

TDM – 12 Aprile 2007

Classification: advanced concepts

Lecture Notes from Chapter 4 & 5
of
Introduction to Data Mining
by
Tan, Steinbach, Kumar

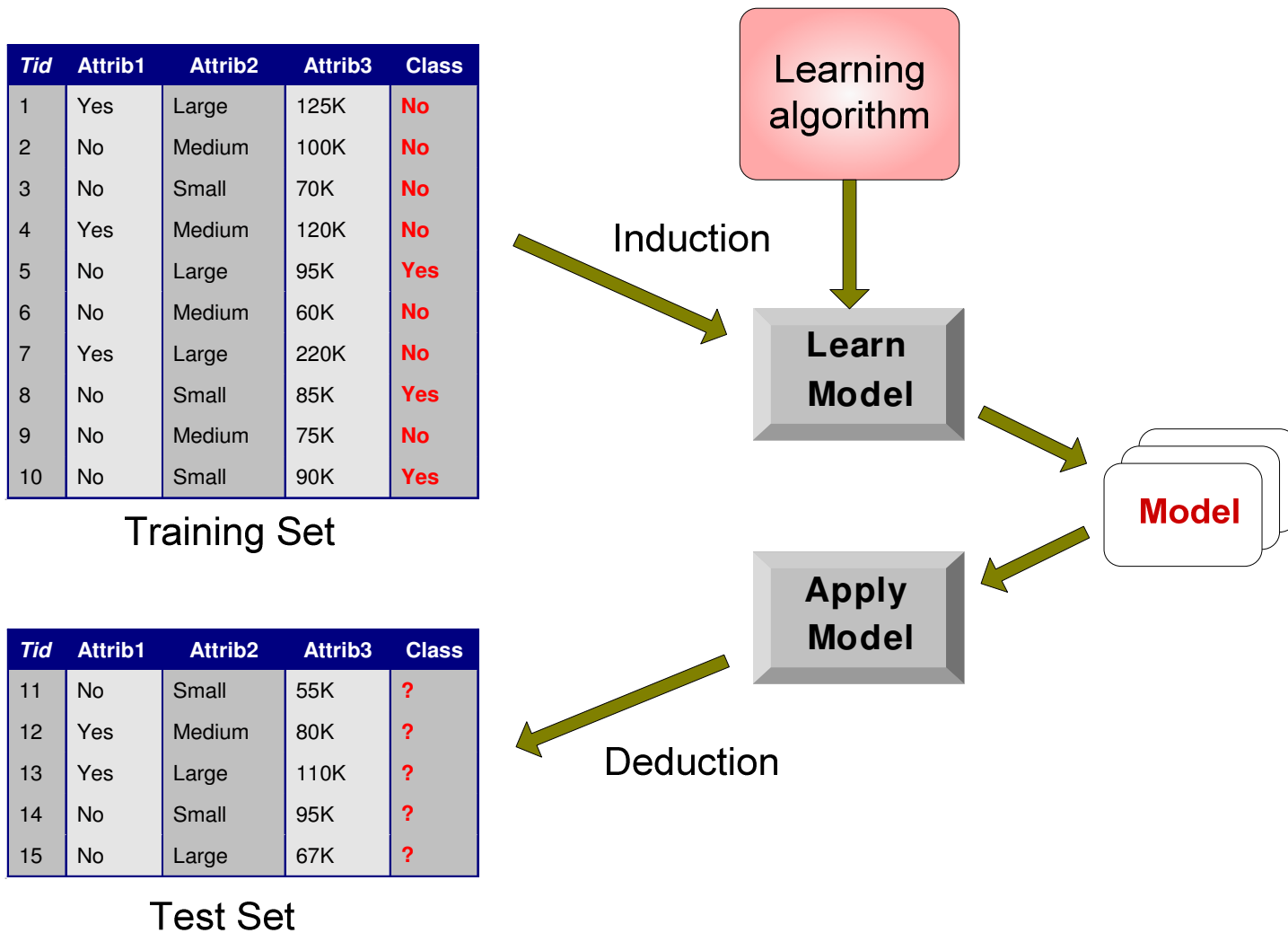


Richiami dall'ultima lezione...

Classification: Definition

- Given a collection of records (*training set*)
 - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
 - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Illustrating Classification Task



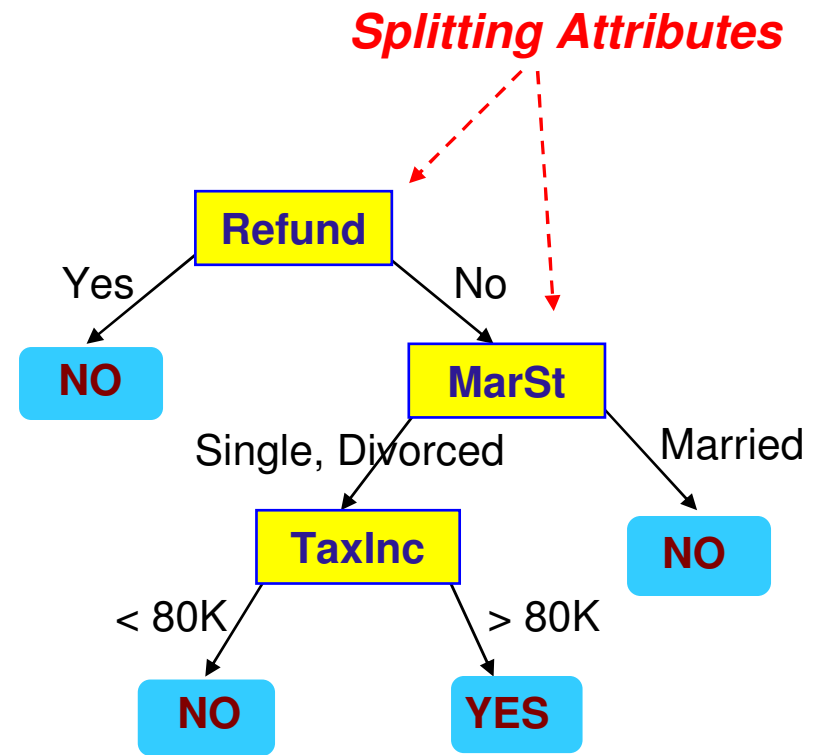
Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines

Example of a Decision Tree

<i>Tid</i>	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

categorical
categorical
continuous
class



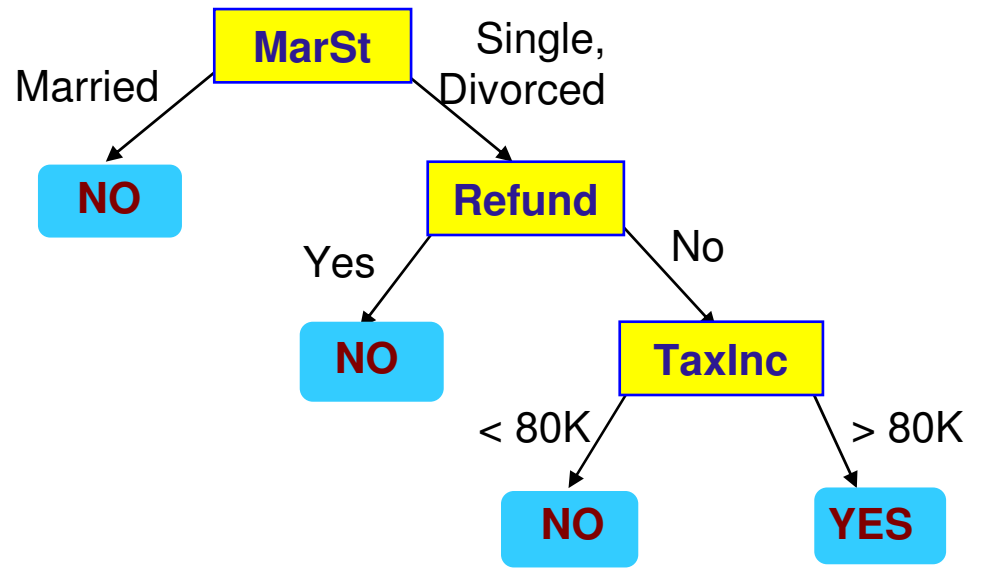
Training Data

Model: Decision Tree

Another Example of Decision Tree

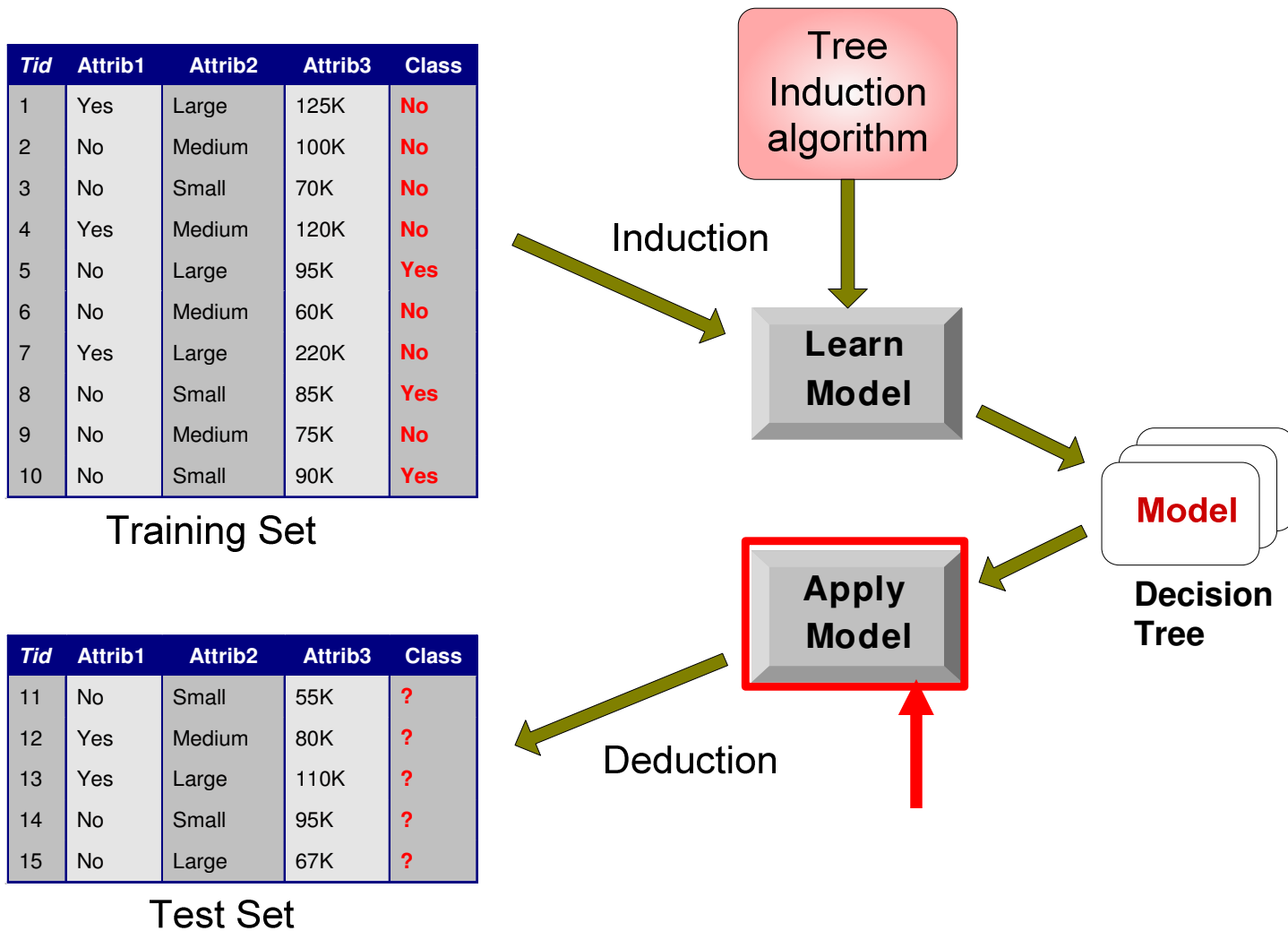
<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

categorical
categorical
continuous
class



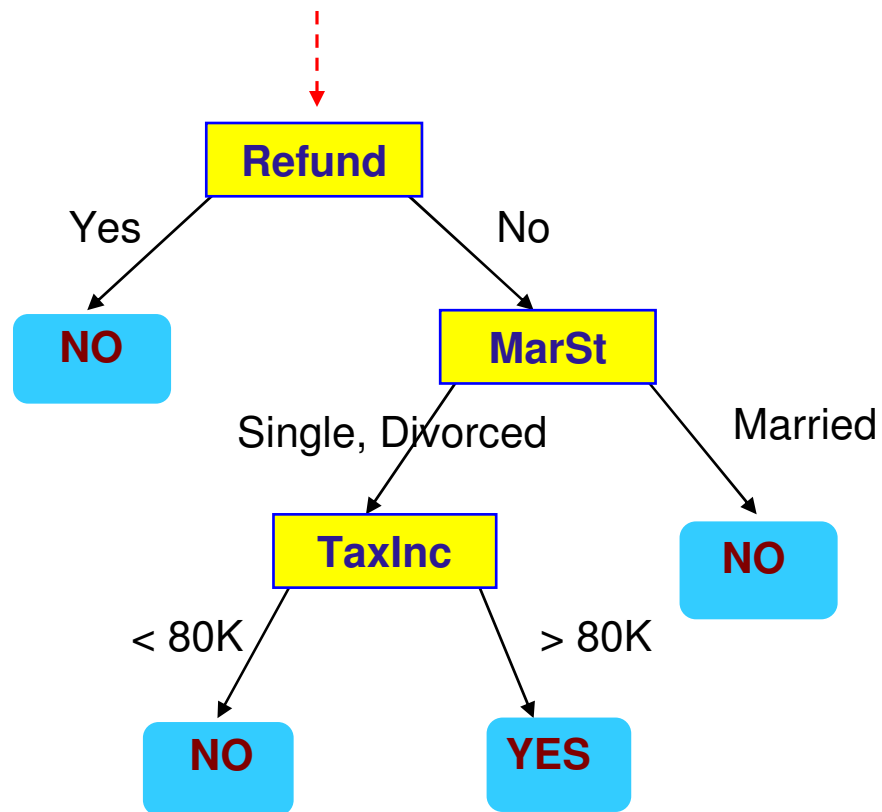
There could be more than one tree that fits the same data!

Decision Tree Classification Task



Apply Model to Test Data

Start from the root of tree.



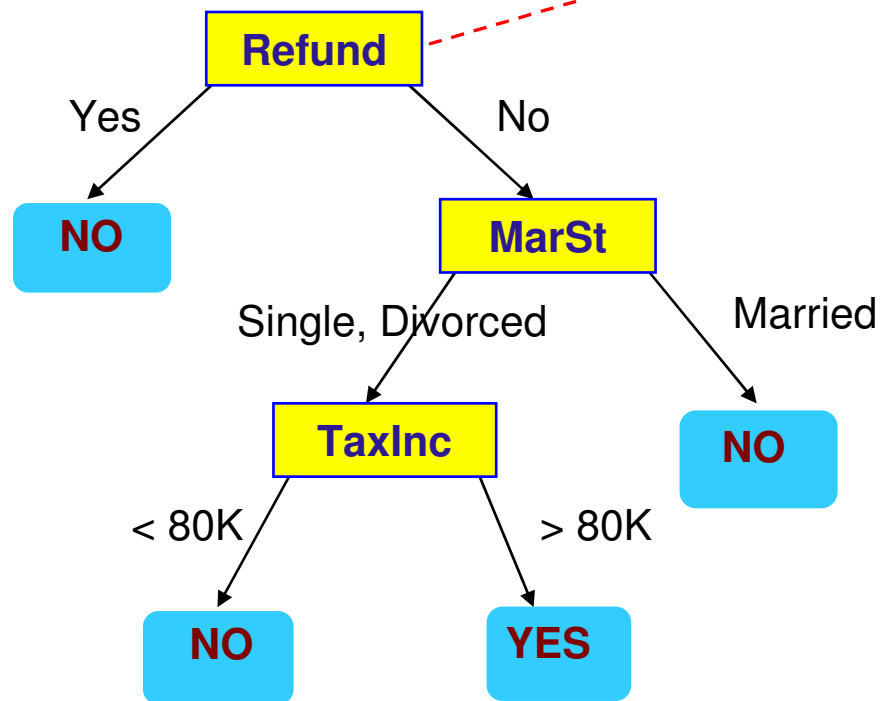
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

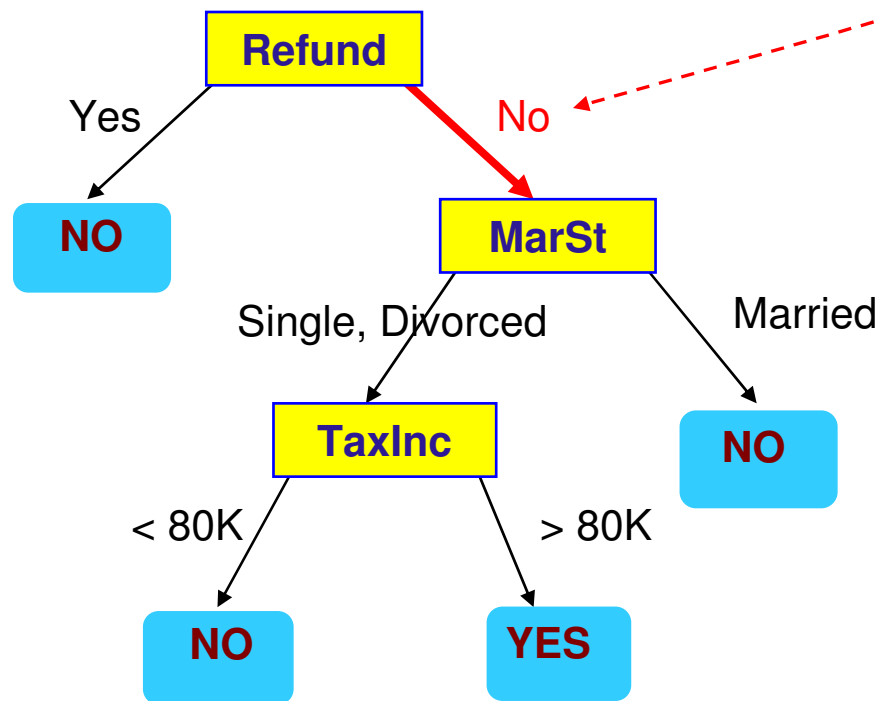
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

