Classification: Alternative Techniques

Dr. Hui Xiong
Rutgers University

Rule-based Classifier
**Rule-Based Classifier**

- Classify records by using a collection of “if…then…” rules

- Rule: \((\text{Condition}) \rightarrow y\)
  - where
    - \(\text{Condition}\) is a conjunctions of attributes
    - \(y\) is the class label
  - \(\text{LHS}:\) rule antecedent or condition
  - \(\text{RHS}:\) 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>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Has Legs</th>
<th>Hibernates</th>
<th>Class Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>python</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>Reptiles</td>
</tr>
<tr>
<td>salmon</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>Fishes</td>
</tr>
<tr>
<td>whale</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>Mammals</td>
</tr>
<tr>
<td>frog</td>
<td>cold-blooded</td>
<td>none</td>
<td>no</td>
<td>semi</td>
<td>yes</td>
<td>yes</td>
<td>Reptiles</td>
</tr>
<tr>
<td>komodo dragon</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>Amphibians</td>
</tr>
<tr>
<td>bat</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>warm-blooded</td>
<td>feathers</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>Birds</td>
</tr>
<tr>
<td>cat</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>guppy</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>Fishes</td>
</tr>
<tr>
<td>alligator</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>no</td>
<td>Reptiles</td>
</tr>
<tr>
<td>penguin</td>
<td>warm-blooded</td>
<td>feathers</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>Birds</td>
</tr>
<tr>
<td>porcupine</td>
<td>warm-blooded</td>
<td>quills</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>eel</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>Fishes</td>
</tr>
<tr>
<td>salamander</td>
<td>cold-blooded</td>
<td>none</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>Amphibians</td>
</tr>
</tbody>
</table>

\(r_1: (\text{Gives Birth} = \text{no}) \land (\text{Aerial Creature} = \text{yes}) \rightarrow \text{Birds}\)

\(r_2: (\text{Gives Birth} = \text{no}) \land (\text{Aquatic Creature} = \text{yes}) \rightarrow \text{Fishes}\)

\(r_3: (\text{Gives Birth} = \text{yes}) \land (\text{Body Temperature} = \text{warm-blooded}) \rightarrow \text{Mammals}\)

\(r_4: (\text{Gives Birth} = \text{no}) \land (\text{Aquatic Creature} = \text{no}) \rightarrow \text{Reptiles}\)

\(r_5: (\text{Aquatic Creature} = \text{semi}) \rightarrow \text{Amphibians}\)
Application of Rule-Based Classifier

- A rule \( r \) covers an instance \( x \) if the attributes of the instance satisfy the condition of the rule

\[
\begin{align*}
\text{r}_1: \quad & \text{(Gives Birth = no)} \land \text{(Aerial Creature = yes)} \rightarrow \text{Birds} \\
\text{r}_2: \quad & \text{(Gives Birth = no)} \land \text{(Aquatic Creature = yes)} \rightarrow \text{Fishes} \\
\text{r}_3: \quad & \text{(Gives Birth = yes)} \land \text{(Body Temperature = warm-blooded)} \rightarrow \text{Mammals} \\
\text{r}_4: \quad & \text{(Gives Birth = no)} \land \text{(Aerial Creature = no)} \rightarrow \text{Reptiles} \\
\text{r}_5: \quad & \text{(Aquatic Creature = semi)} \rightarrow \text{Amphibians}
\end{align*}
\]

<table>
<thead>
<tr>
<th>Name</th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernate</th>
</tr>
</thead>
<tbody>
<tr>
<td>hawk</td>
<td>warm-blooded</td>
<td>feather</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>grizzly bear</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

The rule \( \text{r}_1 \) covers a hawk \( \Rightarrow \) Bird

The rule \( \text{r}_3 \) 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 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>

(\text{Status=Single}) \rightarrow \text{No}

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

- $r_1$: (Gives Birth = no) ∧ (Aerial Creature = yes) --- Birds
- $r_2$: (Gives Birth = no) ∧ (Aquatic Creature = yes) --- Fishes
- $r_3$: (Gives Birth = yes) ∧ (Body Temperature = warm-blooded) --- Mammals
- $r_4$: (Gives Birth = no) ∧ (Aerial Creature = no) --- Reptiles
- $r_5$: (Aquatic Creature = semi) --- Amphibians

