

Data Mining

Classification: Alternative Techniques

Lecture Notes for Chapter 5 (PART 1)



정보보호 및 지능형 IoT 연구실
Information Security & Intelligent IoT

Agenda

Rule Based Classifier

Bayesian Classifier

Artificial Neural Network

Support Vector Machine

Ensemble, Bagging, Boosting

PART 1

PART 2

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:
 - ◆ $(\text{Blood Type}=\text{Warm}) \wedge (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}$
 - ◆ $(\text{Taxable Income} < 50\text{K}) \wedge (\text{Refund}=\text{Yes}) \rightarrow \text{Evade}=\text{No}$

Rule-based Classifier (Example)

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

For a rule $r: A \rightarrow y$

- Coverage of a rule:
 - Fraction of records that satisfy the antecedent of a rule
 - $\text{Coverage}(r) = |A| / |D|$
- Accuracy of a rule:
 - Fraction of records that satisfy both the antecedent and consequent of a rule
 - $\text{Accuracy}(r) = |A \cap y| / |A|$

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) \rightarrow No

Coverage = 40%, Accuracy = 50%

How does 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

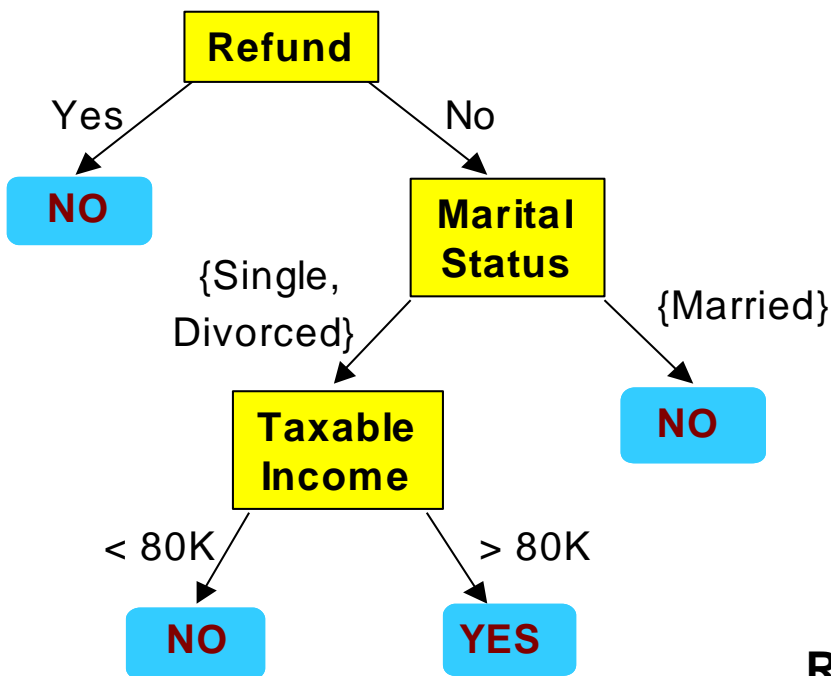
A turtle triggers **both R4 and R5** \rightarrow **weight or order?**

A dogfish shark **triggers none of the rules** \rightarrow **default class**

Characteristics of Rule-Based Classifier

- 상호 배타적 규칙(Mutually exclusive rules)
 - 각 레코드는 하나의 규칙에만 지배를 받아야 함
 - Classifier contains mutually exclusive rules if the rules are independent of each other
 - Every record is covered by at most one rule
- 포괄적 규칙(Exhaustive rules)
 - 분류기 규칙은 모든 가능한 레코드에 적용될 수 있어야 함
 - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
 - Each record is covered by at least one rule

DT에서 규칙 생성



Classification Rules

(Refund=Yes) ==> No

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

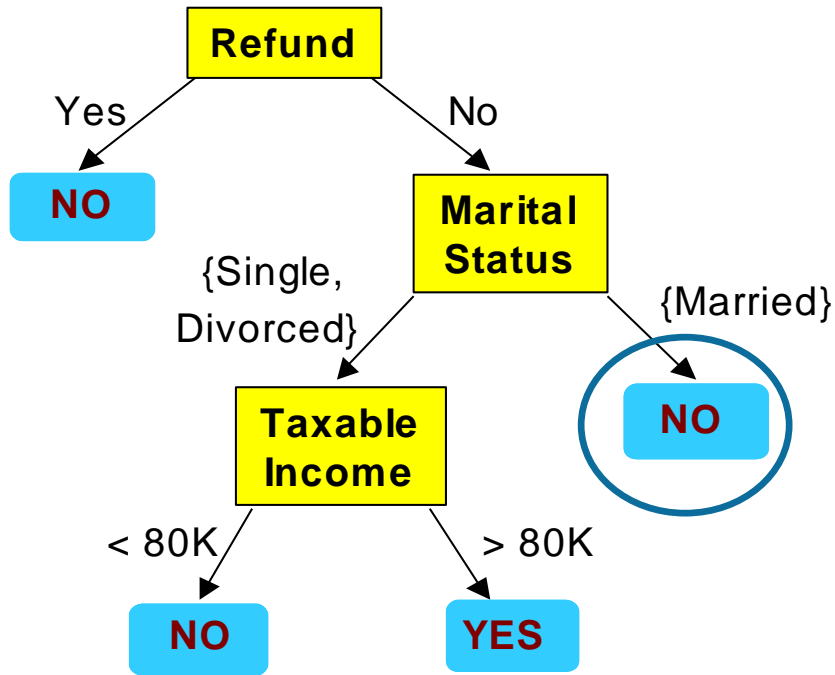
(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

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

Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree

생성된 규칙의 단순화 가능



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: $(\text{Status}=\text{Married}) \rightarrow \text{No}$

규칙 단순화 효과

- 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

순서화된 규칙 집합(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

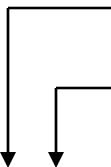
R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

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	?