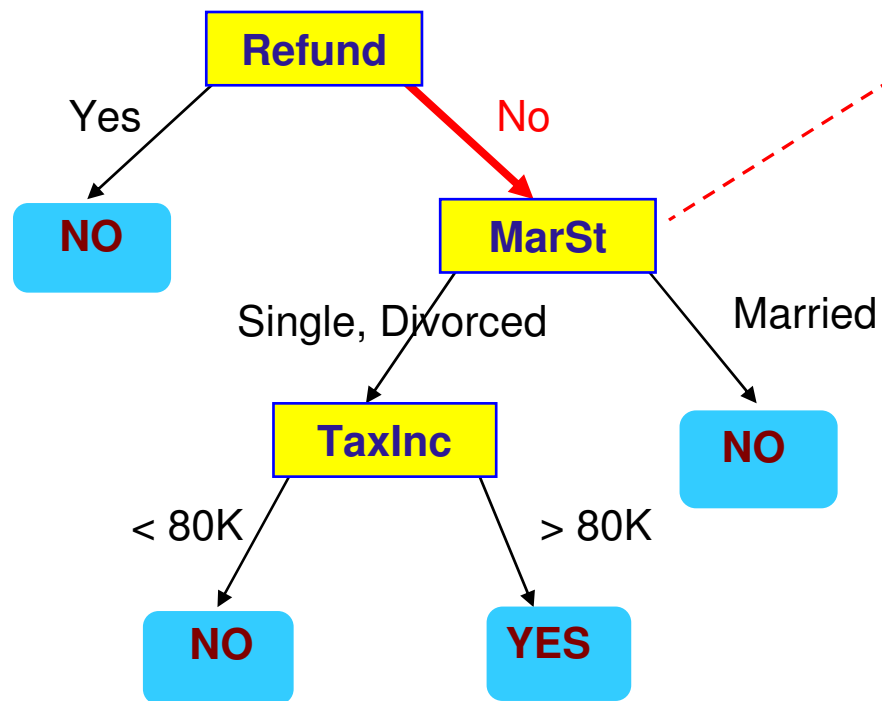
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

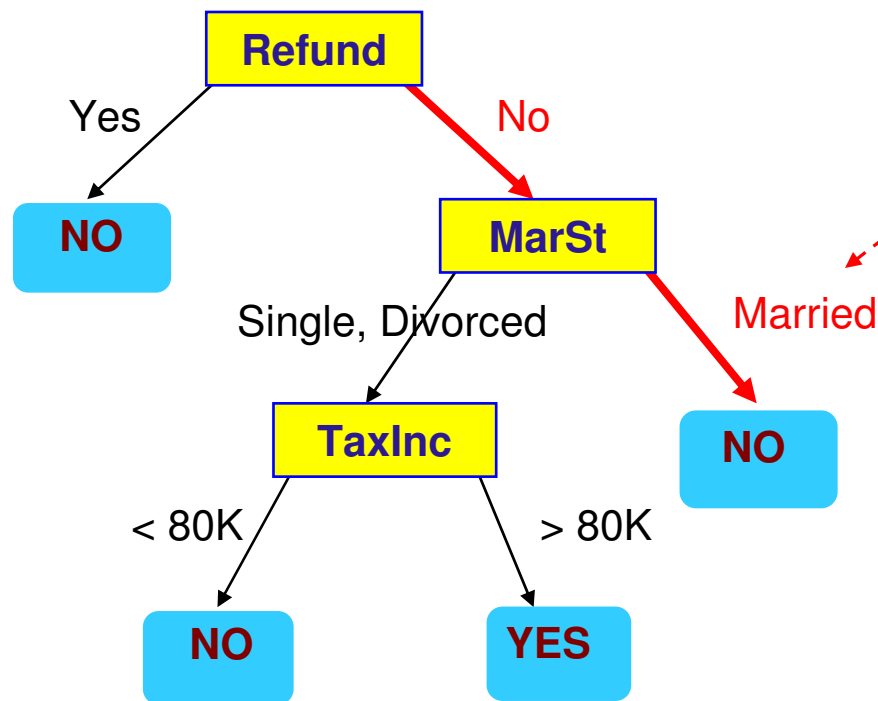
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

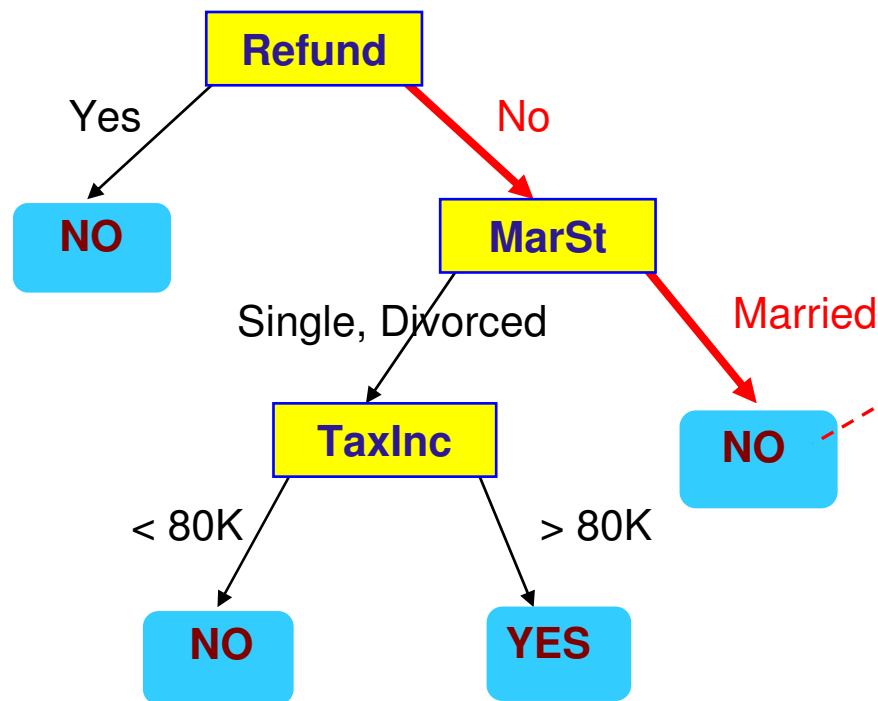
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Assign Cheat to "No"

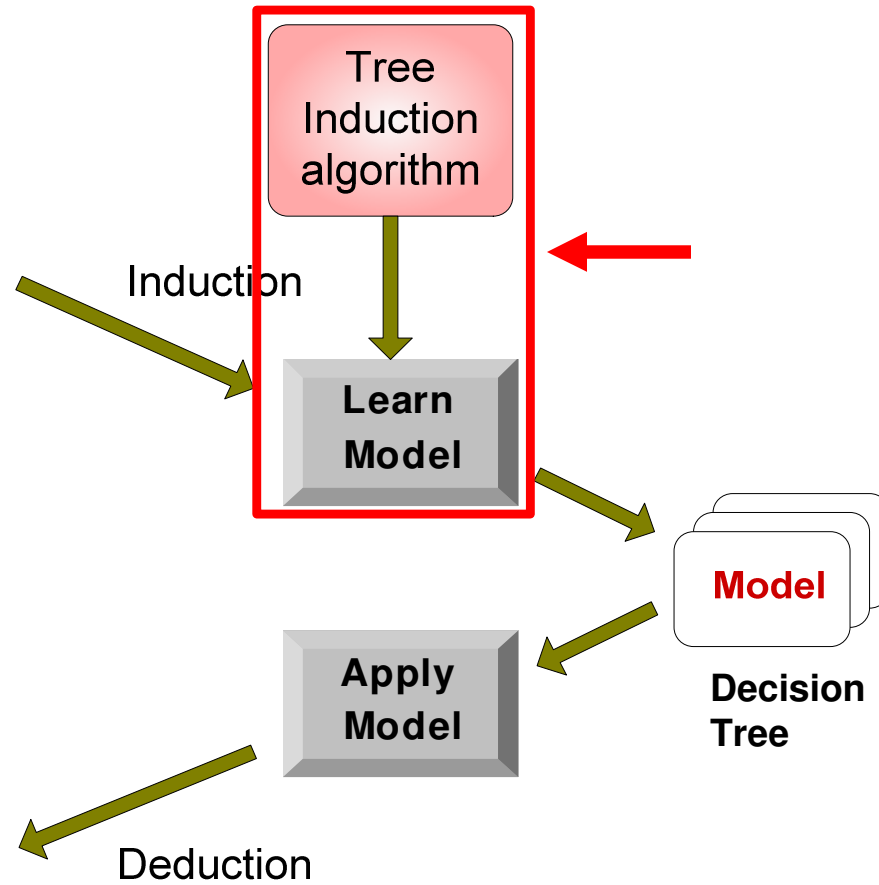
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Decision Tree Induction

- Many Algorithms:
 - Hunt's Algorithm (one of the earliest)
 - CART
 - ID3, C4.5
 - SLIQ, SPRINT

Tree Induction

- Greedy strategy.
 - Split the records based on an attribute test that optimizes certain criterion.
- Issues
 - Determine how to split the records
 - ◆ How to specify the attribute test condition?
 - ◆ How to determine the best split?
 - Determine when to stop splitting

Measures of Node Impurity

- Gini Index
- Entropy
- Misclassification error

Lezione odierna

- Alberi di decisione (concetti avanzati):
 - Criteri di split
 - Criteri di stop
 - Underfitting e Overfitting
 - Pruning
 - Missing values
 - Confusion Matrix
 - Oblique Trees
- Altri modelli di classificazione:
 - Rules-based
 - Nearest Neighbor
 - Bayes

Prossima lezione

- Ripasso di tutta la classificazione, da un punto vista + pratico
- Introduzione ad alcuni strumenti di data mining
- Esempi pratici di costruzione di classificatori

Alternative Splitting Criteria based on INFO

- Entropy at a given node t :

$$Entropy(t) = -\sum_j p(j | t) \log p(j | t)$$

(NOTE: $p(j | t)$ is the relative frequency of class j at node t).

- Measures homogeneity of a node.
 - ◆ Maximum ($\log n_c$) when records are equally distributed among all classes implying least information
 - ◆ Minimum (0.0) when all records belong to one class, implying most information
- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j|t) \log_2 p(j|t)$$

C1	0
C2	6

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$Entropy = -0 \log 0 - 1 \log 1 = -0 - 0 = 0$$

C1	1
C2	5

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$Entropy = - (1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$$

C1	2
C2	4

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$Entropy = - (2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$$

Splitting Based on INFO...

- Information Gain:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^k \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p is split into k partitions;

n_i is number of records in partition i

- Measures Reduction in Entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN)
- Used in ID3 and C4.5
- Disadvantage: Tends to prefer splits that result in large number of partitions, each being small but pure.