<table>
<thead>
<tr>
<th>Name</th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernates</th>
</tr>
</thead>
<tbody>
<tr>
<td>lemur</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>turtle</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>dogfish shark</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

A lemur triggers rule $r_3$, so it is classified as a mammal.
A turtle triggers both $r_4$ and $r_5$.
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: (Refund=No) \land (Status=Married) \rightarrow No

Simplified Rule: (Status=Married) \rightarrow No

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</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>
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

<table>
<thead>
<tr>
<th>Name</th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernate</th>
</tr>
</thead>
<tbody>
<tr>
<td>turtle</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
| r₁: (Gives Birth = no) ∧ (Aquatic Creature = yes) —— Birds
| r₂: (Gives Birth = no) ∧ (Aquatic Creature = yes) —— Fishes
| r₃: (Gives Birth = yes) ∧ (Body Temperature = warm-blooded) —— Mammals
| r₄: (Gives Birth = no) ∧ (Aerial Creature = no) —— Reptiles
| r₅: (Aquatic Creature = semi) —— Amphibians

Introduction to Data Mining 08/26/2006 11
Rule Ordering Schemes

- Rule-based ordering
  - Individual rules are ranked based on their quality/priority

- Class-based ordering
  - Rules that belong to the same class appear together

<table>
<thead>
<tr>
<th>Rule-Based Ordering</th>
<th>Class-Based Ordering</th>
</tr>
</thead>
</table>
| (Skin Cover=feathers, Aerial Creature=yes) \(
  \rightarrow\) Birds |
| (Body temperature=warm-blooded, \(\neg\) Gives Birth=yes) \(
  \rightarrow\) Mammals |
| (Body temperature=warm-blooded, \(\neg\) Gives Birth=no) \(
  \rightarrow\) Birds |
| (Aquatic Creature=semi) \(
  \rightarrow\) Amphibians |
| (Skin Cover=cales, Aerial Creature=no) \(
  \rightarrow\) Reptiles |
| (Skin Cover=cales, Aerial Creature=yes) \(
  \rightarrow\) Fishes |
| (Skin Cover=none) \(
  \rightarrow\) Amphibians |

Building Classification Rules

- Direct Method:
  - Extract rules directly from data
  - e.g.: RIPPER, CN2, 1R, and AQ

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

Algorithm 1.1 Sequential covering algorithm.
1: Let $E$ be the training records and $A$ be the set of attribute-value pairs, $\{(a_j, v_j)\}$.  
2: Let $Y_o$ be an ordered set of classes $\{y_1, y_2, \ldots, y_k\}$.  
3: Let $R = \{\}$ be the initial rule list.  
4: for each class $y \in Y_o - \{y_k\}$ do  
5: \hspace{1em} while stopping condition is not met do  
6: \hspace{2em} $r \leftarrow$ Learn-One-Rule $(E, A, y)$.  
7: \hspace{2em} Remove training records from $E$ that are covered by $r$.  
8: \hspace{1em} Add $r$ to the bottom of the rule list: $R \leftarrow R \cup r$.  
9: \hspace{1em} end while  
10: end for  
11: Insert the default rule, $\{\} \rightarrow y_k$, to the bottom of the rule list $R$.

Example of Sequential Covering

(i) Original Data

(ii) Step 1
Example of Sequential Covering...

Aspects of Sequential Covering

- Rule Growing
  - Rule evaluation

- Instance Elimination

- Stopping Criterion

- Rule Pruning
Rule Growing

- Two common strategies

(a) General-to-specific
(b) Specific-to-general

Rule Evaluation

- Evaluation metric determines which conjunct should be added during rule growing

  - 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 of class c covered by rule

k : Number of classes

p : Prior probability
Rule Growing (Examples)

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

- **RIPPER Algorithm:**
  - Start from an empty rule: {} => class.
  - Add conjuncts that maximize FOIL's information gain measure:
    - R0: {} => class (initial rule)
    - R1: {A} => class (rule after adding conjunct)
    - Gain(R0, R1) = \[ \log \left( \frac{p_1}{p_1 + n_1} \right) / \log \left( \frac{p_0}{p_0 + n_0} \right) \]
    - 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.
Stopping Criterion and Rule Pruning