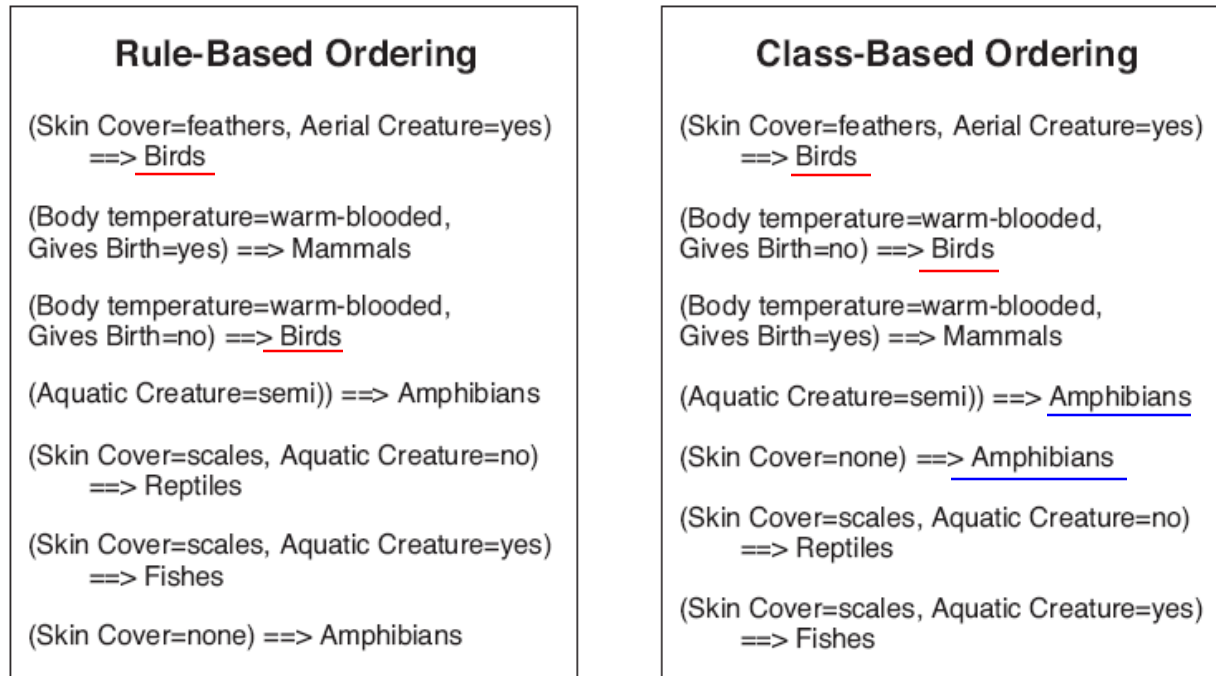
Rule Ordering Schemes

- Rule-based ordering

- Individual rules are ranked based on their quality → 우선순위가 낮은 rule은 해석이 어려움

- Class-based ordering

- Rules that belong to the same class appear together → rule 해석은 용이함. 하지만, 우선 배치된 열등한 rule에 의해 우수한 rule이 간과될 수 있음



같은 클래스에 속하는 rule을 연속적으로 배치함. 클래스 단위 정렬

Figure 5.1. Comparison between rule-based and class-based ordering schemes.

분류 규칙 생성 방법(Building Classification Rules)

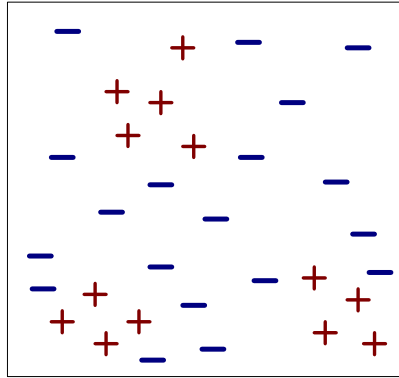
- Rule 기반 분류기를 만들기 위해선 데이터 집합의 속성과 class label 사이의 관계를 식별하는 rule을 추출할 필요 있음. 두가지 방법 존재
- Direct Method:
 - ◆ Extract rules directly from data
 - ◆ e.g.: 순차적 커버링(sequential covering), RIPPER, CN2, Holte's 1R
- Indirect Method:
 - ◆ Extract rules from other classification models (e.g. decision trees, neural networks, etc).
 - ◆ e.g: C4.5rules

Direct Method: 순차적 커버링(Sequential Covering)

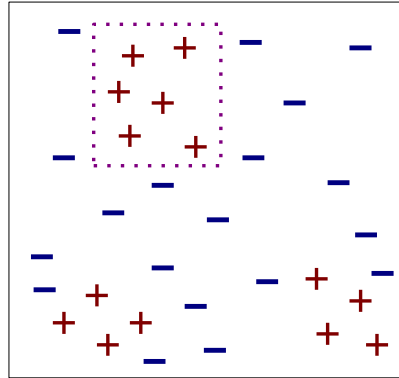
- Rule은 특정 평가 기준에 대해 greedy기법으로 생성됨
- 둘 이상의 클래스를 포함하는 데이터 집합에서 한번에 한 클래스씩 rule을 추출함
 - 예) 먼저 조류를 분류하는 rule 생성 → 다음 포유류 → 양서류 → 파충류 → 어류.. 식으로 분류하는 rule을 생성함
 - 클래스 생성 순서는 클래스 크기, 비용 등을 고려하여 결정

