
数据挖掘

第4章 分类-基于规则的分类器

教师：王东京

学院：计算机学院

邮箱：dongjing.wang@hdu.edu.cn

基于规则的分类器 Rule-Based Classifier

使用一组“if...then...”的规则对记录进行分类：

规则Rule: (条件 $Condition$) $\rightarrow y$

— 其中

- ◆ 条件是属性测试的合取 (conjunction) , 又称为规则前件 (rule antecedent) 或者前提 (precondition)
- ◆ y 是类别标签, 又称为规则后件 (rule consequent)

— 分类规则示例:

- ◆ (体温=恒温) \wedge (卵生=Yes) \rightarrow 鸟类
- ◆ (纳税收入 < 50K) \wedge (退款=Yes) \rightarrow 逃税=No

Rule-based Classifier (Example)

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

R2: (Give Birth = no) \wedge (Live in Water = yes) \rightarrow Fishes

R3: (Give Birth = yes) \wedge (Blood Type = warm) \rightarrow Mammals

R4: (Give Birth = no) \wedge (Can Fly = no) \rightarrow Reptiles

R5: (Live in Water = sometimes) \rightarrow Amphibians



基于规则的分类器的应用

规则 r 覆盖 (covers) 实例 x , 如果 x 的属性符合规则的条件

R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

R2: (Give Birth = no) \wedge (Live in Water = yes) \rightarrow Fishes

R3: (Give Birth = yes) \wedge (Blood Type = warm) \rightarrow Mammals

R4: (Give Birth = no) \wedge (Can Fly = no) \rightarrow Reptiles

R5: (Live in Water = sometimes) \rightarrow Amphibians (两栖动物)

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?



The rule R1 covers a hawk \Rightarrow Bird

The rule R3 covers the grizzly bear \Rightarrow Mammal

规则覆盖率 (Coverage) 和准确率 (Accuracy)

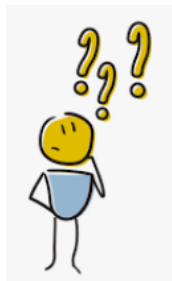
规则涵盖率Coverage:

- 满足规则前件 (rule antecedent) 的记录的比例

规则的准确性Accuracy:

- 满足规则前件的记录中也满足规则后件 (rule consequent) 的比例

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



(Status=Single) → No

覆盖率和准确率?

Coverage = 40%, Accuracy = 50%

下述规则的覆盖率和准确率分别为多少？

(Refund=Yes) → No

- ☐ A 30%, 10%
- ☒ B 30%, 100%
- ☐ C 70%, 10%
- ☐ D 70%, 100%

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

提交

基于规则的分类器的工作方式?

R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

R2: (Give Birth = no) \wedge (Live in Water = yes) \rightarrow Fishes

R3: (Give Birth = yes) \wedge (Blood Type = warm) \rightarrow Mammals

R4: (Give Birth = no) \wedge (Can Fly = no) \rightarrow Reptiles

R5: (Live in Water = sometimes) \rightarrow Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?



A lemur (狐猴) triggers rule R3, so it is classified as a mammal

A turtle (海龟) triggers both R4 and R5 (冲突)

A dogfish shark (角鲨鱼) triggers none of the rules

规则集的性质

Characteristics of Rule Sets: Strategy 1

互斥规则 (Mutually exclusive)

- 如果规则彼此独立，则分类器包含互斥规则
- 每条记录**最多**被一条规则覆盖

穷举规则 (Exhaustive rules)

- 如果分类器考虑了属性值的每种可能组合，则它具有穷举覆盖范围
- 每条记录**至少**受一条规则覆盖

规则集的性质：如果不满足上述条件

规则不是互斥的

- 一条记录可能会触发多条规则
- 解决方案？
 - ◆有序规则集：按顺序
 - ◆无序规则集：使用投票方案

不满足穷举规则

- 一条记录可能不会触发任何规则
- 解决方案？
 - ◆使用默认类

有序规则集 Ordered Rule Set

规则按优先级 (priority) 排序

- 有序规则集称为**决策列表 (decision list)**

将测试记录提交给分类器时？

- 它会被分配所触发的最高等级规则的 (highest ranked rule) 类别标签
- 如果未触发任何规则，则将其分配给默认类

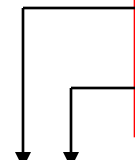
R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

R2: (Give Birth = no) \wedge (Live in Water = yes) \rightarrow Fishes

R3: (Give Birth = yes) \wedge (Blood Type = warm) \rightarrow Mammals

R4: (Give Birth = no) \wedge (Can Fly = no) \rightarrow Reptiles

R5: (Live in Water = sometimes) \rightarrow Amphibians



Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?