- Examples of stopping criterion:
  - If rule does not improve significantly after adding conjunct
  - If rule starts covering examples from another class

- Rule Pruning
  - Similar to post-pruning of decision trees
  - Example: using validation set (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

- Initial rule set is empty

- Repeat
  - Grow a single rule
  - Remove Instances covered by the rule
  - Prune the rule (if necessary)
  - Add rule to the current rule set
**Direct Method: RIPPER**

- For 2-class problem, choose one of the classes as positive class, and the other as negative class
  - Learn the rules for positive class
  - Use negative class as default

- 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**

- Rule growing:
  - Start from an empty rule: \( \{ \} \rightarrow + \)
  - 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
    - Grow a rule to cover 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

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 for every path from root to leaf nodes.
- For each rule, $r: A \rightarrow y$,
  - Consider 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 $r$'s has lower pessimistic error rate.
  - Repeat until pessimistic error rate can no longer be improved.

Indirect Method: C4.5rules

- Use class-based ordering.
  - Rules that predict the same class are grouped together into the same subset.
  - Compute total description length for each class.
  - Classes are ordered in increasing order of their total description length.
Example

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

Characteristics 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
Instance-Based Classifiers

Set of Stored Cases

<table>
<thead>
<tr>
<th>Atr1</th>
<th>-------</th>
<th>AtrN</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
</tbody>
</table>

• Store the training records
• Use training records to predict the class label of unseen cases

Unseen Case

<table>
<thead>
<tr>
<th>Atr1</th>
<th>-------</th>
<th>AtrN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Instance Based Classifiers

- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor
    - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it’s probably a duck
Nearest-Neighbor Classifiers

- Requires three things
  - The set of stored records
  - Distance metric to compute distance between records
  - The value of $k$, the number of nearest neighbors to retrieve

To classify an unknown record:
- Compute distance to other training records
- Identify $k$ nearest neighbors
- Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor

K-nearest neighbors of a record $x$ are data points that have the $k$ smallest distance to $x$
Nearest Neighbor Classification

- Compute distance between two points:
  - Example: Euclidean distance
  \[ d(p, q) = \sqrt{\sum_i (p_i - q_i)^2} \]
- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, \( w = 1/d^2 \)
Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M
Nearest Neighbor Classification…

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

\[
\begin{align*}
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}
& \quad \text{vs} \quad \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\end{align*}
\]

\[ d = 1.4142 \quad \text{vs} \quad d = 1.4142 \]

- Solution: Normalize the vectors to unit length

Nearest neighbor Classification…

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive
Example: PEBLS

- PEBLS: Parallel Exemplar-Based Learning System (Cost & Salzberg)
  - Works with both continuous and nominal features
    - For nominal features, distance between two nominal values is computed using modified value difference metric (MVDM)
  - Each record is assigned a weight factor
  - Number of nearest neighbor, \( k = 1 \)

### Distance between nominal attribute values:

\[
d(V_1, V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|
\]
**Example: PEBLS**

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>Y</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
</tbody>
</table>

Distance between record X and record Y:

\[
\Delta(X, Y) = w_X w_Y \sum_{i=1}^{d} d(X_i, Y_i)^2
\]

where:

\[w_X = \frac{\text{Number of times } X \text{ is used for prediction}}{\text{Number of times } X \text{ predicts correctly}}\]

\[w_X \approx 1 \text{ if } X \text{ makes accurate prediction most of the time}\]

\[w_X > 1 \text{ if } X \text{ is not reliable for making predictions}\]

**Classification: Alternative Techniques**

Bayesian Classifiers
Ensemble Methods

- Construct a set of classifiers from the training data

- Predict class label of test records by combining the predictions made by multiple classifiers
**Why Ensemble Methods work?**

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume errors made by classifiers are uncorrelated
  - Probability that the ensemble classifier makes a wrong prediction:
    \[
    P(X \geq 13) = \sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06
    \]

