Data Mining
Classification: Basic Concepts, Decision Trees, and Model Evaluation

Lecture Notes for Chapter 3

Introduction to Data Mining

contents

- 分类例子
- 分类的基本概念
- 决策树
- 模型的过分拟合
- 评估分类的性能
- 比较分类器的方法

本章的目的

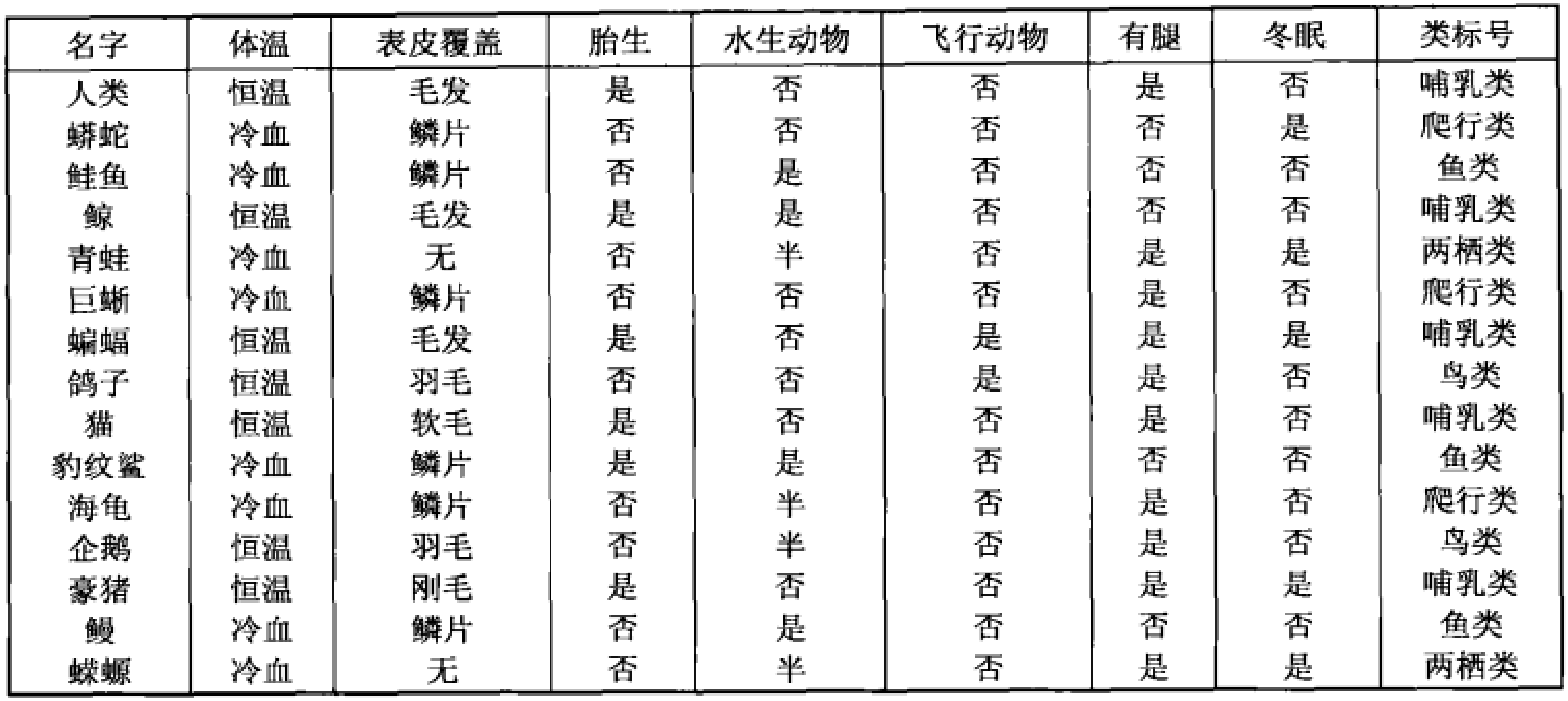
- 分类任务就是确定对象属于那个预定义的目标类。
- 分类问题是一个普遍存在的问题，有许多不同的应用：
  - 根据电子邮件的标题和内容检查出垃圾邮件；
  - 根据web网页的内容进行网页分类；
  - 根据图像的内容和语义对图像进行分类；
  - 根据用户的习惯对客户进行分类。
  - ……
- 本章将以决策树归纳技术作为例子，介绍分类的基本概念、模型的过分拟合和模型的评估和比较方法。

3.1 一个分类的例子

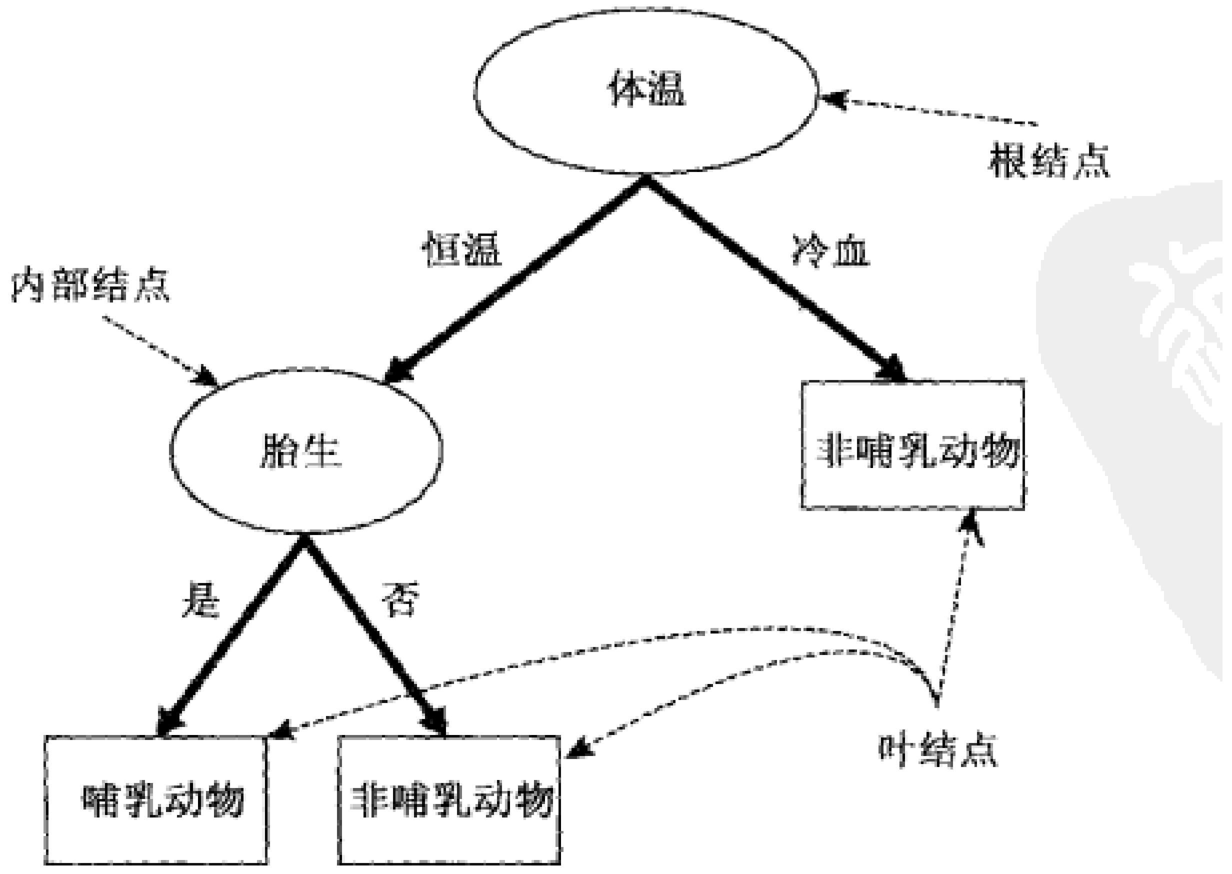
- 分类任务就是确定对象属于那个预定义的目标类。
- 分类（classification）：分类任务就是通过学习得到一个目标函数f（target function）,把每个属性集x映射到一个预先定义的类标号y。
  - 目标函数也称分类模型（classification model）
- 分类模型的作用：
  - 描述性建模
  - 预测性建模

描述性建模：给定训练集

表 4-1　脊椎动物的数据集



名字	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
人类	恒温	毛发	是	否	否	是	否	哺乳类
蟒蛇	冷血	鳞片	否	否	否	否	是	爬行类
鲑鱼	冷血	鳞片	否	是	否	否	否	鱼类
鲸	恒温	毛发	是	是	否	否	否	哺乳类
青蛙	冷血	无	否	半	否	是	是	两栖类
巨蜥	冷血	鳞片	否	否	否	是	否	爬行类
蝙蝠	恒温	毛发	是	否	是	是	是	哺乳类
鸽子	恒温	羽毛	否	否	是	是	否	鸟类
猫	恒温	软毛	是	否	否	是	否	哺乳类
豹纹鲨	冷血	鳞片	是	是	否	否	否	鱼类
海龟	冷血	鳞片	否	半	否	是	否	爬行类
企鹅	恒温	羽毛	否	半	否	是	否	鸟类
豪猪	恒温	刚毛	是	否	否	是	是	哺乳类
鳗	冷血	鳞片	否	是	否	否	否	鱼类
蝾螈	冷血	无	否	半	否	是	是	两栖类


体温
根结点
恒温
冷血
内部结点
胎生
非哺乳动物
是
否
叶结点
哺乳动物
非哺乳动物

预测性建模

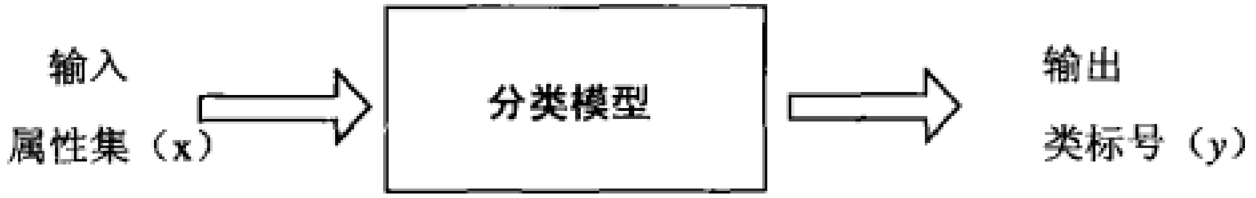


图 4-2　分类器的任务是根据输入属性集 x 确定类标号 y

名字	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
毒蜥	冷血	鳞片	否	否	否	是	是	?

分类模型的适用性

- 分类技术
  - 非常适合预测或描述二元或标称类型的数据集
  - 不适合序数型数据集的分类
  - 分类不考虑隐含在目标类中的序数关系和继承关系

3.2 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.
- 训练集、属性、类标号、模型、测试集

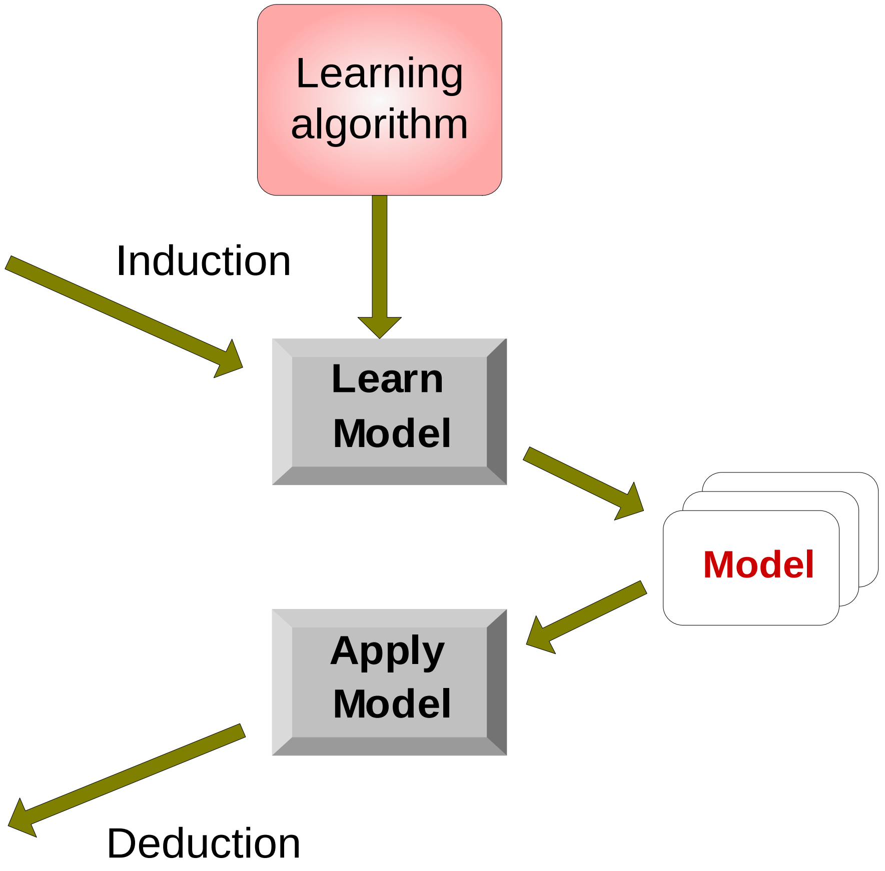
Illustrating Classification Task

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

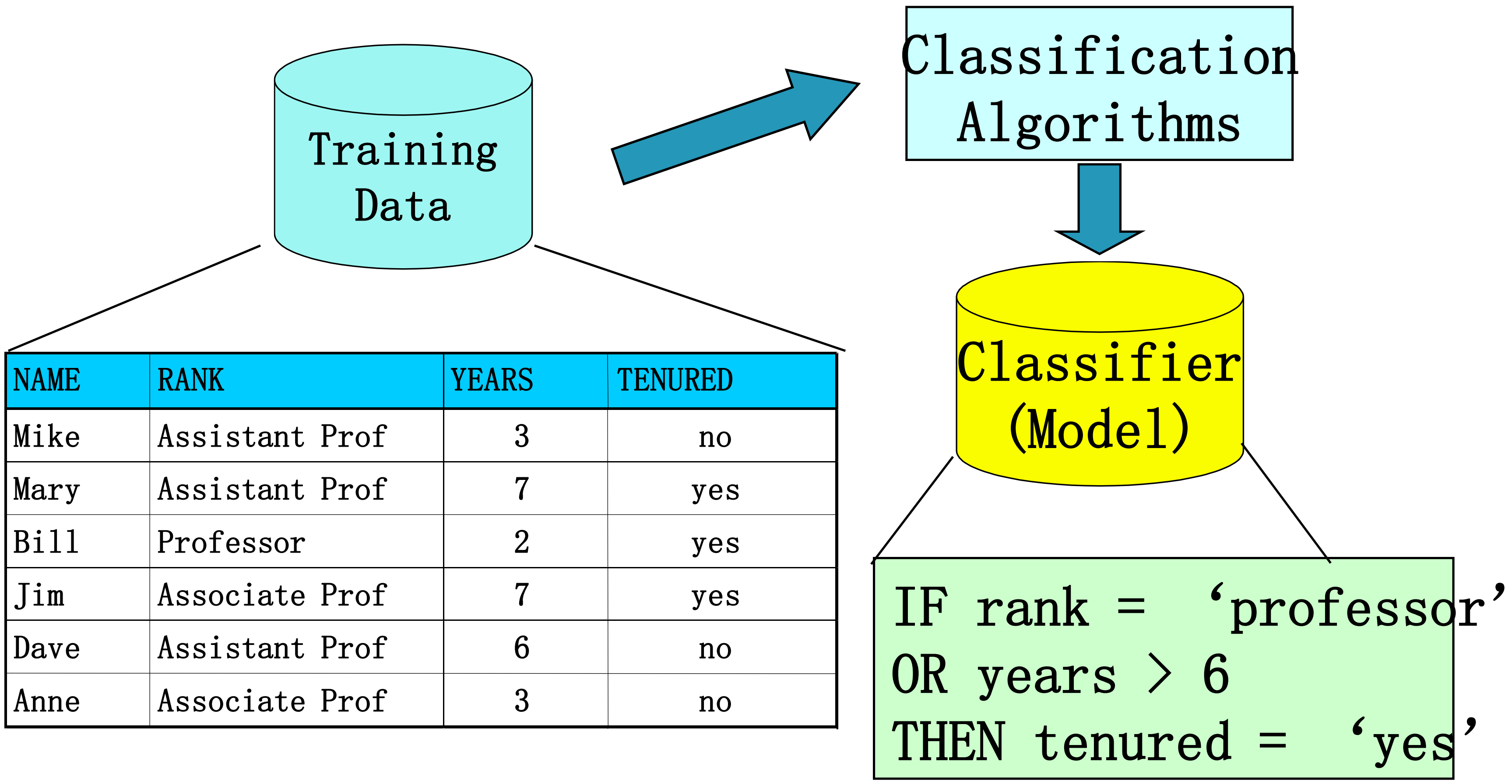
Training Set

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



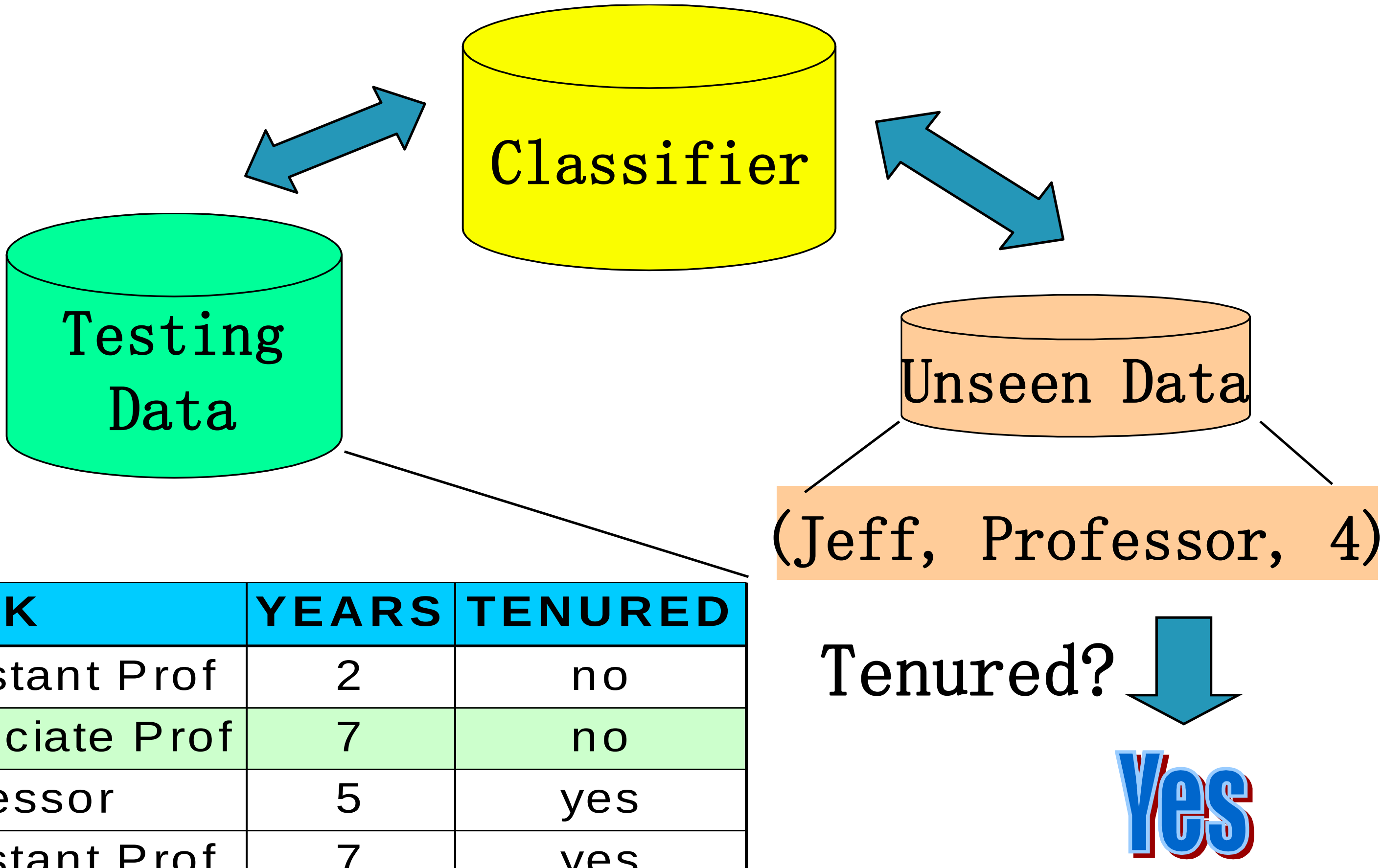
(1) 模型的构建



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

(2) 利用模型分类



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

有监督（指导） vs. 无监督（无指导）的学习

- 有指导的学习 (分类) superwised
  - 指导: 训练数据是已经被标注好类标号的数据，用来进行有指导的分类。
  - 新数据是基于训练集进行分类的。
- 无指导的学习 (聚类) unsuperwised
  - 训练数据的类标号不可知
  - 是观察式学习