规则排序方案 Rule Ordering Schemes

基于规则的排序 Rule-based ordering

- 各个规则根据其**质量**进行排名

基于类的排序 Class-based ordering

- 属于**同一类**的规则一起出现

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

构建分类规则 Building Classification Rules

直接方法 Direct Method:

- ◆ 直接从数据中提取规则
- ◆ Examples: RIPPER, CN2, Holte's 1R

间接方法 Indirect Method:

- ◆ 从其他分类模型中提取规则（例如决策树，神经网络等）
- ◆ Examples: C4.5rules

直接方法: 顺序覆盖 Sequential Covering

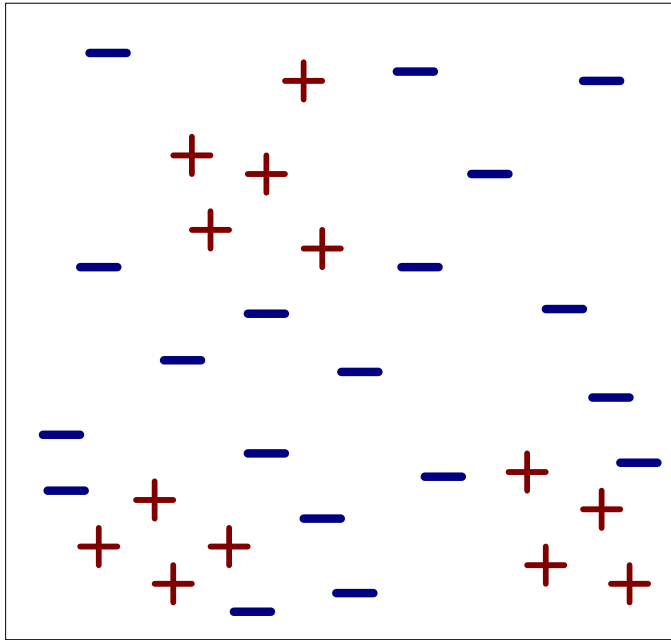
1. 从空规则开始
2. 使用“**学习一个规则 (Learn-One-Rule)**”函数增长规则 (grow a rule)
3. 删除训练集中被规则覆盖的样本记录
4. 重复步骤 (2) 和 (3) , 直到满足停止标准