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

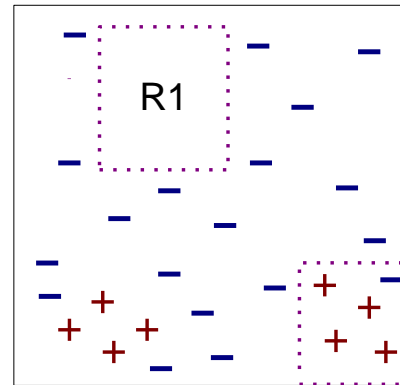
Example of Sequential Covering



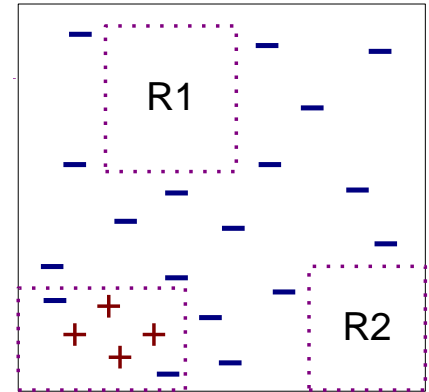
(i) Original Data



(ii) Step 1



(iii) Step 2



(iv) Step 3

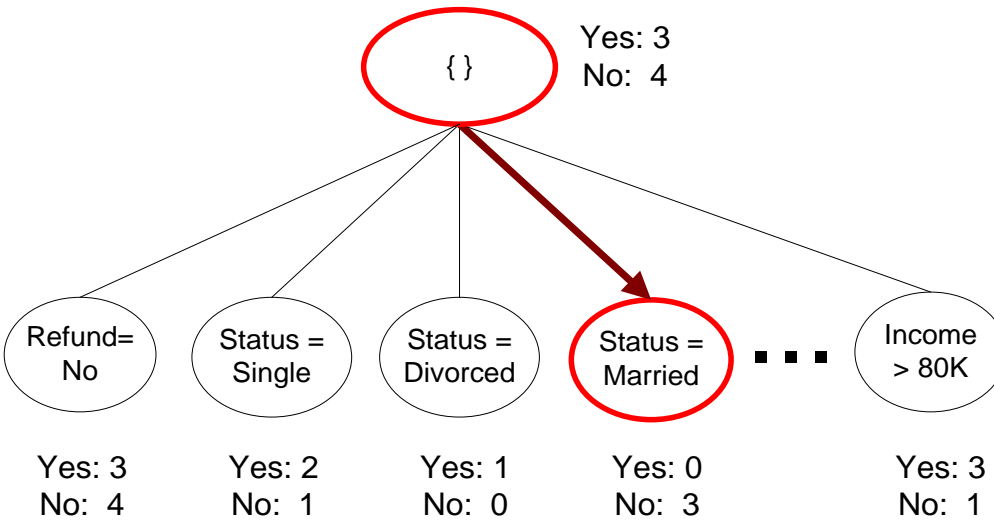
- 먼저 클래스(“+”)를 위한 최선의 rule을 추출함 (step 1) → R1
 - 최선의 rule을 추출하는데 하나의 class만 고려하면서, 정지조건 만족할 때까지 rule을 찾음 → Learn-One-Rule 기법
 - (“+”) 클래스에 대한 rule 이 모두 추출되기전까지는 다른 클래스 (“-”)를 고려하지 않음
- 두번째 긍정적인 부분을 찾음(step 2) → R2
- 세번째로 클래스 (“+”) 관점에서 긍정적인 부분을 찾음 (그림 iv)

Aspects of Sequential Covering

1. Rule Growing
2. Instance Elimination
3. Rule Evaluation
4. Stopping Criterion & Rule Pruning

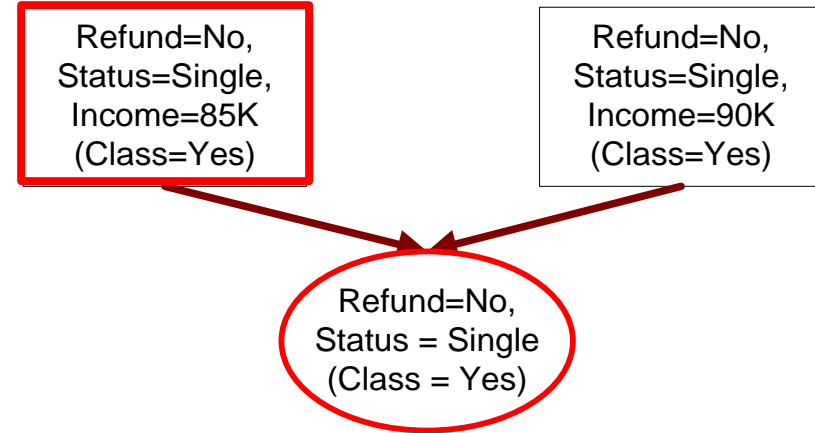
1. Rule Growing

- Two common strategies



(a) General-to-specific

- 초기 규칙 $r: \{ \} \rightarrow y$ 이 생성됨(매우 질이 낮음)
- Rule의 질을 향상시키기 위해 조건 결합항들이 선택됨
- 가장 좋은 조건을 선택함 (Status = Married) 조건 선택시 가장 좋게 분류됨

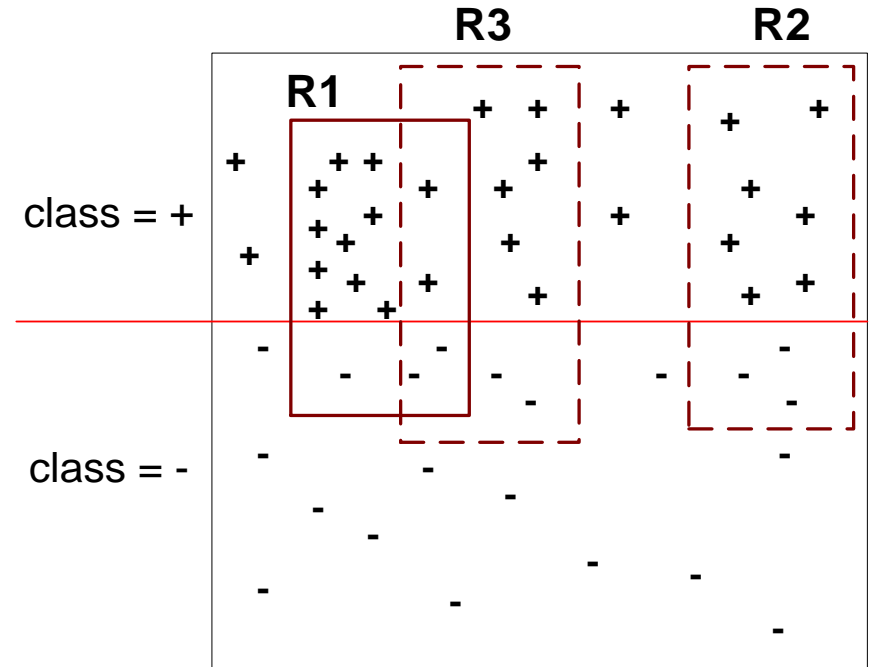


(b) Specific-to-general

- 긍정적인 예 중에서 하나를 rule로 만들기 위해서 선택됨
- 조건항 하나를 제거함으로써 보다 많은 사례를 포함할 수 있도록 일반화함(예에서는 Income 조건항이 제거되어 일반화됨)

