

# Rule-based Classifiers

Anna Monreale  
Computer Science Department

Introduction to Data Mining, 2<sup>nd</sup> Edition  
Chapter 5.1

# Rule-based Classifier

- Classify records by using a collection of “if...then...” rules
- Rule:  $(Condition) \rightarrow y$ 
  - where
    - *Condition* is a conjunction of tests on attributes
    - $y$  is the class label
  - Examples of classification rules:
    - $(\text{Blood Type}=\text{Warm}) \wedge (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}$
    - $(\text{Taxable Income} < 50\text{K}) \wedge (\text{Refund}=\text{Yes}) \rightarrow \text{Evade}=\text{No}$

# Rule-based Classifier (Example)

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
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

# Application of Rule-Based Classifier

- A rule  $r$  **covers** an instance  $x$  if the attributes of the instance satisfy the condition of the rule

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
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

# Rule Coverage and Accuracy

- Coverage of a rule:
  - Fraction of records that satisfy the antecedent of a rule
- Accuracy of a rule:
  - Fraction of records that satisfy the antecedent that also satisfy the consequent of a rule

<i>Tid</i>	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

Coverage = 40%, Accuracy = 50%

# How does a Rule-based Classifier Work?

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 triggers rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules

# Characteristics of Rule Sets: Strategy I

- **Mutually exclusive rules**
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered **by at most one rule**
- **Exhaustive rules**
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered **by at least one rule**

# Characteristics of Rule Sets: Strategy 2

- **Rules are not mutually exclusive**
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes
- **Rules are not exhaustive**
  - A record may not trigger any rules
  - Solution?
    - Use a default class



# Ordered Rule Set

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a **decision list**
- When a test record is presented to the classifier
  - It is assigned to the class label of the **highest ranked rule** it has triggered
  - If none of the rules fired, it is assigned to the default class (typically majority class)

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	?

# Rule Ordering Schemes

- **Rule-based ordering**
  - Individual rules are ranked based on their quality
- **Class-based ordering**
  - Rules that belong to the same class appear together
- **Unordered rules**
  - vote schema

# Rule-based Ordering Scheme

- Individual rules are ranked based on their quality measured by **accuracy, coverage, or size** (number of attribute tests in the rule antecedent)
- The rule set is known as a **decision list**
- The record  $X$  is classified by the rule with the highest priority and any other rule that satisfies is ignored
- Each rule in a decision list implies the **negation** of the rules that come before it in the list
- Rules in a decision list are more difficult to interpret.

## 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 Scheme

- Rules that belong to the same class appear together
- The classes are sorted in order of decreasing “**importance**” such as by decreasing order of prevalence.
- Alternatively, they may be sorted based on the **misclassification cost** per class.
- Within each class, the rules are not ordered

## 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

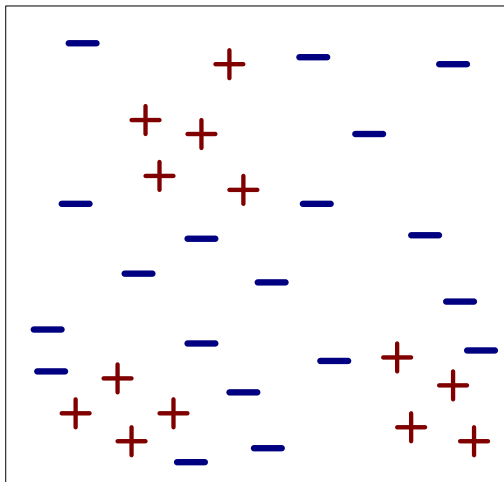
# Building Classification Rules

- **Direct Method:**
  - Extract rules directly from data
  - Examples: RIPPER, CN2, Holte's 1R
- **Indirect Method:**
  - Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  - Examples: C4.5rules

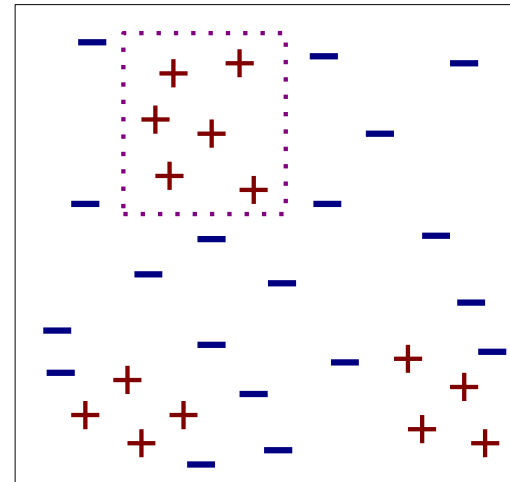
# Direct Method: Sequential Covering

1. Start from an empty rule
2. For each class
  1. Grow a rule using the **Learn-One-Rule function**
  2. Remove training records covered by the rule
  3. Repeat Step until stopping criterion is met