直接方法: 顺序覆盖 Sequential Covering

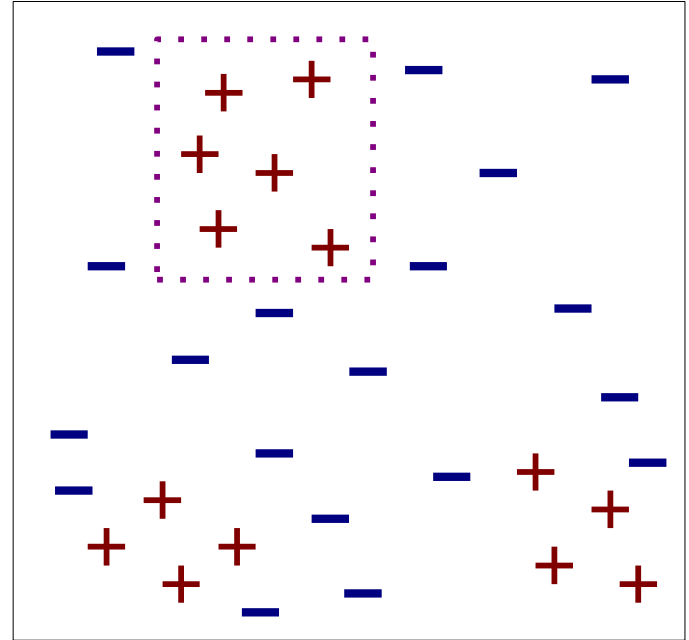
算法 5.1 顺序覆盖算法

- 1: 令 E 是训练记录, A 是属性-值对的集合 $\{(A_j, v_j)\}$
 - 2: 令 Y_o 是类的有序集 $\{y_1, y_2, \dots, y_k\}$
 - 3: 令 $R = \{\}$ 是初始规则列表
 - 4: **for** 每个类 $y \in Y_o - \{y_k\}$ **do**
 - 5: **while** 终止条件不满足 **do**
 - 6: $r \leftarrow \text{Learn-One-Rule}(E, A, y)$
 - 7: 从 E 中删除被 r 覆盖的训练记录
 - 8: 追加 r 到规则列表尾部: $R \leftarrow R \vee r$
 - 9: **end while**
 - 10: **end for**
 - 11: 把默认规则 $\{\} \rightarrow y_k$ 插入到规则列表 R 尾部
-