评价分类方法

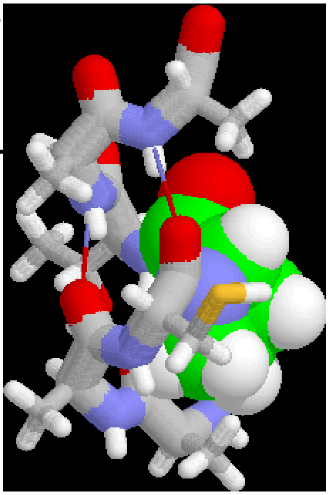
- 预测的准确率
  - 这涉及模型正确地预测新的或先前未见过的数据的类标号的能力
- 速度
  - 构造模型的速度
  - 利用模型进行分类的速度
- 强壮性
  - 给定噪声数据或具有空缺值的数据，模型正确预测的能力
- 可伸缩性
  - 当给定大量数据时，有效地构造模型的能力
- 可解释性
  - 涉及学习模型提供的理解和洞察的层次

Examples of Classification Task

- Predicting tumor cells as benign or malignant

- Classifying credit card transacti as legitimate or fraudulent

- Classifying secondary structures of as alpha-helix, beta-sheet, or rand coil

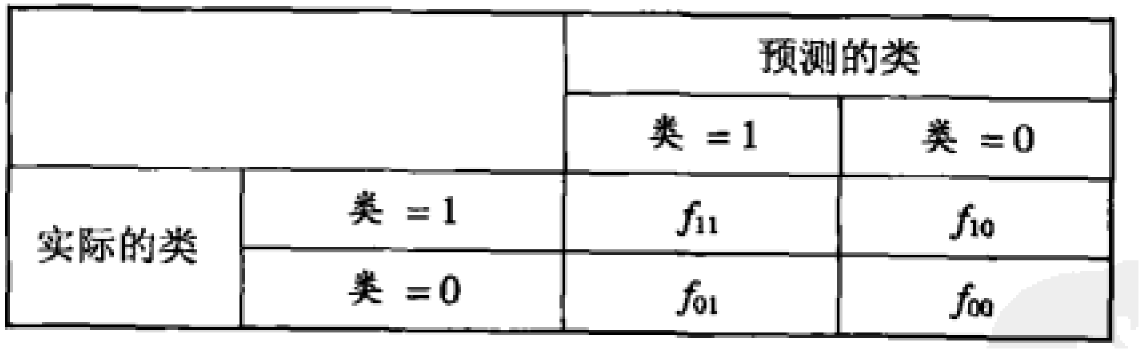


- Categorizing news stories as financ weather, entertainment, sports, etc

评价方法－混淆矩阵

- 分类模型的性能根据模型正确和错误预测的检验记录计数进行评估，这些计数存放在被称作混淆矩阵（confusion matrix）的表格中。
- 混淆矩阵举例：

表 4-2　二类问题的混淆矩阵



		预测的类	
		类 =1	类 =0
实际的类	类 =1	f_{11}	f_{10}
	类 =0	f_{01}	f_{00}

评价方法—性能度量（performance metric）

表 4-2　二类问题的混淆矩阵

		预测的类	
		类 =1	类 =0
实际的类	类 =1	f_{11}	f_{10}
	类 =0	f_{01}	f_{00}

$$准确率 \doteq \frac{正确预测数}{预测总数} = \frac{f_{11}+f_{00}}{f_{11}+f_{10}+f_{01}+f_{00}}$$

$$错误率 = \frac{错误预测数}{预测总数} = \frac{f_{10}+f_{01}}{f_{11}+f_{10}+f_{01}+f_{00}}$$

Classification Techniques-常用分类技术

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

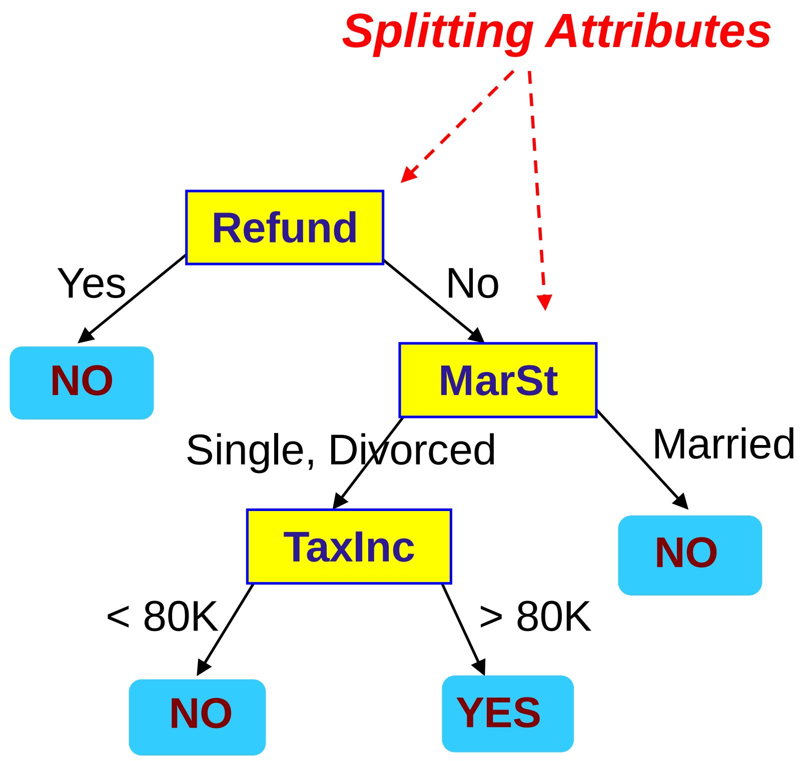
3.3 基于决策树的分类方法

DECISION TREE BASED METHODS

Example of a Decision Tree

Tid	Refund categorical	Marital Status categorical	Taxable Income continuous	Cheat 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

Training Data

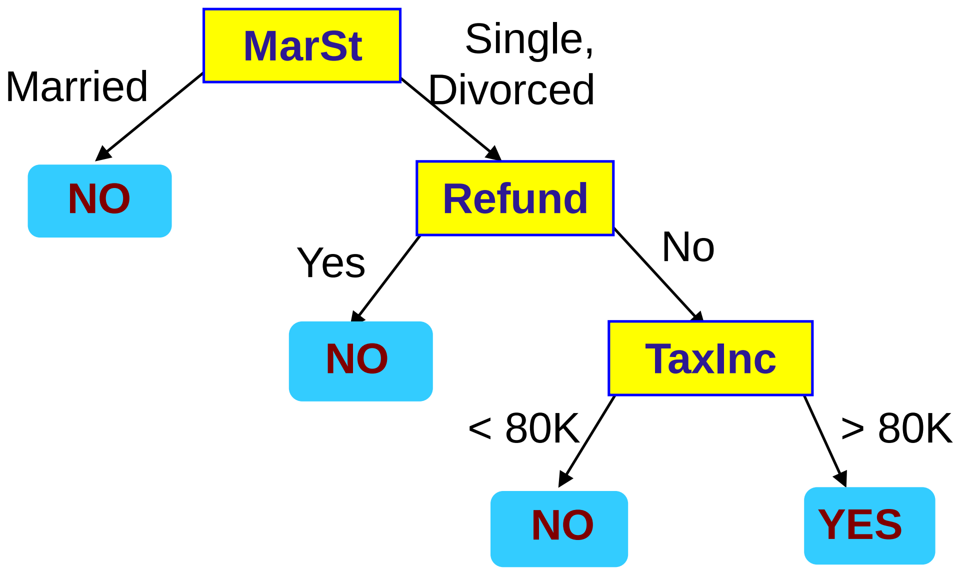


Model: Decision Tree

根节点root node，内部节点internal node，叶节点leaf node

Another Example of Decision Tree

	categorical	categorical	continuous	class
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**



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

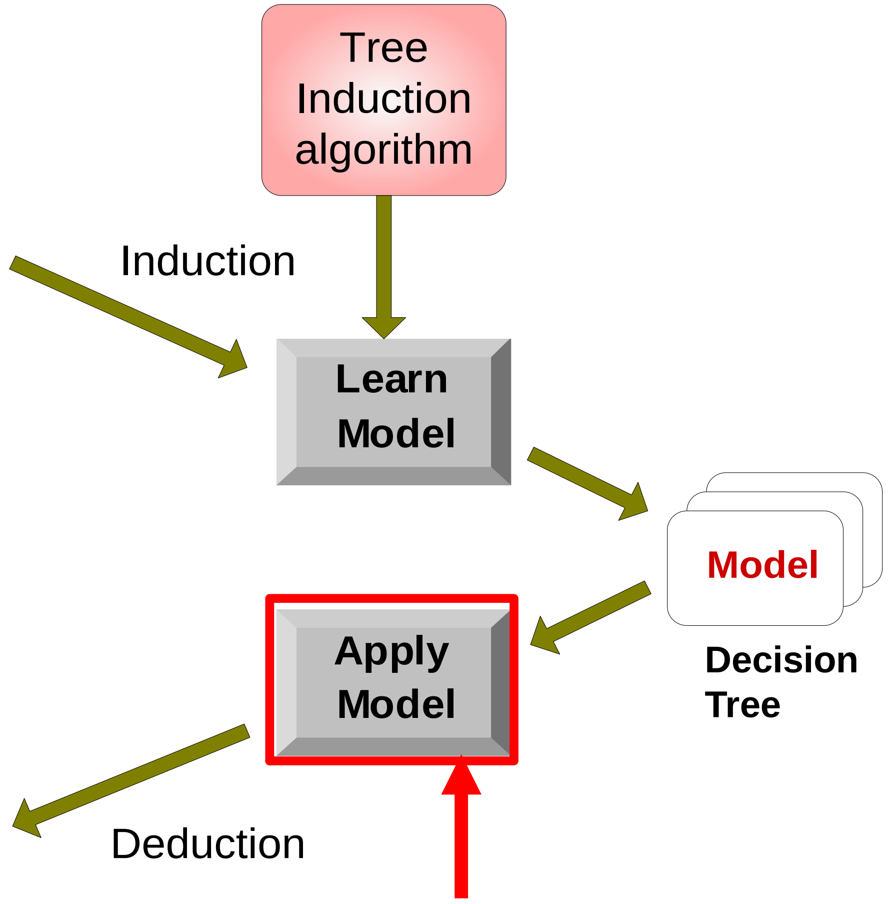
Decision Tree Classification Task

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

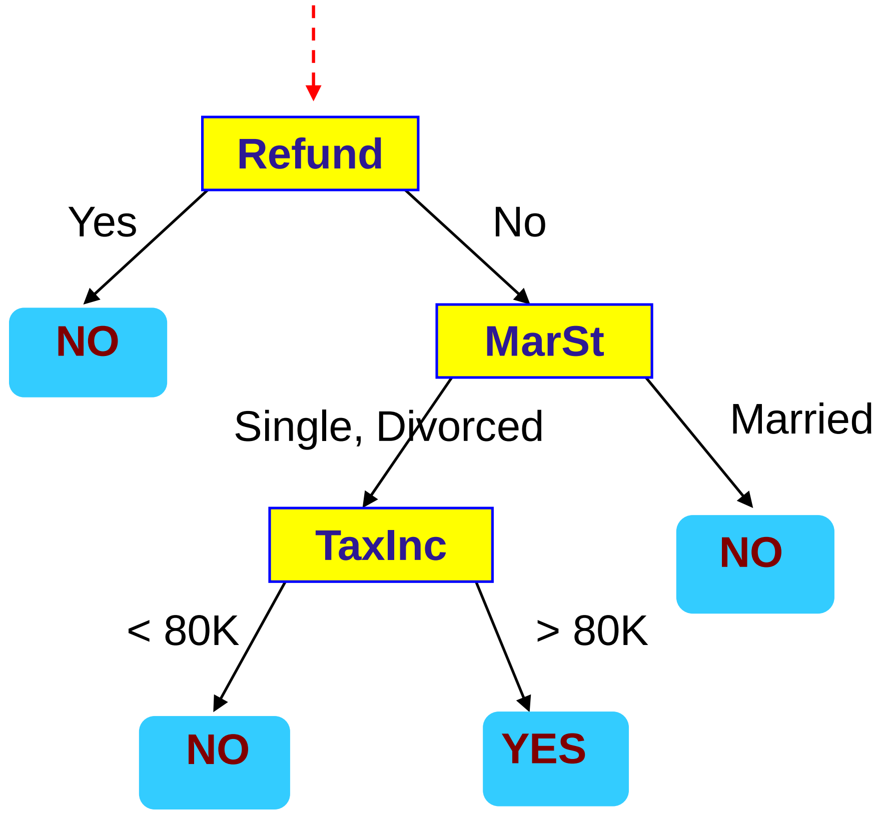
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Apply Model to Test Data

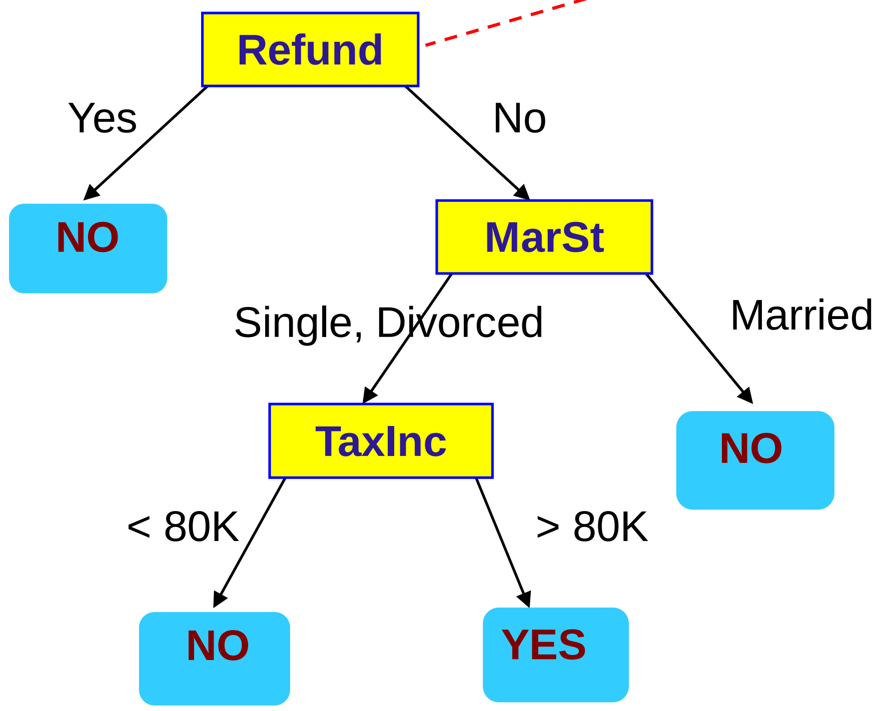
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

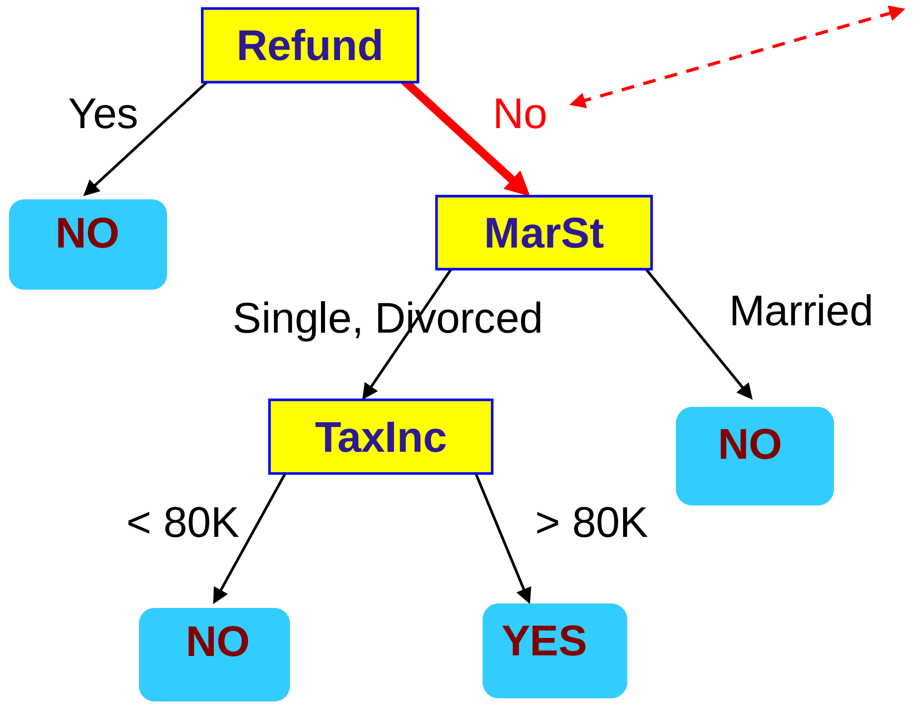
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

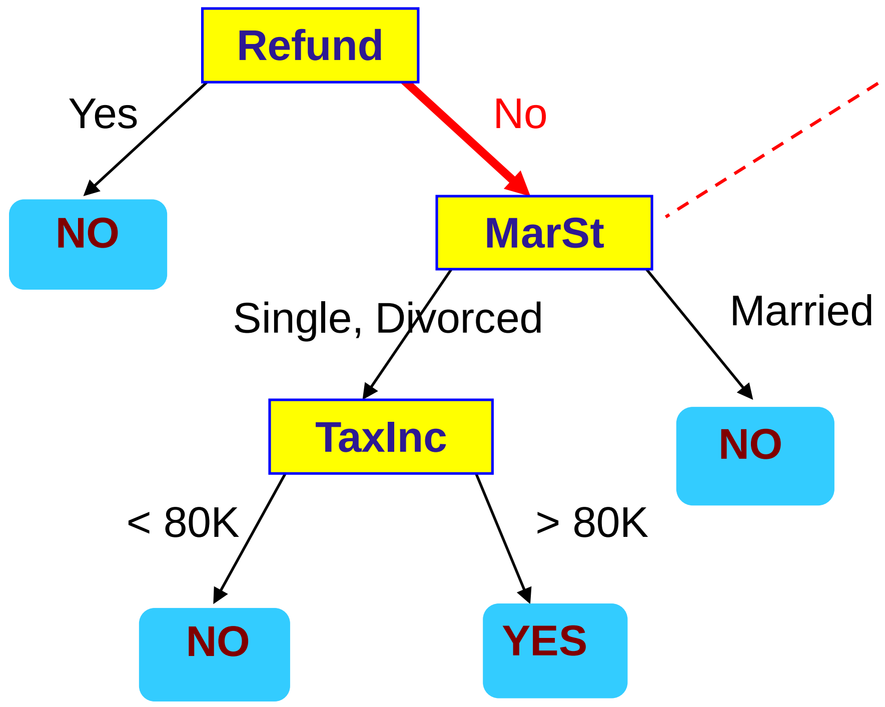
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

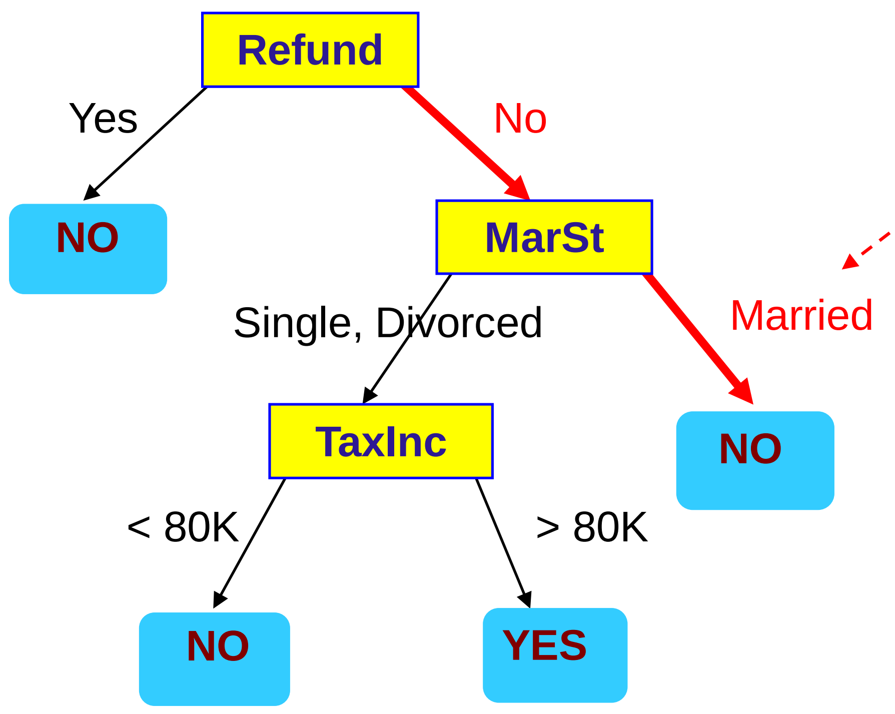
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