# Example of Sequential Covering

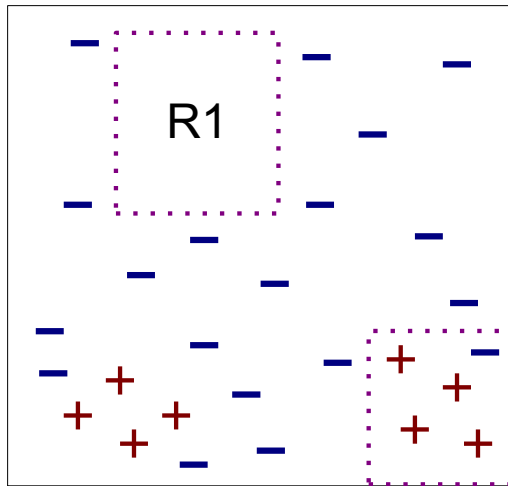


(i) Original Data

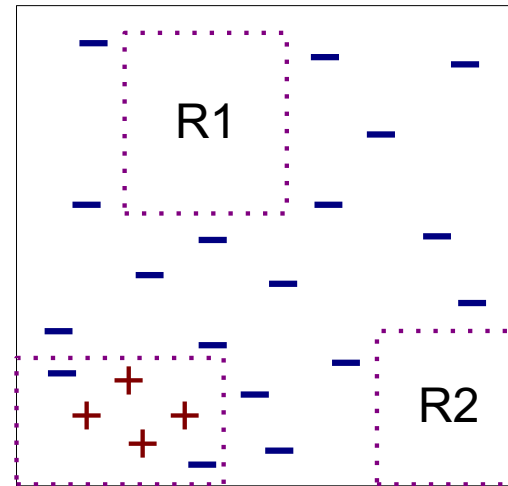


(ii) Step 1

# Example of Sequential Covering...



(iii) Step 2



(iv) Step 3



# Learn-One-Rule Function

- The goal is to extract a classification rule covering many positive records and none (few) negative ones
- Finding optimal rule requires high computational time
- Greedy strategy by refining an initial rule

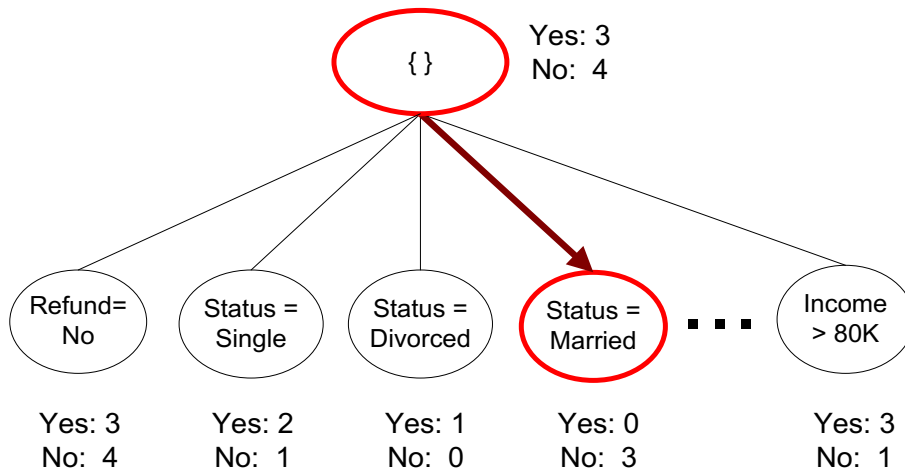
# Greedy approach

- The approach for growing rules is greedy
  - Based on some evaluation measure
- Rules are extracted **one class per time**
- The criterion for deciding the order of the class to consider depends on:
  - Class prevalence
  - Miss classification error for a given class

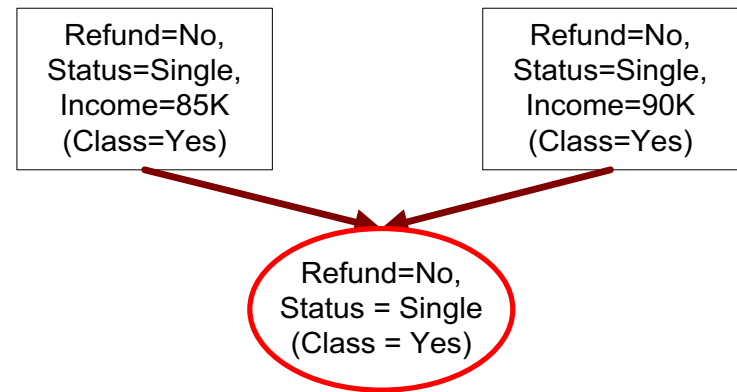
# Rule Growing

Two common strategies:

The initial rule contains the same conjuncts as the attribute values a selected record



(a) General-to-specific



(b) Specific-to-general

## Rule Evaluation for growing rules based on rule coverage

– Accuracy =  $\frac{n_c}{n}$

– Laplace =  $\frac{n_c + 1}{n + k}$

– M-estimate =  $\frac{n_c + kp}{n + k}$

$n$  : Number of instances covered by rule

$n_c$  : Number of instances covered by rule with class  $c$

$k$  : Number of classes

$p$  : Prior probability

# Rule Evaluation for growing rules based on support count

- Foil's Information Gain

- R0: {} => class (initial rule)

- R1: {A} => class (rule after adding conjunct)

- $$Gain(R_0, R_1) = p_1 \times \left[ \log_2 \left( \frac{p_1}{p_1 + n_1} \right) - \log_2 \left( \frac{p_0}{p_0 + n_0} \right) \right]$$

- $p_0$ : number of positive instances covered by R0

- $n_0$ : number of negative instances covered by R0

- $p_1$ : number of positive instances covered by R1

- $n_1$ : number of negative instances covered by R1

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

# Rule Evaluation for growing rules based on statistical test

- Given a rule we can compute the Likelihood ratio statistic

The diagram illustrates the Likelihood ratio statistic formula  $R = 2 \sum_{i=1}^k f_i \log(f_i/e_i)$ . Three callout boxes are connected to the formula by arrows:

- A box labeled "N. of classes" points to the upper limit  $k$  of the summation.
- A box labeled "N. of records covered by the rule" points to the term  $f_i$  in the summation.
- A box labeled "Expected frequency of a rule that makes random predictions." points to the term  $e_i$  in the summation.

- A large  $R$  value suggests that the number of correct predictions made by the rule is significantly larger than that expected by random guessing

# Example

$$R = 2 \sum_{i=1}^k f_i \log(f_i/e_i),$$

**Dataset:** 60 positive records and 100 negative records

**Rule 1:** covers 50 positive records and 5 negative examples,

**Rule 2:** covers 2 positive records and no negative examples.

## Rule 1

- Expected frequency for the positive class is  $e_+ = 55 \times 60 / 160 = 20.625$
- Expected frequency for the negative class is  $e_- = 55 \times 100 / 160 = 34.375$

$$R(r_1) = 2 \times [50 \times \log_2(50/20.625) + 5 \times \log_2(5/34.375)] = 99.9.$$

## Rule 2

- Expected frequency for the positive class is  $e_+ = 2 \times 60 / 160 = 0.75$
- Expected frequency for the negative class is  $e_- = 2 \times 100 / 160 = 1.25$

$$R(r_2) = 2 \times [2 \times \log_2(2/0.75) + 0 \times \log_2(0/1.25)] = 5.66.$$