示例：Sequential Covering

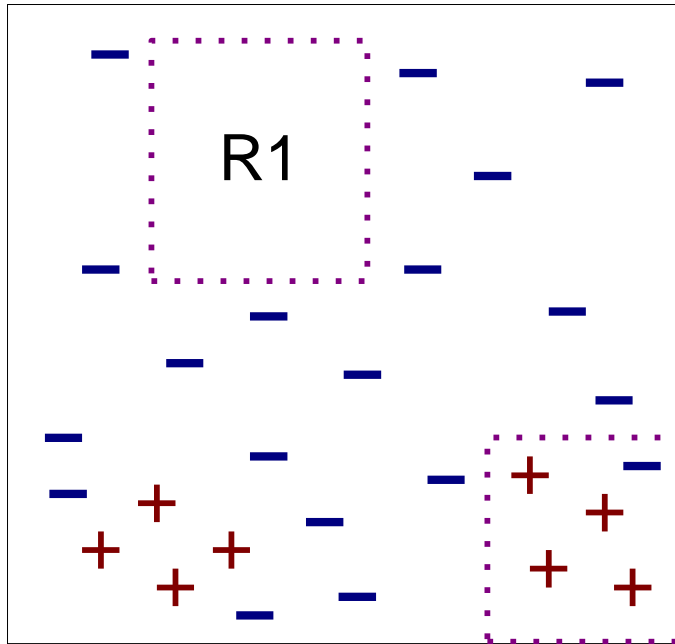


(i) Original Data

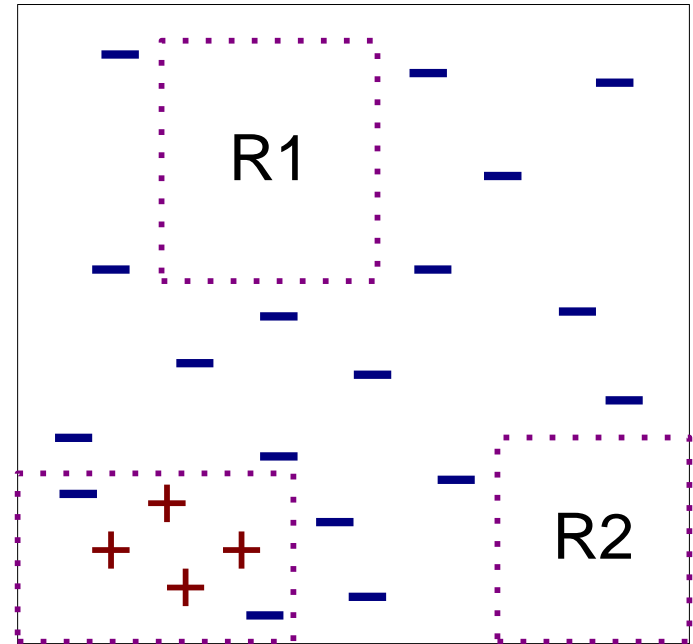


(ii) Step 1

示例：Sequential Covering



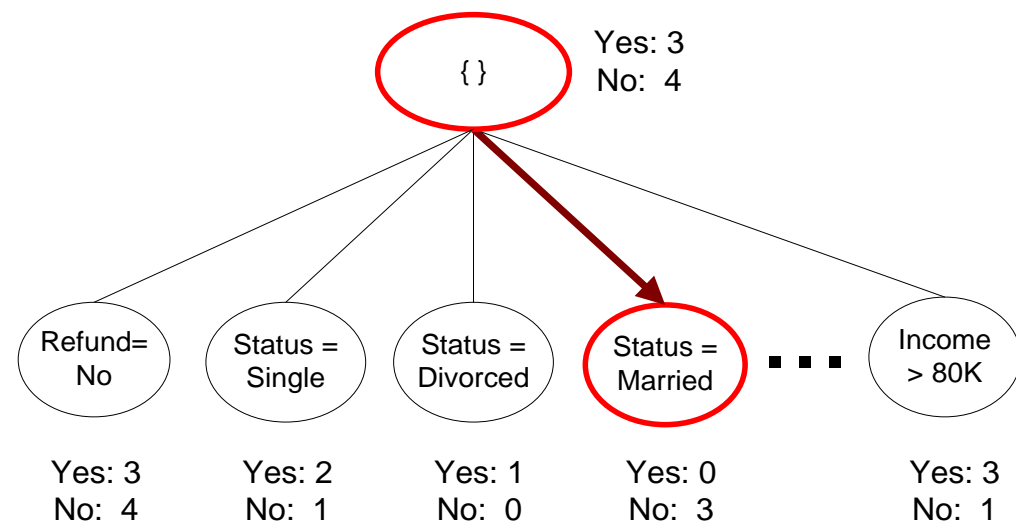
(iii) Step 2



(iv) Step 3

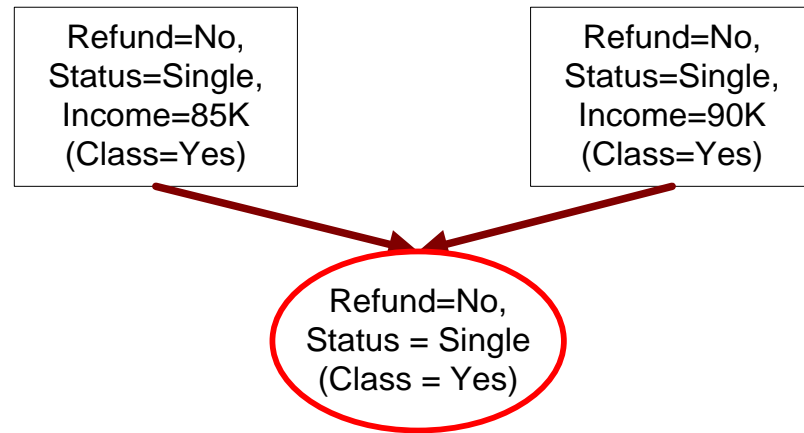
规则增长策略 Rule Growing

Two common strategies



(a) General-to-specific

从一般到特殊的策略



(b) Specific-to-general

从特殊到一般的策略

规则评估 Rule Evaluation

Foil信息增益 (Foil's Information Gain)

FOIL: First Order Inductive Learner – an early rule-based learning algorithm

- $R_0: \{\} \Rightarrow \text{class}$ (初始规则 initial rule)
- $R_1: \{A\} \Rightarrow \text{class}$ (增加合取项之后的规则 rule after adding conjunct)
- $\text{Gain}(R_0, R_1) = p_1 \times \left[\log_2 \left(\frac{p_1}{p_1 + n_1} \right) - \log_2 \left(\frac{p_0}{p_0 + n_0} \right) \right]$
- p_0 : R_0 所覆盖的正样本数
 n_0 : R_0 所覆盖的负样本数
 p_1 : R_1 所覆盖的正样本数
 n_1 : R_1 所覆盖的负样本数

直接方法: RIPPER (选学)

For 2-class problem, choose one of the classes as positive class, and the other as negative class

- Learn rules for positive class
- Negative class will be default class

For multi-class problem

- Order the classes according to increasing class prevalence (fraction of instances that belong to a particular class)
- Learn the rule set for smallest class first, treat the rest as negative class
- Repeat with next smallest class as positive class

RIPPER (选学)

Growing a rule:

