

IF-THEN Rules

Rule-based classifier makes use of a set of IF-THEN rules for classification. We can express a rule in the following form –

IF condition THEN conclusion

Let us consider a rule R1,

```
R1: IF age=youth AND student=yes  
    THEN buy_computer=yes
```

Points to remember –

- The IF part of the rule is called **rule antecedent** or **precondition**.
- The THEN part of the rule is called **rule consequent**.
- The antecedent part the condition consist of one or more attribute tests and these tests are logically ANDed.
- The consequent part consists of class prediction.

Note – We can also write rule R1 as follows:

```
R1: (age = youth) ^ (student = yes)(buys computer = yes)
```

If the condition holds true for a given tuple, then the antecedent is satisfied.

Rule Extraction

Here we will learn how to build a rule-based classifier by extracting IF-THEN rules from a decision tree.

Points to remember –

- One rule is created for each path from the root to the leaf node.
- To form a rule antecedent, each splitting criterion is logically ANDed.
- The leaf node holds the class prediction, forming the rule consequent.

Rule Induction Using Sequential Covering Algorithm

Sequential Covering Algorithm can be used to extract IF-THEN rules from the training data. We do not require to generate a decision tree first. In this algorithm, each rule for a given class covers many of the tuples of that class.

Some of the sequential Covering Algorithms are AQ, CN2, and RIPPER. As per the general strategy the rules are learned one at a time. For each time rules are learned, a tuple covered by the rule is removed and the process continues for the rest of the tuples. This is because the path to each leaf in a decision tree corresponds to a rule.

Note – The Decision tree induction can be considered as learning a set of rules simultaneously.

The Following is the sequential learning Algorithm where rules are learned for one class at a time. When learning a rule from a class C_i , we want the rule to cover all the tuples from class C only and no tuple from any other class.

```
Algorithm: Sequential Covering
```

```

Input:
D, a data set class-labeled tuples,
Att_vals, the set of all attributes and their possible values.

Output: A Set of IF-THEN rules.
Method:
Rule_set={ }; // initial set of rules learned is empty

for each class c do
    repeat
        Rule = Learn_One_Rule(D, Att_vals, c);
        remove tuples covered by Rule form D;
    until termination condition;

    Rule_set=Rule_set+Rule; // add a new rule to rule-set
end for
return Rule_Set;

```

Rule Pruning

The rule is pruned is due to the following reason –

- The Assessment of quality is made on the original set of training data. The rule may perform well on training data but less well on subsequent data. That's why the rule pruning is required.
- The rule is pruned by removing conjunct. The rule R is pruned, if pruned version of R has greater quality than what was assessed on an independent set of tuples.

FOIL is one of the simple and effective method for rule pruning. For a given rule R,

$$\text{FOIL_Prune} = \text{pos} - \text{neg} / \text{pos} + \text{neg}$$

where pos and neg is the number of positive tuples covered by R, respectively.

Note – This value will increase with the accuracy of R on the pruning set. Hence, if the FOIL_Prune value is higher for the pruned version of R, then we prune R.