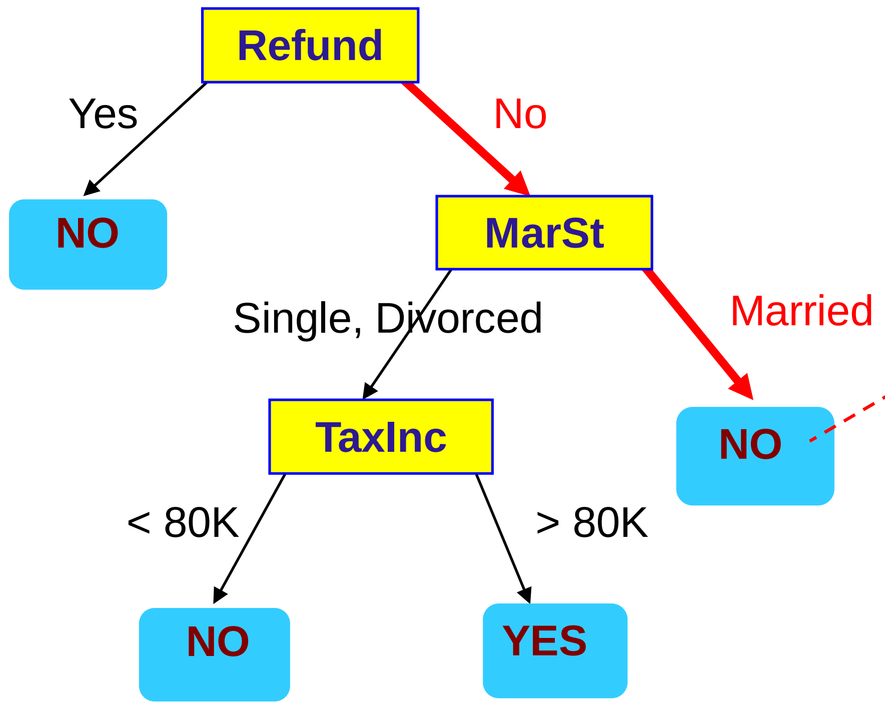
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

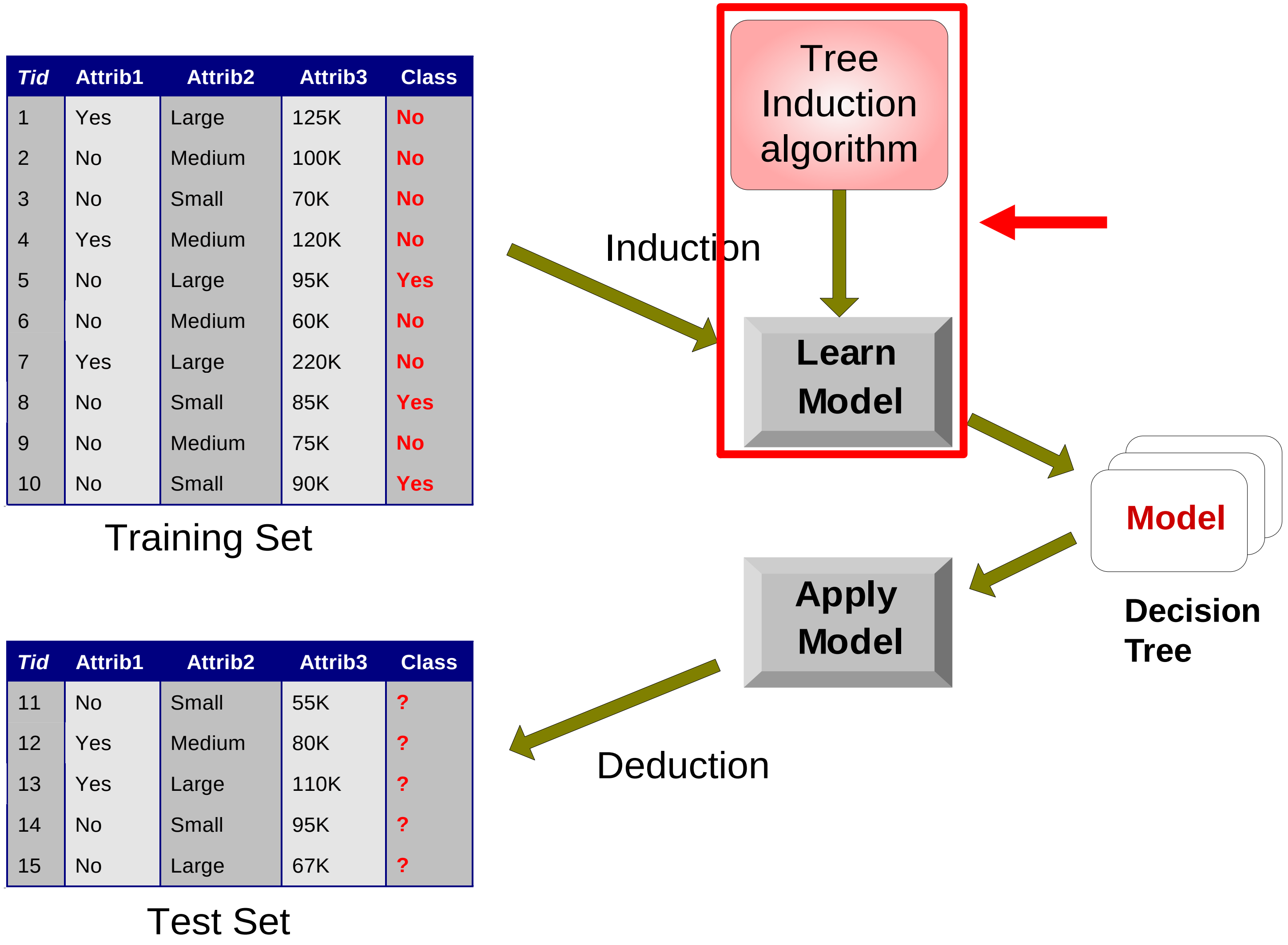


Test Data

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

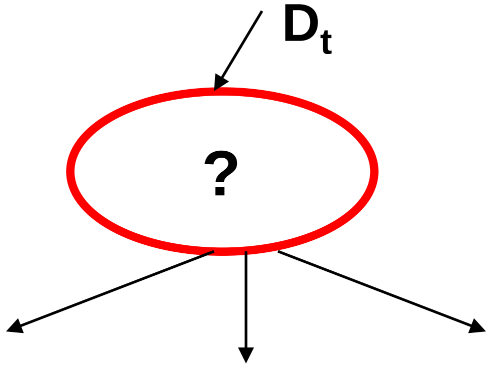
4.3.2 Decision Tree Induction

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

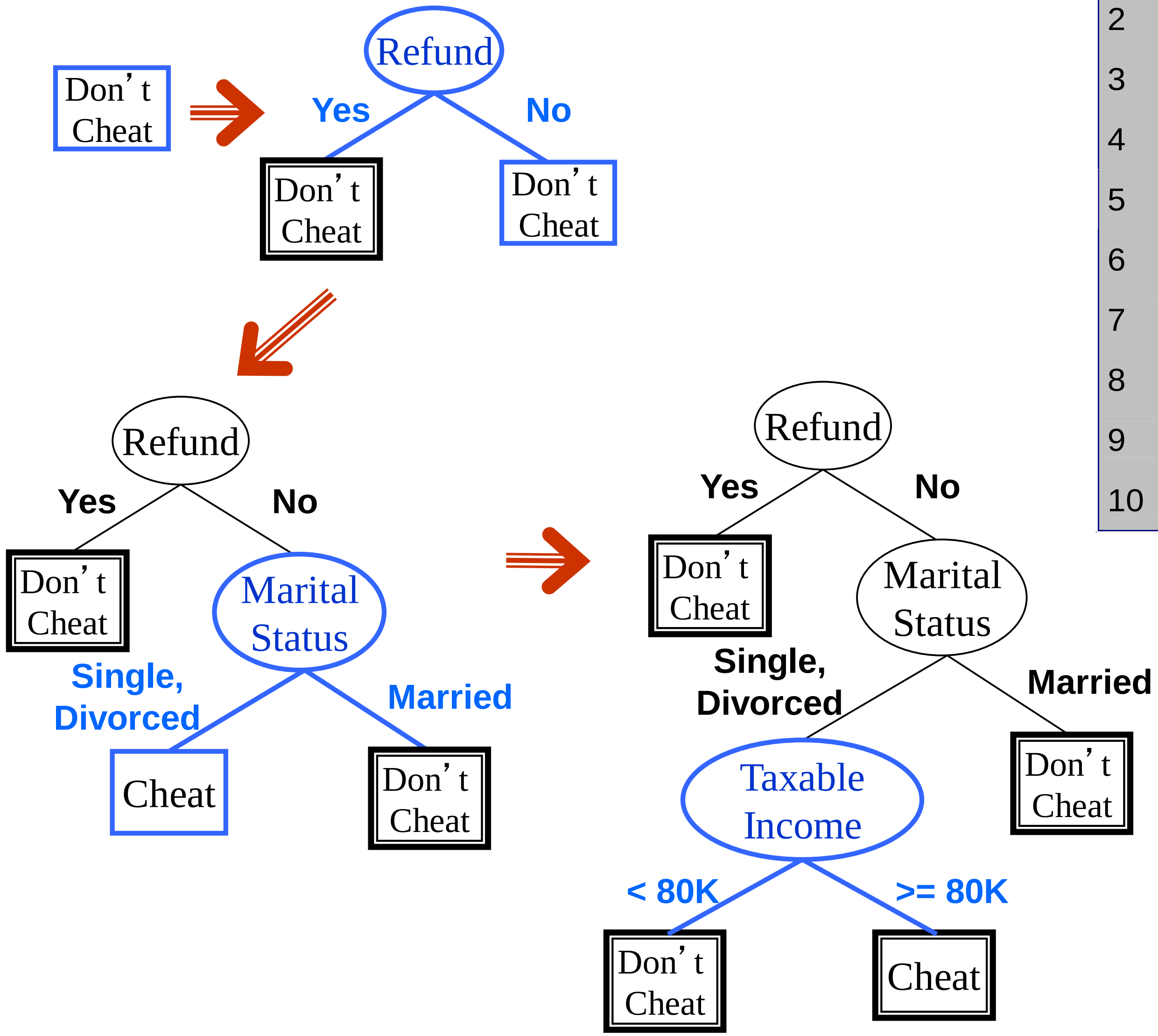
1. General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

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



Hunt's Algorithm



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

2. 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

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

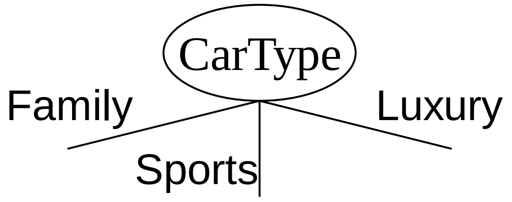
4.3.3 How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

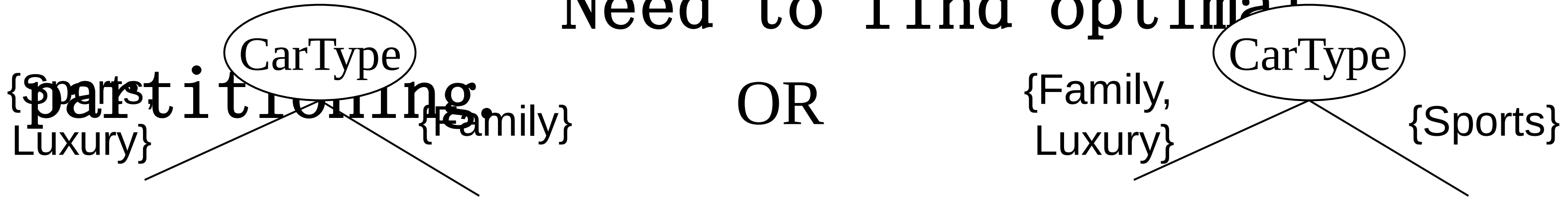
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

1. Splitting Based on Nominal Attributes

- **Multi-way split:** Use as many partitions as distinct values.

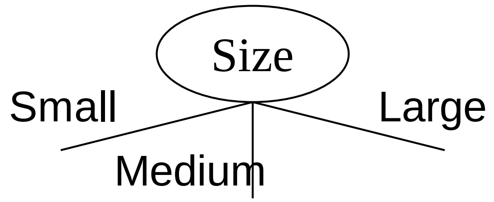


- **Binary split:** Divides values into two subsets. Need to find optimal partitioning.

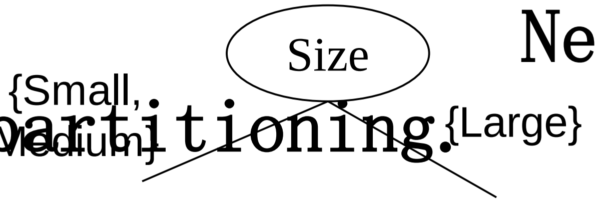


2. Splitting Based on Ordinal Attributes

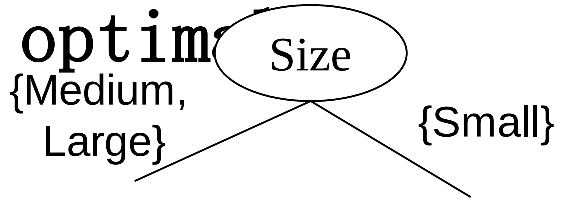
- Multi-way split: Use as many partitions as distinct values.



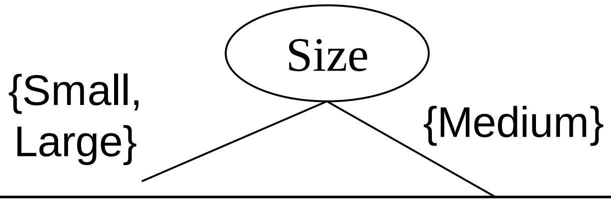
- Binary split: Divides values into two subsets. Need to find optimal partitioning.



OR



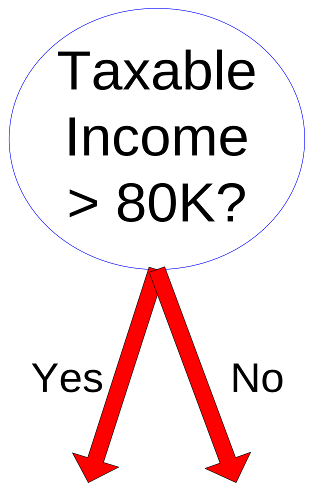
- What about this split?

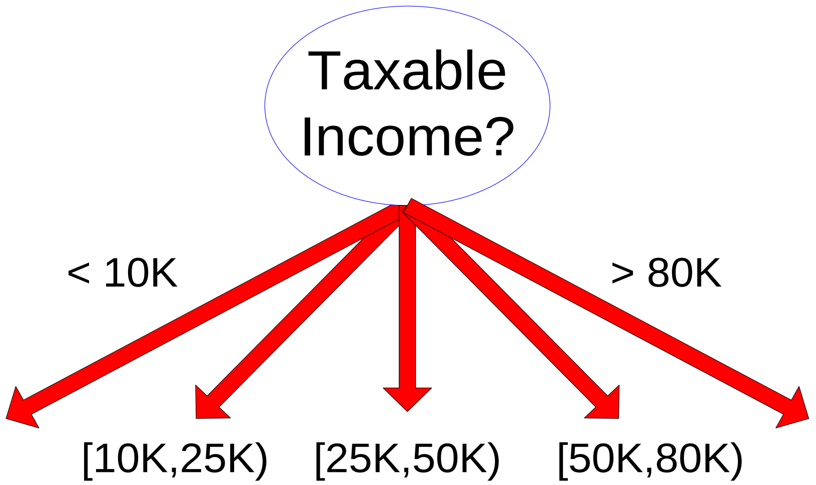


3. Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes





(i) Binary split

(ii) Multi-way split

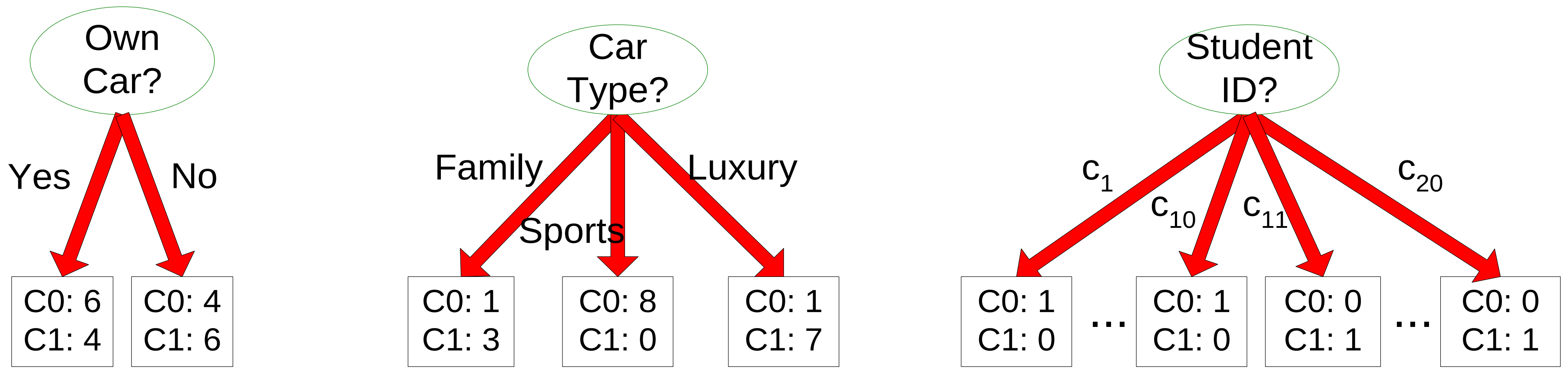
Tree Induction

- 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

4.3.4 How to determine the Best Split

Before Splitting: 10 records of class 0,
10 records of class 1



Which test condition is the best?

分裂属性选择

- 怎样减少不确定性?
  - 通过描述属性可以减少类标属性的不确定性。
  - 如：由番茄、茄子、黄瓜三种蔬菜，现在对蔬菜分类。在不给任何描述时，该蔬菜可能是番茄、茄子、黄瓜三种之一，不确定性大。给出该蔬菜是“长形”形状描述时，该蔬菜可能是茄子、黄瓜两种之一，不确定性减少。在给出该蔬菜是“紫的”颜色描述时，只可能是茄子了，不确定性为零。

1. How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous（同类的） class distribution are preferred
- Need a measure of node impurity（杂质，不纯性）：

C0: 5
C1: 5

C0: 9
C1: 1

Non-homogeneous,

High degree of impurity

Homogeneous,

Low degree of impurity

2. Measures of Node Impurity

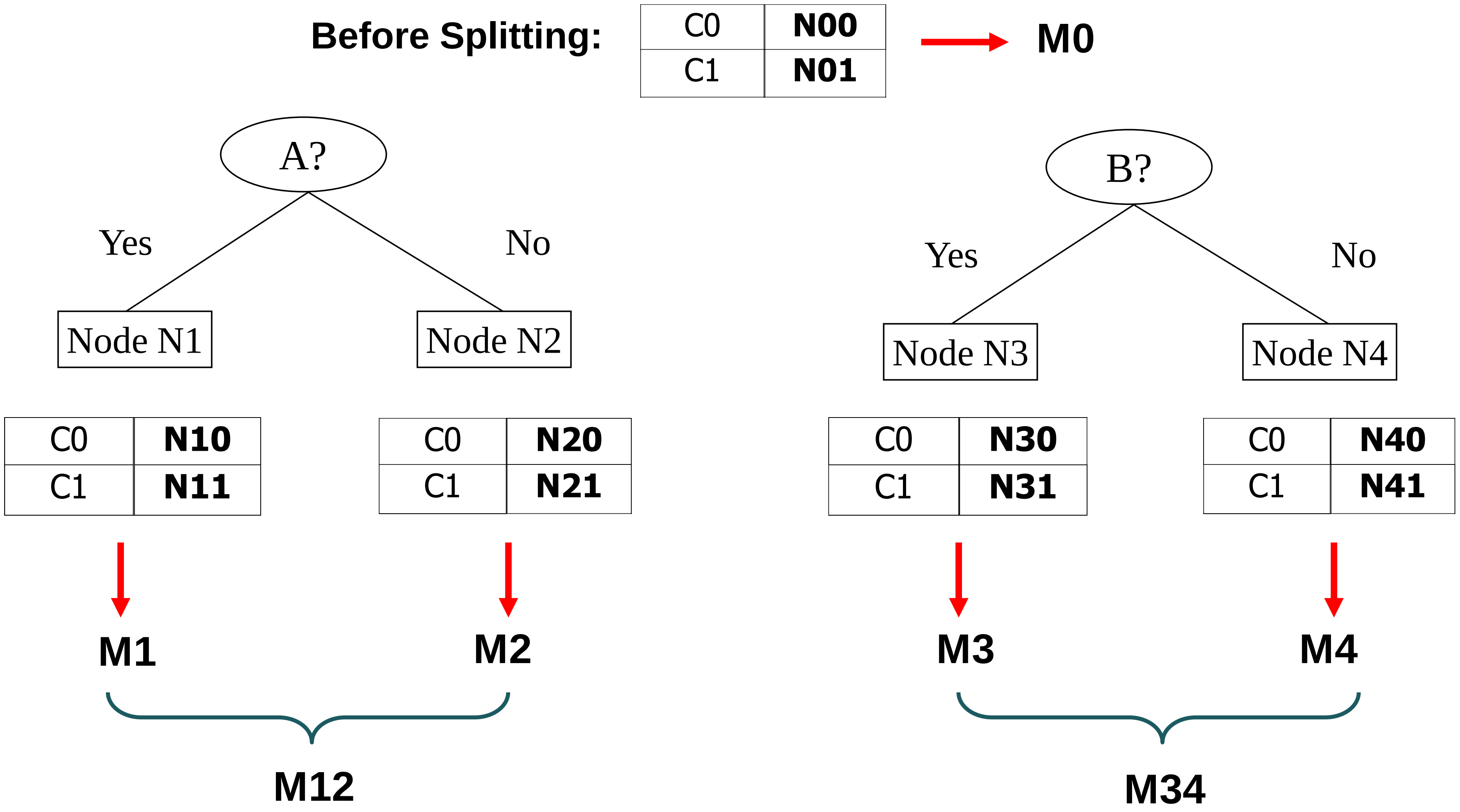
- Gini Index $$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i \mid t)]^2$$

- Entropy $$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i \mid t) \log_2 p(i \mid t)$$

- Misclassification err $$\text{Classification error}(t) = 1 - \max_i [p(i \mid t)]$$

设 $p(i|t)$表示给定结点 t 中属于类 i 的记录所占的比例

How to Find the Best Split



Gain = M0 – M12 vs M0 – M34

Measure of Impurity: GINI

- Gini Index for a given node t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

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

  – 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

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Gini = 1 – P(C1)^2 – P(C2)^2 = 1 – 0 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6   P(C2) = 5/6

Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278

C1	**2**
C2	**4**

P(C1) = 2/6   P(C2) = 4/6

Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444

Splitting Based on GINI

- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

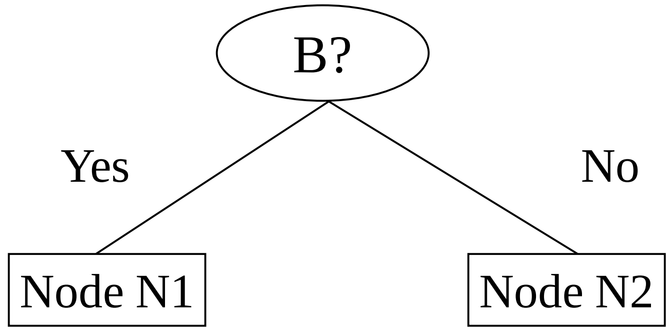
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,
n = number of records at node parent.