- Start from empty rule
- Add conjuncts as long as they improve FOIL's information gain
- Stop when rule no longer covers negative examples
- Prune the rule immediately using incremental reduced error pruning
- Measure for pruning: $v = (p-n)/(p+n)$
 - ◆ p : number of positive examples covered by the rule in the validation set
 - ◆ n : number of negative examples covered by the rule in the validation set
- Pruning method: delete any final sequence of conditions that maximizes v

RIPPER (选学)

Building a Rule Set:

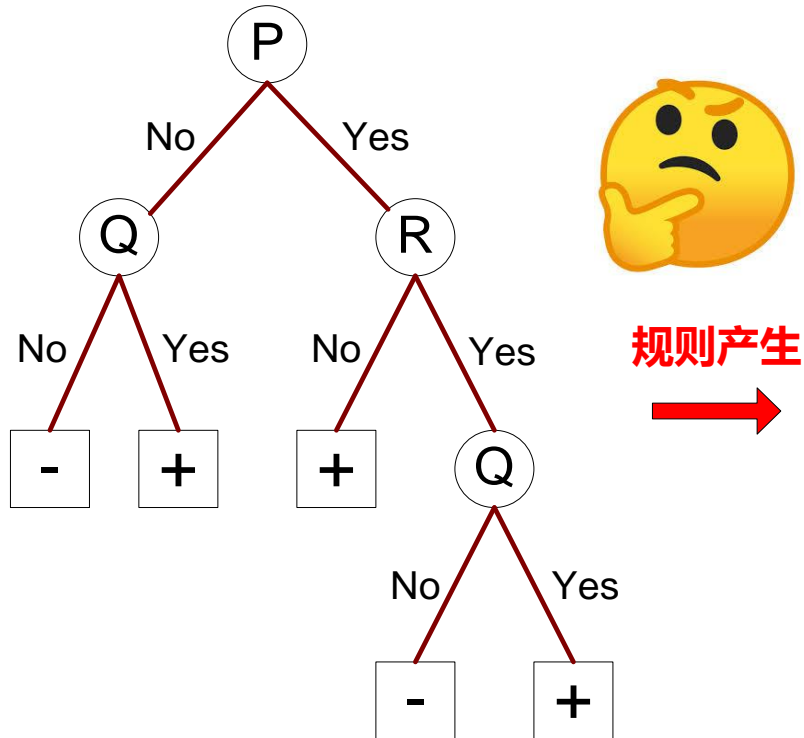
- Use sequential covering algorithm
 - ◆ Finds the best rule that covers the current set of positive examples
 - ◆ Eliminate both positive and negative examples covered by the rule
- Each time a rule is added to the rule set, compute the new description length
 - ◆ Stop adding new rules when the new description length is d bits longer than the smallest description length obtained so far

RIPPER (选学)

Optimize the rule set:

- For each rule r in the rule set R
 - ◆ Consider 2 alternative rules:
 - Replacement rule (r^*): grow new rule from scratch
 - Revised rule(r'): add conjuncts to extend the rule r
 - ◆ Compare the rule set for r against the rule set for r^* and r'
 - ◆ Choose rule set that minimizes MDL principle
- Repeat rule generation and rule optimization for the remaining positive examples