# Direct Method: RIPPER

- 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



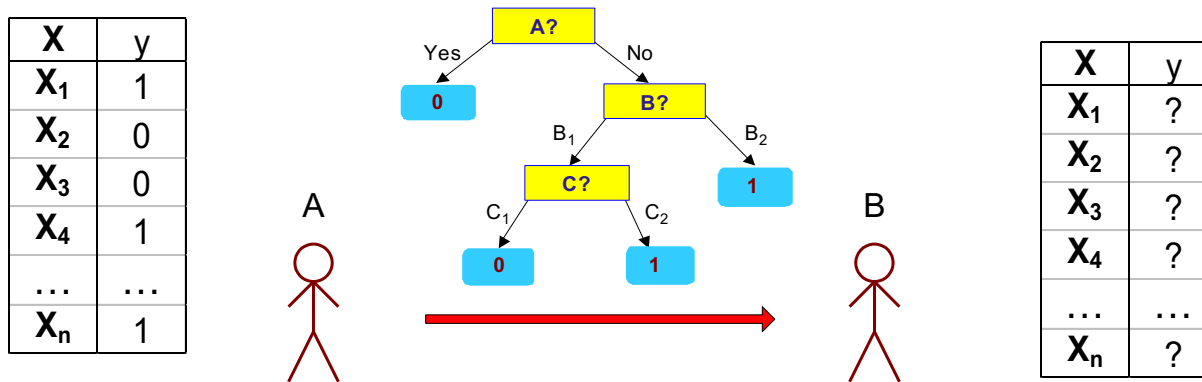
# Direct Method: RIPPER

- 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  $v$

# Direct Method: RIPPER

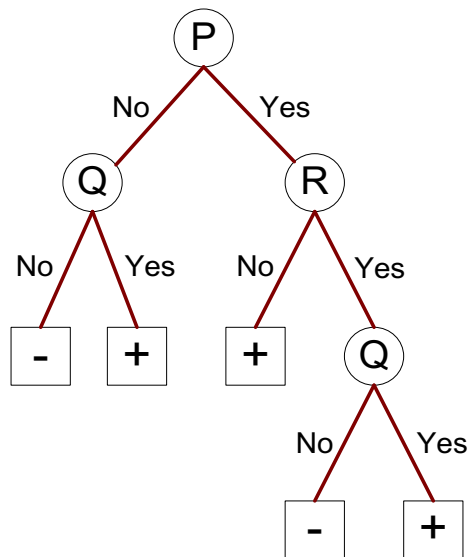
- Building a Rule Set:
  - Use sequential covering algorithm
    - Finds the best rule that covers the current set of positive examples
    - Eliminate both positive and negative examples covered by the rule
  - Each time a rule is added to the rule set, compute the new description length
    - Stop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

# Minimum Description Length (MDL)



- $Cost(Model, Data) = Cost(Data|Model) + \alpha \times Cost(Model)$ 
  - Cost is the number of bits needed for encoding.
  - Search for the least costly model.
- $Cost(Data|Model)$  encodes the misclassification errors.
- $Cost(Model)$  uses node encoding (number of children) plus splitting condition encoding.

# 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) ==> +

# Indirect Method: C4.5rules

- 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**
  - Remove **duplicate** rules
  - Repeat until we can no longer improve generalization error
  - When choosing a **default class**, C4.5 does not choose the majority class, because this class will likely have many rules for its tuples. Instead, it selects **the class that contains the most training tuples that were not covered by any rule.**

# Pessimistic Error Estimate

- **Pessimistic Error Estimate** of a rule set  $T$  with  $k$  rules:

$$err_{gen}(T) = err(T) + \Omega \times \frac{k}{N_{train}}$$

- $err(T)$ : error rate on all training records
- $\Omega$ : trade-off hyper-parameter relative cost of adding a rule
- $k$ : number of rules nodes
- $N_{train}$ : total number of training records

# Indirect Method: C4.5rules

- 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
  - Description length =  $L(error) + g L(model)$
  - $g$  is a parameter that takes into account the presence of redundant attributes in a rule set (default value = 0.5)

# Advantages of Rule-Based Classifiers

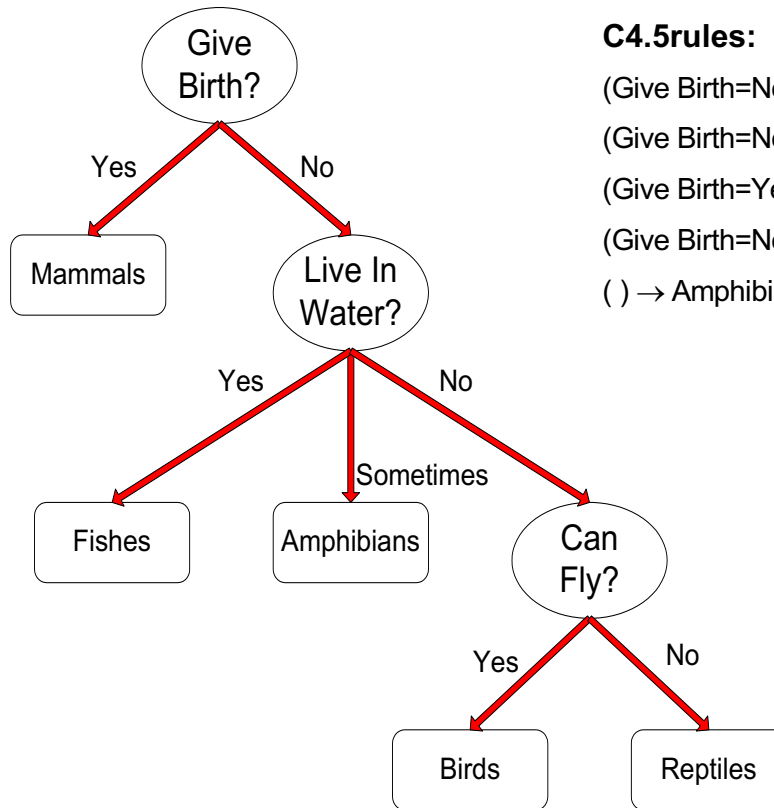
- Has characteristics quite similar to decision trees
  - As highly expressive as decision trees
  - Easy to interpret
  - Performance comparable to decision trees
  - Can handle redundant attributes
- Better suited for handling imbalanced classes
- Harder to handle missing values in the test set



# Example

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



## 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

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

# References

- Rule-Based Classifiers.  
Chapter 5.1. Introduction to  
Data Mining.