Splitting Based on INFO...

- Gain Ratio:

$$\text{GainRatio}_{split} = \frac{\text{GAIN}_{split}}{\text{SplitINFO}} \quad \text{SplitINFO} = -\sum_{i=1}^k \frac{n_i}{n} \log \frac{n_i}{n}$$

Parent Node, p is split into k partitions

n_i is the number of records in partition i

- Adjusts Information Gain by the entropy of the partitioning (SplitINFO). Higher entropy partitioning (large number of small partitions) is penalized!
- Used in C4.5
- Designed to overcome the disadvantage of Information Gain

Splitting Criteria based on Classification Error

- Classification error at a node t :

$$Error(t) = 1 - \max_i P(i | t)$$

- Measures misclassification error made by a node.
 - ◆ Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
 - ◆ Minimum (0.0) when all records belong to one class, implying most interesting information

Examples for Computing Error

$$Error(t) = 1 - \max_i P(i | t)$$

C1	0
C2	6

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$Error = 1 - \max(0, 1) = 1 - 1 = 0$$

C1	1
C2	5

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$Error = 1 - \max(1/6, 5/6) = 1 - 5/6 = 1/6$$

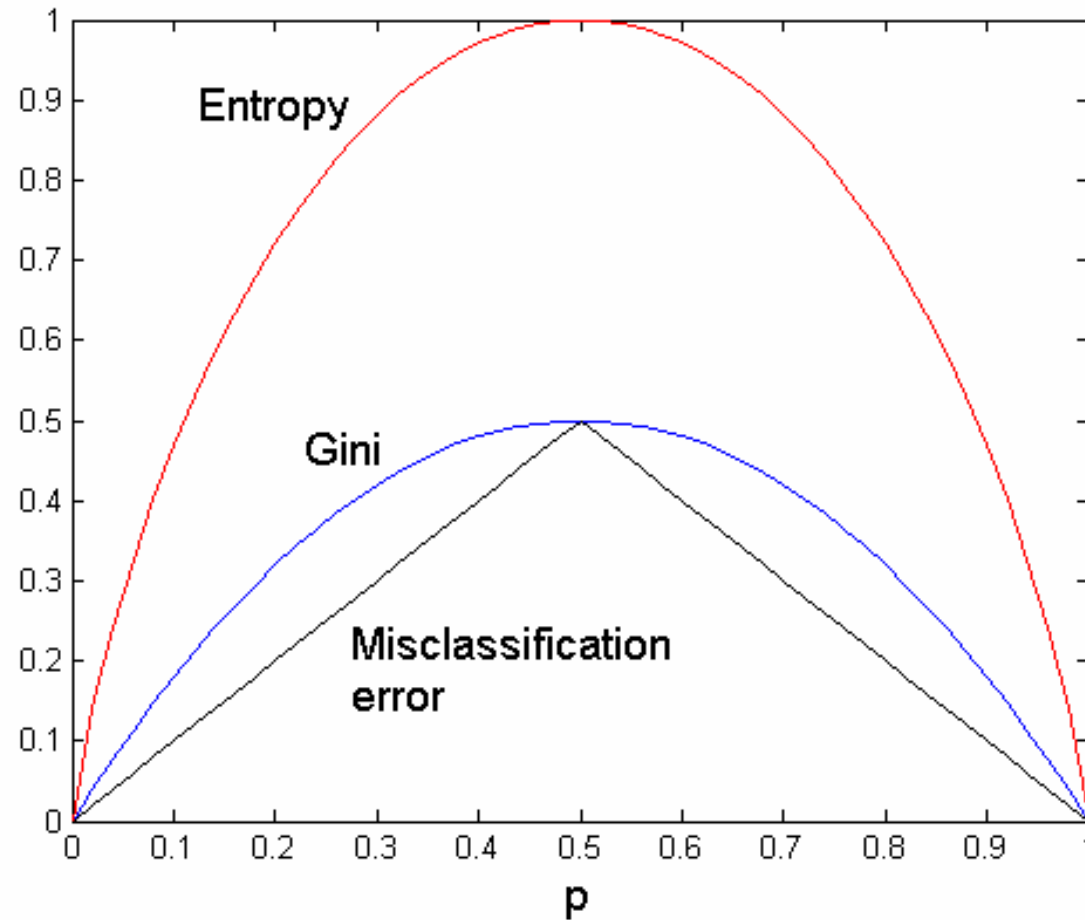
C1	2
C2	4

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$Error = 1 - \max(2/6, 4/6) = 1 - 4/6 = 1/3$$

Comparison among Splitting Criteria

For a 2-class problem:



Tree Induction

- Greedy strategy.
 - Split the records based on an attribute test that optimizes certain criterion.
- Issues
 - Determine how to split the records
 - ◆ How to specify the attribute test condition?
 - ◆ How to determine the best split?
 - **Determine when to stop splitting**

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination (to be discussed later)

Decision Tree Based Classification

- Advantages:
 - Inexpensive to construct
 - Extremely fast at classifying unknown records
 - Easy to interpret for small-sized trees
 - Accuracy is comparable to other classification techniques for many simple data sets

Example: C4.5

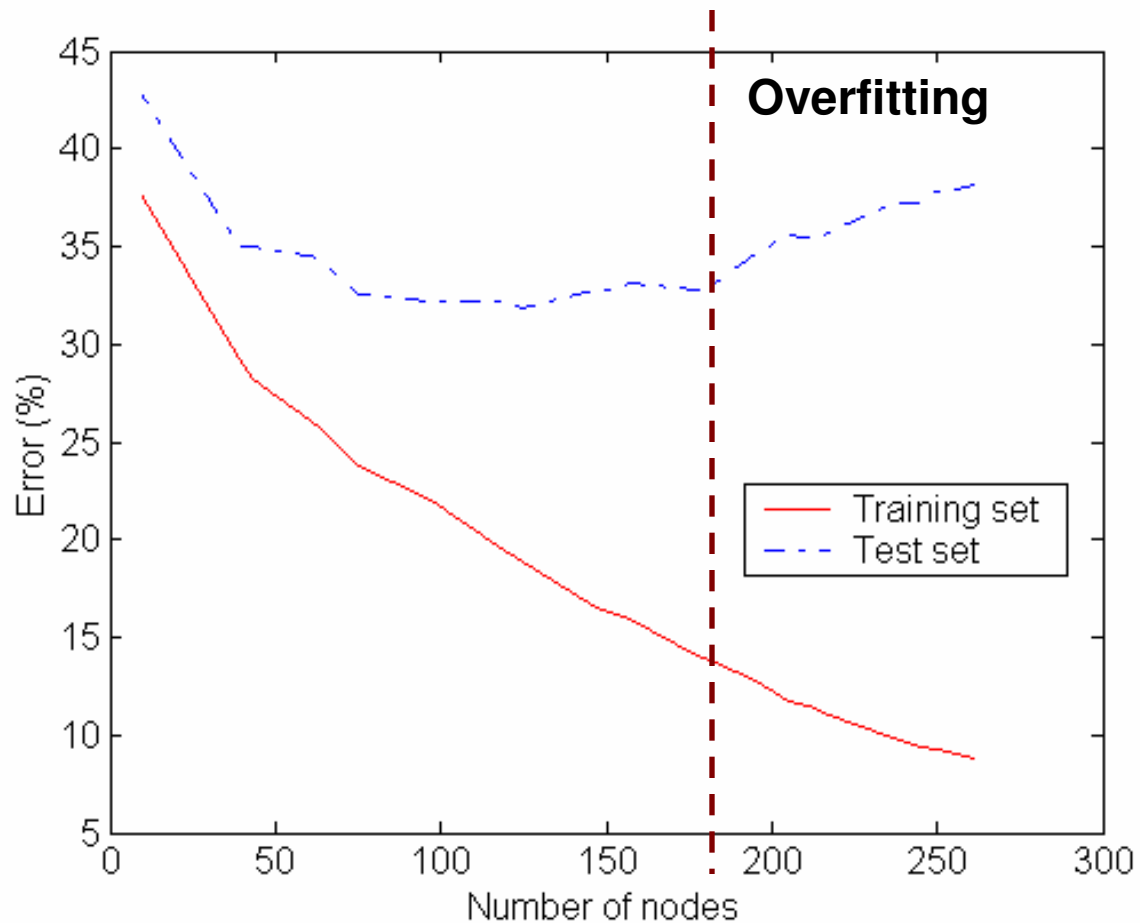
- Simple depth-first construction.
- Uses Information Gain
- Sorts Continuous Attributes at each node.
- Needs entire data to fit in memory.
- Unsuitable for Large Datasets.
 - Needs out-of-core sorting.

