Supervised Learning
Rule-based Classification
The Principle

- The model learned in Rule-Based classification is represented as set of **IF-THEN** rules

  **IF** condition **THEN** conclusion

- Example

  **R1**: **IF** age=youth AND student=yes **THEN** buys_computer=yes

- Terminology
  - The “IF” part is known as the **rule antecedent** or **precondition**: consists in one or more attributes
  - The “THEN” part is known as **rule consequent**: contains a class prediction
  - If the condition in a rule antecedent holds true we say the condition is **satisfied** or the **rule covers** the tuple
Example

<table>
<thead>
<tr>
<th>RID</th>
<th>age</th>
<th>student</th>
<th>credit-rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>youth</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>5</td>
<td>middle-aged</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>senior</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

**R1:** IF age=youth AND student=yes THEN buys_computer=yes

The condition is satisfied = The rule covers the tuple
How to Assess The Rules

- A rule $R$ can be assessed by
  - Coverage
  - Accuracy

- Methodology
  - Class labeled dataset $D$ (a set of tuples)
  - Consider
    - $n_{covers}$: the number of tuples covered by $R$
    - $n_{correct}$: the number of tuples correctly classified by $R$
    - $|D|$: the total number of tuples in $D$

$$coverage(R) = \frac{n_{covers}}{|D|}$$

$$accuracy(R) = \frac{n_{correct}}{n_{covers}}$$
**Example**

<table>
<thead>
<tr>
<th>RID</th>
<th>age</th>
<th>student</th>
<th>credit-rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>youth</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>5</td>
<td>middle-aged</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>senior</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

**R1:** IF age=youth AND student=yes THEN buys_computer=yes

\[ n_{covers} = ? \]
\[ n_{correct} = ? \]
**Example**

<table>
<thead>
<tr>
<th>RID</th>
<th>age</th>
<th>student</th>
<th>credit-rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>youth</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>5</td>
<td>middle-aged</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>senior</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

**R1:** IF age=youth AND student=yes THEN buys_computer=yes

\[ n_{covers} = 3 \]

\[ n_{correct} = ? \]
**Example**

<table>
<thead>
<tr>
<th>RID</th>
<th>age</th>
<th>student</th>
<th>credit-rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>youth</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>5</td>
<td>middle-aged</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>senior</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

**R1:** IF age=youth AND student=yes THEN buys_computer=yes

\[
n_{\text{covers}} = 3 \\
n_{\text{correct}} = 2
\]

\[
\text{coverage}(R1) = \frac{n_{\text{covers}}}{|D|} = \frac{3}{7} \\
\text{accuracy}(R1) = \frac{n_{\text{correct}}}{n_{\text{covers}}} = \frac{2}{3}
\]
How to use Rules for Classification

- **Predict the class label for tuple X**
  - If a rule $R$ is satisfied by $X$, the rule is said to be **triggered**
  - If a rule $R$ is the only one satisfied by $X$, the rule **fires** by returning the class prediction of $X$

- **Important**
  - Triggering $\neq$ Firing
  - More than one rule can be satisfied

- **Problems**
  - What if no rule is satisfied by $X$?
  - **Solution**: use a default rule that fires, for example, the most frequent class
  - If more than one rule are triggered, what if each rule specifies a different class?
Conflicting Rules

- **R1**: IF age=youth AND student=yes THEN buys_computer=yes
  
- **R2**: IF income=low THEN buys_computer=no

- Need a conflict resolution strategy
  - **Size ordering approach**
    - Give priority to the rule having the toughest requirement
    - **Toughness** is measured by the rule antecedent size
    - The triggering rule with the most attribute sets is fired
  
- **Rule ordering approach**
  - Prioritize the rules beforehand
  - **Class-based** ordering
  - **Rule-based** ordering
Rule Extraction From a Decision Tree

- One rule is created for each path from the root to a leave node.
- Each splitting criterion along a given path is logically ANDed to form the rule antecedent (IF part).
- The leaf node holds the class prediction (the rule consequent).