**General Approach**

1. **Step 1:** Create Multiple Data Sets
2. **Step 2:** Build Multiple Classifiers
3. **Step 3:** Combine Classifiers
Types of Ensemble Methods

- Bayesian ensemble
  - Example: Mixture of Gaussian
- Manipulate data distribution
  - Example: Resampling method
- Manipulate input features
  - Example: Feature subset selection
- Manipulate class labels
  - Example: error-correcting output coding
- Introduce randomness into learning algorithm
  - Example: Random forests

### 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 \((1 - 1/n)^n\) of being selected
Bagging Algorithm

**Algorithm 5.6 Bagging Algorithm**

1. Let $k$ be the number of bootstrap samples.
2. for $i = 1$ to $k$ do
3. Create a bootstrap sample of size $n$, $D_i$.
4. Train a base classifier $C_i$ on the bootstrap sample $D_i$.
5. end for
6. $C^*(x) = \arg \max_y \sum_i \delta(C_i(x) = y)$. \{\(\delta(\cdot) = 1\) if its argument is true, and 0 otherwise.\}

Bagging Example

- Consider 1-dimensional data set:

<table>
<thead>
<tr>
<th>$x$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y$</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Classifier is a decision stump
  - Decision rule: $x \leq k$ versus $x > k$
  - Split point $k$ is chosen based on entropy

\[
\begin{array}{c}
\text{True} \\
\mathbf{y_{left}} \\
\text{False} \\
\mathbf{y_{right}}
\end{array}
\]
### Bagging Example

#### Bagging Round 1:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.2</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.9</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.35 \Rightarrow y = 1 \]
\[ x > 0.35 \Rightarrow y = -1 \]

#### Bagging Round 2:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.5</th>
<th>0.9</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.7 \Rightarrow y = 1 \]
\[ x > 0.7 \Rightarrow y = -1 \]

#### Bagging Round 3:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.4</th>
<th>0.5</th>
<th>0.7</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.35 \Rightarrow y = 1 \]
\[ x > 0.35 \Rightarrow y = -1 \]

#### Bagging Round 4:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.1</th>
<th>0.2</th>
<th>0.4</th>
<th>0.4</th>
<th>0.5</th>
<th>0.5</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.3 \Rightarrow y = 1 \]
\[ x > 0.3 \Rightarrow y = -1 \]

#### Bagging Round 5:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.1</th>
<th>0.2</th>
<th>0.5</th>
<th>0.6</th>
<th>0.6</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.35 \Rightarrow y = 1 \]
\[ x > 0.35 \Rightarrow y = -1 \]

---

### Bagging Example

#### Bagging Round 6:

<table>
<thead>
<tr>
<th>x</th>
<th>0.2</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.75 \Rightarrow y = -1 \]
\[ x > 0.75 \Rightarrow y = 1 \]

#### Bagging Round 7:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.4</th>
<th>0.4</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>0.9</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.75 \Rightarrow y = -1 \]
\[ x > 0.75 \Rightarrow y = 1 \]

#### Bagging Round 8:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.2</th>
<th>0.5</th>
<th>0.5</th>
<th>0.5</th>
<th>0.7</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.75 \Rightarrow y = -1 \]
\[ x > 0.75 \Rightarrow y = 1 \]

#### Bagging Round 9:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.3</th>
<th>0.4</th>
<th>0.4</th>
<th>0.6</th>
<th>0.7</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.75 \Rightarrow y = -1 \]
\[ x > 0.75 \Rightarrow y = 1 \]

#### Bagging Round 10:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.1</th>
<th>0.1</th>
<th>0.1</th>
<th>0.3</th>
<th>0.3</th>
<th>0.8</th>
<th>0.8</th>
<th>0.9</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x \leq 0.05 \Rightarrow y = 1 \]
\[ x > 0.05 \Rightarrow y = -1 \]
## Bagging Example

- **Summary of Training sets:**

<table>
<thead>
<tr>
<th>Round</th>
<th>Split Point</th>
<th>Left Class</th>
<th>Right Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.35</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>0.7</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0.35</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>0.3</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>5</td>
<td>0.35</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>6</td>
<td>0.75</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0.75</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0.75</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>0.75</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>0.05</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### Bagging Example