Binary Attributes: Computing GINI Index

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for.

	Parent
C1	6
C2	6
Gini = 0.500	



Gini(N1)
$= 1 – (5/7)^2 – (2/7)^2$
= 0.238

Gini(N2)
$= 1 – (1/5)^2 – (4/5)^2$
= 0.528

	N1	N2
C1	5	1
C2	2	4
Gini=0.371		

Gini(Children)
= 7/12 * 0.2380 +
5/12 * 0.0.32
= 0.371

Categorical Attributes: Computing Gini Index

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

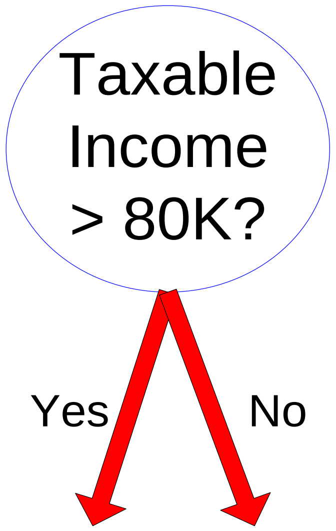
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

3. Continuous Attributes: Computing Gini Index

- Use Binary Decisions based on one value
- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values
- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, A < v and A ≥ v
- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work.

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



Continuous Attributes: Computing Gini Index...

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
	Taxable Income									
Sorted Values →	60	70	75	85	90	95	100	120	125	220

Split Positions →	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

I. 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 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

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

C1	**2**
C2	**4**

P(C1) = 2/6 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…why we need INFO

- Gain Ratio:

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

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

3. Splitting Criteria based on Classification Error

- Classification error at a node t :

$$\boxed{Error(t) = 1 - \max_{i} P(i \mid t)}$$

- 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

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

C1	**0**
C2	**6**

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6   P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

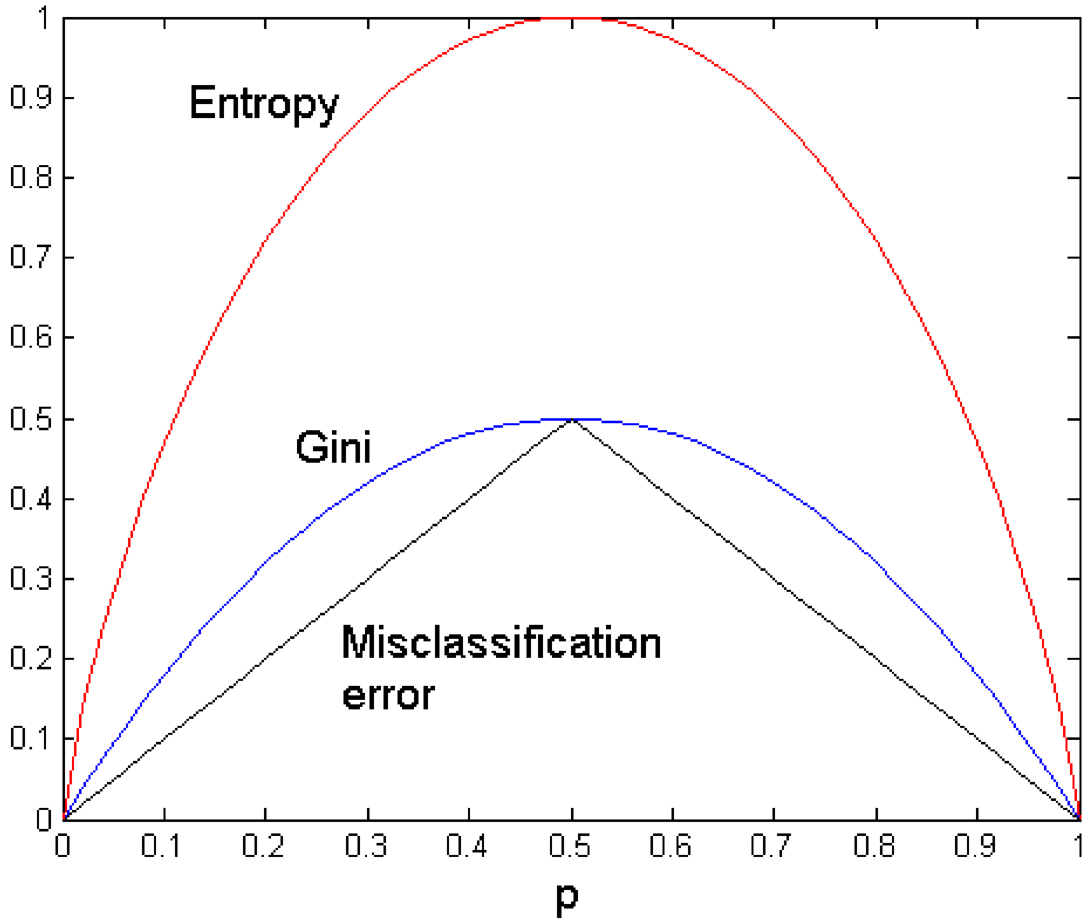
C1	**2**
C2	**4**

P(C1) = 2/6   P(C2) = 4/6

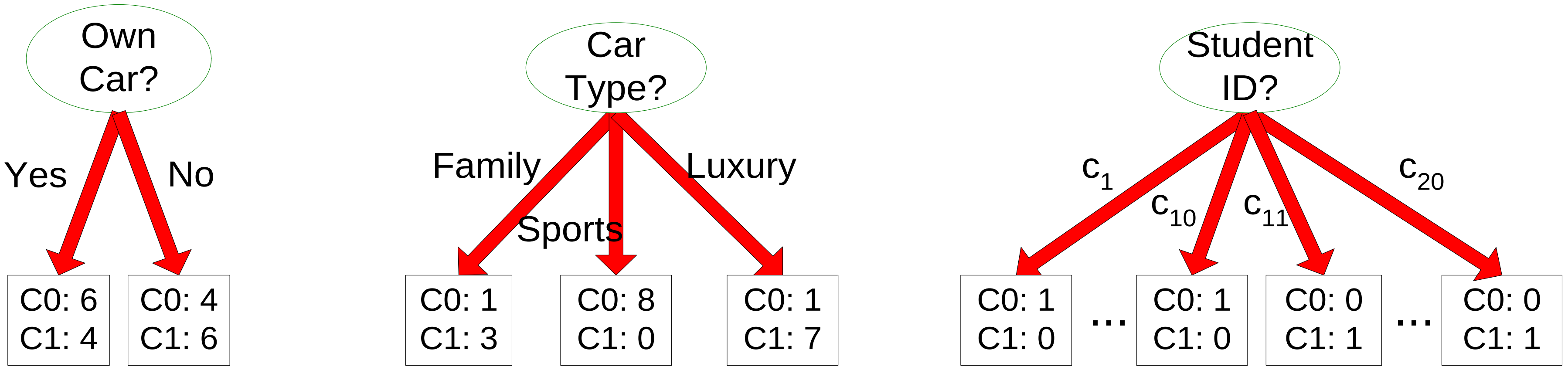
Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

5. Comparison among Splitting Criteria

For a 2-class problem:



Compare all the measures you have learned!



课后习题1: 生成决策树

计数	年龄	收入	学生	信誉	归类：买计算机?
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

课后习题2： 根据商场顾客数据库(训练样本集合)生成决策树

rid	age	income	student	credit_rating	buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30 – 40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30 – 40	Low	Yes	Excellent	Yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30 – 40	Medium	No	Excellent	Yes
13	30 – 40	High	Yes	Fair	Yes
14	>40	Medium	No	Excellent	No

3.3.5 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

决策树算法

ID3 决策树建立算法

1 决定分类属性；

2 对目前的数据表，建立一个节点N

3 如果数据库中的数据都属于同一个类，N就是树叶，在树叶上标出所属的类

4 如果数据表中没有其他属性可以考虑，则N也是树叶，按照少数服从多数的原则在树叶上标出所属类别

5 否则，根据平均信息期望值E或GAIN值选出一个最佳属性作为节点N的测试属性

6 节点属性选定后，对于该属性中的每个值：
 从N生成一个分支，并将数据表中与该分支有关的数据收集形成分支节点的数据表，在表中删除节点属性那一栏
 如果分支数据表非空，则运用以上算法从该节点建立子树。

ID3算法的性能分析

- ID3算法的假设空间包含所有的决策树，它是关于现有属性的有限离散值函数的一个完整空间。所以ID3算法避免了搜索不完整假设空间的一个主要风险：假设空间可能不包含目标函数。
- ID3算法在搜索的每一步都使用当前的所有训练样例，大大降低了对个别训练样例错误的敏感性。因此，通过修改终止准则，可以容易地扩展到处理含有噪声的训练数据。
- ID3算法在搜索过程中不进行回溯。所以，它易受无回溯的爬山搜索中的常见风险影响：收敛到局部最优而不是全局最优。

决策树算法

决策树的数据准备

原始表

姓名	年龄	收入	学生	信誉	电话	地址	邮编	买计算机
张三	23	4000	是	良	281-322-0328	2714 Ave. M	77388	买
李四	34	2800	否	优	713-239-7830	5606 Holly Cr	78766	买
王二	70	1900	否	优	281-242-3222	2000 Bell Blvd.	70244	不买
赵五	18	900	是	良	281-550-0544	100 Main Street	70244	买
刘兰	34	2500	否	优	713-239-7430	606 Holly Ct	78566	买
杨俊	27	8900	否	优	281-355-7990	233 Rice Blvd.	70388	不买
张毅	38	9500	否	优	281-556-0544	399 Sugar Rd.	78244	买
。。。	。。							
。。。								

决策树算法

决策树的数据准备

整理后的数据表

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
。。。					

- **Data cleaning**
  删除/减少**noise**，
  补填**missing values**
- **Data transformation**
  数据标准化（**data normalization**）
  数据归纳（**generalize data to higher-level concepts using concept hierarchies**）
  例如：年龄归纳为老、中、青三类
  控制每个属性的可能值不超过七种
  （最好不超过五种）
- **Relevance analysis**
  对于与问题无关的属性：删
  对于属性的可能值大于七种
  又不能归纳的属性：删

决策树算法

决策树的数据准备

处理连续属性值

决策树算法比较适合处理离散数值的属性。实际应用中属性是连续的或者离散的情况都比较常见。

在应用连续属性值时，在一个树结点可以将属性Ai的值划分为几个区间。然后信息增益的计算就可以采用和离散值处理一样的方法。原则上可以将Ai的属性划分为任意数目的空间。C4.5中采用的是二元分割（Binary Split）。需要找出一个合适的分割阈值。

参考C4.5算法 Top 10 algorithms in data mining
Knowledge Information System 2008 14:1–37

决策树算法

ID3算法小结

ID3算法是一种经典的决策树学习算法，由Quinlan于1979年提出。ID3算法的基本思想是，以信息熵为度量，用于决策树节点的属性选择，每次优先选取信息量最多的属性，亦即能使熵值变为最小的属性，以构造一颗熵值下降最快的决策树，到叶子节点处的熵值为0。此时，每个叶子节点对应的实例集中的实例属于同一类。

决策树算法

ID3算法实际应用-在电信行业应用实例（1）

通过ID3算法来实现客户流失的预警分析，找出客户流失的特征，以帮助电信公司有针对性地改善客户关系，避免客户流失

利用决策树方法进行数据挖掘，一般有如下步骤：数据预处理、决策树挖掘操作，模式评估和应用。

电信运营商的客户流失有三方面的含义：一是指客户从一个电信运营商转网到其他电信运营商，这是流失分析的重点。二是指客户月平均消费量降低，从高价值客户成为低价值客户。三、指客户自然流失和被动流失。

在客户流失分析中有两个核心变量：财务原因／非财务原因、主动流失／被动流失。

客户流失可以相应分为四种类型：其中非财务原因主动流失的客户往往是高价值的客户。他们会正常支付服务费用，并容易对市场活动有所响应。这种客户是电信企业真正需要保住的客户。

ID3算法实际应用-在电信行业应用实例（2）

数据预处理

数据挖掘的处理对象是大量的数据，这些数据一般存储在数据库系统中（该用户相关数据存储在其CRM中），是长期积累的结果。但往往不适合直接挖掘，需要做数据的预处理工作，一般包括数据的选择(选择相关的数据)、净化(消除冗余数据)、转换、归约等。数据预处理工作准备是否充分，对于挖掘算法的效率乃至正确性都有关键性的影响。

该公司经过多年的电脑化管理，已有大量的客户个人基本信息(文中简称为客户信息表)。在客户信息表中，有很多属性，如姓名用户号码、用户标识、用户身份证号码(转化为年龄)、在网时间（竣工时间）、地址、职业、用户类别、客户流失（用户状态）等等，数据准备时必须除掉表中一些不必要的属性，一般可采用面向属性的归纳等方法去掉不相关或弱相关属性。

决策树算法

ID3算法实际应用-在电信行业应用实例（3）

属性删除：将有大量不同取值且无概化操作符的属性或者可用其它属性来代替它的较高层概念的那些属性删除。比如客户信息表中的用户标识、身份证号码等，它们的取值太多且无法在该取值域内找到概化操作符，应将其删除，得到表1。

表1　客户信息表

年龄	学历	职业	缴费方式	在网时长	费用变化率	客户流失
58	大学	公务员	托收	13	10%	NO
47	高中	工人	营业厅缴费	9	42%	NO
26	研究生	公务员	充值卡	2	63%	YES
28	大学	公务员	营业厅缴费	5	2.91%	NO
32	初中	工人	营业厅缴费	3	2.3%	NO
42	高中	无业人员	充值卡	2	100%	YES
68	初中	无业人员	营业厅缴费	9	2.3%	NO

决策树算法

ID3算法实际应用-在电信行业应用实例（4）

属性概化：用属性概化阈值控制技术沿属性概念分层上卷或下钻进行概化。文化程度分为3类：W1初中以下(含初中)，W2高中(含中专)，W3大学(专科、本科及以上)；职业类别：按工作性质来分共分3类：Z1一Z3；
缴费方式：托收：T1,营业厅缴费：T2,充值卡：T3。

连续型属性概化为区间值：表中年龄、费用变化率和在网时间为连续型数据，由于建立决策树时，用离散型数据进行处理速度最快，因此对连续型数据进行离散化处理，根据专家经验和实际计算信息增益，在“在网时长”属性中，通过检测每个划分，得到在阈值为5年时信息增益最大，从而确定最好的划分是在5年处，则这个属性的范围就变为｛<=5,>5：H1,H2｝。而在“年龄”属性中，信息增益有两个锋值，分别在40和50处，因而该属性的范围变为{<=40,>40-<=50,>50}即变为{青年，中年，老年：N1,N2,N3}；费用变化率：指（（当月话费－近3个月的平均话费）/近3个月的平均话费）×%>0，F1:<＝30%，F2：30%-99%, F3:＝100%变为｛F1,F2,F3｝。

决策树算法

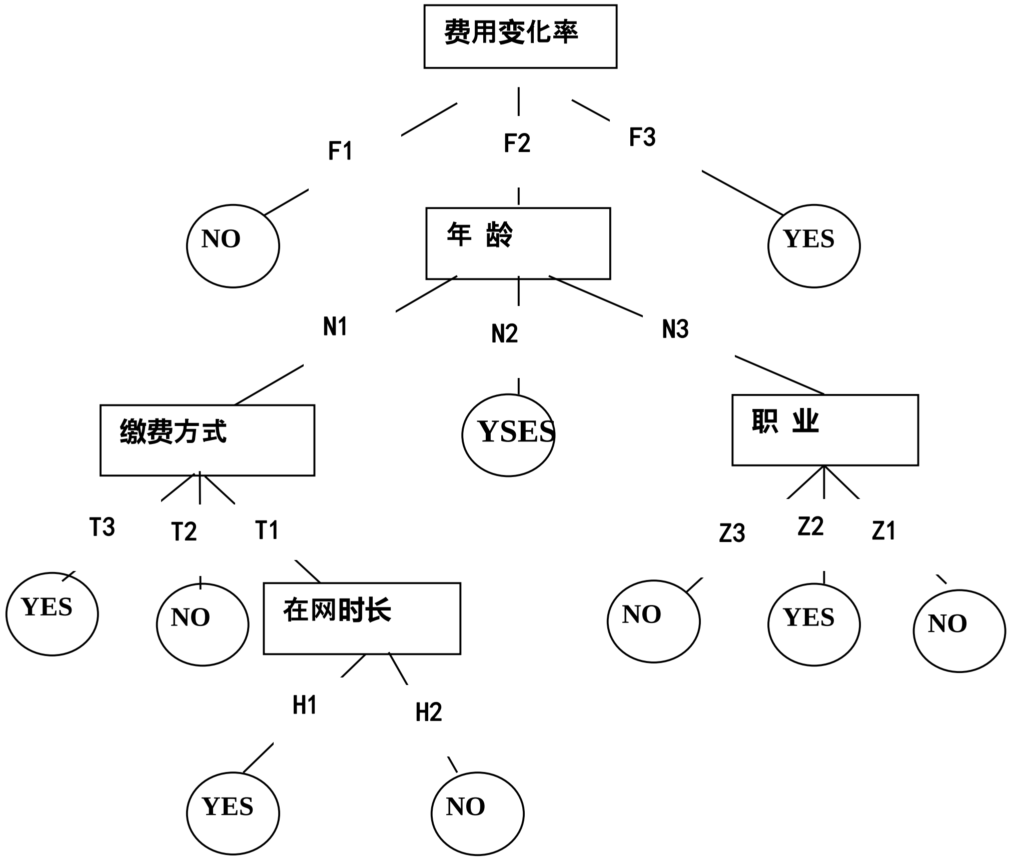
ID3算法实际应用-在电信行业应用实例（5）

表2　转化后的客户信息表

年龄	学历	职业	缴费方式	开户时间	费用变化率	客户流失
N3	W3	Z1	T1	H2	F1	NO
N2	W2	Z2	T2	H2	F2	NO
N1	W3	Z1	T3	H1	F2	YES
N1	W3	Z1	T2	H1	F1	NO
N1	W1	Z2	T2	H1	F1	NO
N2	W2	Z3	T3	H1	F3	YES
N3	W1	Z3	T1	H2	F1	NO

决策树算法

ID3算法实际应用-在电信行业应用实例（6）



在图中，NO表示客户不流失，YES表示客户流失。从图可以看出，客户费用变化率为100%的客户肯定已经流失；而费用变化率低于30%的客户；即每月资费相对稳定的客户一般不会流失，费用变化率在30%～99%的客户有可能流失，其中年龄在40～50岁之间的客户流失的可能性非常大，而年龄低于40岁的客户，用充值卡缴费的客户和在网时间较短的客户容易流失；年龄较大的客户，则工人容易流失。

C4.5算法对ID3的主要改进

- C4.5算法是从ID3算法演变而来，除了拥有ID3算法的功能外， C4.5克服了ID3在应用中的不足，主要体现在：
  - 用信息增益比例/信息增益率来选择属性，克服了用信息增益选择属性时偏向于选择取值多的属性的不足；
  - 能够完成对连续属性的离散化处理；
  - 可以处理具有缺少属性值的训练样本；
  - 在数构造过程中或者构造完成之后，进行剪枝以避免树的过度拟合；
  - C4.5采用的知识表示形式为决策树，并最终可以形成产生式规则。

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

C4.5算法例子

样本数据

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	85	false	No
Sunny	Hot	90	true	No
Overcast	Hot	78	false	Yes
Rain	Mild	96	false	Yes
Rain	Cool	80	false	Yes
Rain	Cool	70	true	No
Overcast	Cool	65	true	Yes
Sunny	Mild	95	false	No
Sunny	Cool	70	false	Yes
Rain	Mild	80	false	Yes
Sunny	Mild	70	true	Yes
Overcast	Mild	90	true	Yes
Overcast	Hot	75	false	Yes
Rain	Mild	80	true	No

1）首先对*Humidity*进行属性离散化，针对上面的训练集合，通过检测每个划分而确定最好的划分在75处，则这个属性的范围就变为{（<=75，>75）}。

2）计算目标属性*PlayTennis*分类的期望信息：

$$I(s_1, s_2) = I(9,5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

3）计算每个属性的GainRatio：

$$GainRatio\ (\text{Outlook}) = \frac{0.246}{1.577} = 0.156;$$

$$GainRatio\ (\text{windy}) = 0.049;$$

$$GainRatio\ (\text{Temperatu re}) = 0.0248;$$

$$GainRatio\ (\text{Humidity}) = 0.0483$$

4）选取最大的*GainRatio*，根据*Outlook*的取值，得到三个分支。

5）再扩展各分枝节点，得到最终的决策树。

4.4 Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

决策树研究问题

理想的决策树有三种：

(1)叶子结点数最少；

(2)叶子结点深度最小；

(3)叶子结点数最少且叶子结点深度最小。

然而，洪家荣等人已经证明了要找到这种最优的决策树是NP难题。因此，决策树优化的目的就是要找到尽可能趋向于最优的决策树。

决策树研究问题

关于过渡拟合

　　上述的决策树算法增长树的每一个分支的深度，直到恰好能对训练样例比较完美地分类。实际应用中，当数据中有噪声或训练样例的数量太少以至于不能产生目标函数的有代表性的采样时，该策略可能会遇到困难。

　　在以上情况发生时，这个简单的算法产生的树会过渡拟合训练样例（过渡拟合：Over Fitting）.