2. Instance Elimination

- Why do we need to eliminate instances?
 - Otherwise, the next rule is identical to previous rule
- Why do we remove “+” instances?
 - Ensure that the next rule is different
- Why do we remove “-” instances?
 - Prevent underestimating accuracy of rule



3. Rule Evaluation

- 규칙의 성장과정에서 어떤 결합항을 추가(혹은 제거)할지를 결정할 평가 기준이 필요함

- Metrics:

- Accuracy = $\frac{f_+}{n}$

n : Number of instances covered by rule

- Laplace = $\frac{f_+^n + 1}{n + k}$

f_+ : Number of positive instances covered by rule

k : Number of classes

- M-estimate = $\frac{f_+ + kp_+}{n + k}$

p_+ : Prior probability for positive class

- FOIL's information gain

- = $p_1(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0})$

p_1 : Number of positive instances covered by new rule

n_1 : Number of negative instances covered by new rule

3. Rule Evaluation (예제)

- Consider a training set that contains 60 positive examples and 100 negative examples.

Rule r1 covers 50 positive examples and 5 negative examples

Rule r2 covers 2 positive examples and no negative examples

Accuracy of r1=50/55=90.9%, accuracy of r2=2/2=100%

Laplace measure for r1=(50+1)/(55+2)=89.47%,
r2=(2+1)/(2+2)=75%

Foil's information gain for r1=63.87, r2=2.83 (textbook's numbers are wrong)

$$p_1 \left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0} \right)$$

- r1 정보이득 계산:
 $50 * (\log_2 (50 / (50 + 5)) - \log_2 (60 / (60+100))) = 63.87$

- r2 정보이득 계산:
 $2 * (\log_2 (2 / (2+0)) - \log_2 (60 / (60+100))) = 2.83$
--> r1이 r2보다 더 좋은 규칙

4. 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

RIPPER(repeated incremental pruning to produce error reduction)

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

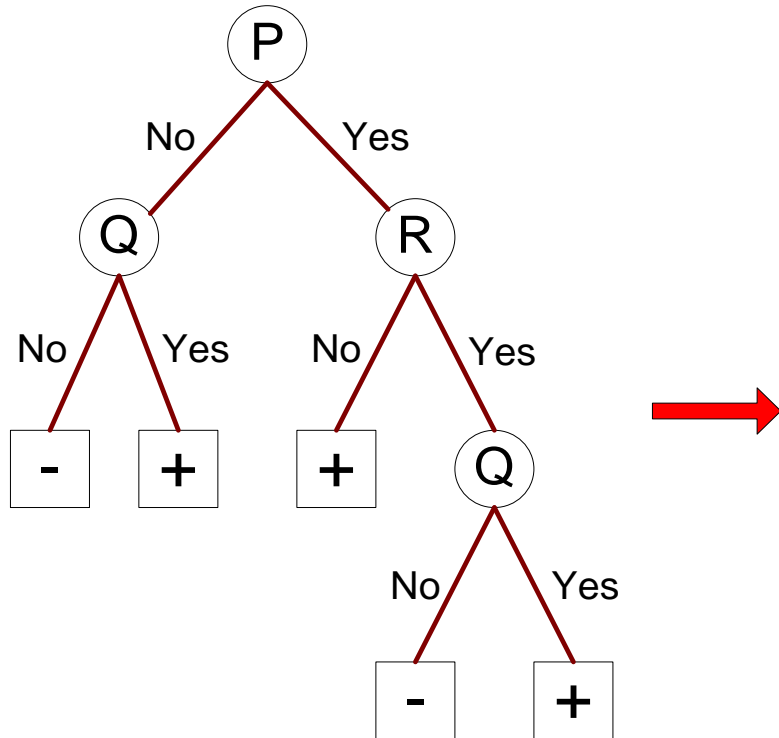
Direct Method: RIPPER

- 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(의사결정 트리 등 사용)



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 r 's has lower pessimistic error rate
 - Repeat until we can no longer improve generalization error