- Assume test set is the same as the original data
- Use majority vote to determine class of ensemble classifier

<table>
<thead>
<tr>
<th>Round</th>
<th>(x=0.1)</th>
<th>(x=0.2)</th>
<th>(x=0.3)</th>
<th>(x=0.4)</th>
<th>(x=0.5)</th>
<th>(x=0.6)</th>
<th>(x=0.7)</th>
<th>(x=0.8)</th>
<th>(x=0.9)</th>
<th>(x=1.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
</tr>
<tr>
<td>Sign</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
</tr>
<tr>
<td>Sign</td>
</tr>
</tbody>
</table>
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 each 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>6</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>1</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
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:

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

AdaBoost Algorithm

- 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:

$$C^*(x) = \arg \max_y \sum_{j=1}^{T} \alpha_j \delta(C_j(x) = y)$$
AdaBoost Algorithm

Algorithm 5.7 AdaBoost Algorithm
1. \( w = \{w_j = 1/n \mid j = 1, 2, \cdots, n\} \). [Initialize the weights for all instances]
2. Let \( k \) be the number of boosting rounds.
3. for \( i = 1 \) to \( k \) do
4. Create training set \( D_i \) by sampling (with replacement) from \( D \) according to \( w \).
5. Train a base classifier \( C_i \) on \( D_i \).
6. Apply \( C_i \) to all instances in the original training set, \( D \).
7. \( e_i = \frac{1}{n}\left(\sum_j w_j \delta(C_i(x_j) \neq y_j)\right) \) [Calculate the weighted error]
8. if \( e_i > 0.5 \) then
9. \( w = \{w_j = 1/n \mid j = 1, 2, \cdots, n\} \). [Reset the weights for all \( n \) instances]
11. end if
12. \( \alpha_i = \frac{1}{2} \ln \frac{1 - e_i}{e_i} \)
13. Update the weight of each instance according to equation (5.88).
14. end for
15. \( C^*(x) = \arg\max_y \sum_{j=1}^n \alpha_j \delta(C_j(x) = y) \).

AdaBoost Example

- Consider 1-dimensional data set:

  Original Data:

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Classifier is a decision stump
  - Decision rule: \( x \leq k \) versus \( x > k \)
  - Split point \( k \) is chosen based on entropy

  ![Decision Stump Diagram]
AdaBoost Example

- Training sets for the first 3 boosting rounds:

**Boosting Round 1:**

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.7</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Boosting Round 2:**

<table>
<thead>
<tr>
<th>x</th>
<th>0.1</th>
<th>0.1</th>
<th>0.2</th>
<th>0.2</th>
<th>0.2</th>
<th>0.3</th>
<th>0.3</th>
<th>0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Boosting Round 3:**

<table>
<thead>
<tr>
<th>x</th>
<th>0.2</th>
<th>0.2</th>
<th>0.4</th>
<th>0.4</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

- Summary:

<table>
<thead>
<tr>
<th>Round</th>
<th>Split Point</th>
<th>Left Class</th>
<th>Right Class</th>
<th>alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.75</td>
<td>-1</td>
<td>1</td>
<td>1.738</td>
</tr>
<tr>
<td>2</td>
<td>0.05</td>
<td>1</td>
<td>1</td>
<td>2.7784</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>1</td>
<td>-1</td>
<td>4.1195</td>
</tr>
</tbody>
</table>

AdaBoost Example

- Weights

<table>
<thead>
<tr>
<th>Round</th>
<th>x=0.1</th>
<th>x=0.2</th>
<th>x=0.3</th>
<th>x=0.4</th>
<th>x=0.5</th>
<th>x=0.6</th>
<th>x=0.7</th>
<th>x=0.8</th>
<th>x=0.9</th>
<th>x=1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>0.311</td>
<td>0.311</td>
<td>0.311</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>0.029</td>
<td>0.029</td>
<td>0.029</td>
<td>0.228</td>
<td>0.228</td>
<td>0.228</td>
<td>0.228</td>
<td>0.009</td>
<td>0.009</td>
<td>0.009</td>
</tr>
</tbody>
</table>