决策树研究问题

关于过渡拟合

对于一个假设，当存在其它的假设对训练样例的拟合比它差，但事实上在实例的整个分布上（包含训练集合以外的实例）表现得却更好时，则称该假设过度拟合训练样例。

过度拟合：给定一个假设空间H，一个假设h∈H，如果存在其它的假设h1 ∈H ,使得在训练样例上h的错误率比h1小，但在整个实例发布上h1的错误率比h小，则称假设h过度拟合训练数据

过度拟合产生的原因：噪声，训练样例太小等

决策树研究问题

关于过渡拟合

对学习算法是否成功的真正测试是看它对于训练中未见到的数据的执行性能。

训练过程应该包含训练样本和验证样本。验证样本用于测试训练后的性能。如果验证结果差，则需要考虑采用不同的结构重新进行训练，例如使用更大的样本集，或者改变从连续值到离散值得数据转换等。

通常应该建立一个验证过程，在训练最终完成后用来检测训练结果的泛化能力。

决策树研究问题

关于过渡拟合

分类模型的误差

一般可以将分类模型的误差分为:

1、训练误差（Training Error）；

2、泛化误差（Generalization Error）

决策树研究问题

关于过渡拟合

分类模型的误差

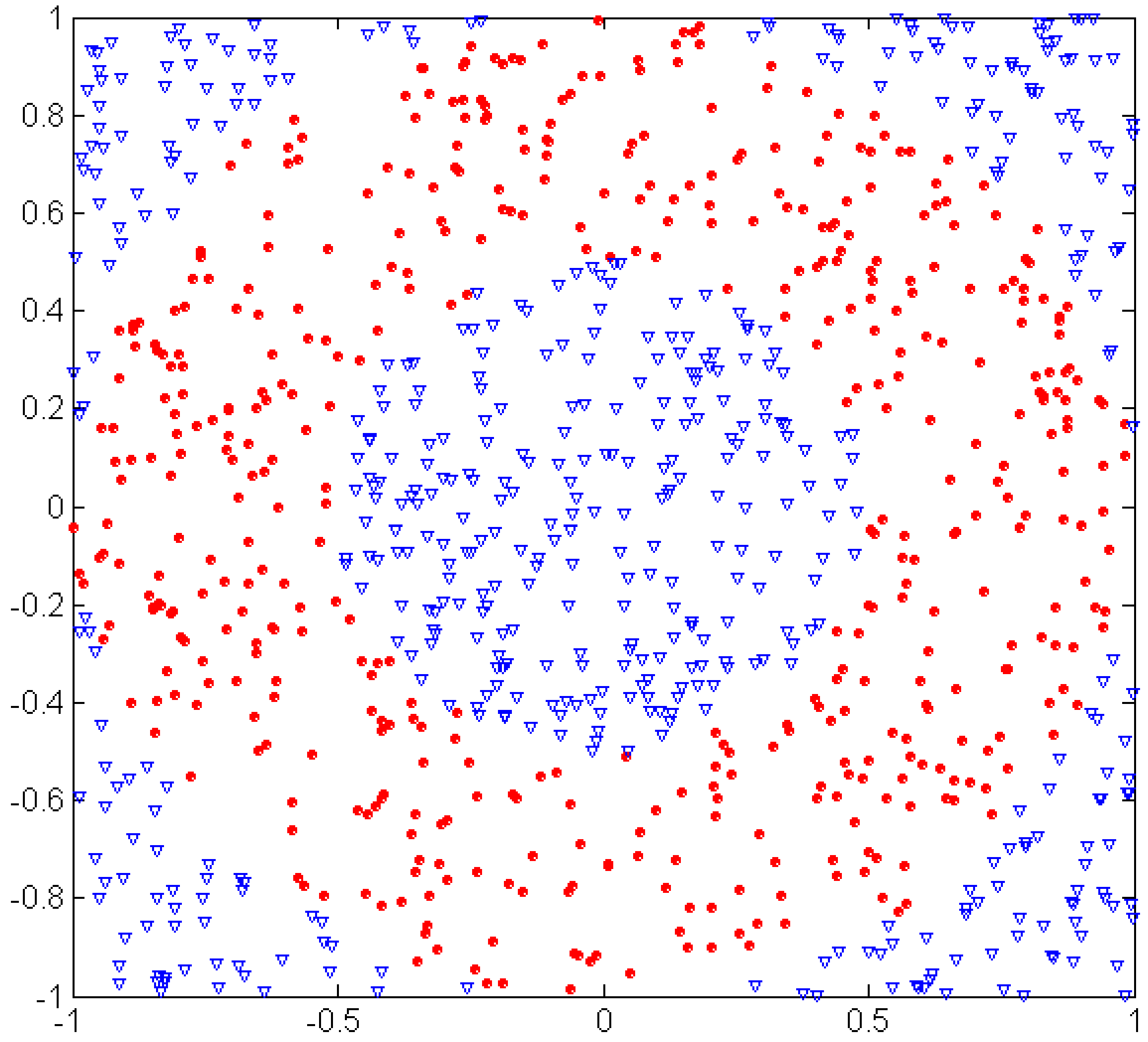
训练误差是在训练记录上误分类样本比例；

泛化误差是模型在未知记录上的期望误差；

一个好的模型不仅要能够很好地拟合训练数据，而且对未知样本也要能够准确地分类。

一个好的分类模型必须具有低的训练误差和泛化误差。因为一个具有低训练误差的模型，其泛化误差可能比具有较高训练误差的模型高。（训练误差低，泛化误差高，称为过渡拟合）

Underfitting and Overfitting (Example)



500 circular and 500 triangular data points.

Circular points:

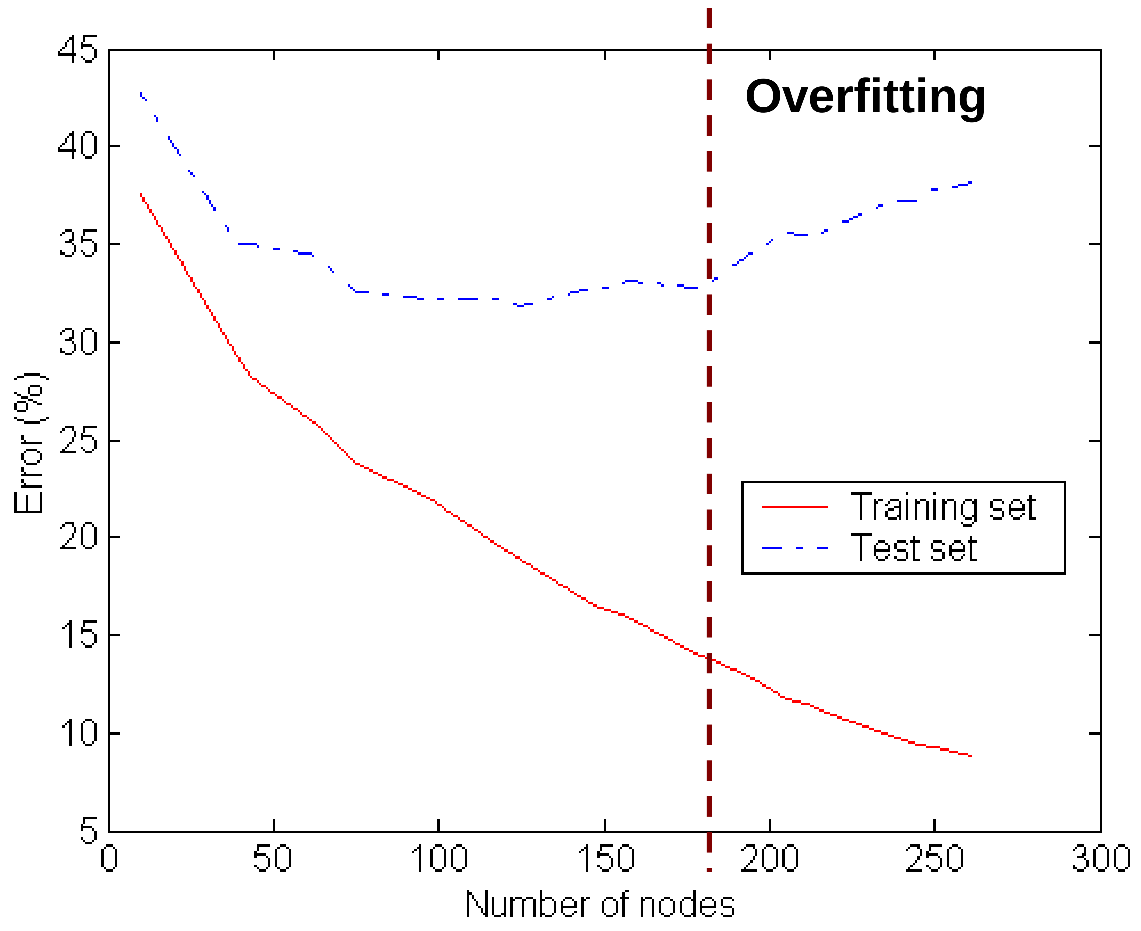
$0.5 \le \mathrm{sqrt}(x_1^2+x_2^2) \le 1$

Triangular points:

$\mathrm{sqrt}(x_1^2+x_2^2) > 0.5$ or

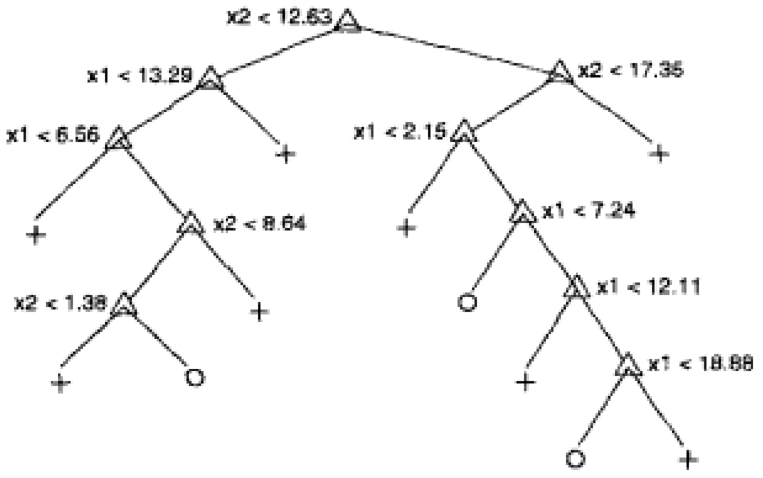
$\mathrm{sqrt}(x_1^2+x_2^2) < 1$

Underfitting and Overfitting

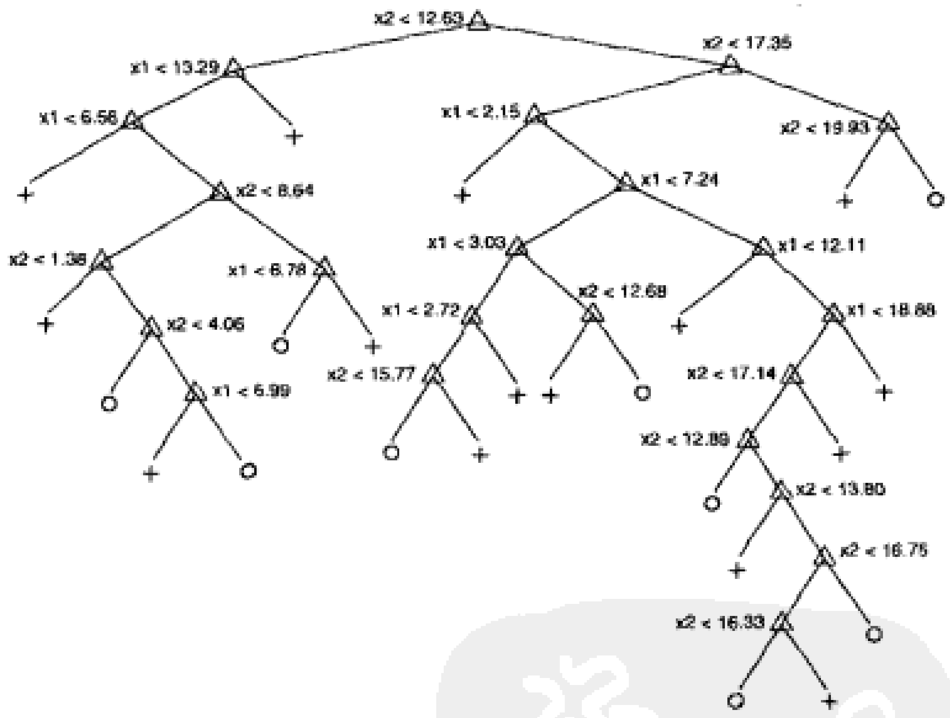


Underfitting: when model is too simple, both training and test errors are large

不同模型复杂度的决策树



(a) 包含 11 个叶结点的决策树



(b) 包含 24 个叶结点的决策树

图 4-24 具有不同模型复杂度的决策树

决策树研究问题

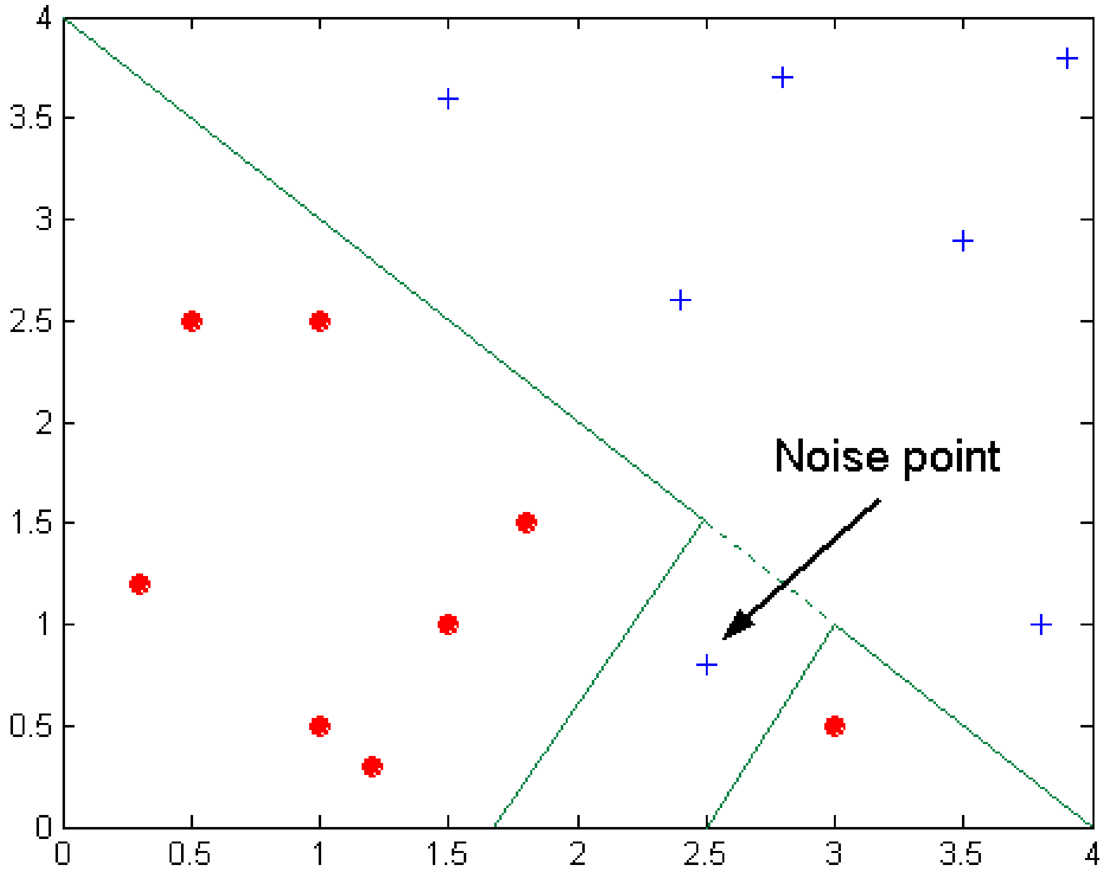
关于过渡拟合

模型过渡拟合的潜在因素

（1）噪声导致的过渡拟合;
错误的类别值/类标签，属性值等

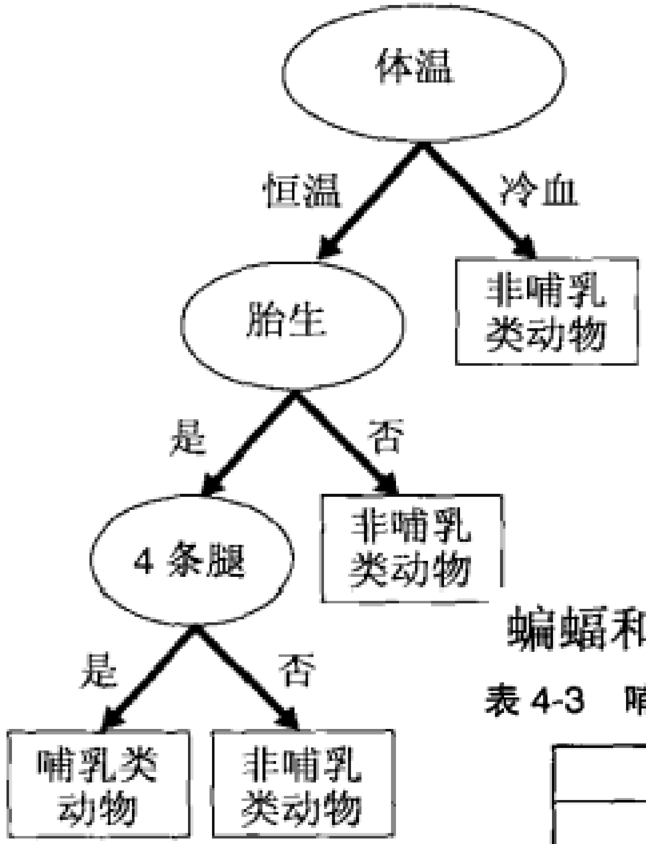
（2）缺乏代表性样本所导致的过渡拟合
根据少量训练记录作出的分类决策模型容易受过渡拟合的影响。由于训练样本缺乏代表性的样本，在没有多少训练记录的情况下，学习算法仍然继续细化模型就会导致过渡拟合。