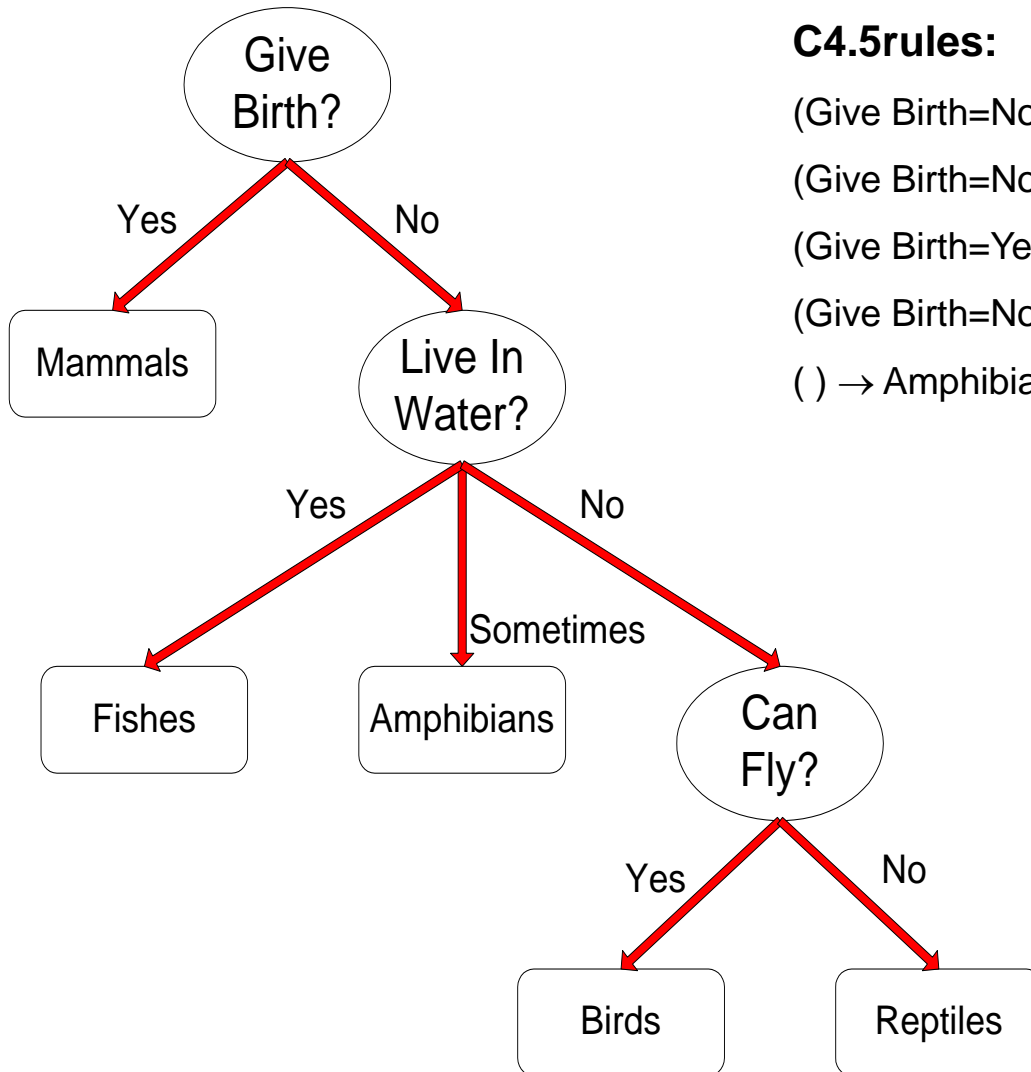
Indirect Method: C4.5rules

- 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 = $L(\text{error}) + g L(\text{model})$
 - ◆ g is a parameter that takes into account the presence of redundant attributes in a rule set (default value = 0.5)
- $L(\text{error})$: 잘못 분류된 instance 인코딩에 필요한 bit 길이
- $L(\text{model})$: 모델을 인코딩하는데 필요한 비트수
- Description Length가 가장 적은 클래스를 선택함

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	0	1
	Reptiles	1	0	3	0	0	0
	Birds	1	0	0	3	0	0
	Mammals	0	0	1	0	0	6

RIPPER:

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

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

- Instance based learning(memory based learning 이라고도함) :
 - explicit generalization을 하지 않고 이전의 데이터(memory에 있는 데이터)를 토대로 새로운 데이터를 분류함
 - 사전 모델을 만들지 않음(소극적 학습기:lazy learner) → Rote Classifier, KNN 등
 - ◆ Cf. DT는 귀납적으로 모델을 만듦(적극적 학습기:eager learner)

Instance Based Classifiers

- Examples:

- Rote-learner

- ◆ 훈련 데이터 전체를 암기한 후, 시험 사례 속성이 훈련 데이터와 정확히 일치할 때만 분류를 수행함

- 단점: 일부 시험 항목들이 어떠한 훈련 예와도 일치하지 않을 경우 → 분류 안됨 (Nearest neighbor(인접 이웃) 찾기로 어느정도 해결함)

- ◆ Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly

- Nearest neighbor

- ◆ 시험 사례의 속성과 상대적으로 유사한 훈련 예를 찾음

- ◆ Uses k “closest” points (nearest neighbors) for performing classification

Instance Based Classifiers

- Rote learner 사례

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

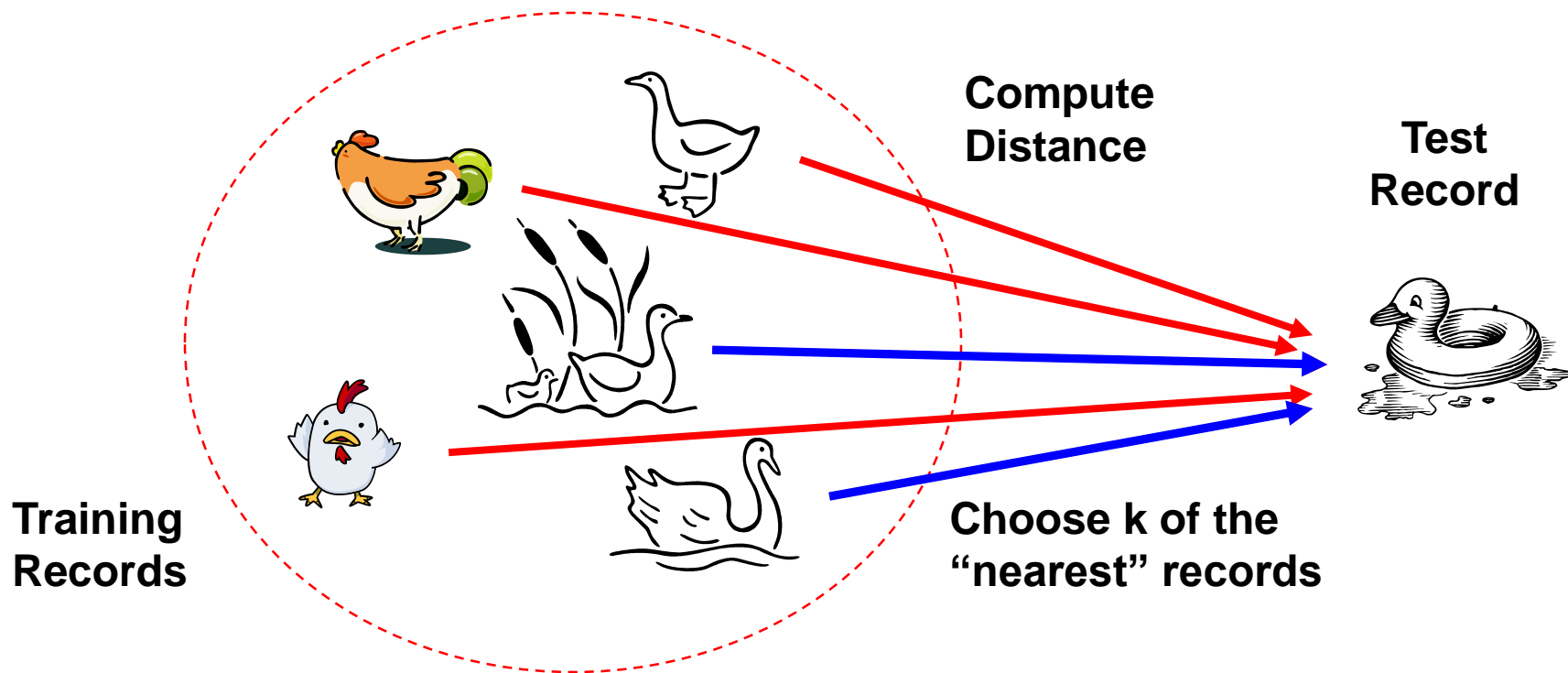
today	cool	sunny	normal	false	yes
-------	------	-------	--------	-------	-----