- You can download the software from:
<http://www.cse.unsw.edu.au/~quinlan/c4.5r8.tar.gz>

Practical Issues of Classification

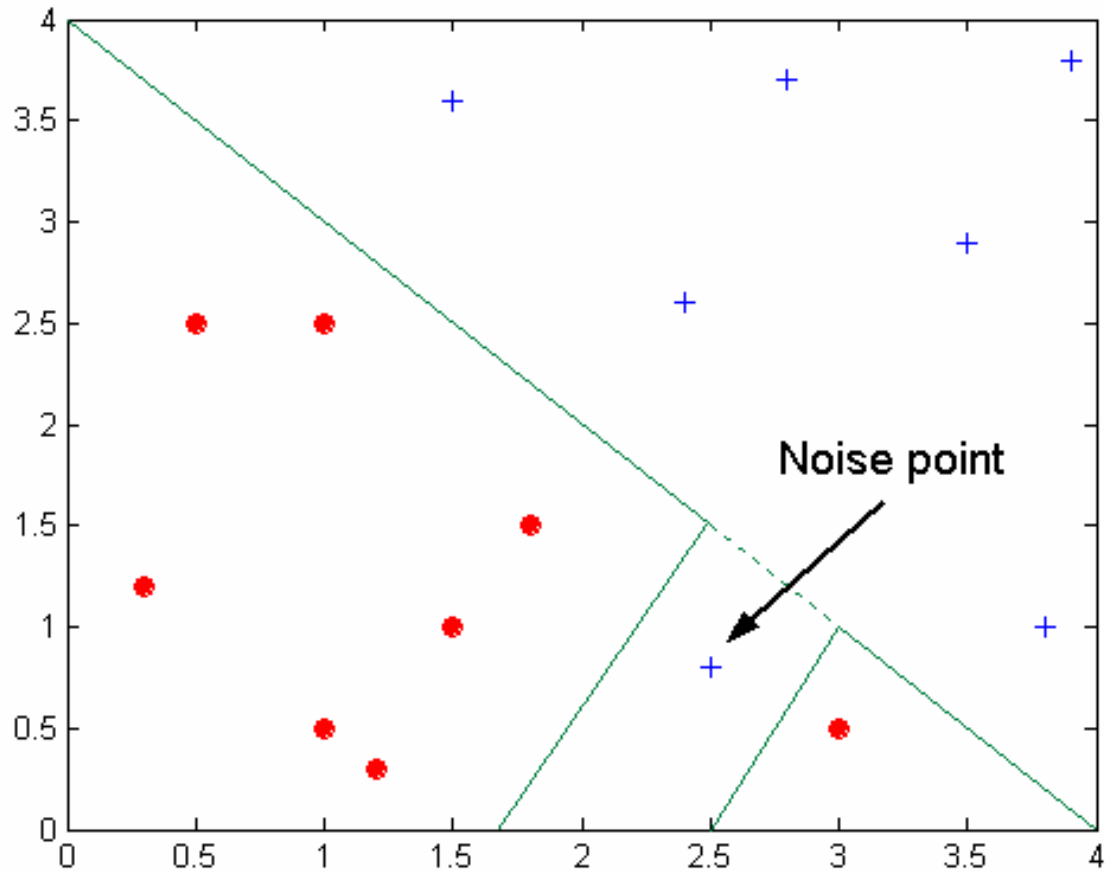
- Underfitting and Overfitting
- Missing Values
- Costs of Classification

Underfitting and Overfitting



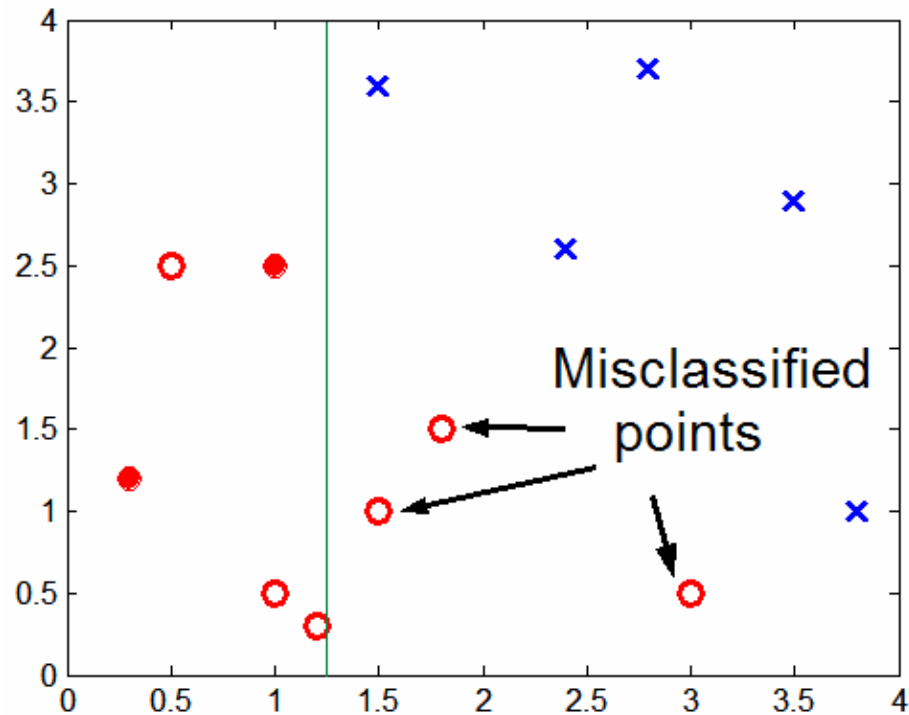
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task**

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

Occam's Razor

- Given two models of similar generalization errors, one should prefer the simpler model over the more complex model
- For complex models, there is a greater chance that it was fitted accidentally by errors in data
- Therefore, one should include model complexity when evaluating a model

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)

- Stop the algorithm before it becomes a fully-grown tree
- Typical stopping conditions for a node:
 - ◆ Stop if all instances belong to the same class
 - ◆ Stop if all the attribute values are the same
- More restrictive conditions:
 - ◆ Stop if number of instances is less than some user-specified threshold
 - ◆ Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
 - ◆ Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

How to Address Overfitting...

- **Post-pruning**
 - Grow decision tree to its entirety
 - Trim the nodes of the decision tree in a bottom-up fashion
 - If generalization error improves after trimming, replace sub-tree by a leaf node.
 - Class label of leaf node is determined from majority class of instances in the sub-tree

Example of Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

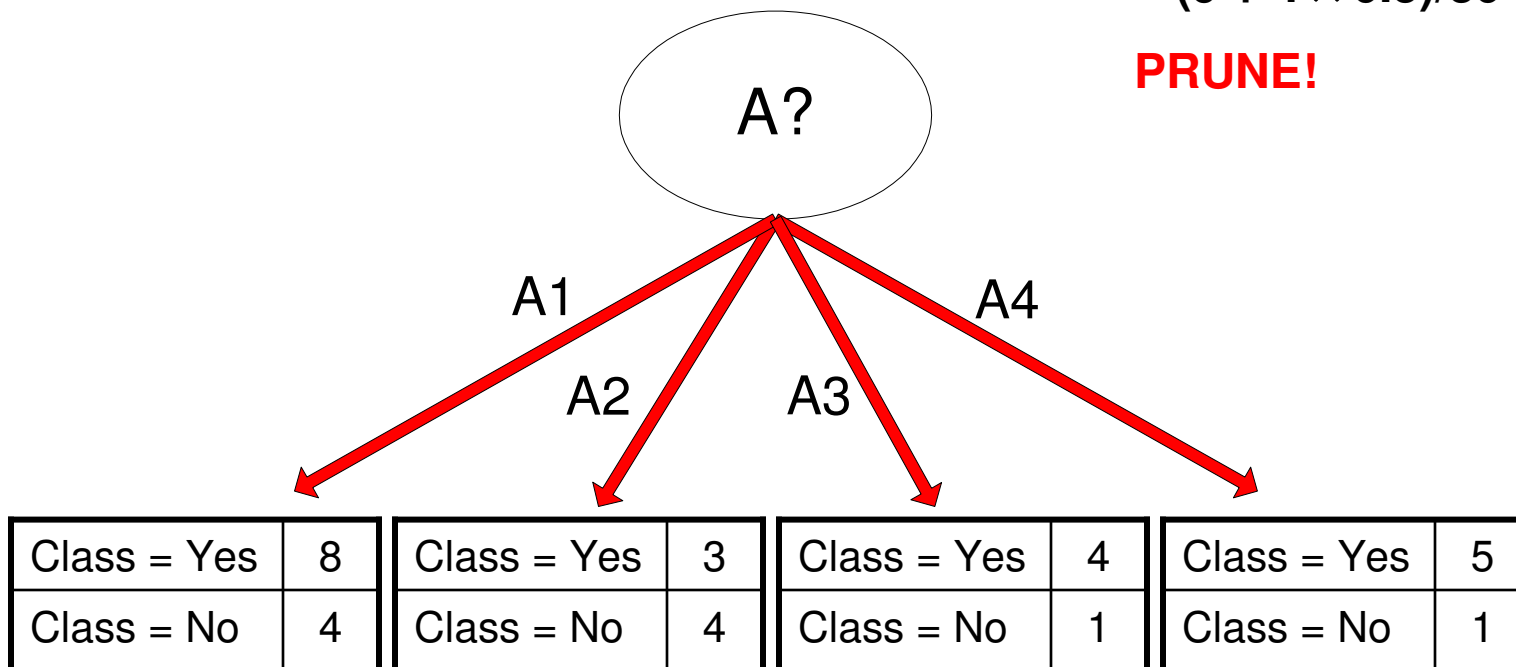
Training Error (Before splitting) = 10/30

Pessimistic error = $(10 + 0.5)/30 = 10.5/30$

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)
= $(9 + 4 \times 0.5)/30 = 11/30$

PRUNE!