**R1:** IF age=youth AND student=no  
THEN buys_computer=no

**R2:** IF age=youth AND student=yes  
THEN buys_computer=yes

**R3:** IF age=middle-aged  
THEN buys_computer=yes

**R4:** IF age=senior AND credit_rating=excellent  
THEN buys_computer=yes

**R5:** IF age=senior AND credit_rating=fair  
THEN buys_computer=no
Decision tree rules are mutually **exclusive** and **exhaustive**

**Exclusive**
- No rule conflict, no two rules triggered for the same tuple
- One rule per leaf and any tuple is mapped to only one leaf

**Exhaustive**
- One rule for each attribute-value combination
- The set of rules does not require a default rule

**Note:** The order of rules does not matter when extracted from a decision tree

**Pruning Rules**
- Any rule that does not improve accuracy can be pruned
- Pruning may generate non-Mutually exclusive and non-exhaustive rules: C4.5 uses class-based ordering
IF-THEN rules are **directly** extracted from training data

Rules are learned sequentially (one at a time)  
*Note:* In decision trees rules are learned simultaneously

Each rule for a given class ideally covers many tuples of that class and hopefully no tuples from other classes

When a rule is learned, the tuples covered by the rule are removed (need of accurate rules but not necessarily high coverage)

The process repeats on the remaining tuples until a stopping condition:

- No tuples left
- The quality measure of a rule is below a threshold
How Are Rules learned?

- In a **general-to-specific** manner

**Example**

- In loan-application data, customers have *(age, income, education level, residence, credit-rating, and term of the loan)*
- Two classes: \textit{loan\_decision=accept} and \textit{loan\_decision=reject}

**Start with a general rule for class accept:**

\[
\text{IF} \quad \text{THEN} \quad \text{loan\_decision=accept}
\]

- Consider each possible attribute test that may be added to the rule
- Adopt a greedy depth-first strategy choosing the rule with high quality (use beam search where the k best attributes are maintained)
- Repeat the process till the rule reached an acceptable quality level

\[
\text{IF} \quad \text{income=high} \quad \text{AND} \quad \text{credit\_rating=excellent} \quad \text{THEN} \quad \text{loan\_decision=accept}
\]
Accuracy seems to be natural as a quality measure, but accuracy alone is not enough.

- R1: correctly classifies 18 tuples out of 20 (accuracy=90%)
- R2: correctly classifies 2 tuples out of 2 (accuracy=100%)

Coverage alone is not enough (cover many tuples of ≠ classes)

Use Entropy
Rule Quality Measure

- Using **Entropy** (Information Gain)

  \[ R: \text{IF} \text{ condition} \text{ THEN} \text{ class}=c \]

- If logically ANDing a given attribute test to \textbf{condition} we obtain \textbf{condition’}

  \[ R’: \text{IF} \text{ condition’} \text{ THEN} \text{ class}=c \]

- Test the potential rule \textbf{R’} using entropy
- Compute the \textbf{entropy} based on probabilities \( p_i \), where \( p_i \) is the probability of a class \( C_i \) in \( D \)
- \( D \) is the set of tuples covered by \textbf{R’}
- Entropy prefers conditions that cover a large number of tuples of a single class and few tuples of other classes
Rule Quality Measures

- Using **FOIL_Gain (First Order Inductive Learner- Gain)**

  \[
  R: \text{IF condition THEN class} = c
  \]

- If logically ANDing a given attribute test to condition we obtain condition’

  \[
  R': \text{IF condition'} \text{ THEN class} = c
  \]

- The FOIL Gain is computed by

  \[
  \text{FOIL_GAIN} = \text{pos'} \times \left( \log_2 \frac{\text{pos'}}{\text{pos'} + \text{neg'}} - \log_2 \frac{\text{pos}}{\text{pos} + \text{neg}} \right)
  \]

- \(\text{pos, pos'}\): the number of positive tuples covered by R and R’
- \(\text{neg, neg'}\): the number of negative tuples covered by R and R’

- It favors rules that have high accuracy and cover many positive tuples
Rule-based classification builds a model that is a set of rules.

Rules can be extracted from a decision tree or directly from training data.

Rule quality measures are important to assess the rules and to define orders for conflict resolution.