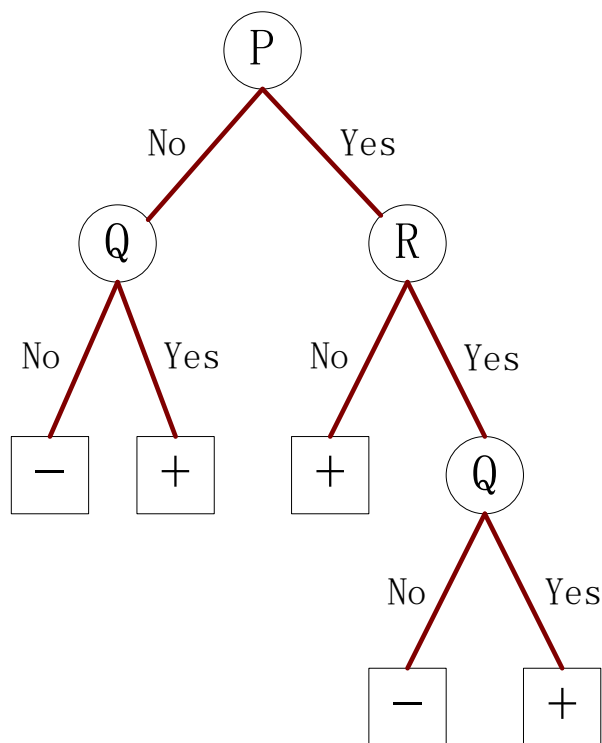
间接方法 Indirect Methods



Rule Set

- r1: (P=No, Q=No) ==> -
- r2: (P=No, Q=Yes) ==> +
- r3: (P=Yes, R=No) ==> +
- r4: (P=Yes, R=Yes, Q=No) ==> -
- r5: (P=Yes, R=Yes, Q=Yes) ==> +

间接方法 Indirect Methods



Rule Set

r1: (P=No,Q=No) ==> -

r2: (P=No,Q=Yes) ==> +

r3: (P=Yes,R=No) ==> +

r4: (P=Yes,R=Yes,Q=No) ==> -

r5: (P=Yes,R=Yes,Q=Yes) ==> +



$r2': (Q = \text{Yes}) \rightarrow +$

$r3: (P = \text{Yes}) \wedge (R = \text{No}) \rightarrow +$

Indirect Method: C4.5 rules (选学)

Extract rules from an unpruned decision tree

For each rule, $r: A \rightarrow y$,

- consider an alternative rule $r': A' \rightarrow y$ where A' is obtained by removing one of the conjuncts in A
- Compare the pessimistic error rate for r against all r 's
- Prune if one of the alternative rules has lower pessimistic error rate
- Repeat until we can no longer improve generalization error

Indirect Method: C4.5 rules (选学)

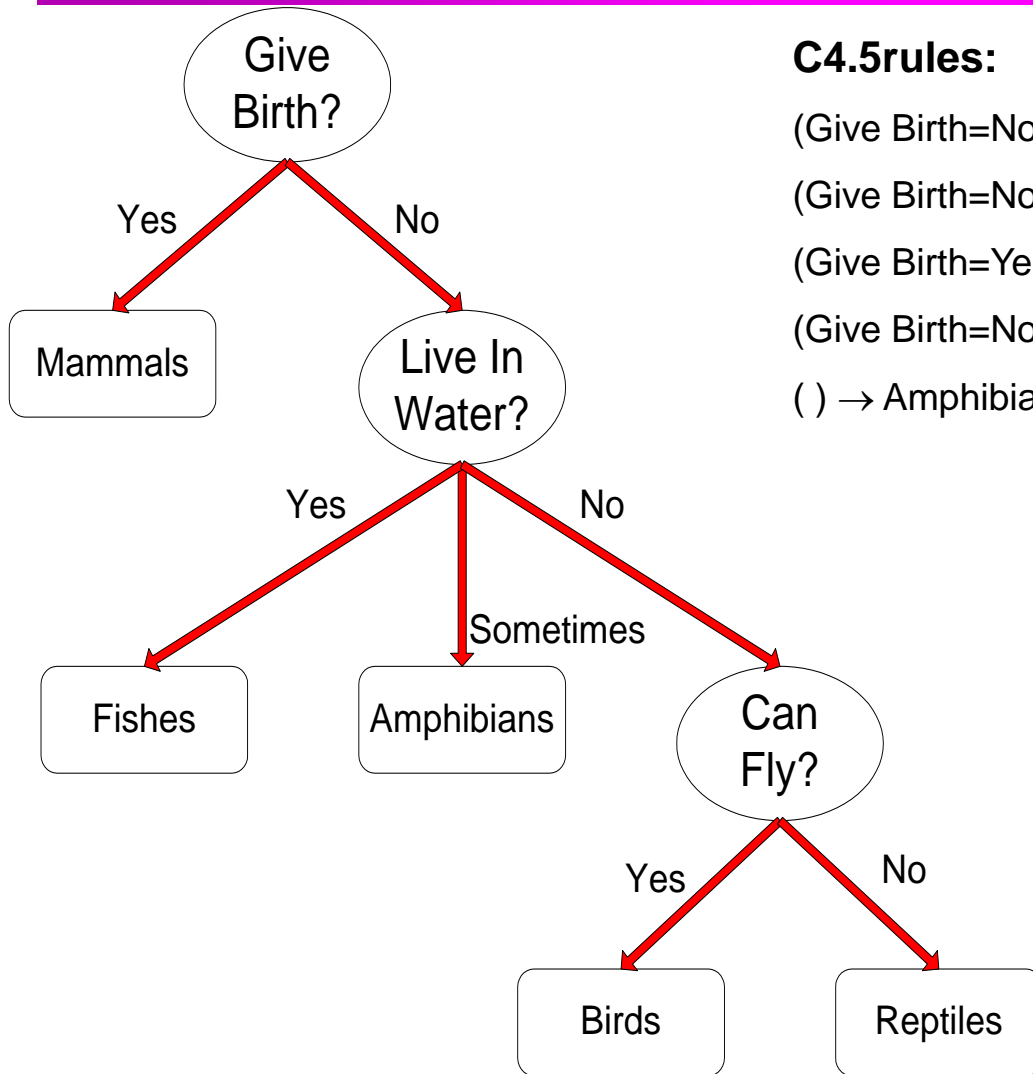
Instead of ordering the rules, order subsets of rules (**class ordering**)