* 기존 속성 중에서 오늘의 날씨(온도, 시야, 습도, 바람)과 정확히 일치하는 속성을 찾아, class가 결정됨(play golf: yes)

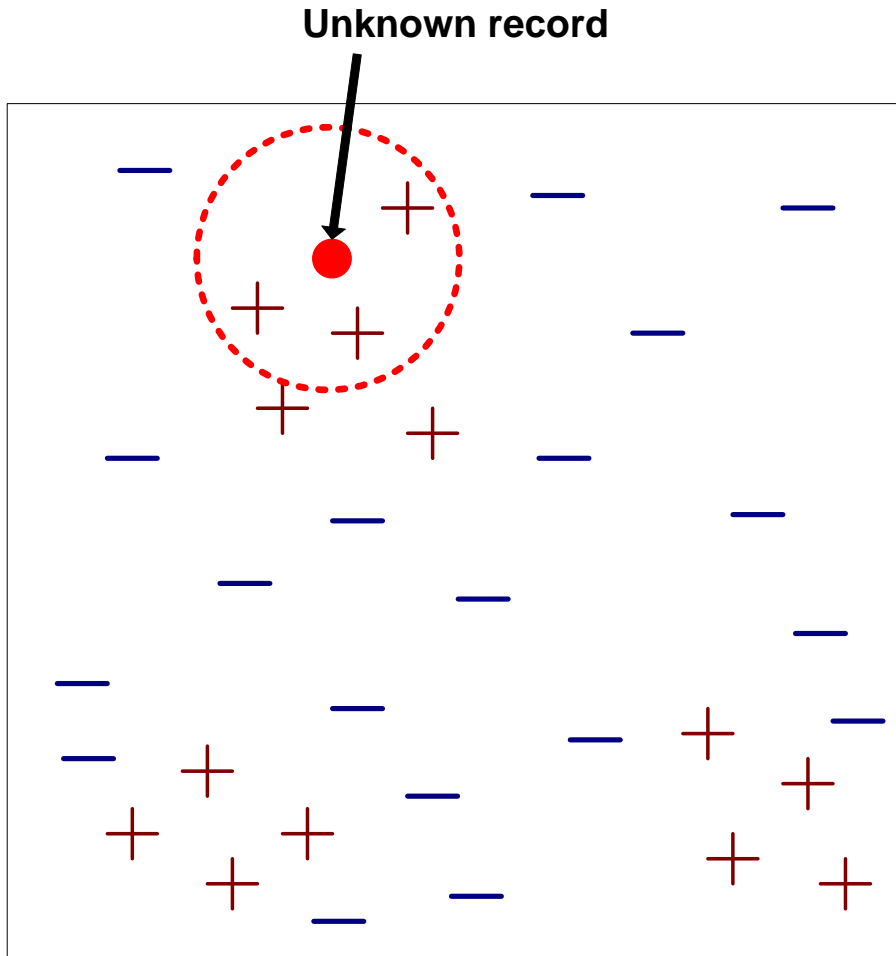
참고: Darmstadt Knowledge Engineering Group

Nearest Neighbor Classifiers

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

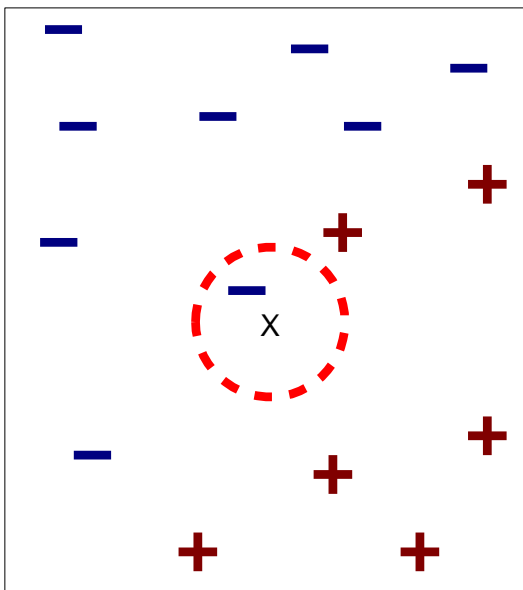


Nearest-Neighbor Classifiers

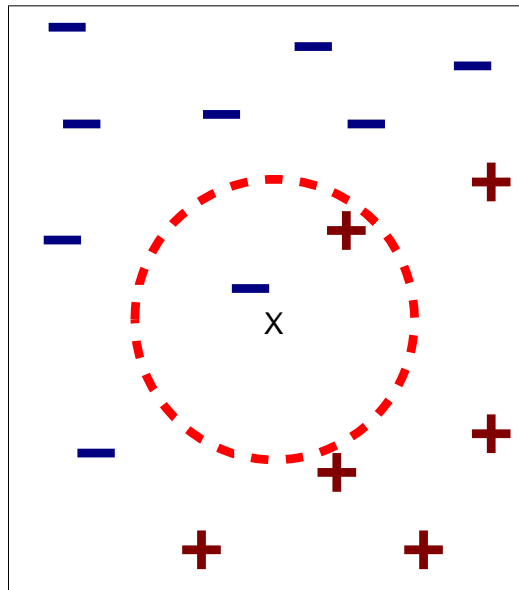


- Requires three things
 - The set of stored records
 - Distance Metric to compute distance between records
 - The value of k , the number of nearest neighbors to retrieve
- To classify an unknown record:
 - Compute **distance** to other training records
 - **Identify** k nearest neighbors
 - **Use class labels of nearest neighbors** to determine the class label of unknown record (e.g., by taking majority vote)