Handling Missing Attribute Values

- Missing values affect decision tree construction in three different ways:
 - Affects how impurity measures are computed
 - Affects how to distribute instance with missing value to child nodes
 - Affects how a test instance with missing value is classified

Computing Impurity Measure

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

Missing value

Before Splitting:

Entropy(Parent)

$$= -0.3 \log(0.3) - (0.7) \log(0.7) = 0.8813$$

	Class = Yes	Class = No
Refund=Yes	0	3
Refund=No	2	4
Refund=?	1	0

Split on Refund:

Entropy(Refund=Yes) = 0

Entropy(Refund=No)

$$= -(2/6) \log(2/6) - (4/6) \log(4/6) = 0.9183$$

Entropy(Children)

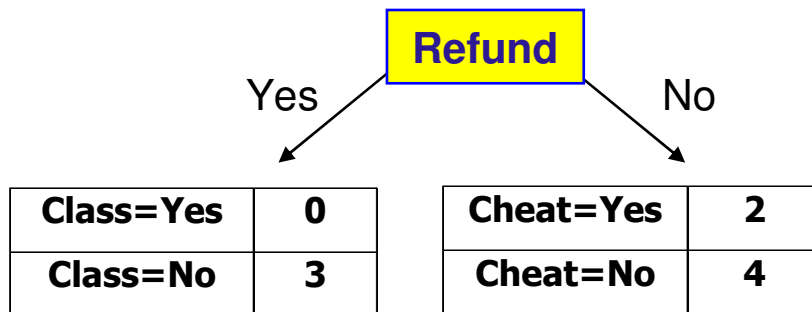
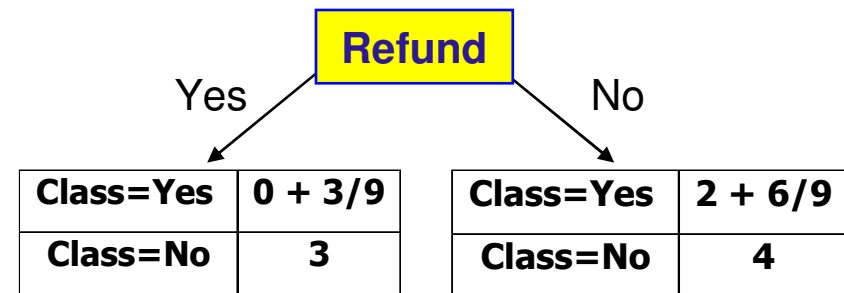
$$= 0.3 (0) + 0.6 (0.9183) = 0.551$$

$$\text{Gain} = 0.9 \times (0.8813 - 0.551) = 0.3303$$

Distribute Instances

<i>Tid</i>	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No

<i>Tid</i>	Refund	Marital Status	Taxable Income	Class
10	?	Single	90K	Yes



Probability that Refund=Yes is 3/9

Probability that Refund=No is 6/9

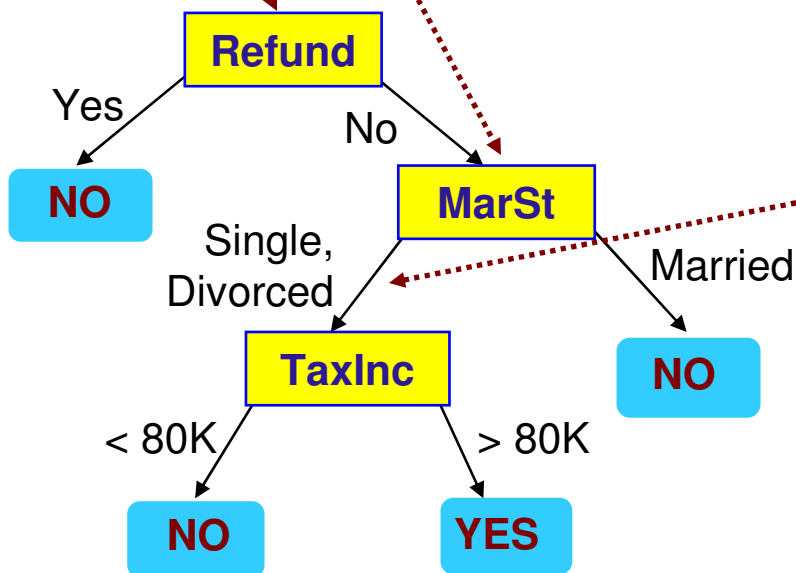
Assign record to the left child with weight = 3/9 and to the right child with weight = 6/9

Classify Instances

New record:

Tid	Refund	Marital Status	Taxable Income	Class
11	No	?	85K	?

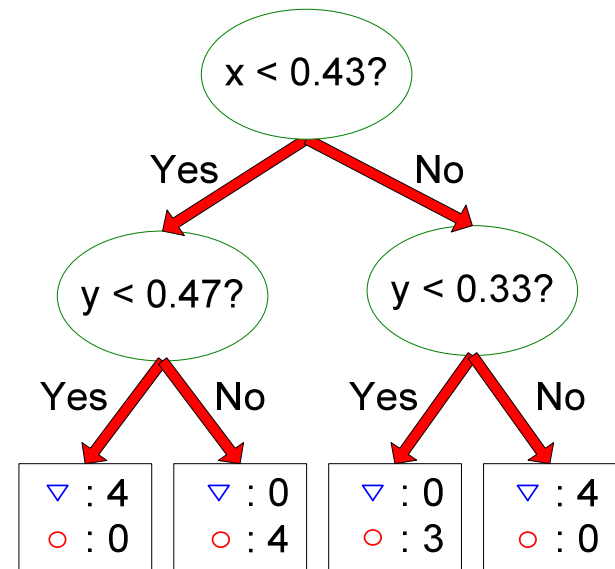
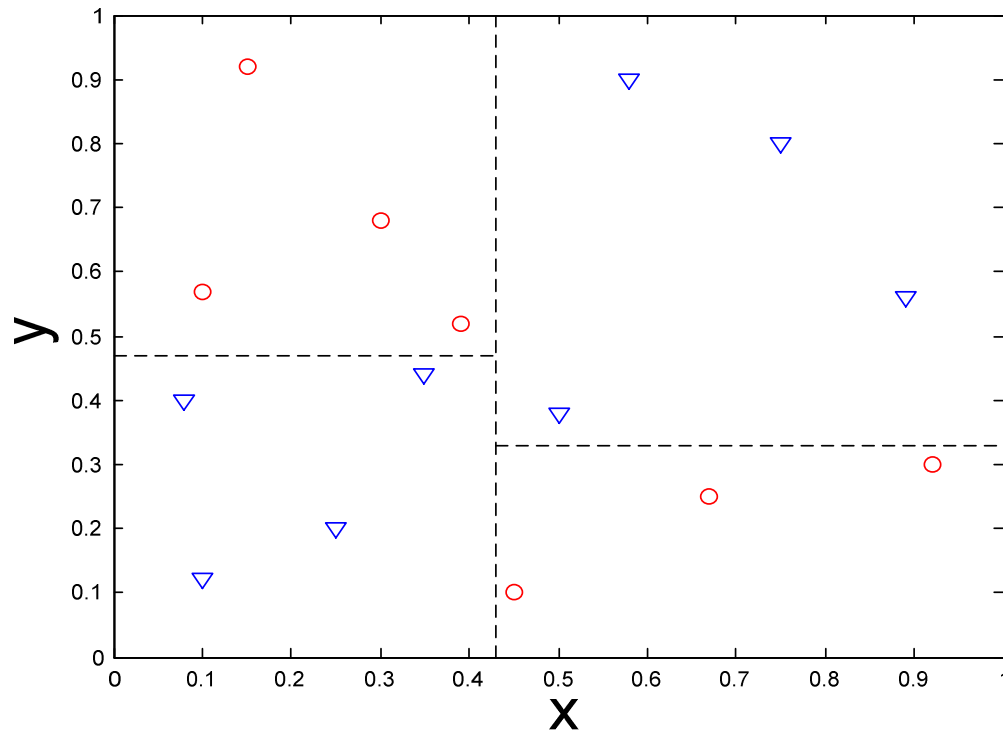
	Married	Single	Divorced	Total
Class=No	3	1	0	4
Class=Yes	6/9	1	1	2.67
Total	3.67	2	1	6.67



Probability that Marital Status = Married is $3.67/6.67$

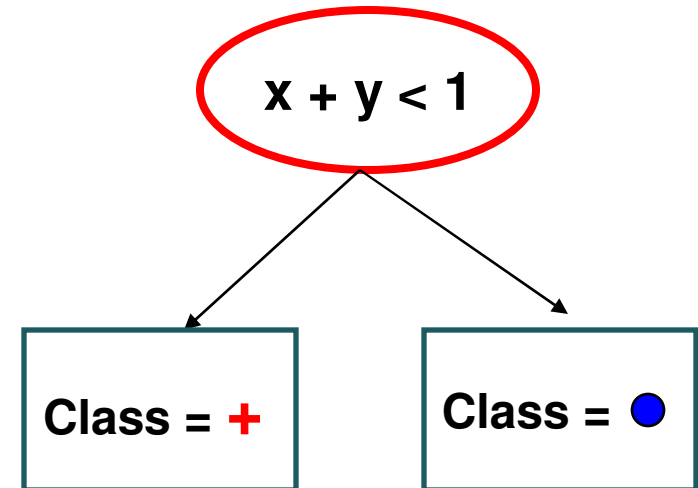
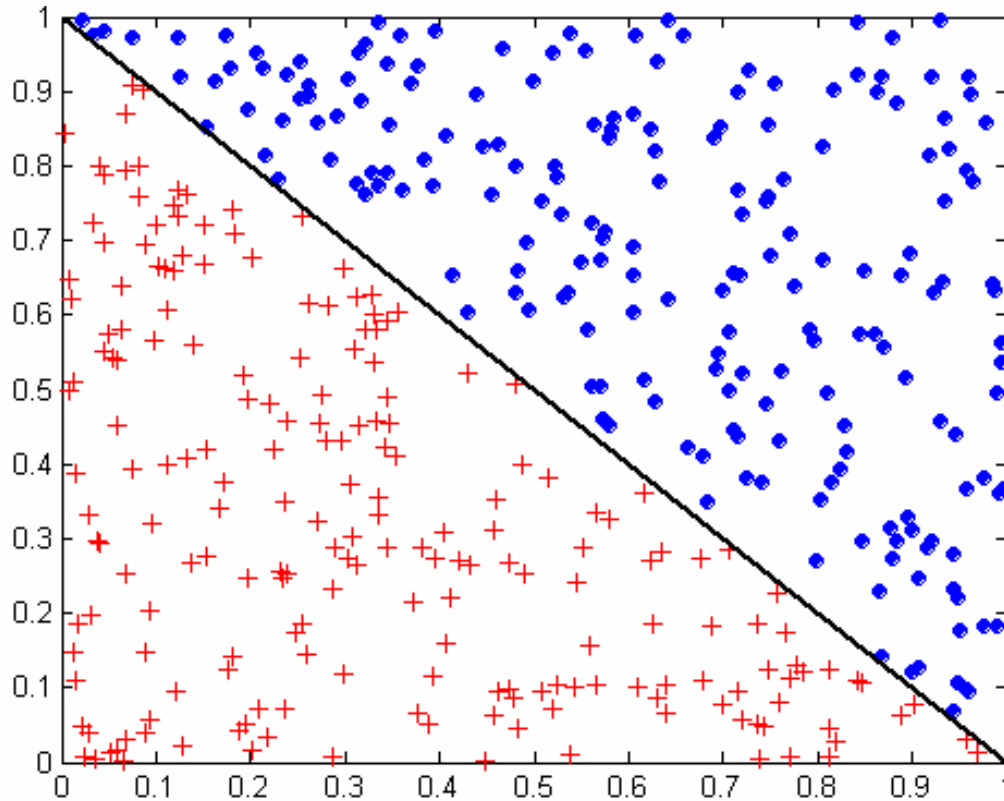
Probability that Marital Status = {Single, Divorced} is $3/6.67$

Decision Boundary



- Border line between two neighboring regions of different classes is known as decision boundary
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

Oblique Decision Trees



- Test condition may involve multiple attributes
- More expressive representation
- Finding optimal test condition is computationally expensive

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
 - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

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

Metrics for Performance Evaluation...

