Advanced classification methods
Ensemble Methods

• Construct a set of classifiers from the training data

• Predict class label of previously unseen records by aggregating predictions made by multiple classifiers
General Idea

Original Training data

Step 1: Create Multiple Data Sets

D

D_1

D_2

D_{t-1}

D_t

Step 2: Build Multiple Classifiers

C_1

C_2

C_{t-1}

C_t

Step 3: Combine Classifiers

C^*
Why does it work?

• Suppose there are 25 base classifiers
  – Each classifier has error rate, \( \varepsilon = 0.35 \)
  – Assume classifiers are independent
  – Probability that the ensemble classifier makes a wrong prediction:

\[
\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1 - \varepsilon)^{25-i} = 0.06
\]
Examples of Ensemble Methods

• How to generate an ensemble of classifiers?
  – Bagging

  – Boosting
Bagging

- Sampling with replacement

<table>
<thead>
<tr>
<th>Original Data</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagging (Round 1)</td>
<td>7</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>2</td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Bagging (Round 2)</td>
<td>1</td>
<td>4</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Bagging (Round 3)</td>
<td>1</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>

- Build classifier on each bootstrap sample

- Each sample has probability \((1 - \frac{1}{n})^n\) of being selected
Boosting

• An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  – Initially, all N records are assigned equal weights
  – Unlike bagging, weights may change at the end of boosting round
Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

<table>
<thead>
<tr>
<th>Original Data</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boosting (Round 1)</td>
<td>7</td>
<td>3</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>4</td>
<td>10</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>Boosting (Round 2)</td>
<td>5</td>
<td>4</td>
<td>9</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Boosting (Round 3)</td>
<td><strong>4</strong></td>
<td><strong>4</strong></td>
<td>8</td>
<td><strong>10</strong></td>
<td><strong>4</strong></td>
<td>5</td>
<td><strong>4</strong></td>
<td>6</td>
<td>3</td>
<td><strong>4</strong></td>
</tr>
</tbody>
</table>

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds
Example: AdaBoost

• Base classifiers: $C_1, C_2, \ldots, C_T$

• Error rate:

$$\varepsilon_i = \frac{1}{N} \sum_{j=1}^{N} w_j \delta(C_i(x_j) \neq y_j)$$

• Importance of a classifier:

$$\alpha_i = \frac{1}{2} \ln \left( \frac{1 - \varepsilon_i}{\varepsilon_i} \right)$$
Example: AdaBoost

- **Weight update:**
  \[
  w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \begin{cases} 
    \exp^{-\alpha_j} & \text{if } C_j(x_i) = y_i \\
    \exp^{\alpha_j} & \text{if } C_j(x_i) \neq y_i 
  \end{cases}
  \]
  where \(Z_j\) is the normalization factor

- If any intermediate rounds produce error rate higher than 50%, the weights are reverted back to \(1/n\) and the resampling procedure is repeated

- **Classification:**
  \[
  C^*(x) = \arg \max_y \sum_{j=1}^T \alpha_j \delta(C_j(x) = y)
  \]
Illustrating AdaBoost

Initial weights for each data point

Original Data

Data points for training

Boosting Round 1

$B_1 = 1.9459$
Illustrating AdaBoost

Boosting Round 1

Boosting Round 2

Boosting Round 3

Overall
Rule-Based Classifier

• Classify records by using a collection of “if... then...” rules

• Rule: \((\text{Condition}) \rightarrow y\)
  – where
    • \(\text{Condition}\) is a conjunction of attributes
    • \(y\) is the class label
  – \(LHS\): rule antecedent or condition
  – \(RHS\): rule consequent
  – Examples of classification rules:
    • \((\text{Blood Type}=\text{Warm}) \land (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}\)
    • \((\text{Taxable Income} < 50\text{K}) \land (\text{Refund}=\text{Yes}) \rightarrow \text{Evade}=\text{No}\)
Rule-based Classifier (Example)