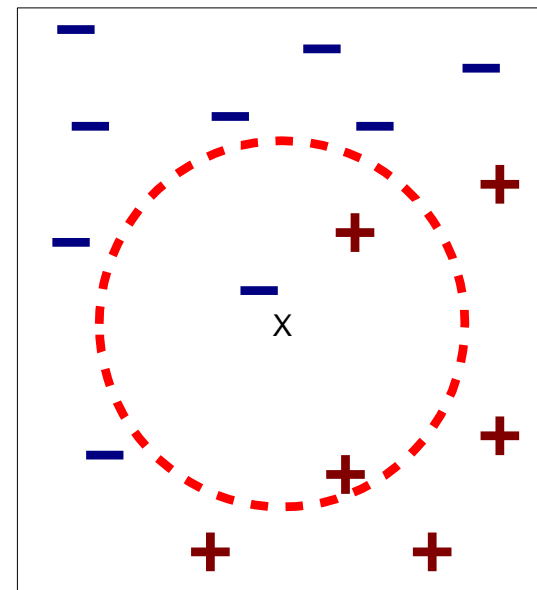
Definition of Nearest Neighbor



(a) 1-nearest neighbor



(b) 2-nearest neighbor

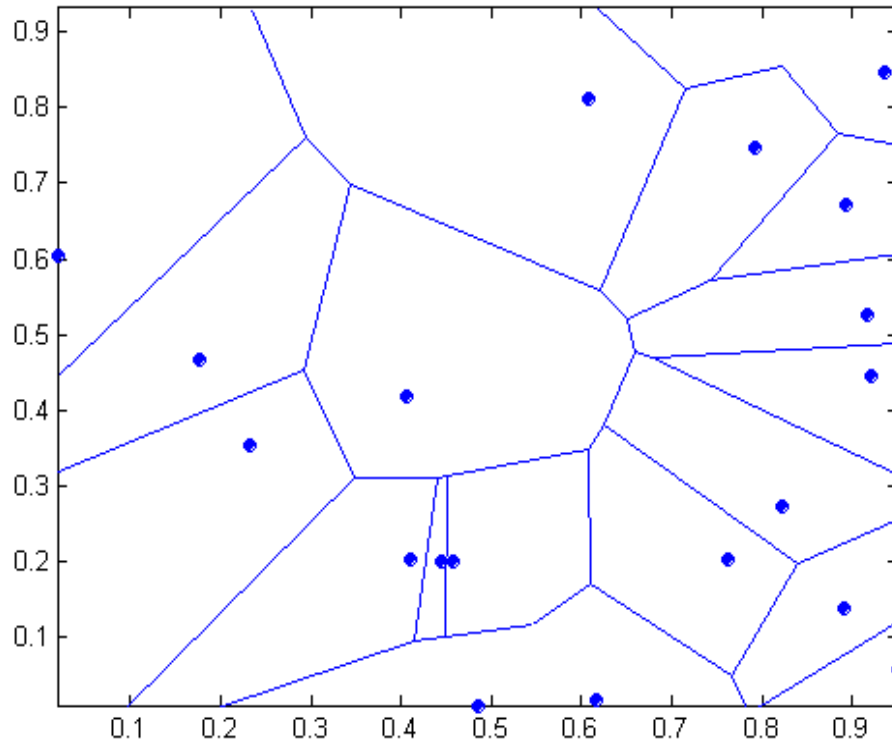


(c) 3-nearest neighbor

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

1 nearest-neighbor

Voronoi Diagram



- 점과 가까운 영역을 표시함
- boundaries of these regions correspond to potential decision boundaries of 1NN classifier

Nearest Neighbor Classification

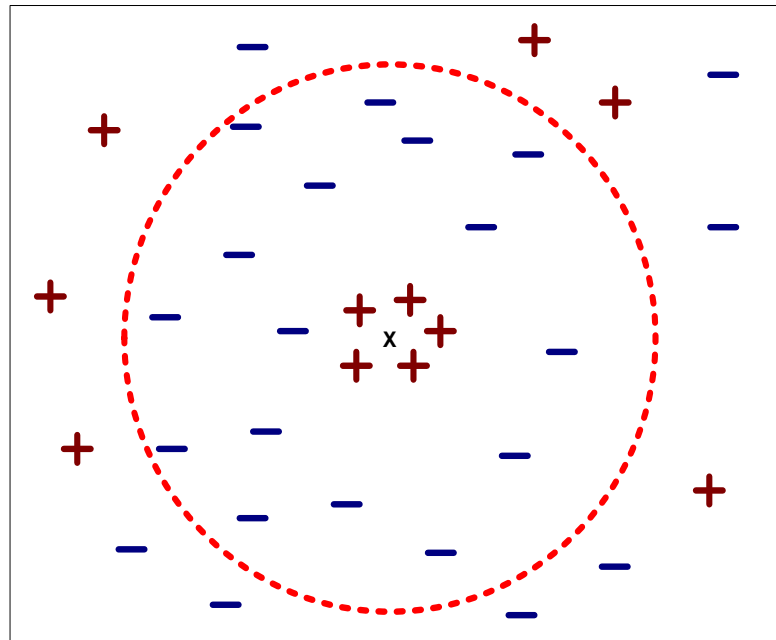
- Compute distance between two points:
 - Euclidean distance

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

- 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

1 1 1 1 1 1 1 1 1 1 0

0 1 1 1 1 1 1 1 1 1 1

$d = 1.4142$

VS

1 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 1

$d = 1.4142$

- ◆ 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, $k = 1$

Example: PEBLS

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

Distance between nominal attribute values:

$d(\text{Single}, \text{Married})$

$$= |2/4 - 0/4| + |2/4 - 4/4| = 1$$

$d(\text{Single}, \text{Divorced})$

$$= |2/4 - 1/2| + |2/4 - 1/2| = 0$$

$d(\text{Married}, \text{Divorced})$

$$= |0/4 - 1/2| + |4/4 - 1/2| = 1$$

$d(\text{Refund}=\text{Yes}, \text{Refund}=\text{No})$

$$= |0/3 - 3/7| + |3/3 - 4/7| = 6/7$$

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

Class	Refund	
	Yes	No
Yes	0	3
No	3	4

$$d(V_1, V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

Example: PEBLS

<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
X	Yes	Single	125K	No
Y	No	Married	100K	No

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

$w_X \cong 1$ if X makes accurate prediction most of the time

$w_X > 1$ if X is not reliable for making predictions

Contents



Bayesian Classifier

Bayes Classifier

- A probabilistic framework for solving classification problems

- Conditional Probability:

$$P(C | A) = \frac{P(A, C)}{P(A)}$$

$$P(A | C) = \frac{P(A, C)}{P(C)}$$

- Bayes theorem:

$$P(C | A) = \frac{P(A | C)P(C)}{P(A)}$$

Example of Bayes Theorem

- Given:
 - A doctor knows that meningitis causes stiff neck 50% of the time
 - Prior probability of any patient having meningitis is 1/50,000
 - Prior probability of any patient having stiff neck is 1/20
- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M | S) = \frac{P(S | M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Bayesian Classifiers

- Consider each attribute and class label as random variables
- Given a record with attributes (A_1, A_2, \dots, A_n)
 - Goal is to predict class C
 - Specifically, we want to find the value of C that maximizes $P(C | A_1, A_2, \dots, A_n)$
- Can we estimate $P(C | A_1, A_2, \dots, A_n)$ directly from data?