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

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

Limitation of Accuracy

- Consider a 2-class problem
 - Number of Class 0 examples = 9990
 - Number of Class 1 examples = 10
- If model predicts everything to be class 0, accuracy is $9990/10000 = 99.9\%$
 - Accuracy is misleading because model does not detect any class 1 example

Cost Matrix

	PREDICTED CLASS		
	$C(i j)$	Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	$C(\text{Yes} \text{Yes})$	$C(\text{No} \text{Yes})$
	Class=No	$C(\text{Yes} \text{No})$	$C(\text{No} \text{No})$

$C(i|j)$: Cost of misclassifying class j example as class i

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(i j)	+	-
	ACTUAL CLASS	+	-
	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
	ACTUAL CLASS	+	-
	+	150	40
	-	60	250

Accuracy = 80%

Cost = 3910

Model M_2	PREDICTED CLASS		
		+	-
	ACTUAL CLASS	+	-
	+	250	45
	-	5	200

Accuracy = 90%

Cost = 4255

Cost vs Accuracy

Count	PREDICTED CLASS		
	Class=Yes	Class=No	
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

Accuracy is proportional to cost if

1. $C(\text{Yes}|\text{No})=C(\text{No}|\text{Yes}) = q$
2. $C(\text{Yes}|\text{Yes})=C(\text{No}|\text{No}) = p$

$$N = a + b + c + d$$

$$\text{Accuracy} = (a + d)/N$$

Cost	PREDICTED CLASS		
	Class=Yes	Class=No	
ACTUAL CLASS	Class=Yes	p	q
	Class=No	q	p

$$\begin{aligned} \text{Cost} &= p(a + d) + q(b + c) \\ &= p(a + d) + q(N - a - d) \\ &= qN - (q - p)(a + d) \\ &= N[q - (q - p) \times \text{Accuracy}] \end{aligned}$$

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

- 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

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

(Status=Single) → No

Coverage = 40%, Accuracy = 50%

How does Rule-based Classifier Work?

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

R2: (Give Birth = no) \wedge (Live in Water = yes) \rightarrow Fishes

R3: (Give Birth = yes) \wedge (Blood Type = warm) \rightarrow Mammals

R4: (Give Birth = no) \wedge (Can Fly = no) \rightarrow Reptiles

R5: (Live in Water = sometimes) \rightarrow Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur triggers rule R3, so it is classified as a mammal

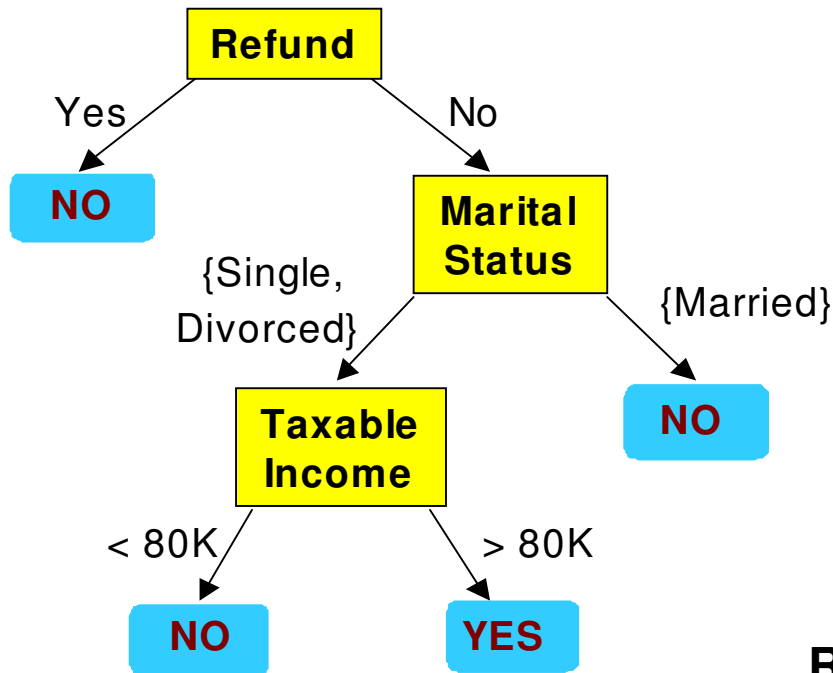
A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules

Characteristics of Rule-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

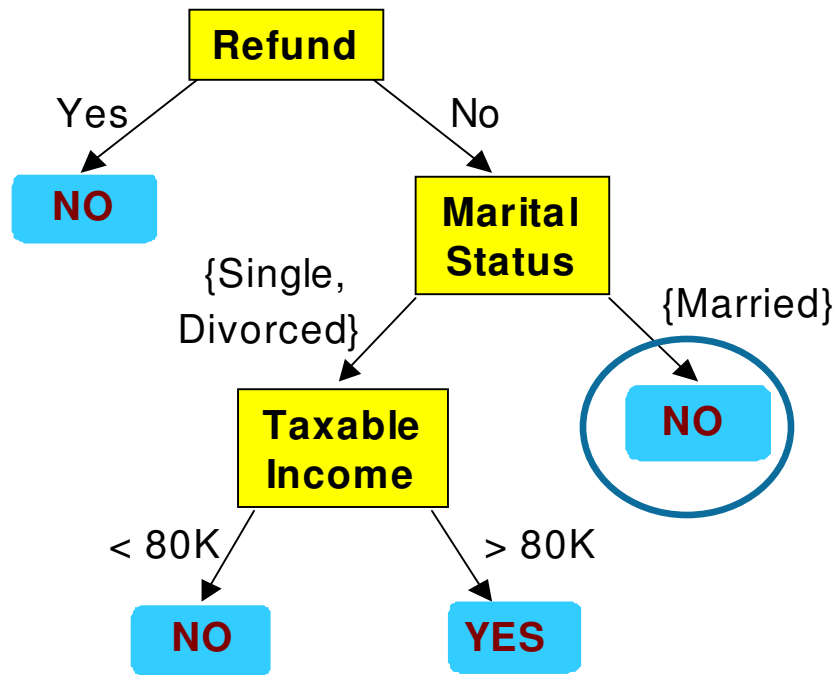
(Refund=No, Marital Status={Single, Divorced}, Taxable Income>80K) ==> Yes

(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



<i>Tid</i>	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

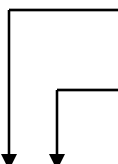
R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

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	?

Rule Ordering Schemes

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

Rule-based Ordering

(Refund=Yes) ==> No

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

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

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

Class-based Ordering

(Refund=Yes) ==> No

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

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

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

Building Classification Rules

- Direct Method:
 - ◆ Extract rules directly from data
 - ◆ e.g.: RIPPER, CN2, Holte's 1R

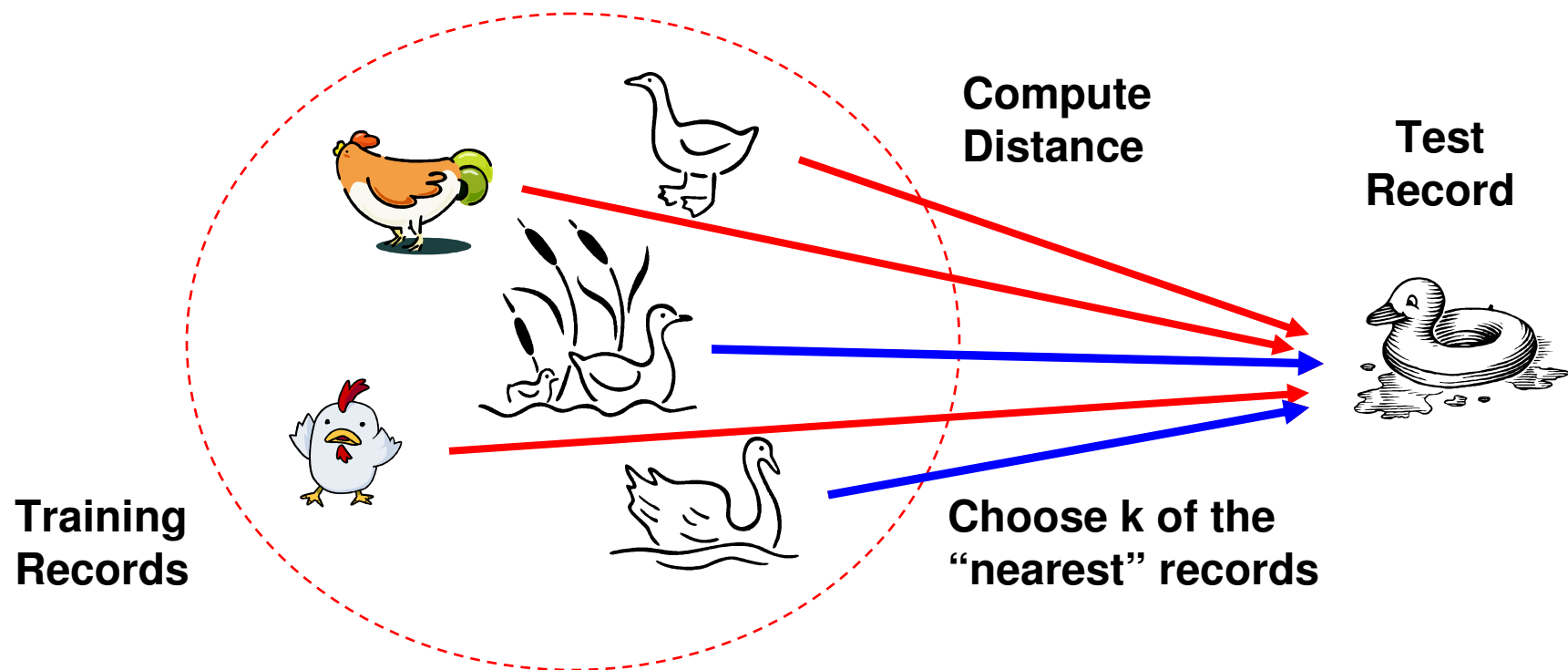
- Indirect Method:
 - ◆ Extract rules from other classification models (e.g. decision trees, neural networks, etc).
 - ◆ e.g: C4.5rules