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

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds
R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes
R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals
R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles
R5: (Live in Water = sometimes) → Amphibians
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: \((\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{yes}) \rightarrow \text{Birds}\)
R2: \((\text{Give Birth} = \text{no}) \land (\text{Live in Water} = \text{yes}) \rightarrow \text{Fishes}\)
R3: \((\text{Give Birth} = \text{yes}) \land (\text{Blood Type} = \text{warm}) \rightarrow \text{Mammals}\)
R4: \((\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{no}) \rightarrow \text{Reptiles}\)
R5: \((\text{Live in Water} = \text{sometimes}) \rightarrow \text{Amphibians}\)

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>hawk</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>?</td>
</tr>
<tr>
<td>grizzly bear</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>?</td>
</tr>
</tbody>
</table>

The rule R1 covers a hawk \( \rightarrow \text{Bird} \)
The rule R3 covers the grizzly bear \( \rightarrow \text{Mammal} \)
### 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 both the antecedent and consequent of a rule

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

(Status=Single) $\rightarrow$ No

Coverage = 40%, Accuracy = 50%
How does Rule-based Classifier Work?

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>lemur</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>?</td>
</tr>
<tr>
<td>turtle</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>?</td>
</tr>
<tr>
<td>dogfish shark</td>
<td>cold</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>?</td>
</tr>
</tbody>
</table>

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-Based Classifier

• 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
From Decision Trees To Rules

Classification Rules

(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

Rules are mutually exclusive and exhaustive
Rule set contains as much information as the tree
Rules Can Be Simplified

Initial Rule: \((\text{Refund}=\text{No}) \land (\text{Status}=\text{Married}) \rightarrow \text{No}\)

Simplified Rule: \((\text{Status}=\text{Married}) \rightarrow \text{No}\)
Effect of Rule Simplification

• Rules are no longer mutually exclusive
  – A record may trigger more than one rule
  – Solution?
    • Ordered rule set
    • Unordered rule set – use voting schemes

• Rules are no longer 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

R1: \((\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{yes}) \rightarrow \text{Birds}\)
R2: \((\text{Give Birth} = \text{no}) \land (\text{Live in Water} = \text{yes}) \rightarrow \text{Fishes}\)
R3: \((\text{Give Birth} = \text{yes}) \land (\text{Blood Type} = \text{warm}) \rightarrow \text{Mammals}\)
R4: \((\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{no}) \rightarrow \text{Reptiles}\)
R5: \((\text{Live in Water} = \text{sometimes}) \rightarrow \text{Amphibians}\)

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>turtle</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>?</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Rule-based Ordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Refund=Yes) ==&gt; No</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Single,Divorced}, Taxable Income&lt;80K) ==&gt; No</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Single,Divorced}, Taxable Income&gt;80K) ==&gt; Yes</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Married}) ==&gt; No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class-based Ordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Refund=Yes) ==&gt; No</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Single,Divorced}, Taxable Income&lt;80K) ==&gt; No</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Married}) ==&gt; No</td>
</tr>
<tr>
<td>(Refund=No, Marital Status={Single,Divorced}, Taxable Income&gt;80K) ==&gt; Yes</td>
</tr>
</tbody>
</table>
Building Classification Rules

• Direct Method:
  • Extract rules directly from data
  • e.g.: RIPPER, CN2, Holte’s 1R

• Indirect Method:
  • Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  • e.g: C4.5rules
Direct Method: Sequential Covering

1. Start from an empty rule
2. Grow a rule using the Learn-One-Rule function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) 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
Aspects of Sequential Covering

• Rule Growing

• Instance Elimination

• Rule Evaluation

• Stopping Criterion

• Rule Pruning
Rule Growing

- Two common strategies

(a) General-to-specific

(b) Specific-to-general
Rule Growing (Examples)

- **CN2 Algorithm:**
  - Start from an empty conjunct: {}
  - Add conjuncts that minimizes the entropy measure: \{A\}, \{A,B\}, ... 
  - Determine the rule consequent by taking majority class of instances covered by the rule