Bayesian Classifiers

- Approach:
 - compute the posterior probability $P(C | A_1, A_2, \dots, A_n)$ for all values of C using the Bayes theorem

$$P(C | A_1 A_2 \dots A_n) = \frac{P(A_1 A_2 \dots A_n | C) P(C)}{P(A_1 A_2 \dots A_n)}$$

- Choose value of C that maximizes $P(C | A_1, A_2, \dots, A_n)$
 - Equivalent to choosing value of C that maximizes $P(A_1, A_2, \dots, A_n | C) P(C)$
- How to estimate $P(A_1, A_2, \dots, A_n | C)$?

Naïve Bayes Classifier

- Assume independence among attributes A_i when class is given:
 - $P(A_1, A_2, \dots, A_n | C) = P(A_1 | C_j) P(A_2 | C_j) \dots P(A_n | C_j)$
 - Can estimate $P(A_i | C_j)$ for all A_i and C_j .
 - New point is classified to C_j if $P(C_j) \prod P(A_i | C_j)$ is maximal.

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Class: $P(C) = N_c/N$

- e.g., $P(\text{No}) = 7/10$,
 $P(\text{Yes}) = 3/10$

- For discrete attributes:

$$P(A_i | C_k) = |A_{ik}| / N_{C_k}$$

- where $|A_{ik}|$ is number of instances having attribute A_i and belongs to class C_k
- Examples:

$$P(\text{Status}=\text{Married}|\text{No}) = 4/7$$
$$P(\text{Refund}=\text{Yes}|\text{Yes})=0$$

How to Estimate Probabilities from Data?

- For continuous attributes:
 - **Discretize** the range into bins
 - ◆ one ordinal attribute per bin
 - ◆ violates independence assumption ^k
 - **Two-way split:** $(A < v)$ or $(A > v)$
 - ◆ choose only one of the two splits as new attribute
 - **Probability density estimation:**
 - ◆ Assume attribute follows a normal distribution
 - ◆ Use data to estimate parameters of distribution (e.g., mean and standard deviation)
 - ◆ Once probability distribution is known, can use it to estimate the conditional probability $P(A_i|c)$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Normal distribution:

$$P(A_i | c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(A_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

- One for each (A_i, c_i) pair

- For (Income, Class=No):

- If Class=No

◆ sample mean = 110

◆ sample variance = 2975

$$P(\text{Income} = 120 | \text{No}) = \frac{1}{\sqrt{2\pi(54.54)}} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Example of Naïve Bayes Classifier

$$P(C | A_1 A_2 \dots A_n) = \frac{P(A_1 A_2 \dots A_n | C) P(C)}{P(A_1 A_2 \dots A_n)}$$

Given a Test Record:

$X = (\text{Refund}=\text{No}, \text{Married}, \text{Income}=120\text{K})$

naive Bayes Classifier:

$$P(\text{Refund}=\text{Yes}|\text{No}) = 3/7$$

$$P(\text{Refund}=\text{No}|\text{No}) = 4/7$$

$$P(\text{Refund}=\text{Yes}|\text{Yes}) = 0$$

$$P(\text{Refund}=\text{No}|\text{Yes}) = 1$$

$$P(\text{Marital Status}=\text{Single}|\text{No}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{No})=1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{No}) = 4/7$$

$$P(\text{Marital Status}=\text{Single}|\text{Yes}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{Yes})=1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{Yes}) = 0$$

For taxable income:

If class=No: sample mean=110

sample variance=2975

If class=Yes: sample mean=90

sample variance=25

- $P(X|\text{Class}=\text{No}) = P(\text{Refund}=\text{No}|\text{Class}=\text{No})$
 $\times P(\text{Married}|\text{Class}=\text{No})$
 $\times P(\text{Income}=120\text{K}|\text{Class}=\text{No})$
 $= 4/7 \times 4/7 \times 0.0072 = \underline{0.0024}$
- $P(X|\text{Class}=\text{Yes}) = P(\text{Refund}=\text{No}|\text{Class}=\text{Yes})$
 $\times P(\text{Married}|\text{Class}=\text{Yes})$
 $\times P(\text{Income}=120\text{K}|\text{Class}=\text{Yes})$
 $= 1 \times 0 \times 1.2 \times 10^{-9} = \underline{0}$

Since $\underline{P(X|\text{No})P(\text{No})} > P(X|\text{Yes})P(\text{Yes})$

Therefore $P(\text{No}|X) > P(\text{Yes}|X)$

$\Rightarrow \text{Class} = \text{No}$

Naïve Bayes Classifier

- If one of the conditional probability is zero, then the entire expression becomes zero
- Probability estimation:

$$\text{Original: } P(A_i | C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace: } P(A_i | C) = \frac{N_{ic} + 1}{N_c + c}$$

$$\text{m-estimate: } P(A_i | C) = \frac{N_{ic} + mp}{N_c + m}$$

c: number of classes

p: prior probability

m: parameter

Example of Naïve Bayes Classifier

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

A: attributes

M: mammals

N: non-mammals

- 1. Give birth=yes
- 2. Can fly=no
- 3. Live in water = yes
- 4. Have legs = no

$$P(A|M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

$$P(A|M)P(M) > P(A|N)P(N)$$

=> Mammals

Naïve Bayes (Summary)

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
 - Use other techniques such as Bayesian Belief Networks (BBN)