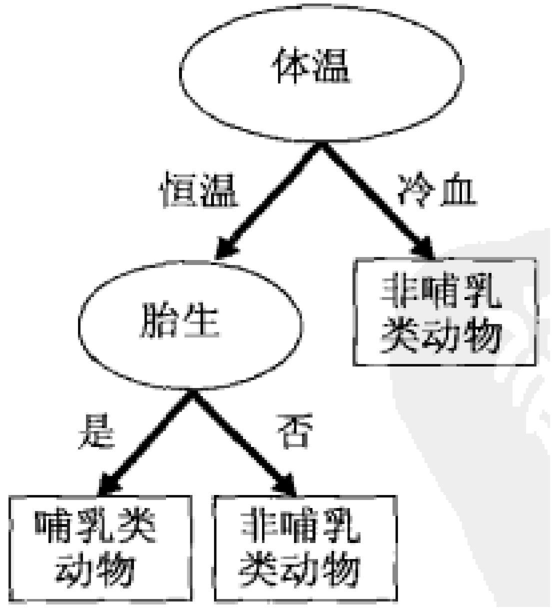
4.4.1Overfitting due to Noise



Decision boundary is distorted by noise point

噪声导致的过分拟合举例



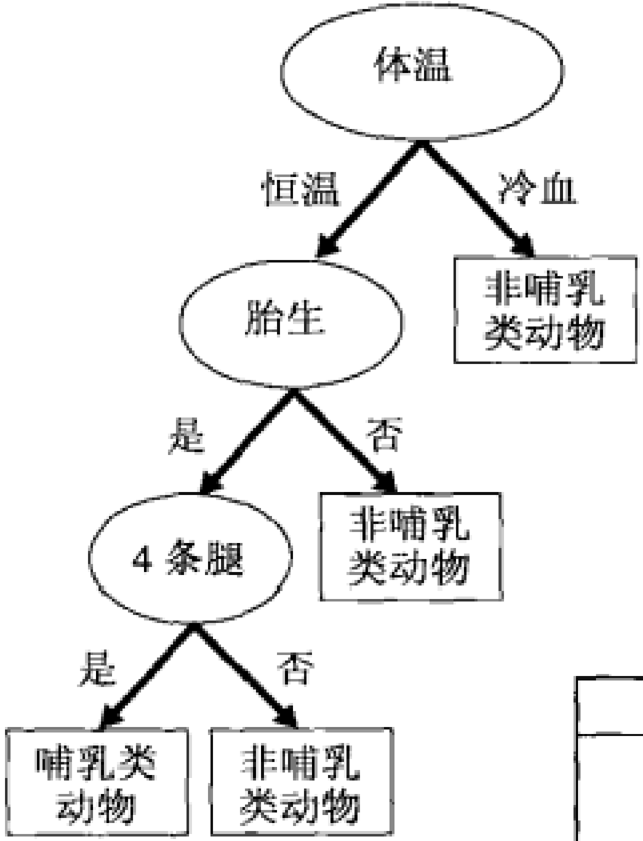


蝙蝠和鲸被错误地标记为非哺乳类动物，而不是哺乳类动物

表 4-3 哺乳类动物分类的训练数据集样本。打星号的类标号代表错误标记的记录

名称	体温	胎生	4 条腿	冬眠	类标号
豪猪	恒温	是	是	是	是
猫	恒温	是	是	否	是
蝙蝠	恒温	是	否	是	否*
鲸	恒温	是	否	否	否*
蝾螈	冷血	否	是	是	否
科莫多巨蜥	冷血	否	是	否	否
蟒蛇	冷血	否	否	是	否
鲑鱼	冷血	否	否	否	否
鹰	恒温	否	否	否	否
虹鳉	冷血	是	否	否	否

噪声导致的过分拟合举例2



人和海豚

针鼹:

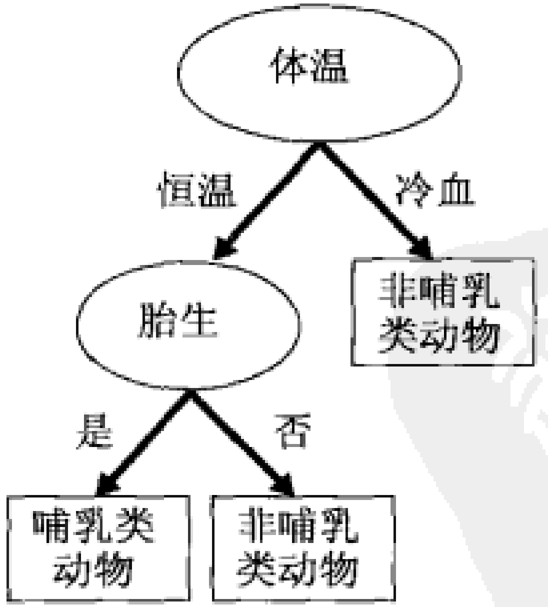
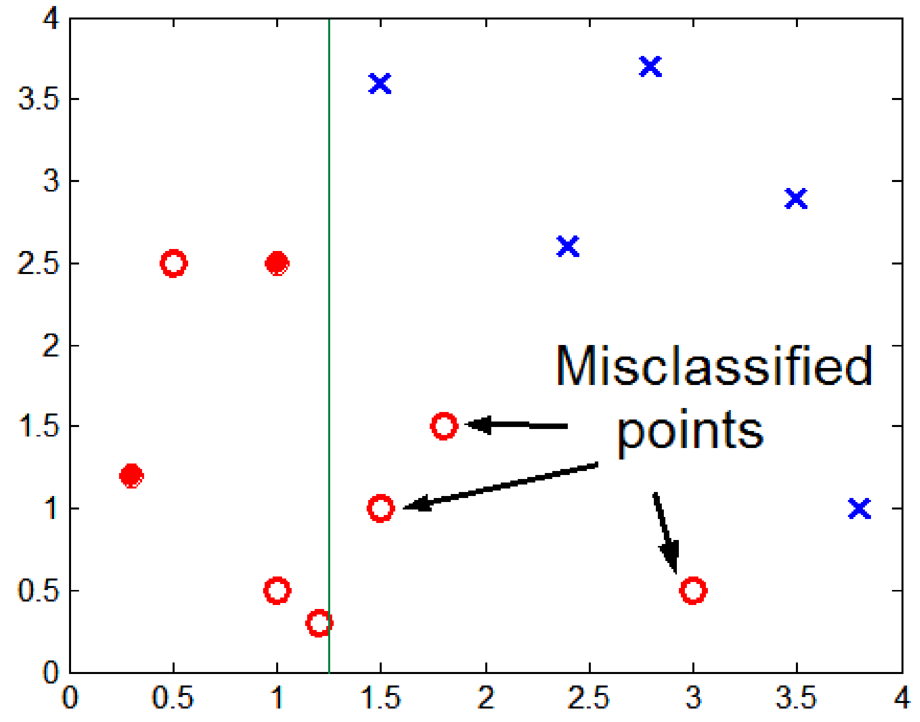


表 4-4　哺乳类动物分类的检验数据集样本

名称	体温	胎生	4 条腿	冬眠	类标号
人	恒温	是	否	否	是
鸽子	恒温	否	否	否	否
象	恒温	是	是	否	是
豹纹鲨	冷血	是	否	否	否
海龟	冷血	否	是	否	否
企鹅	冷血	否	否	否	否
鳗	冷血	否	否	否	否
海豚	恒温	是	否	否	是
针鼹	恒温	否	是	是	是
希拉毒蜥	冷血	否	是	是	否

4.4.2 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

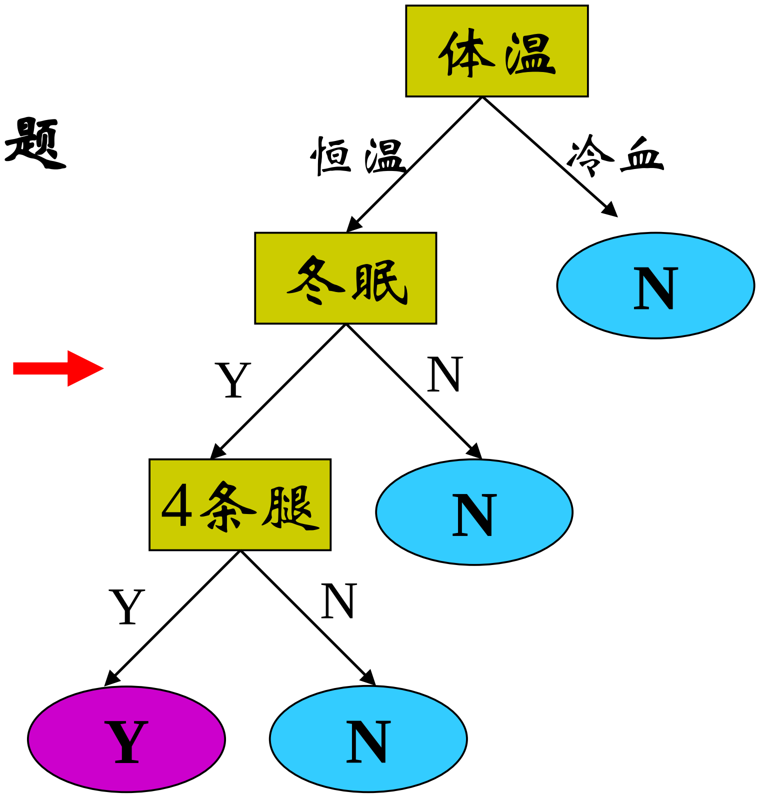
- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

关于过渡拟合　　　　决策树研究问题

模型过渡拟合的潜在因素

哺乳动物分类的训练样例

名称	体温	胎生	4条腿	冬眠	哺乳动物
蝾螈	冷血	N	Y	Y	N
虹鳉	冷血	Y	N	N	N
鹰	恒温	N	N	N	N
弱夜鹰	恒温	N	N	Y	N
鸭嘴兽	恒温	Y	Y	Y	Y



哺乳动物分类的测试样例

名称	体温	胎生	4条腿	冬眠	哺乳动物
人	恒温	Y	N	N	Y
大象	恒温	Y	Y	N	Y
鸽子	恒温	N	N	N	N

按照训练模型。人和大象都不是哺乳动物。决策树作出这样的判断是因为只有一个训练样例具有这些特点（鹰，恒温，不冬眠）被划分为非哺乳动物。

该例清楚表明，当决策树的叶节点没有足够的代表性时，可能会预测错误。

4.4.3 过分拟合与多重比较过程

为了理解多重比较过程，考虑预测未来 10 个交易日股市是升还是降的任务。

每个分析家做出 8 次正确预测的概率

$$\frac{C_{10}^{8}+C_{10}^{9}+C_{10}^{10}}{2^{10}}=0.0547$$

假设我们想从 50 个股票分析家中选择一个投资顾问，策略是选择在未来的 10 个交易日做出最多正确预测的分析家。该策略的缺点是，即使所有的分析家都用随机猜测做出预测，至少有一个分析家做出 8 次正确预测的概率是：

大量的候选属性和少量的训练记录最后导致了模型的过分拟合。

设 T_0 是初始决策树，T_x 是插入属性 x 的内部结点后的决策树。原则上，如果观察到的增益 $\Delta(T_0, T_x)$大于某个预先定义的阈值α，就可以将 x 添加到树中。如果只有一个属性测试条件，则可以通过选择足够大的阈值α来避免插入错误的结点。然而，在实践中，可用的属性测试条件不止一个，并且决策树算法必须从候选集$\{x_1, x_2,\cdots, x_k\}$中选择最佳属性 $x_{\max}$ 来划分数据。在这种情况下，算法实际上是使用多重比较过程来决定是否需要扩展决策树。更具体地说，这是测试 $\Delta(T_0, T_{x_{\max}})$ $> \alpha$，而不是测试$\Delta(T_0, T_x) > \alpha$。随着候选个数 k 的增加，找到 $\Delta(T_0, T_{x_{\max}}) > \alpha$ 的几率也在增大。除非根据 k 修改增益函数Δ或阈值α，否则算法会不经意间在模型上增加一些欺骗性的结点，导致模型过分拟合。

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

4.4.4 Estimating Generalization Errors

- Re-substitution errors(再带入误差，训练误差)：
  - error on training (Σ e(t))
- Generalization errors (泛化误差)：
  - error on testing (Σ e’ (t))
- Methods for estimating generalization errors:
  - Optimistic approach（乐观法）： e’ (t) = e(t)
  - Pessimistic approach: (悲观法)
    - ◆ For each leaf node: e’ (t) = (e(t)+0.5)
    - ◆ Total errors: e’ (T) = e(T) + N × 0.5 (N: number of leaf nodes)
    - ◆ For a tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):

      Training error = 10/1000 = 1%

      Generalization error = (10 + 30×0.5)/1000 = 2.5%
  - Reduced error pruning (REP):
    - ◆ uses validation data set to estimate generalization error

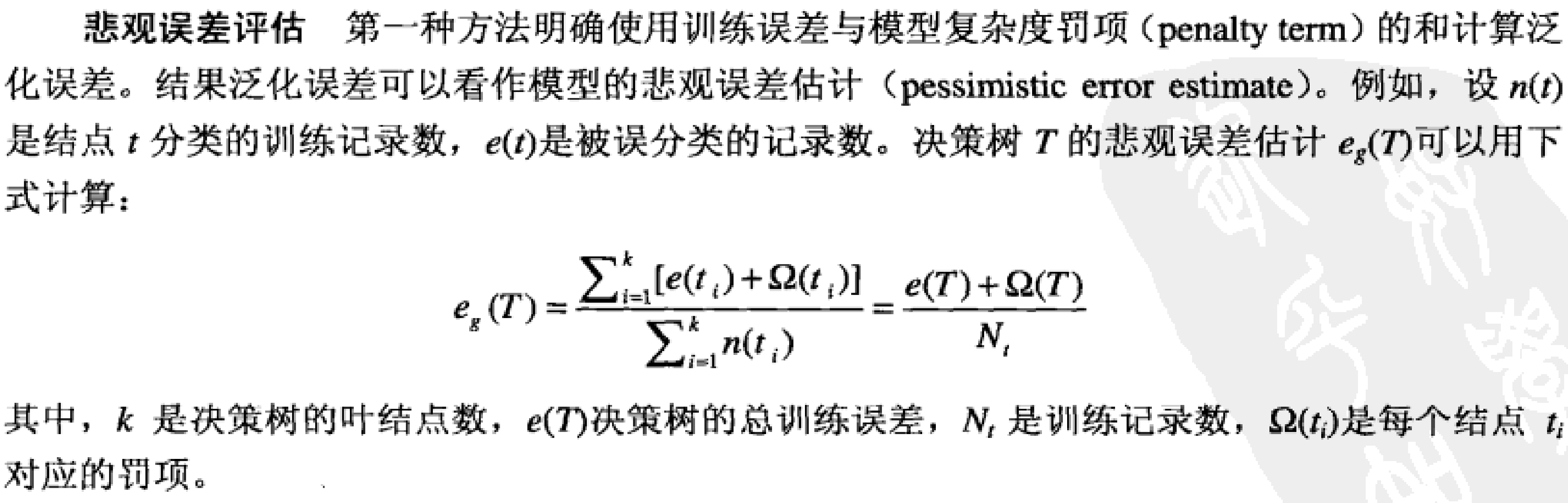
Occam's Razor 奥卡姆剃刀

- 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

1. 悲观误差评估

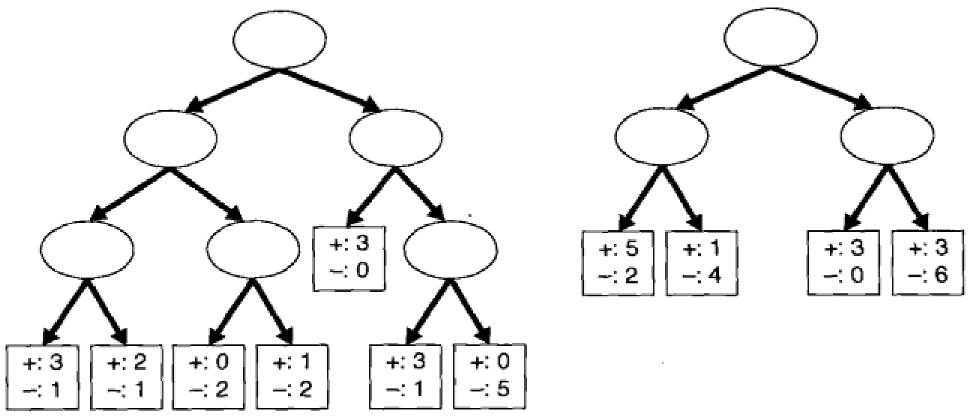


悲观误差评估　第一种方法明确使用训练误差与模型复杂度罚项（penalty term）的和计算泛化误差。结果泛化误差可以看作模型的悲观误差估计（pessimistic error estimate）。例如，设 $n(t)$ 是结点 t 分类的训练记录数，$e(t)$是被误分类的记录数。决策树 T 的悲观误差估计 $e_g(T)$可以用下式计算：

$$e_g(T)=\frac{\sum_{i=1}^{k}[e(t_i)+\Omega(t_i)]}{\sum_{i=1}^{k}n(t_i)}=\frac{e(T)+\Omega(T)}{N_t}$$

其中，k 是决策树的叶结点数，$e(T)$决策树的总训练误差，N_t 是训练记录数，$\Omega(t_i)$是每个结点 t_i 对应的罚项。

Example 1



决策树 T_L　　决策树 T_R

例 4.2　考虑图 4-27 中的二叉决策树。如果罚项等于 0.5，左边的决策树的悲观误差估计为：

$$e_g(T_L) = \frac{4+7\times0.5}{24} = \frac{7.5}{24} = 0.3125$$

右边的决策树的悲观误差估计为：

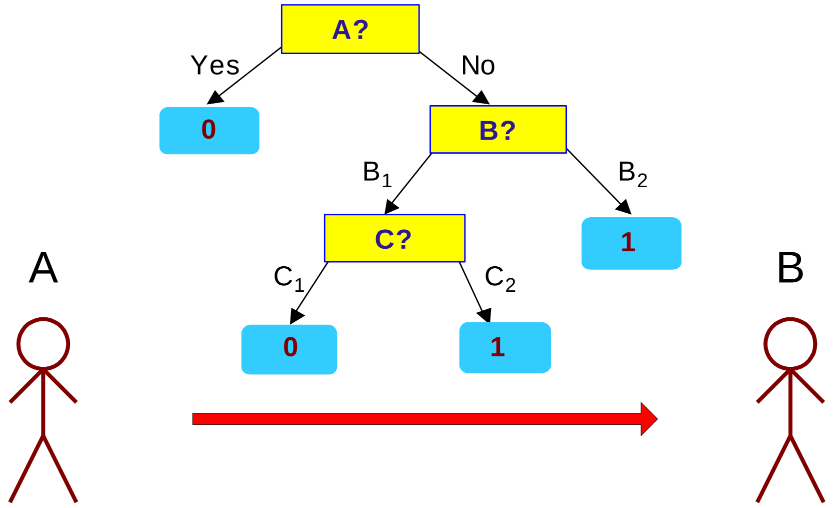
$$e_g(T_R) = \frac{6+4\times0.5}{24} = \frac{8}{24} = 0.3333$$

这样，左边的决策树比右边的决策树具有更好的悲观误差估计。对二叉树来说，0.5 的罚项意味着只要至少能够改善一个训练记录的分类，结点就应当扩展，因为扩展一个结点等价于总误差增加 0.5，代价比犯一个训练错误小。

如果对于所有的结点 t，$\Omega(t) = 1$，左边的决策树的悲观误差估计为 $e_g(T_L) = 11/24 = 0.458$，右边的决策树的悲观误差估计为 $e_g(T_R) = 10/24 = 0.417$。因此，右边的决策树比左边的决策树具有更好的悲观错误率。这样，除非能够减少一个以上训练记录的误分类，否则结点不应当扩展。□

2. Minimum Description Length (MDL)

X	y
X_1	1
X_2	0
X_3	0
X_4	1
…	…
X_n	1



X	y
X_1	?
X_2	?
X_3	?
X_4	?
…	…
X_n	?

- Cost(Model, Data) = Cost(Data|Model) + Cost(Model)
  - Cost is the number of bits needed for encoding.
  - Search for the least costly model.
- Cost(Data|Model) encodes the misclassification errors.
- Cost(Model) uses node encoding (number of children) plus splitting condition encoding.

决策树研究问题

关于过渡拟合

解决过度拟合的手段：

1 及早停止树增长；

2 后修剪法。

决策树研究问题

关于过渡拟合

1 及早停止树增长

由于决策树学习要从候选集合众选择满足给定标准的最大化属性，并且不回溯，也就是我们常说的爬山策略，其选择往往会是局部最优而不是全局最优。树结构越复杂，则过渡拟合发生的可能性越大。因此，要选择简单的模型。

Occan法则（又称Occan剃刀 Occan Razor）:具有相同泛化误差的两个模型，较简单的模型比复杂的模型更可取。

4.4.5 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
  - Can use MDL for post-pruning

Example of Post-Pruning

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

Training Error (Before splitting) = 10/30

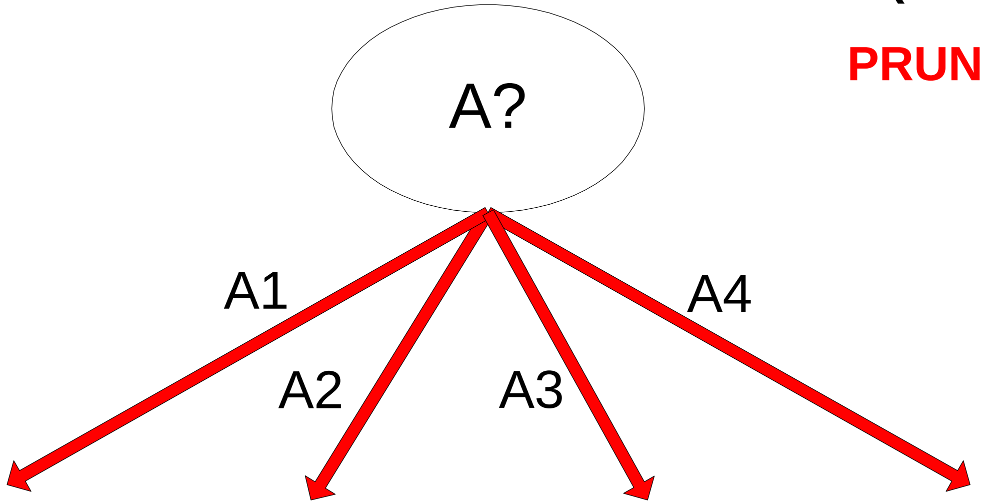
Pessimistic error = (10 + 0.5)/30 = 10.5/30

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

= $(9 + 4 \times 0.5)/30 = 11/30$

PRUNE!