Advantages of Rule-Based Classifiers

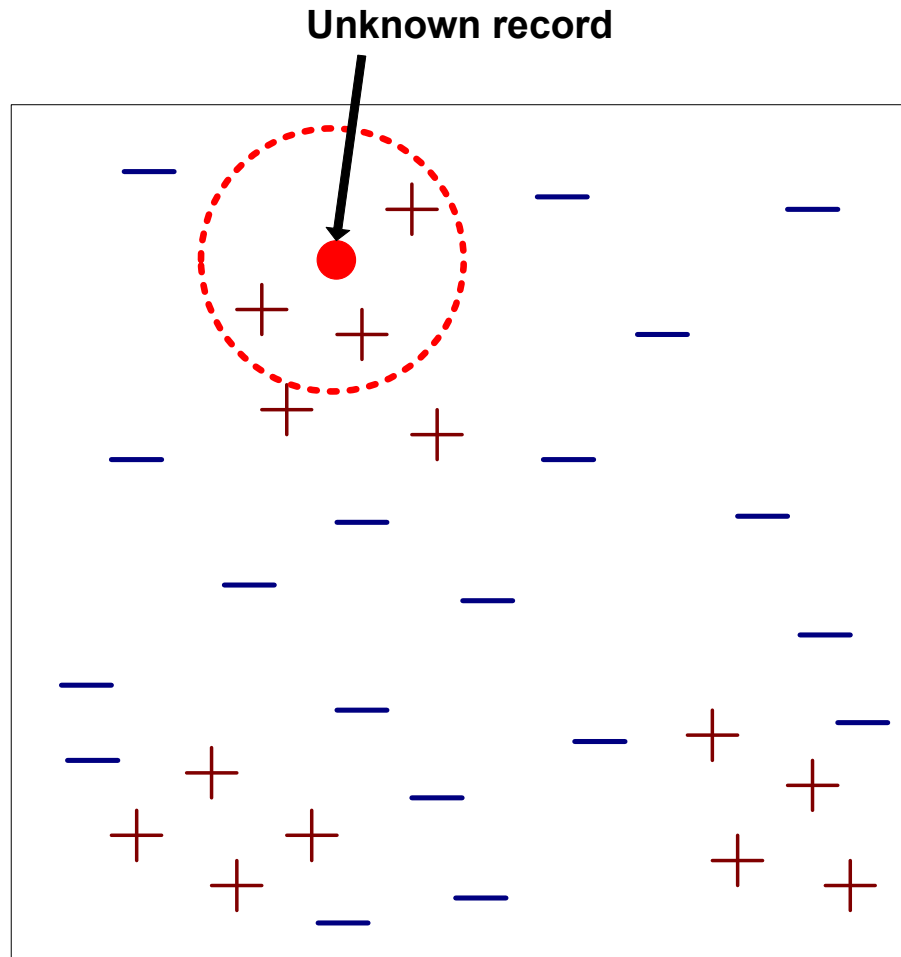
- As highly expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

Nearest Neighbor Classifiers

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

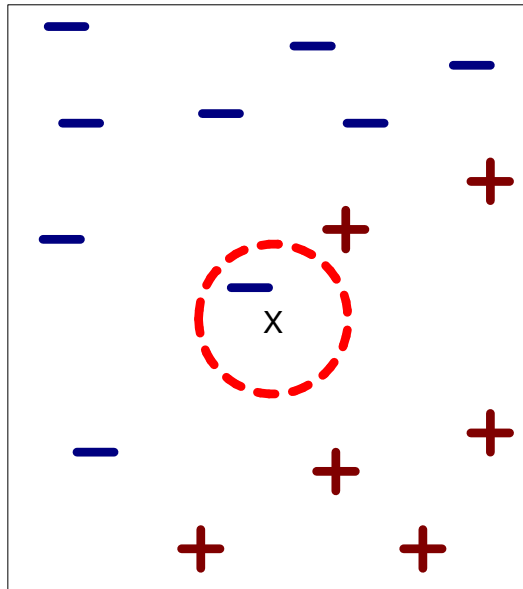


Nearest-Neighbor Classifiers

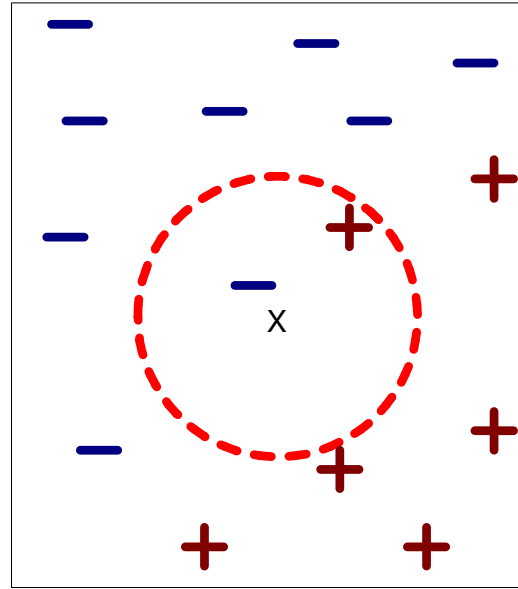


- Requires three things
 - The set of stored records
 - Distance Metric to compute distance between records
 - The value of k , the number of nearest neighbors to retrieve
- To classify an unknown record:
 - Compute distance to other training records
 - Identify k nearest neighbors
 - Use class labels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

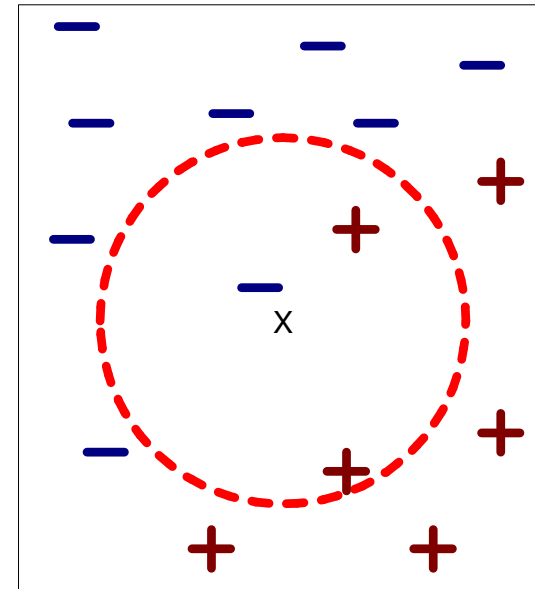
Definition of Nearest Neighbor



(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

Nearest Neighbor Classification

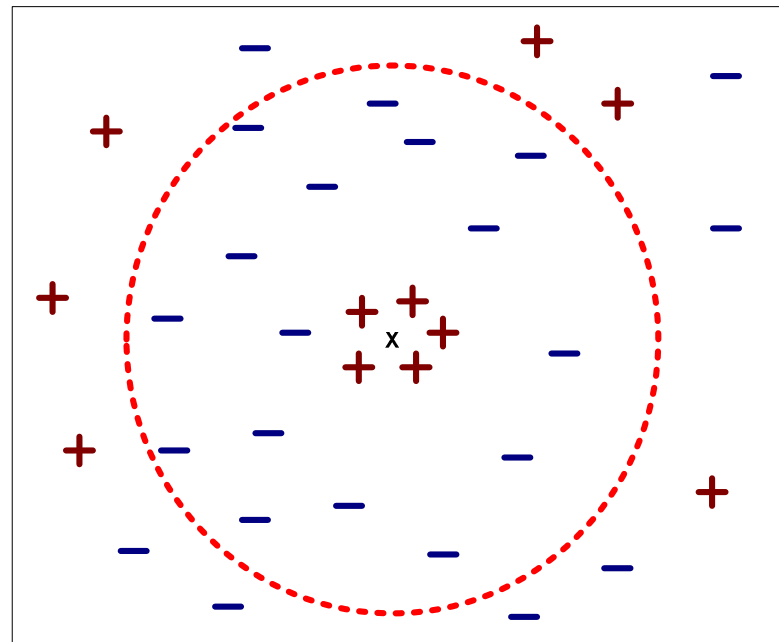
- Compute distance between two points:
 - Euclidean distance

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

- 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...

- 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

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

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

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

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

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

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

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)