1.1. Rule Based Classification

Rule-based classifier makes use of a set of IF-THEN rules for classification. We can express a rule in the following from − IF condition THEN conclusion.

Let us consider a rule R1, R1: IF age = youth AND student = yes, THEN buy_computer = yes.

The IF part of the rule is called rule condition and the THEN part of the rule is called rule consequent. The condition part consists of one or more attribute tests and these tests are logically related (AND, OR). The consequent part consists of class prediction.

1.1.1. One Rule Classification

Consider the following dataset example. **Name** **Blood_type** **Give_birth** **Can_Fly** ** Live_in_water** **Class** human warm yes no no mammals python cold no no no reptiles pigeon warm no yes no birds eagle warm no yes no birds whale warm yes no yes mammals frog cold no no sometimes amphibians salmon cold no no yes fishes

We can generate a large number of rules with the data and check the error rates and pick the set of rules with the least error rates.

Here are a few rules.

```
R1:(Give Birth = no) →Birds
R2:(Can Fly = no) →Reptiles
R3:(Give Birth = no) ∧ (Can Fly = yes) →Birds
R4:(Give Birth = yes) ∧ (Blood Type = warm) →Mammals
R5:(Give Birth = no) ∧ (Can Fly = no) →Reptiles
R6:(Give Birth = no) ∧(Live in Water = yes) →Fishes
```

Observe the above set of rules. The rules R1 and R2 contain only one variable while R3, R4, R5 have 2 variables.

Rule R1 has an error rate of 33% as it would classify python as a bird, rule R2 has an error rate of 25% as it will misclassify mammals and amphibians as reptiles. Rule R5 has an error rate of 50% as it would classify amphibians as reptiles, instead we can modify the rule as

Name **Blood_type** **Give_birth** **Can_Fly** ** Live_in_water** **Class** leopardshark cold yes no yes fishes

Even though leopard shark is a fish, it can't be validated with the help of any of the rules created and therefore can't be classified.

1.1.2. Rule Extraction

The rule extraction from a decision tree is pretty straight forward. The leaf nodes contain all the classes and hence is the consequent part of the rule. Every path from the root node leading to the leaf node is a rule whose corresponding consequent part is stored in that leaf node.

1.1.3. Strategies for Learning Rules

General-to-Specific

Start with an empty rule. Add constraints to eliminate negative examples. Stop when only positive examples are covered.

Specific-to-General

Start with a rule that identifies a single random instance. Remove constraints to cover more positive examples. Stop when further generalization starts covering negatives.

1.1.4. Rule Pruning

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 altering the conjunct. The rule R is pruned, if pruned version of R has greater quality when assessed on an independent set of tuples(cross validation sets).

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

FOIL_Prune = pos - neg / pos + neg

where pos and neg is the number of positive and negative tuples covered by R, respectively. Postive tuples are those correctly predicted and negative are those incorrectly predicted. 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.

1.1.5. Pros and Cons

Questionnaire

1. Make a rule based classification for the following data

Age Spectacle_prescription Astigmatism Tear_production_rate Recommended_Lenses young myope no reduced none young myope no normal soft young myope yes reduced none young myope yes normal hard young hypermetrope no reduced none young hypermetrope no normal soft young hypermetrope yes reduced none young hypermetrope yes normal hard pre-presbyopic myope no reduced none pre-presbyopic myope no normal soft pre-presbyopic myope no normal soft pre-presbyopic myope yes reduced none pre-presbyopic myope yes normal hard pre-presbyopic hypermetrope no reduced none pre-presbyopic hypermetrope no normal soft pre-presbyopic hypermetrope yes reduced none pre-presbyopic hypermetrope yes normal none presbyopic myope no reduced none presbyopic myope no normal none presbyopic myope yes reduced none presbyopic myope yes normal hard presbyopic hypermetrope no reduced none presbyopic hypermetrope no normal soft presbyopic hypermetrope yes reduced none presbyopic hypermetrope yes normal noneRule we seek: If ? then recommendation = hard. 1st set of rules with one variable. Age = young 2/8 Age = pre-presbyopic 1/8 Age = presbyopic 1/8 Spectacle_prescription = myope 3/12 Spectacle_prescription = hypermetrope 1/12 Astigmatism = no 0/12 Astimagtism = yes 4/12 Tear_production_rate = reduced 0/12 Tear_production_rate = normal 4/12

The numbers on the right show the fraction of “correct” instances in the set singled out by that given condition. In this case, correct means that their recommendation is “hard.”

Updated Rule: If Astigmatism = yes, then recommendation = hard. The rule isn’t very accurate, getting only 4 out of 12 that it covers. So, it needs further refinement.

Current state: If Astigmatism = yes and ?, then recommendation = hard.

Age = young 2/4 Age = pre-presbyopic 1/4 Age = presbyopic 1/4 Spectacle_prescription = myope 3/6 Spectacle_prescription = hypermetrope 1/6 Tear_production_rate = reduced 0/6 Tear_production_rate = normal 4/6

Updated Rule: If Astigmatism = yes and Tear_production_rate = normal, then recommendation = hard. The rule covers satisfactry part of the given dataset, still let's try adding one more variable.

Current state: If Astigmatism = yes and Tear_production_rate = normal and ?, then recommendation = hard.

Age = young 2/2 Age = pre-presbyopic 1/2 Age = presbyopic 1/2 Spectacle_prescription = myope 3/3 Spectacle_prescription = hypermetrope 1/3

Final Rule: If Astigmatism = yes and Tear_production_rate = normal and Spectacle_prescription = myope, then recommendation = hard