- **RIPPER Algorithm:**
  - Start from an empty rule: \{\} => class
  - Add conjuncts that maximizes FOIL’s information gain measure:
    - R0: \{\} => class (initial rule)
    - R1: \{A\} => class (rule after adding conjunct)
    - Gain(R0, R1) = t \{ \log (p1/(p1+n1)) – \log (p0/(p0 + n0)) \}
    - where t: number of positive instances covered by both R0 and R1
    - p0: number of positive instances covered by R0
    - n0: number of negative instances covered by R0
    - p1: number of positive instances covered by R1
    - n1: number of negative instances covered by R1
Instance Elimination

- Why do we need to eliminate instances?
  - Otherwise, the next rule is identical to previous rule

- Why do we remove positive instances?
  - Ensure that the next rule is different

- Why do we remove negative instances?
  - Prevent underestimating accuracy of rule
  - Compare rules R2 and R3 in the diagram
Rule Evaluation

• Metrics:
  - 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
\( k \) : Number of classes
\( p \) : Prior probability
Stopping Criterion and Rule Pruning

• **Stopping criterion**
  – Compute the gain
  – If gain is not significant, discard the new rule

• **Rule Pruning**
  – Similar to post-pruning of decision trees
  – Reduced Error Pruning:
    • Remove one of the conjuncts in the rule
    • Compare error rate on validation set before and after pruning
    • If error improves, prune the conjunct
Summary of Direct Method

- Grow a single rule
- Remove Instances from rule
- Prune the rule (if necessary)
- Add rule to Current Rule Set
- Repeat
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
Direct Method: RIPPER

• 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
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 → y,
  – consider an alternative rule r’: A’ → 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 r’s has lower pessimistic error rate
  – Repeat until we can no longer improve generalization error
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 = $L(\text{error}) + g \cdot L(\text{model})$
    • $g$ is a parameter that takes into account the presence of redundant attributes in a rule set (default value = 0.5)
### Example

<table>
<thead>
<tr>
<th>Name</th>
<th>Give Birth</th>
<th>Lay Eggs</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Have Legs</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>python</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>salmon</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>fishes</td>
</tr>
<tr>
<td>whale</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>frog</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>amphibians</td>
</tr>
<tr>
<td>komodo</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>reptiles</td>
</tr>
<tr>
<td>bat</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>birds</td>
</tr>
<tr>
<td>cat</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>leopard shark</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>fishes</td>
</tr>
<tr>
<td>turtle</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>reptiles</td>
</tr>
<tr>
<td>penguin</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>birds</td>
</tr>
<tr>
<td>porcupine</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>eel</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>fishes</td>
</tr>
<tr>
<td>salamander</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>amphibians</td>
</tr>
<tr>
<td>gila monster</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>reptiles</td>
</tr>
<tr>
<td>platypus</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>owl</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>birds</td>
</tr>
<tr>
<td>dolphin</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>eagle</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>birds</td>
</tr>
</tbody>
</table>
C4.5 versus C4.5rules versus RIPPER

C4.5rules:

(Give Birth=No, Can Fly=Yes) \rightarrow Birds

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

(Give Birth=Yes) \rightarrow Mammals

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

RIPPER:

(Live in Water=Yes) \rightarrow Fishes

(Have Legs=No) \rightarrow Reptiles

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

\rightarrow Reptiles

(Can Fly=Yes, Give Birth=No) \rightarrow Birds

() \rightarrow Mammals
## C4.5 versus C4.5rules versus RIPPER

### C4.5 and C4.5rules:

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>Amphibians</th>
<th>Fishes</th>
<th>Reptiles</th>
<th>Birds</th>
<th>Mammals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amphibians</td>
<td></td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Fishes</td>
<td></td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Reptiles</td>
<td></td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Birds</td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Mammals</td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

### RIPPER:

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>Amphibians</th>
<th>Fishes</th>
<th>Reptiles</th>
<th>Birds</th>
<th>Mammals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amphibians</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Fishes</td>
<td></td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Reptiles</td>
<td></td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Birds</td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Mammals</td>
<td></td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>
Advantages of Rule-Based Classifiers

• As highly expressive as decision trees
• Easy to interpret
• Easy to generate
• Can classify new instances rapidly
• Performance comparable to decision trees