- Each subset is a collection of rules with the same rule consequent (class)
- Compute description length of each subset
 - ◆ $\text{Description length} = L(\text{error}) + g L(\text{model})$
 - ◆ g is a parameter that takes into account the presence of redundant attributes in a rule set (default value = 0.5)

Example (选学)

Name	Give Birth	Lay Eggs	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	no	yes	mammals
python	no	yes	no	no	no	reptiles
salmon	no	yes	no	yes	no	fishes
whale	yes	no	no	yes	no	mammals
frog	no	yes	no	sometimes	yes	amphibians
komodo	no	yes	no	no	yes	reptiles
bat	yes	no	yes	no	yes	mammals
pigeon	no	yes	yes	no	yes	birds
cat	yes	no	no	no	yes	mammals
leopard shark	yes	no	no	yes	no	fishes
turtle	no	yes	no	sometimes	yes	reptiles
penguin	no	yes	no	sometimes	yes	birds
porcupine	yes	no	no	no	yes	mammals
eel	no	yes	no	yes	no	fishes
salamander	no	yes	no	sometimes	yes	amphibians
gila monster	no	yes	no	no	yes	reptiles
platypus	no	yes	no	no	yes	mammals
owl	no	yes	yes	no	yes	birds
dolphin	yes	no	no	yes	no	mammals
eagle	no	yes	yes	no	yes	birds

C4.5 versus C4.5rules versus RIPPER (选学)



C4.5rules:

(Give Birth=No, Can Fly=Yes) → Birds

(Give Birth=No, Live in Water=Yes) → Fishes

(Give Birth=Yes) → Mammals

(Give Birth=No, Can Fly=No, Live in Water=No) → Reptiles

() → Amphibians

RIPPER:

(Live in Water=Yes) → Fishes

(Have Legs=No) → Reptiles

(Give Birth=No, Can Fly=No, Live In Water=No) → Reptiles

(Can Fly=Yes, Give Birth=No) → Birds

() → Mammals

C4.5 versus C4.5rules versus RIPPER (选学)

C4.5 and C4.5rules:

		PREDICTED CLASS				
		Amphibians	Fishes	Reptiles	Birds	Mammals
ACTUAL CLASS	Amphibians	2	0	0	0	0
	Fishes	0	2	0	0	1
	Reptiles	1	0	3	0	0
	Birds	1	0	0	3	0
	Mammals	0	0	1	0	6

RIPPER:

		PREDICTED CLASS				
		Amphibians	Fishes	Reptiles	Birds	Mammals
ACTUAL CLASS	Amphibians	0	0	0	0	2
	Fishes	0	3	0	0	0
	Reptiles	0	0	3	0	1
	Birds	0	0	1	2	1
	Mammals	0	2	1	0	4

基于规则的分类器的特点

有与决策树非常相似的特征

- 与决策树一样具有很强的表达能力
- 容易解释
- 性能可媲美决策树
- 可以处理冗余属性

更适合处理不平衡的类别

难以处理测试集中的缺失值