Class = Yes	8
Class = No	4

Class = Yes	3
Class = No	4

Class = Yes	4
Class = No	1

Class = Yes	5
Class = No	1

Examples of Post-pruning

- Optimistic error?

  Don't prune for both cases

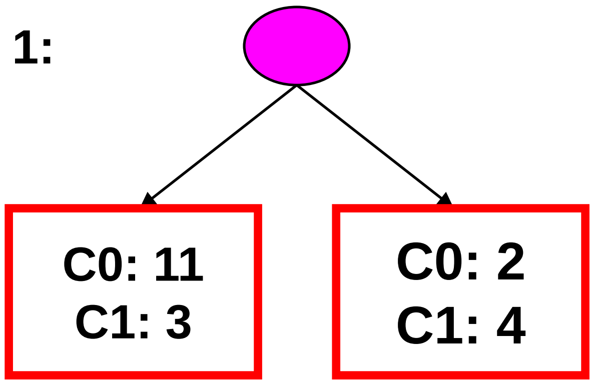
- Pessimistic error?

  Don't prune case 1, prune case 2

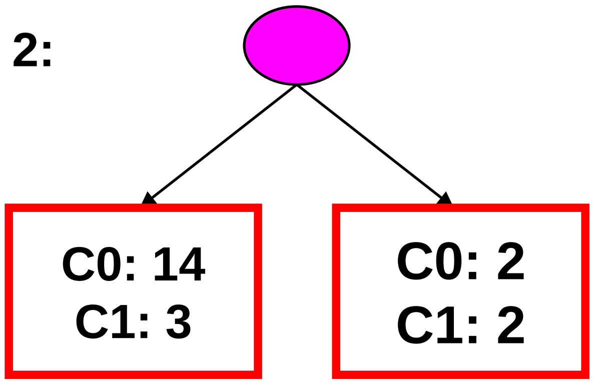
- Reduced error pruning?

  Depends on validation set

Case 1:



Case 2:



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

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	?	Single	90K	Yes

$\searrow$ **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

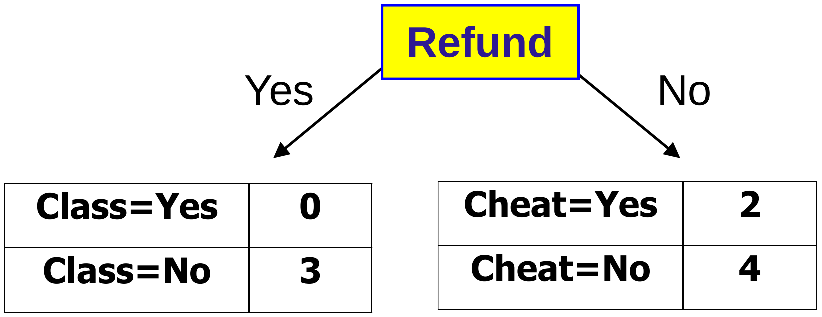
Entropy(Refund=No)
= -(2/6)log(2/6) – (4/6)log(4/6) = 0.9183

Entropy(Children)
= 0.3 (0) + 0.6 (0.9183) = 0.551

Gain = 0.9 $\times$ (0.8813 – 0.551) = 0.3303

Distribute Instances

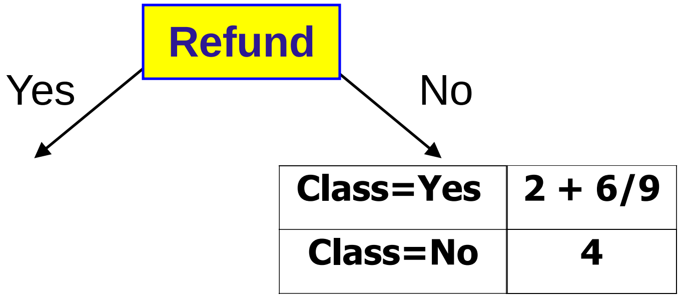
Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No



Class=Yes	0
Class=No	3

Cheat=Yes	2
Cheat=No	4

Tid	Refund	Marital Status	Taxable Income	Class
10	?	Single	90K	Yes



Class=Yes	2 + 6/9
Class=No	4

Probability that Refund=Yes is 3/9

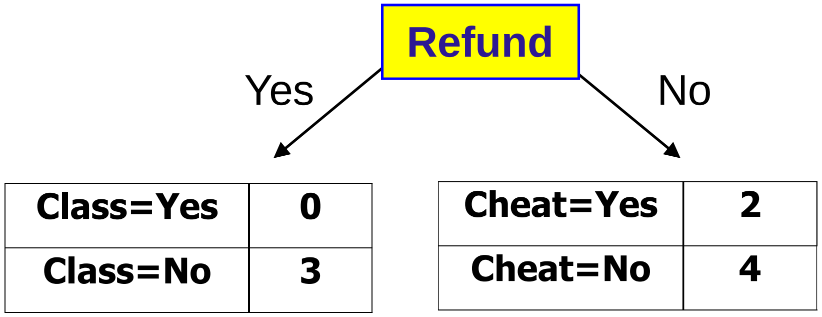
Probability that Refund=No is 6/9

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

Distribute Instances

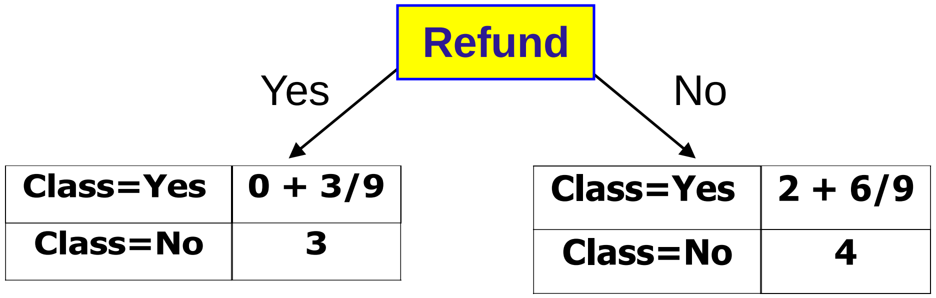
Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No

Tid	Refund	Marital Status	Taxable Income	Class
10	?	Single	90K	Yes



Class=Yes	0
Class=No	3

Cheat=Yes	2
Cheat=No	4



Class=Yes	0 + 3/9
Class=No	3

Class=Yes	2 + 6/9
Class=No	4

Probability that Refund=Yes is 3/9

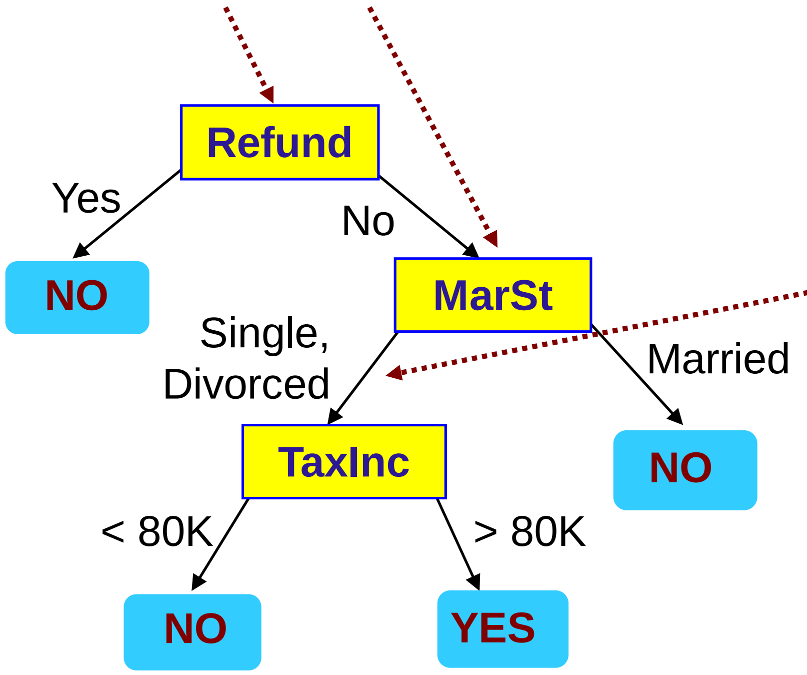
Probability that Refund=No is 6/9

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

Other Issues

- Data Fragmentation
- Search Strategy
- Expressiveness
- Tree Replication

Data Fragmentation

- Number of instances gets smaller as you traverse down the tree

- Number of instances at the leaf nodes could be too small to make any statistically significant decision

Search Strategy

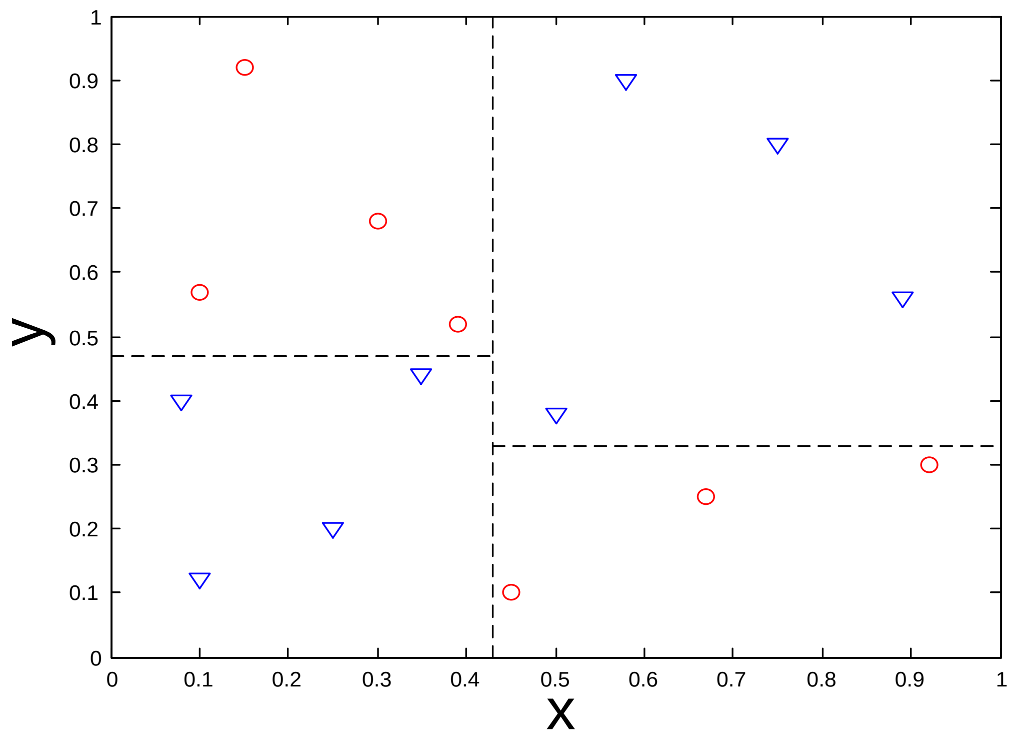
- Finding an optimal decision tree is NP-hard
- The algorithm presented so far uses a greedy, top-down, recursive partitioning strategy to induce a reasonable solution
- Other strategies?
  - Bottom-up
  - Bi-directional

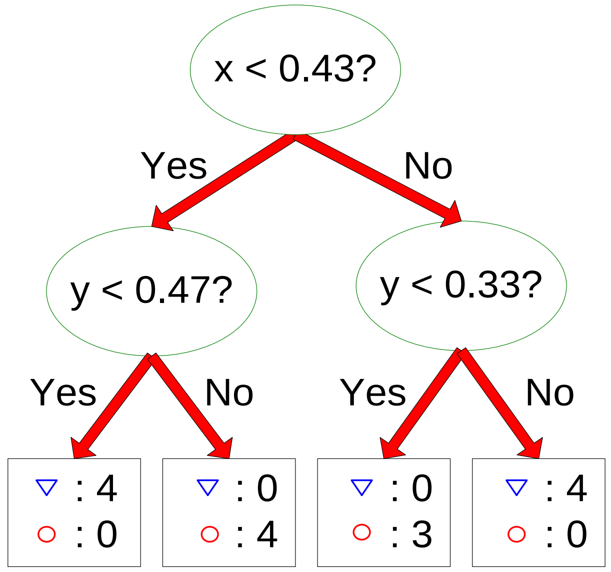
Expressiveness

- Decision tree provides expressive representation for learning discrete-valued function
  - But they do not generalize well to certain types of Boolean functions
    - Example: parity function:
      - Class = 1 if there is an even number of Boolean attributes with truth value = True
      - Class = 0 if there is an odd number of Boolean attributes with truth value = True
    - For accurate modeling, must have a complete tree

- Not expressive enough for modeling continuous variables
  - Particularly when test condition involves only a single attribute at-a-time

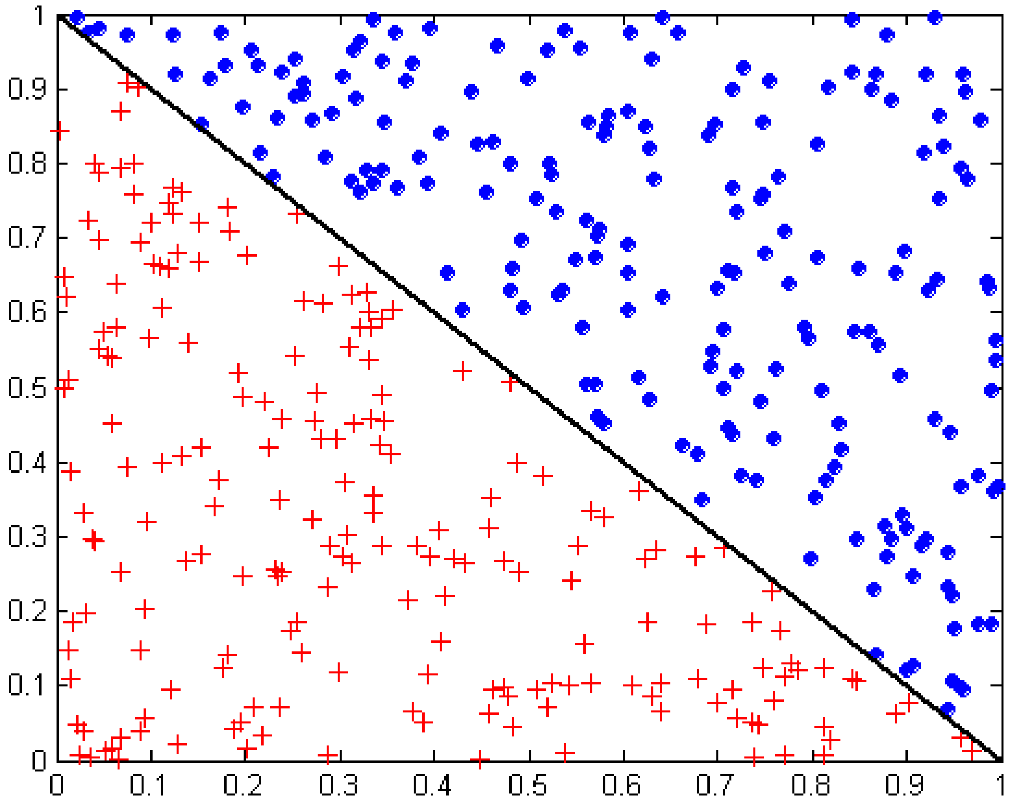
Decision Boundary

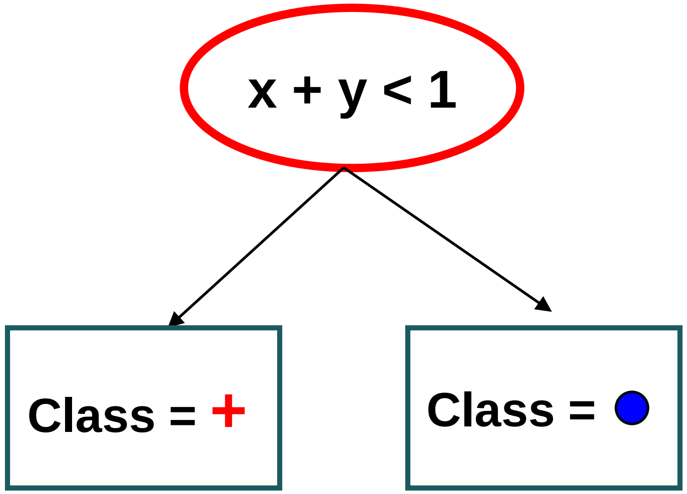




- **Border line between two neighboring regions of different classes is known as decision boundary**
- **Decision boundary is parallel to axes because test condition involves a single attribute at-a-time**

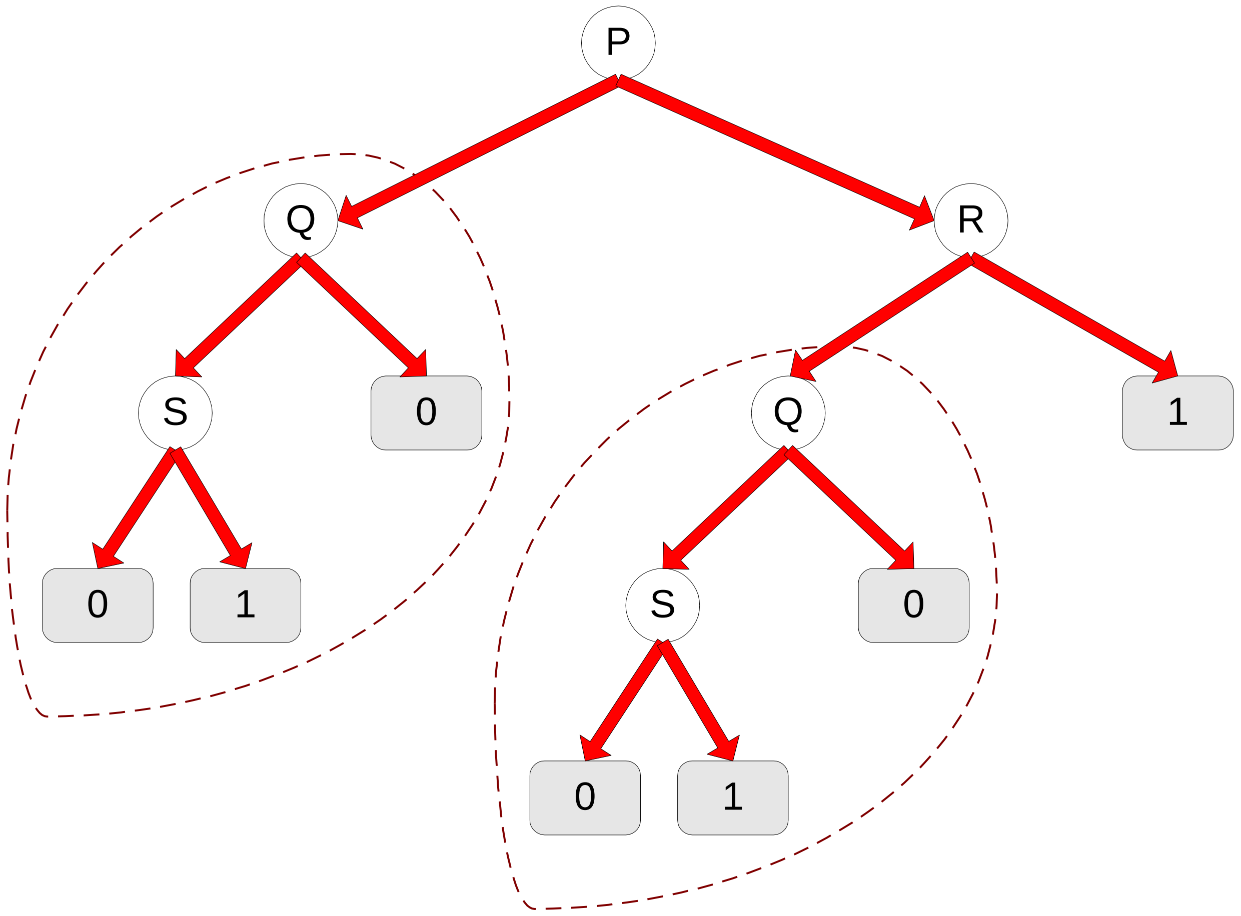
Oblique Decision Trees





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

Tree Replication



• **Same subtree appears in multiple branches**

4.5 Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

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.
- Confus

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	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		
ACTUAL CLASS		Class=Yes	Class=No
	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		
ACTUAL CLASS	C(i|j)	**Class=Yes**	**Class=No**
	Class=Yes	C(Yes|Yes)	C(No|Yes)
	Class=No	C(Yes|No)	C(No|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

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

Accuracy = 80%

Cost = 3910

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(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

N = a + b + c + d

Accuracy = (a + d)/N

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

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

Cost-Sensitive Measures

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

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

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

Precision is biased towards C(Yes|Yes) & C(Yes|No)

Recall is biased towards C(Yes|Yes) & C(No|Yes)