- Classification

<table>
<thead>
<tr>
<th>Round</th>
<th>x=0.1</th>
<th>x=0.2</th>
<th>x=0.3</th>
<th>x=0.4</th>
<th>x=0.5</th>
<th>x=0.6</th>
<th>x=0.7</th>
<th>x=0.8</th>
<th>x=0.9</th>
<th>x=1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

| Sum   | 5.16  | 5.16  | 5.16  | -3.08 | -3.08 | -3.08 | -3.08 | 0.397 | 0.397 | 0.397 |
|       | 0.01  | 0.01  | 0.01  | 0.228 | 0.228 | 0.228 | 0.228 | 0.009 | 0.009 | 0.009 |

<table>
<thead>
<tr>
<th>Predicted Class</th>
<th>Sum</th>
<th>Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5.16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-3.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-3.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-3.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.397</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.397</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.397</td>
<td></td>
</tr>
</tbody>
</table>

Introduction to Data Mining 08/26/2006 67
Classification: Alternative Techniques

Imbalanced Class Problem

Class Imbalance Problem

- Lots of classification problems where the classes are skewed (more records from one class than another)
  - Credit card fraud
  - Intrusion detection
  - Defective products in manufacturing assembly line
Challenges

- Evaluation measures such as accuracy is not well-suited for imbalanced class
- Detecting the rare class is like finding needle in a haystack

Confusion Matrix

- Confusion Matrix:

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class=Yes</td>
</tr>
<tr>
<td>Class=Yes</td>
<td>a</td>
</tr>
<tr>
<td>Class=No</td>
<td>c</td>
</tr>
</tbody>
</table>

a: TP (true positive)  
b: FN (false negative)  
c: FP (false positive)  
d: TN (true negative)
### Accuracy

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class=Yes</td>
</tr>
<tr>
<td>Class=Yes</td>
<td>a (TP)</td>
</tr>
<tr>
<td>Class=No</td>
<td>c (FP)</td>
</tr>
</tbody>
</table>

- Most widely-used metric:

\[
\text{Accuracy} = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}
\]

### Problem with Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If a model predicts everything to be class 0, accuracy is 9990/10000 = 99.9%
  - This is misleading because the model does not detect any class 1 example
  - Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)
Alternative Measures

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>Class=Yes</th>
<th>Class=No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class=Yes</td>
<td>a (TP)</td>
<td>b (FN)</td>
<td></td>
</tr>
<tr>
<td>Class=No</td>
<td>c (FP)</td>
<td>d (TN)</td>
<td></td>
</tr>
</tbody>
</table>

Precision \(p\) = \(\frac{a}{a + c}\)

Recall \(r\) = \(\frac{a}{a + b}\)

\[ F - \text{measure (F)} = \frac{2rp}{r + p} = \frac{2\frac{a}{a+b}}{a+b+c} \]

ROC (Receiver Operating Characteristic)

- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - TPR = \(\frac{TP}{TP+FN}\), FPR = \(\frac{FP}{TN+FP}\)
  - Performance of a model represented as a point in an ROC curve
  - Changing the threshold parameter of classifier changes the location of the point
**ROC Curve**

(TPR,FPR):
- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal

- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

**ROC (Receiver Operating Characteristic)**

- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record

- Many classifiers produce only discrete outputs (i.e., predicted class)
  - How to get continuous-valued outputs?
    - Decision trees, rule-based classifiers, neural networks, Bayesian classifiers, k-nearest neighbors, SVM
Example: Decision Trees

Decision Tree

Continuous-valued outputs

ROC Curve Example

<table>
<thead>
<tr>
<th>$\alpha = 0.3$</th>
<th>Predicted Class</th>
<th>$\alpha = 0.7$</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class $\circ$</td>
<td>Class $\circ$</td>
<td>Class $\circ$</td>
</tr>
<tr>
<td>Actual Class</td>
<td>645</td>
<td>200</td>
<td>183</td>
</tr>
<tr>
<td></td>
<td>298</td>
<td>3488</td>
<td>78</td>
</tr>
</tbody>
</table>

