Rule Induction

Pedro Larrañaga, Concha Bielza, Jose Luis Moreno

Computational Intelligence Group
Artificial Intelligence Department
Universidad Politécnica de Madrid

Machine Learning

Master in Data Science + Master in HMDA

Outline

Outline

Rule induction

General characteristics

- One of the most transparent supervised classification methods
- Rule induction models are easy to understand and apply
- Expert systems as rules elicited from an expert in a given domain. Rule induction aims to extract these rules automatically
- Usually rules are expressions of the form
  IF $(X_j = x_j \text{ AND } X_i = x_i \text{ AND} \cdots \text{ AND } X_k = x_k)$ *THEN* $C = c$

  where $(X_j = x_j$ AND $X_i = x_i$ AND $\cdots$ AND $X_k = x_k)$ is called the antecedent of the rule, and $C = c$ is the consequent of the rule
- More general than classification trees: each classification tree can be transformed into a rule induction model; however, the opposite is not always true

Expert systems -> rules are given by the expert
Rule induction -> rules are induced using a model and the data

Outline

IREP algorithm

IREP algorithm

- Repeated incremental pruning to produce error reduction (RIPPER) (Cohen, 1995) proposed a number of modifications of the algorithm called **incremental reduced error pruning** (IREP) (Fürnkranz and Widmer, 1994)
- The training data is split into a growing set and a pruning set Split in a partition: some data for growing the rule and some for pruning the rule
- First, an initial rule set is formed that overfits the growing set, using some heuristic method
- This overlarge rule set is then repeatedly simplified by applying one of a set of pruning operators. Typical pruning operators would be to delete any single literal or any single rule. The preferred pruning operator yields the greatest error reduction on the pruning set
- Simplification ends when applying any pruning operator that would increase the error on the pruning set
- In the Boolean case, an antecedent of a rule is simply a conjunction of literals (e.g. $X_{36} = 5$ AND $X_{56} = 6$ AND $X_{26} < 0.5$ AND $X_4 > 0.9$), and a rule set is a disjunction of rules
- IREP greedily builds up a rule set in, one rule at a time It introduces one rule at a time. Once we have the rule, we delete the instances covered by the rule
- After a rule is found, all instances covered by the rule (both positive and negative) are deleted from the growing set (line 8). This process is repeated until there are no positive instances (line 1) or until the rule found by IREP has an unacceptably large error rate (line 5)
(very bad rule)

The growing and the pruning sets are also partitioned in positive and negative samples

IREP algorithm

IREP algorithm

Algorithm 1: The IREP algorithm

Input : A split of the data set on Pos and Neg, an empty Ruleset
Output: A Ruleset

```
while Pos ≠ ∅ do   while we have positive samples
    /* grow and prune a new rule */
    Split (Pos, Neg) into (GrowPos, GrowNeg) and (PrunePos, PruneNeg)
    Rule = GrowRule(GrowPos, GrowNeg)
    Rule = PruneRule(Rule, PrunePos, PruneNeg)
    if The error rate of Rule on (PrunePos, PruneNeg) exceeds 50% then return Ruleset
    else if then
        Add Rule to Ruleset
        Remove instances covered by Rule from (Pos, Neg)
    endif
endwhile
```

IREP algorithm

IREP algorithm

- To build a rule: First, the positive (Pos) and negative (Neg) uncovered instances are randomly partitioned into two subsets: a growing set and a pruning set (line 2). The four disjoint subsets are denoted by: GrowPos (positive instances used for growing the rules); GrowNeg (negative instances used for growing the rules); PrunePos (positive instances used for pruning the rules); PruneNeg (negative instances used for pruning the rules)
- Next, a rule is "grown" (line 3). GrowRule begins with an empty conjunction of literals, and considers adding any literal of the form $X_i = x_i$, $X_i < x_i$, $X_i > x_i$. GrowRule repeatedly adds the literal that maximizes the FOIL criterion that is improved until the rule covers no negative instances from the growing data set
- Given a rule R and a more specific rule R' obtained from R after adding a literal, the FOIL criterion is:

$$\text{FOIL}\left(R, R', \text{GrowPos}, \text{GrowNeg}\right) = co\left[-\log_2\left(\frac{pos}{pos+neg}\right) + \log_2\left(\frac{pos'}{pos'+neg'}\right)\right]$$

R=current ruleset
R'=R+a new literal
We will keep the literal that gives the best value for this expression

where co denotes the percentage of positive instances covered by R and also covered by R' in GrowPos, pos is the number of positive instances covered by R in GrowPos (similarly for pos' and R'), neg is the number of negative instances covered by R in GrowNeg (similarly for neg' and R')

- After growing a rule, the rule is immediately pruned (line 4) by considering deleting any final sequence of literals from the rule output by the growing phase, choosing the deletion that maximizes the function

$$v(R, \text{PrunePos}, \text{PruneNeg}) = \frac{pos_R + (|\text{PruneNeg}| - neg_R)}{|\text{PrunePos}| + |\text{PruneNeg}|}$$

where $|\cdot|$ denotes cardinality and pos_R (neg_R) is the number of instances in PrunePos (PruneNeg) covered by rule R. This is repeated until no deletion improves the value of v

RIPPER algorithm

RIPPER algorithm

RIPPER differs from the IREP in:

1. An alternative metric to v for assessing the value of rules in the pruning phase
2. A new heuristic for determining when to stop adding rules to a rule set
3. A postpass that improves a rule set

Methods:
- OneR
- Cn2
- AQ
- Genetic based ML (GBML)

References

- J. Fürnkranz and G. Widmer (1994). Incremental reduced error pruning. *Machine Learning: Proceedings of the 11th Annual Conference*. Morgan Kaufmann, 70-77
- W. W. Cohen (1995). Fast effective rule induction. *Machine Learning: Proceedings of the 12th Annual Conference*. Morgan Kaufmann, 115-123