

# Data Mining

## Classification: Alternative Techniques

---

---

Lecture Notes for Chapter 5

Introduction to Data Mining  
by  
Tan, Steinbach, Kumar

## Rule-Based Classifier

---

---

- Classify records by using a collection of “if...then...” rules
- Rule: (*Condition*)  $\rightarrow y$ 
  - where
    - ◆ *Condition* is a conjunctions of attributes
    - ◆  $y$  is the class label
  - *LHS*: rule antecedent or condition
  - *RHS*: rule consequent
  - Examples of classification rules:
    - ◆ (Blood Type=Warm)  $\wedge$  (Lay Eggs=Yes)  $\rightarrow$  Birds
    - ◆ (Taxable Income < 50K)  $\wedge$  (Refund=Yes)  $\rightarrow$  Evade=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

## 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 both the antecedent and consequent of a rule

| 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

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

| 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

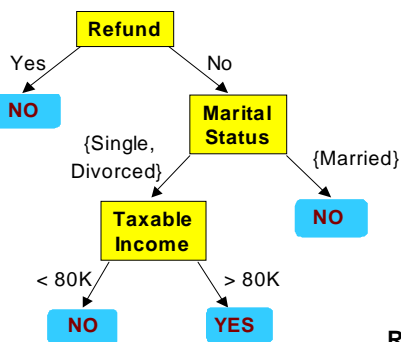
A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules

## Characteristics of Rule-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

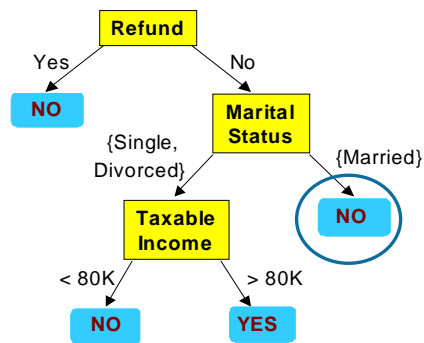


### Classification Rules

(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



| Tid | Refund | Marital Status | Taxable Income | Cheat |
|-----|--------|----------------|----------------|-------|
| 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   |

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

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: (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     | ?     |

## 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

## 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

## Instance-Based Classifiers

Set of Stored Cases

| Atr1 | ..... | AtrN | Class |
|------|-------|------|-------|
|      |       |      | A     |
|      |       |      | B     |
|      |       |      | B     |
|      |       |      | C     |
|      |       |      | A     |
|      |       |      | C     |
|      |       |      | B     |

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

Unseen Case

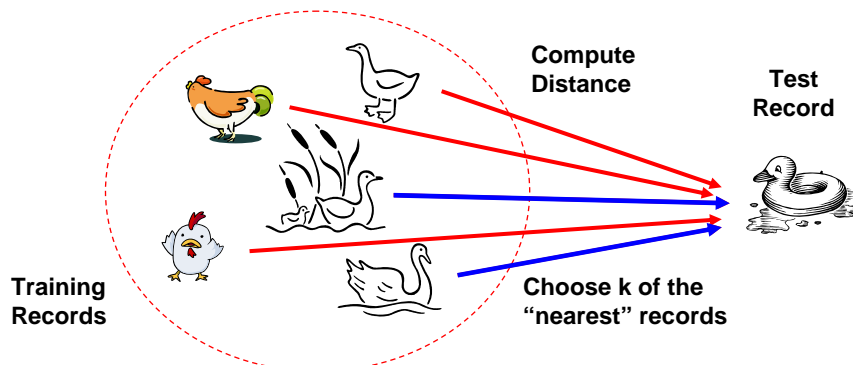
| Atr1 | ..... | AtrN |
|------|-------|------|
|      |       |      |

## 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

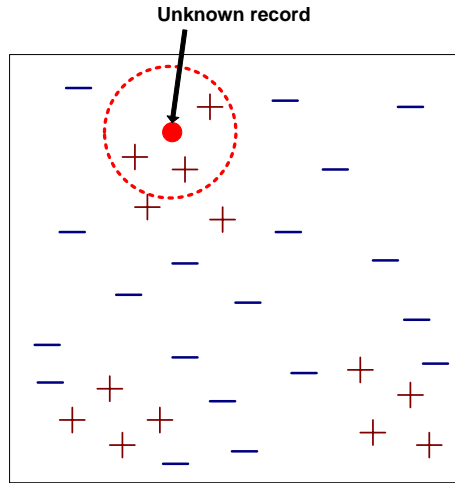
## Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