Introduction to Data Mining 08/26/2006 79

Introduction to Data Mining 08/26/2006 80
Using ROC for Model Comparison

- No model consistently outperform the other
  - $M_1$ is better for small FPR
  - $M_2$ is better for large FPR

- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

How to Construct an ROC curve

| Instance | score($+|A$) | True Class |
|----------|--------------|------------|
| 1        | 0.95         | +          |
| 2        | 0.93         | +          |
| 3        | 0.87         | -          |
| 4        | 0.85         | -          |
| 5        | 0.85         | -          |
| 6        | 0.85         | +          |
| 7        | 0.76         | -          |
| 8        | 0.53         | +          |
| 9        | 0.43         | -          |
| 10       | 0.25         | +          |

- Use classifier that produces continuous-valued output for each test instance score($+|A$)
- Sort the instances according to score($+|A$) in decreasing order
- Apply threshold at each unique value of score($+|A$)
- Count the number of TP, FP, TN, FN at each threshold
  - $TPR = TP/(TP+FN)$
  - $FPR = FP/(FP + TN)$
### How to construct an ROC curve

<table>
<thead>
<tr>
<th>Threshold</th>
<th>0.25</th>
<th>0.43</th>
<th>0.53</th>
<th>0.76</th>
<th>0.85</th>
<th>0.85</th>
<th>0.87</th>
<th>0.93</th>
<th>0.95</th>
<th>1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>FP</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TN</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>FN</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TPR</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>FPR</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Handling Class Imbalanced Problem

- **Class-based ordering** (e.g. RIPPER)
  - Rules for rare class have higher priority

- **Cost-sensitive classification**
  - Misclassifying rare class as majority class is more expensive than misclassifying majority as rare class

- **Sampling-based approaches**
### Cost Matrix

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>( f(i, j) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class=Yes</td>
<td>Class=Yes</td>
<td>( f(Yes, Yes) )</td>
</tr>
<tr>
<td>Class=Yes</td>
<td>Class=No</td>
<td>( f(Yes, No) )</td>
</tr>
<tr>
<td>Class=No</td>
<td>Class=Yes</td>
<td>( f(No, Yes) )</td>
</tr>
<tr>
<td>Class=No</td>
<td>Class=No</td>
<td>( f(No, No) )</td>
</tr>
</tbody>
</table>

\[ C(i, j): \text{Cost of misclassifying class } i \text{ example as class } j \]

\[ \text{Cost} = \sum C(i, j) \times f(i, j) \]

### Computing Cost of Classification

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>( C(i, j) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>+</td>
<td>150</td>
</tr>
<tr>
<td>+</td>
<td>-</td>
<td>40</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>60</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>250</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>( C(i, j) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>+</td>
<td>250</td>
</tr>
<tr>
<td>+</td>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>5</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>200</td>
</tr>
</tbody>
</table>

**Model \( M_1 \)**

- Accuracy = 80%
- Cost = 3910

**Model \( M_2 \)**

- Accuracy = 90%
- Cost = 4255
Cost Sensitive Classification

- Example: Bayesian classifier
  - Given a test record $x$:
    - Compute $p(i|x)$ for each class $i$
    - Decision rule: classify node as class $k$ if
      \[
      k = \arg \max_i p(i \mid x)
      \]
    - For 2-class, classify $x$ as + if $p(+) > p(-)$
      - This decision rule implicitly assumes that
        $C(+) = C(-) = 0$ and $C(+) = C(-)$

Cost Sensitive Classification

- General decision rule:
  - Classify test record $x$ as class $k$ if
    \[
    k = \arg \min_j \sum_i p(i \mid x) \times C(i, j)
    \]
  - 2-class:
    - Cost($+$) = $p(+) C(+) + p(-) C(-)$
    - Cost($-$) = $p(+) C(-) + p(-) C(+)$
    - Decision rule: classify $x$ as + if Cost($+$) < Cost($-$)
      - if $C(+) = C(-) = 0$:
        \[
        p(+) > \frac{C(-)}{C(-) + C(+)}
        \]
### Sampling-based Approaches

- Modify the distribution of training data so that rare class is well-represented in training set
  - Undersample the majority class
  - Oversample the rare class

- Advantages and disadvantages