F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

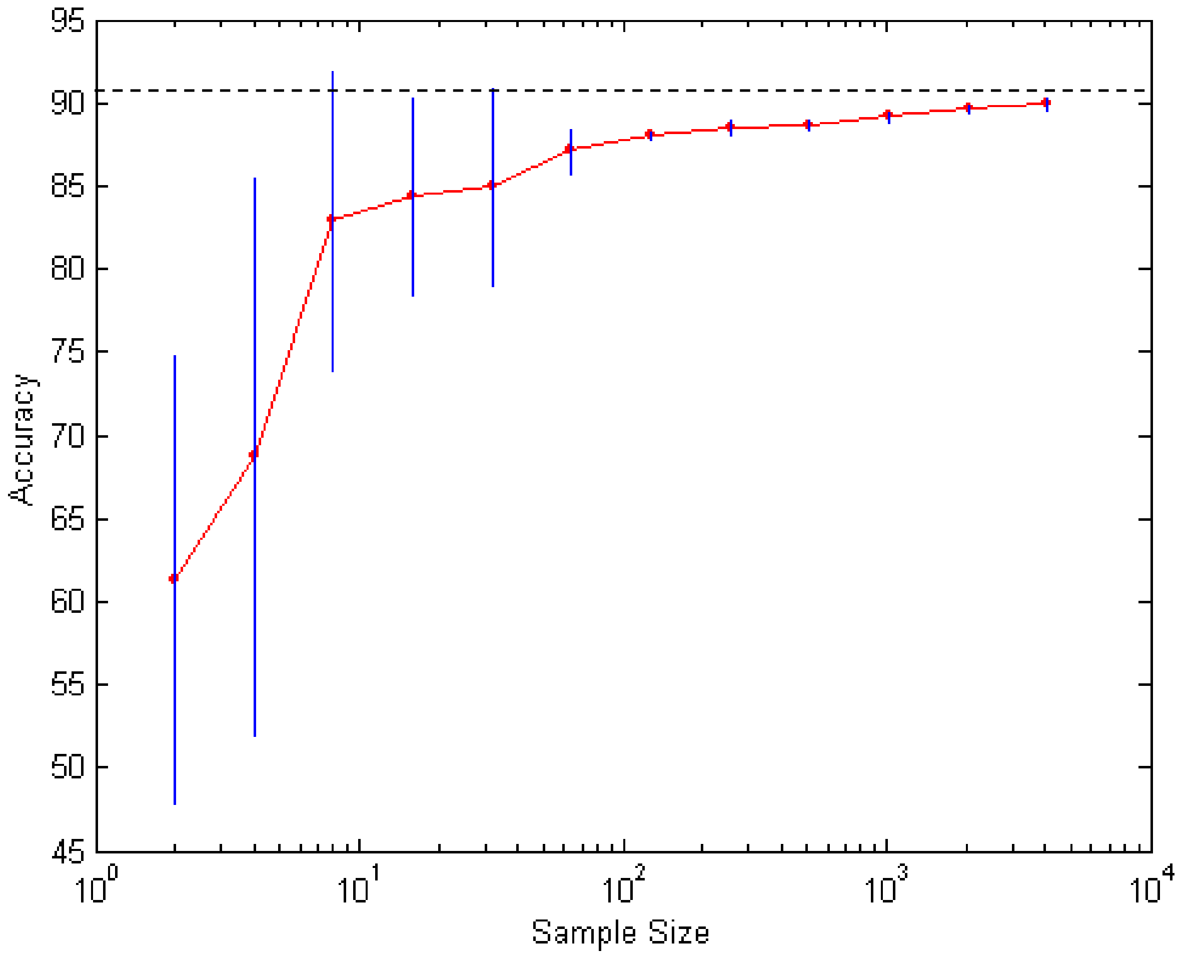
Methods for Performance Evaluation

How to obtain a reliable estimate of performance?

Performance of a model may depend on other factors besides the learning algorithm:

- Class distribution
- Cost of misclassification
- Size of training and test sets

Learning Curve



Learning curve shows how accuracy changes with varying sample size

Requires a sampling schedule for creating learning curve:

- Arithmetic sampling (Langley, et al)
- Geometric sampling (Provost et al)

Effect of small sample size:

- Bias in the estimate
- Variance of estimate

Methods of Estimation

- Holdout（保持法）
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling（随机抽样）
  - Repeated holdout
- Cross validation（交叉验证）
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
  - oversampling vs undersampling
- Bootstrap（自举法）
  - Sampling with replacement

Model Evaluation

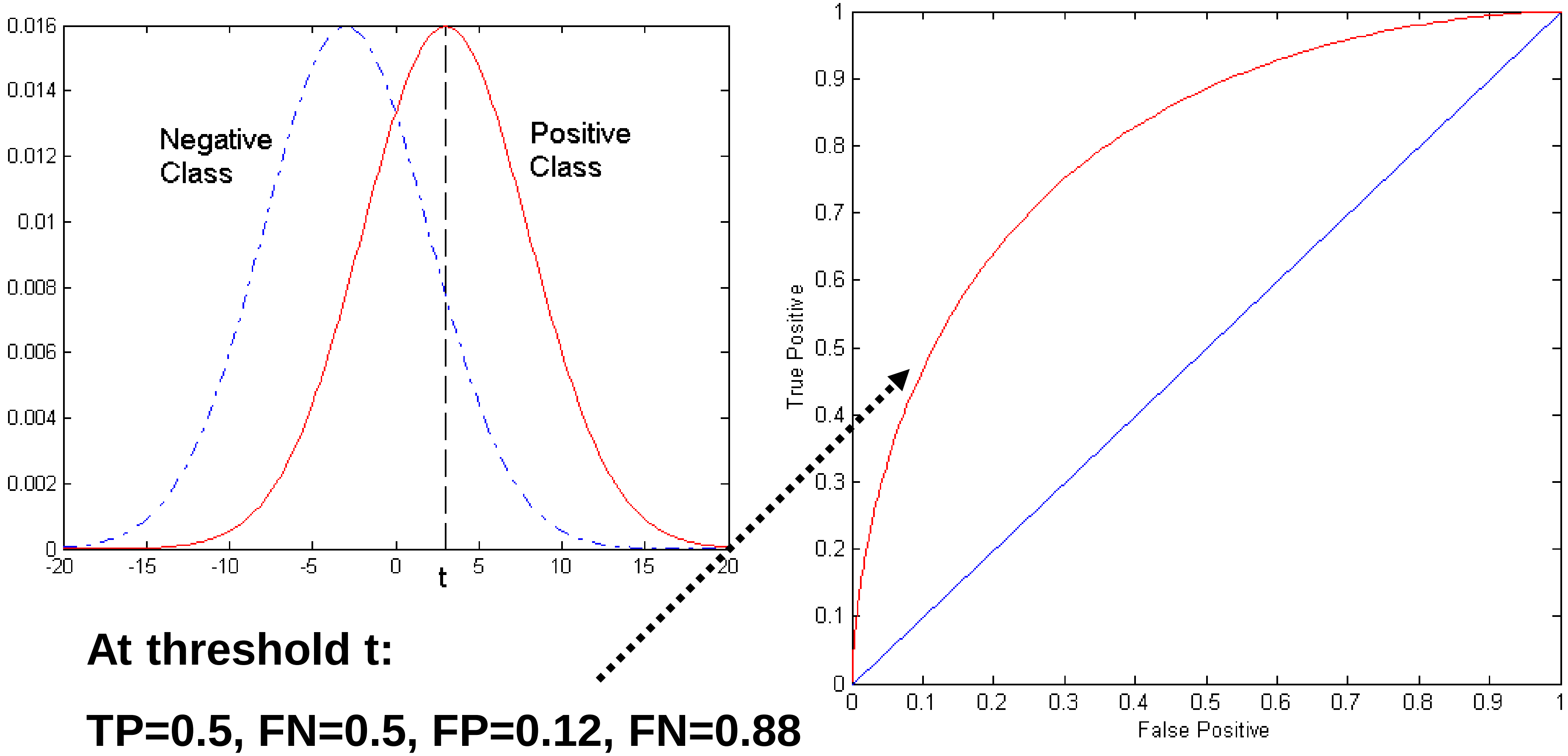
- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)
- any points located at x > t is classified as positive



At threshold t:

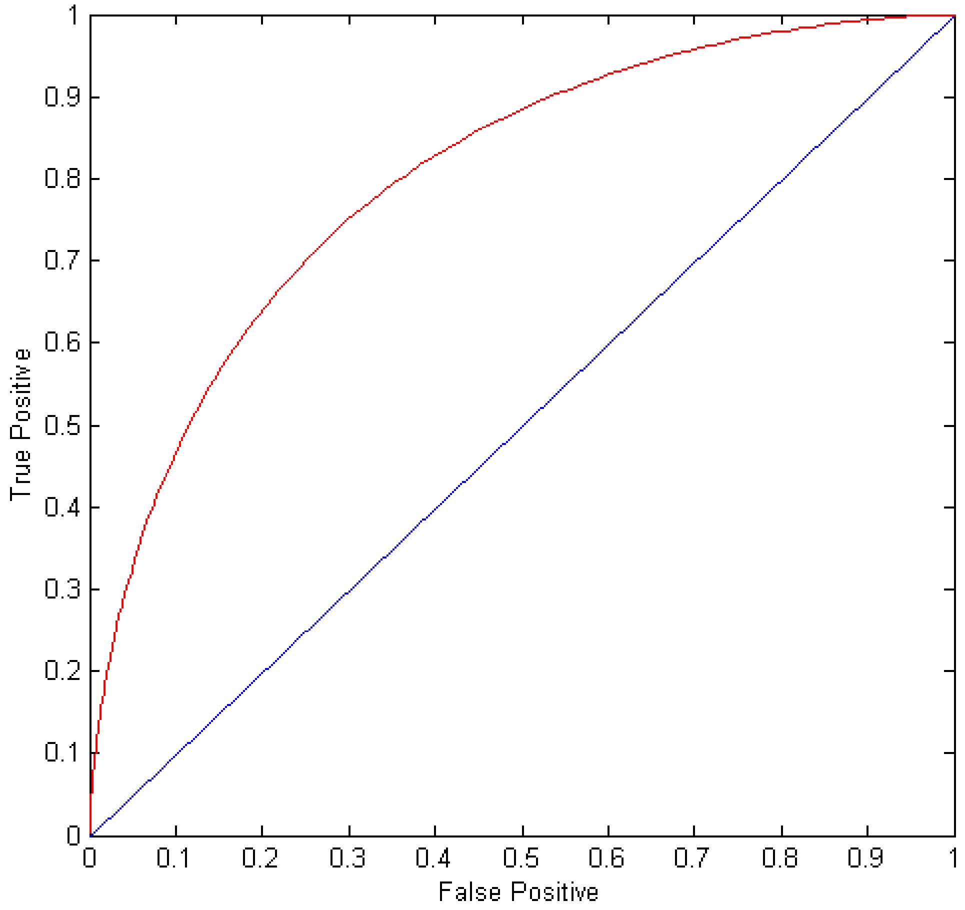
TP=0.5, FN=0.5, FP=0.12, FN=0.88

ROC Curve

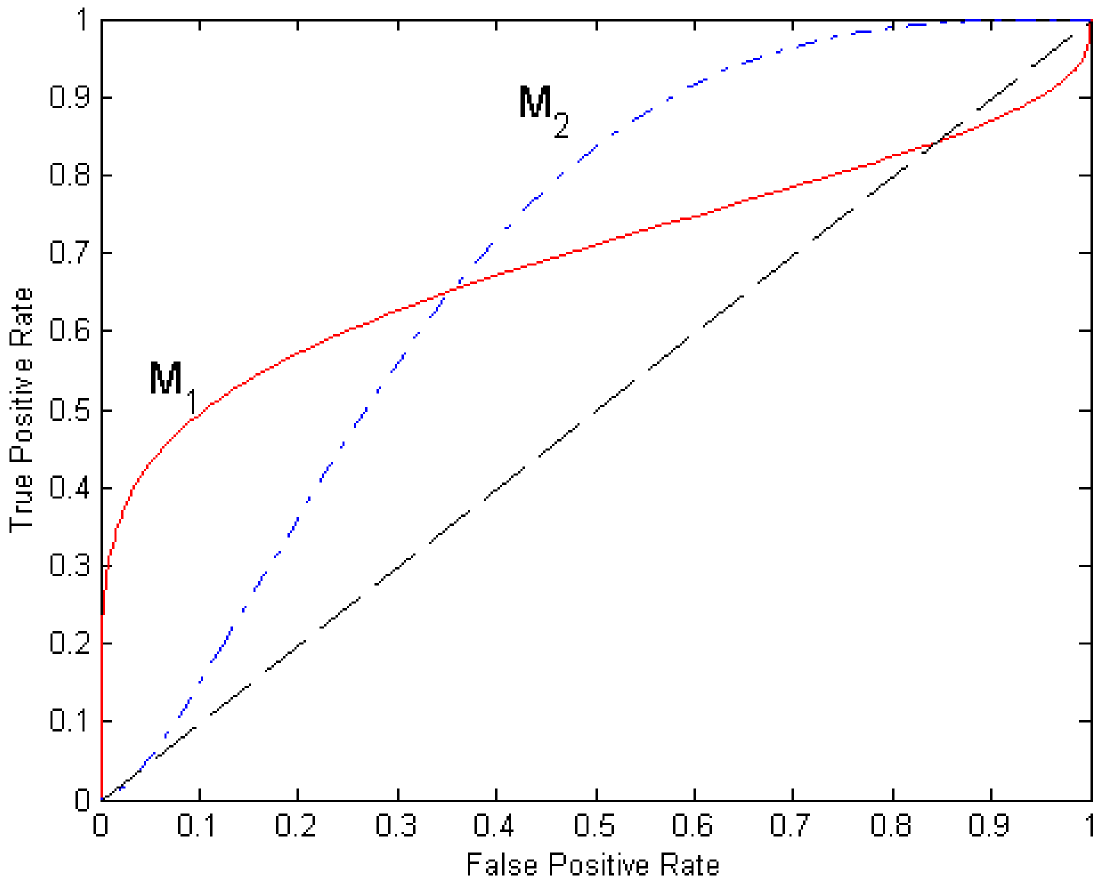
(TP, FP):

- (0, 0): declare everythi
  to be negativ
  class
- (1, 1): declare everythi
  to be positive
  class
- (1, 0): ideal

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



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

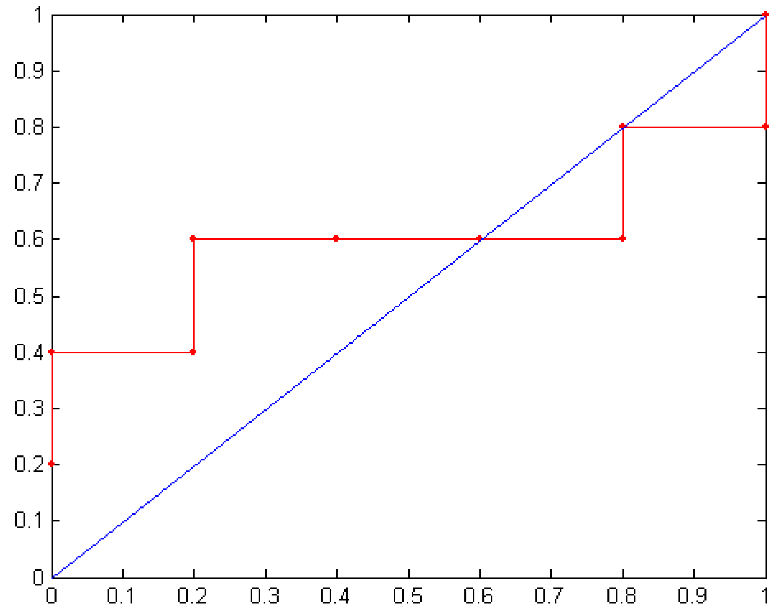
Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use classifier that produces posterior probability for each test instance P(+|A)
- Sort the instances according to P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, TPR = TP/(TP+FN)
- FP rate, FPR = FP/(FP + TN)

How to construct an ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Test of Significance

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances
- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Confidence Interval for Accuracy

- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
  - Possible outcomes for prediction: correct or wrong
  - Collection of Bernoulli trials has a Binomial distribution:
    - x ~ Bin(N, p) x: number of correct predictions
    - e.g: Toss a fair coin 50 times, how many heads would turn up?
      Expected number of heads = N×p = 50 × 0.5 = 25

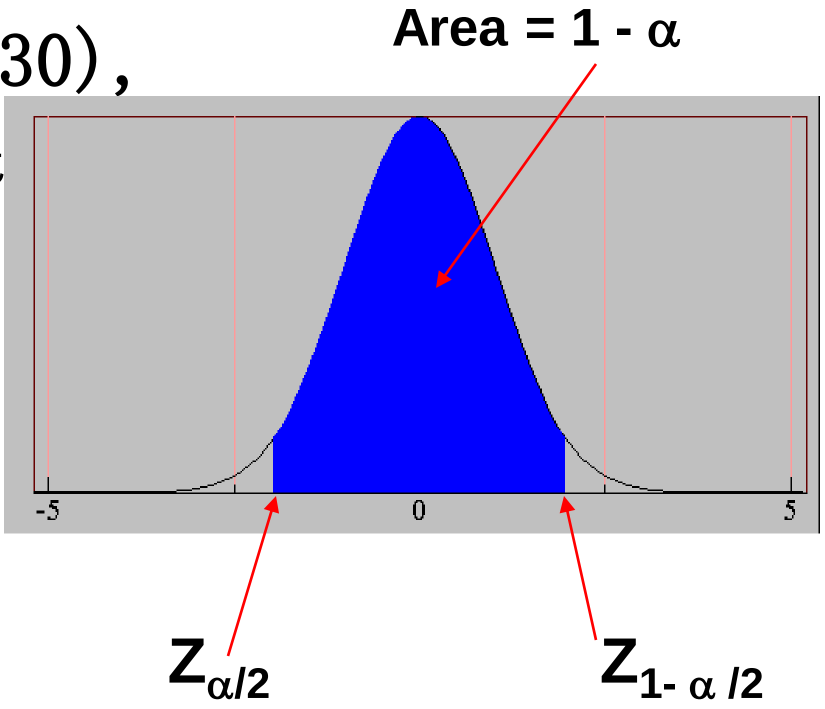
- Given x (# of correct predictions) or equivalently, acc=x/N, and N (# of test instances),

[illegible]

Confidence Interval for Accuracy

- For large test sets (N > 30),
  - acc has a normal distribution with mean p and variance p(1-p)/N

$$P\left(Z_{\alpha/2} < \frac{acc - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2}\right) = 1 - \alpha$$



- Confidence Interval for p:

$$p = \frac{2 \times N \times acc + Z_{\alpha/2}^2 \pm \sqrt{Z_{\alpha/2}^2 + 4 \times N \times acc - 4 \times N \times acc^2}}{2(N + Z_{\alpha/2}^2)}$$

Confidence Interval for Accuracy

- Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:
  - N=100, acc = 0.8
  - Let 1-α = 0.95 (95% confidence)
  - From probability table, $Z_{\alpha/2}$=1.96

1-α	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65

N	50	100	500	1000	5000
p(lower)	0.670	0.711	0.763	0.774	0.789
p(upper)	0.888	0.866	0.833	0.824	0.811

Comparing Performance of 2 Models

- Given two models, say M1 and M2, which is better?
  - M1 is tested on D1 (size=n1), found error rate = e_1
  - M2 is tested on D2 (size=n2), found error rate = e_2
  - Assume D1 and D2 are independent
  - If n1 and n2 are sufficiently large, then

$$e_1 \sim N(\mu_1, \sigma_1)$$

$$e_2 \sim N(\mu_2, \sigma_2)$$

  - Approximate:

$$\hat{\sigma}_i = \frac{e_i(1-e_i)}{n_i}$$

Comparing Performance of 2 Models

- To test if performance difference is statistically significant: d = e1 – e2
  - d ~ $N(d_t, \sigma_t)$ where d_t is the true difference
  - Since D1 and D2 are independent, their variance adds up:

$$\sigma_t^2 = \sigma_1^2 + \sigma_2^2 \cong \hat{\sigma}_1^2 + \hat{\sigma}_2^2$$
$$= \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}$$

  - At (1-α) confidence level, $d_t = d \pm Z_{\alpha/2} \hat{\sigma}_t$

An Illustrative Example

- Given: M1: n1 = 30, e1 = 0.15
  M2: n2 = 5000, e2 = 0.25
- d = |e2 - e1| = 0.1 (2-sided test)

$$\hat{\sigma}_d = \frac{0.15(1-0.15)}{30} + \frac{0.25(1-0.25)}{5000} = 0.0043$$

- At 95% confidence level, $Z_{\alpha/2}$=1.96

$$d_t = 0.100 \pm 1.96 \times \sqrt{0.0043} = 0.100 \pm 0.128$$

=> Interval contains 0 => difference may not be statistically significant

Comparing Performance of 2 Algorithms

- Each learning algorithm may produce k models:
  - L1 may produce M11 , M12, …, M1k
  - L2 may produce M21 , M22, …, M2k
- If models are generated on the same test sets D1, D2, …, Dk (e.g., via cross-validation)
  - For each set: compute $d_j = e_{1j} - e_{2j}$
  - d_j has mean d_t and variance σ_t
  - Estimate:

$$\hat{\sigma}_t^2 = \frac{\sum_{j=1}^{k}(d_j - \bar{d})^2}{k(k-1)}$$

$$d_t = d \pm t_{1-\alpha,k-1}\,\hat{\sigma}_t$$