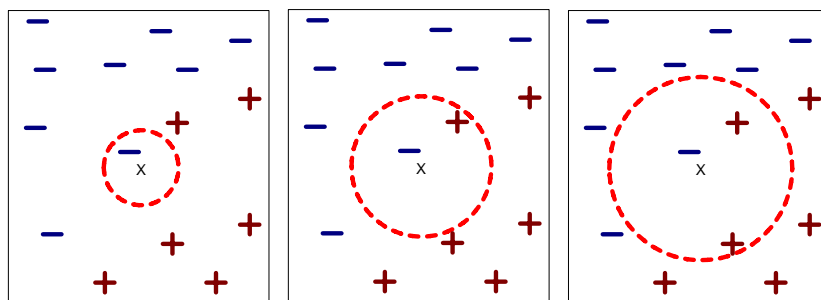


# Nearest-Neighbor Classifiers



- 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



(a) 1-nearest neighbor

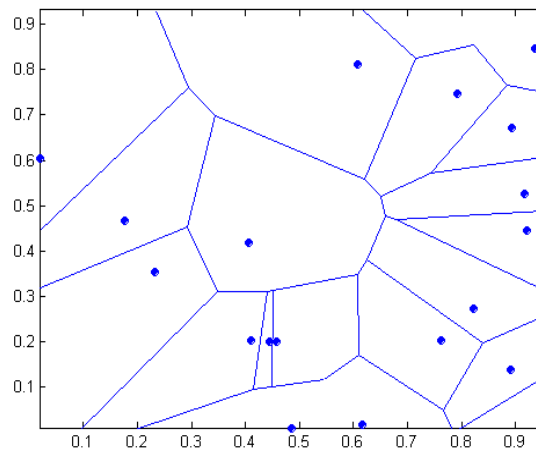
(b) 2-nearest neighbor

(c) 3-nearest neighbor

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

# 1 nearest-neighbor

## Voronoi Diagram



## Nearest Neighbor Classification

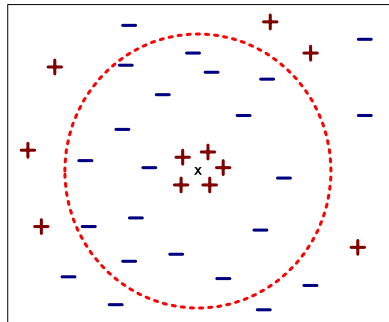
- Compute distance between two points:
  - Euclidean distance

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- 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



## Nearest Neighbor Classification...

- 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

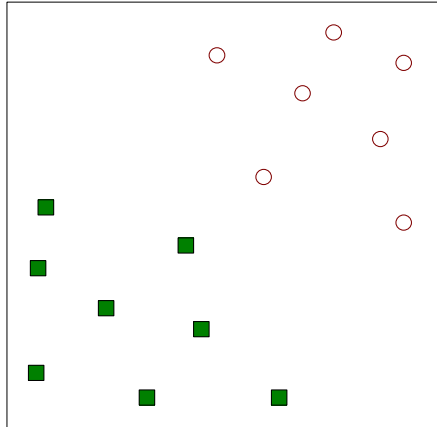
|   |    |   |
|---|----|---|
| $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$ | VS | $\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$ |
| $\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ |    | $\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$ |
| $d = 1.4142$  |    | $d = 1.4142$  |

- ◆ Solution: Normalize the vectors to unit length

## Nearest neighbor Classification...

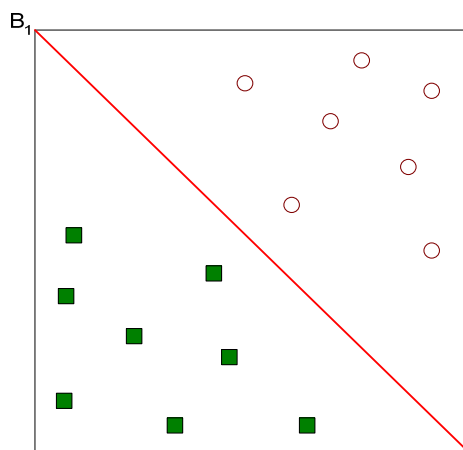
- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive

## Support Vector Machines



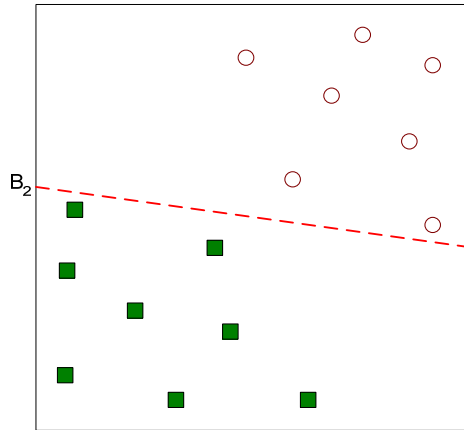
- Find a linear hyperplane (decision boundary) that will separate the data

## Support Vector Machines



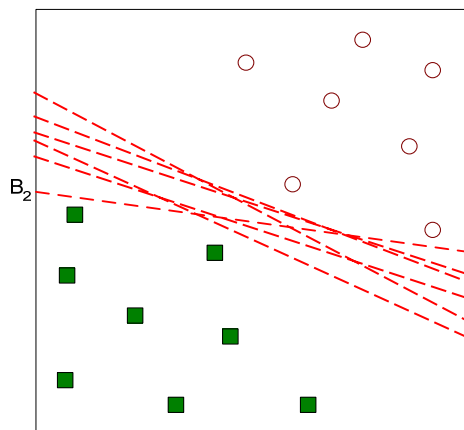
- One Possible Solution

# Support Vector Machines



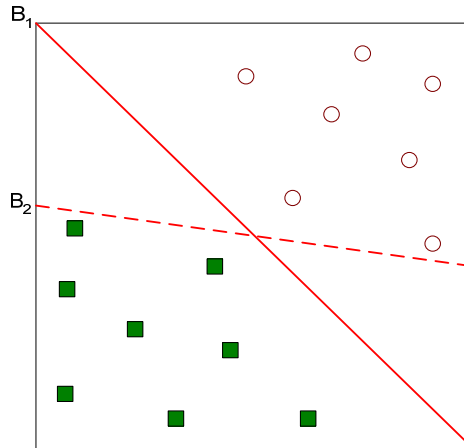
- Another possible solution

# Support Vector Machines



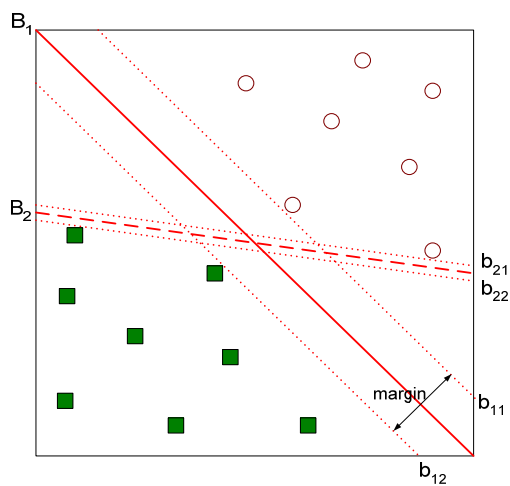
- Other possible solutions

# Support Vector Machines



- Which one is better? B1 or B2?
- How do you define better?

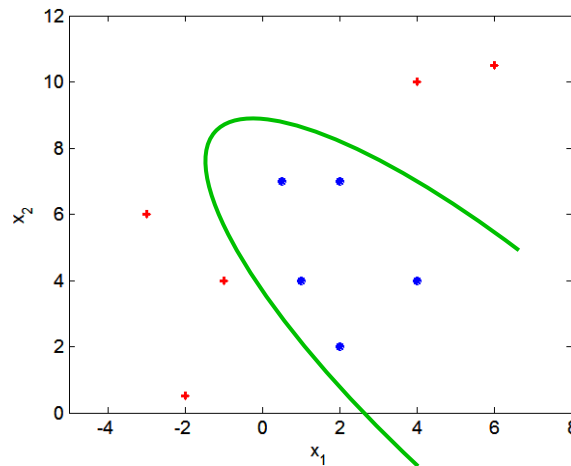
# Support Vector Machines



- Find hyperplane **maximizes** the margin  $\Rightarrow$  B1 is better than B2

## Nonlinear Support Vector Machines

- What if decision boundary is not linear?



## Nonlinear Support Vector Machines

- Transform data into higher dimensional space

