set for that attribute
```

Example: Weather Dataset

- Attributes:
  - Outlook (sunny, overcast, rainy)
  - Temperature (hot, mild, cool)
  - Humidity (high, normal)
  - Windy (true, false)
- Class:
  - Play (yes/no)

Example 1: Attribute = Outlook

Value	Majority Class	Errors
sunny	no	2/5
overcast	yes	0/4
rainy	yes	2/5

Total errors = 4

Resulting rule set:

- sunny -> no
- overcast -> yes
- rainy -> yes

Example 2: Attribute = Humidity

Value	Majority Class	Errors
high	no	3/7
normal	yes	1/7

Total errors = 4

Selecting the Best Attribute

Attribute	Total Errors
Outlook	4
Temperature	5
Humidity	4
Windy	5

- 1R selects any attribute tied for the lowest error (e.g., **Outlook**).

Example: Final 1R Rule

```
If Outlook = sunny      -> Play = no
If Outlook = overcast   -> Play = yes
If Outlook = rainy      -> Play = yes
```

Handling Missing Values

- Nominal Attributes
  - Treat `missing` as an **additional value**:

`Outlook = missing → majority class`

Handling Numeric Attributes

- 1R discretizes numeric values:
  1. Sort values
  2. Insert breakpoints where class labels change
  3. Merge intervals so each has a **minimum support**
- Example rule:

```
Temperature <= 77.5 -> yes
Temperature > 77.5 -> no
```

Overfitting in 1R

- Attributes with **many distinct values** can create zero-error rules.
- Example: ID numbers.
- Mitigation:
  - Require minimum number of instances per value
  - Merge intervals
  - Ignore very high-cardinality attributes

Why 1R Works Well

- Findings (Holte, 1993):
  - Very close to accuracy of full decision trees
  - Often off by only **a few percentage points**
  - Incredibly **fast**
  - Extremely **interpretable**

Benefits of 1R

- Simple and intuitive
- Very fast
- Good accuracy
- Interpretable
- Perfect baseline model

Limitations

- Only one attribute used
- Cannot capture complex interactions
- Sensitive to attributes with many values
- Might oversimplify real patterns

Summary

- 1R creates rules based on a single attribute
- Produces a one-level decision tree
- Uses majority classes for predictions
- Strong baseline before more complex algorithms