

DATA MINING - 1DL105, 1DL111

Fall 2007

An introductory class in data mining

<http://user.it.uu.se/~udbl/dm-ht2007/>

alt. <http://www.it.uu.se/edu/course/homepage/infoutv/ht07>

Kjell Orsborn

Uppsala Database Laboratory

Department of Information Technology, Uppsala University,
Uppsala, Sweden



UPPSALA
UNIVERSITET

Data Mining

Classification: Basic Concepts, Decision Trees, and Model Evaluation

(Tan, Steinbach, Kumar ch. 4)

Kjell Orsborn

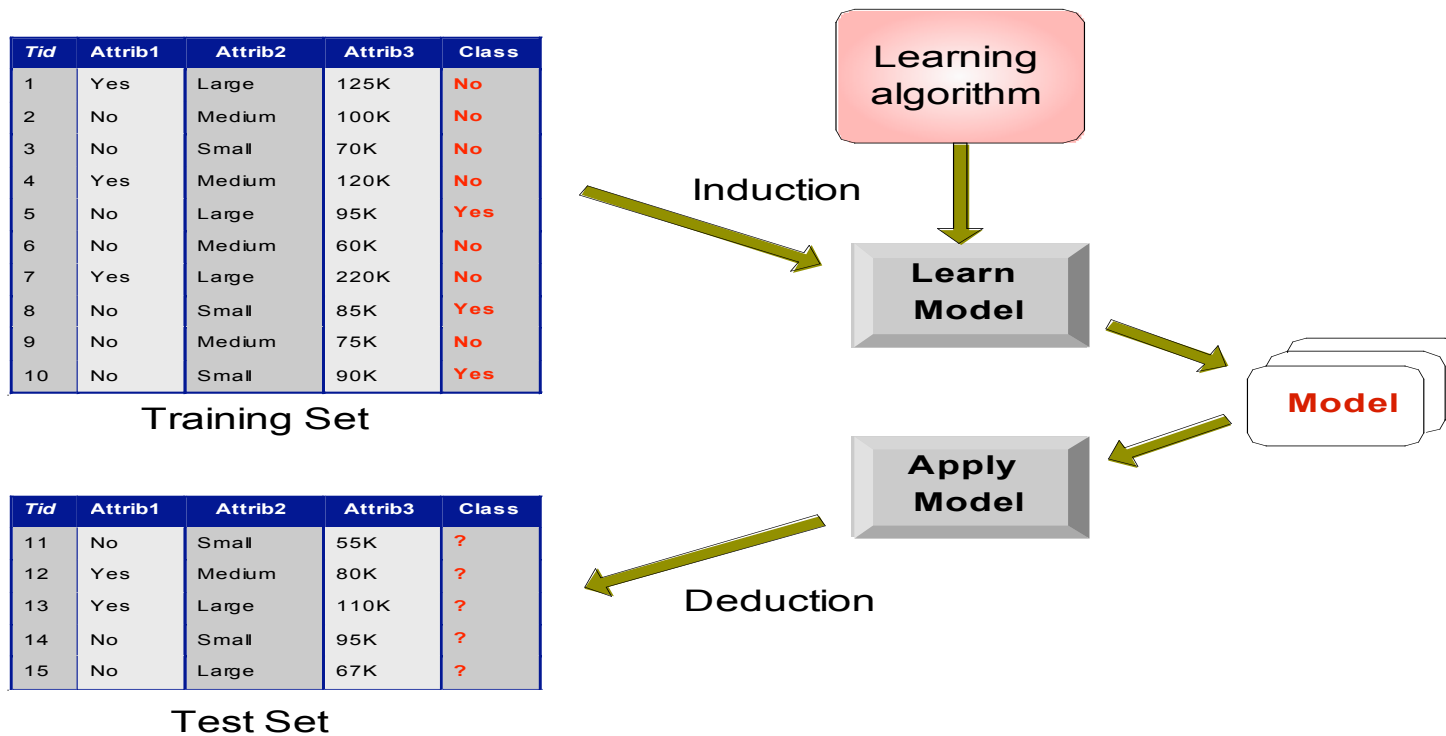
Department of Information Technology
Uppsala University, Uppsala, Sweden

Definition of classification

- 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.

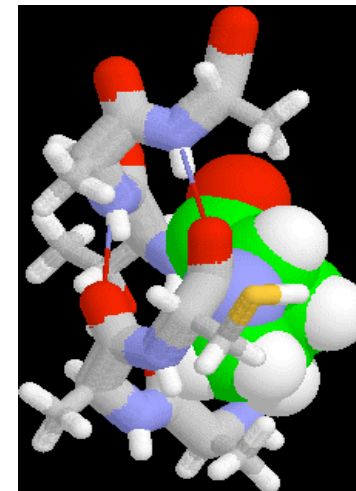


Illustrating Classification Task



Examples of classification task

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc



Classification techniques

- Nearest Neighbour methods
- Decision Tree methods
- Rule-based methods
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines

Instance-based classifiers (ch 5.2)

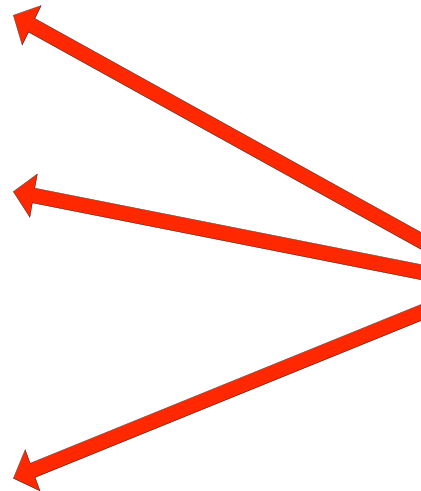
Set of Stored Cases

Atr1	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

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

Unseen Case

Atr1	AtrN

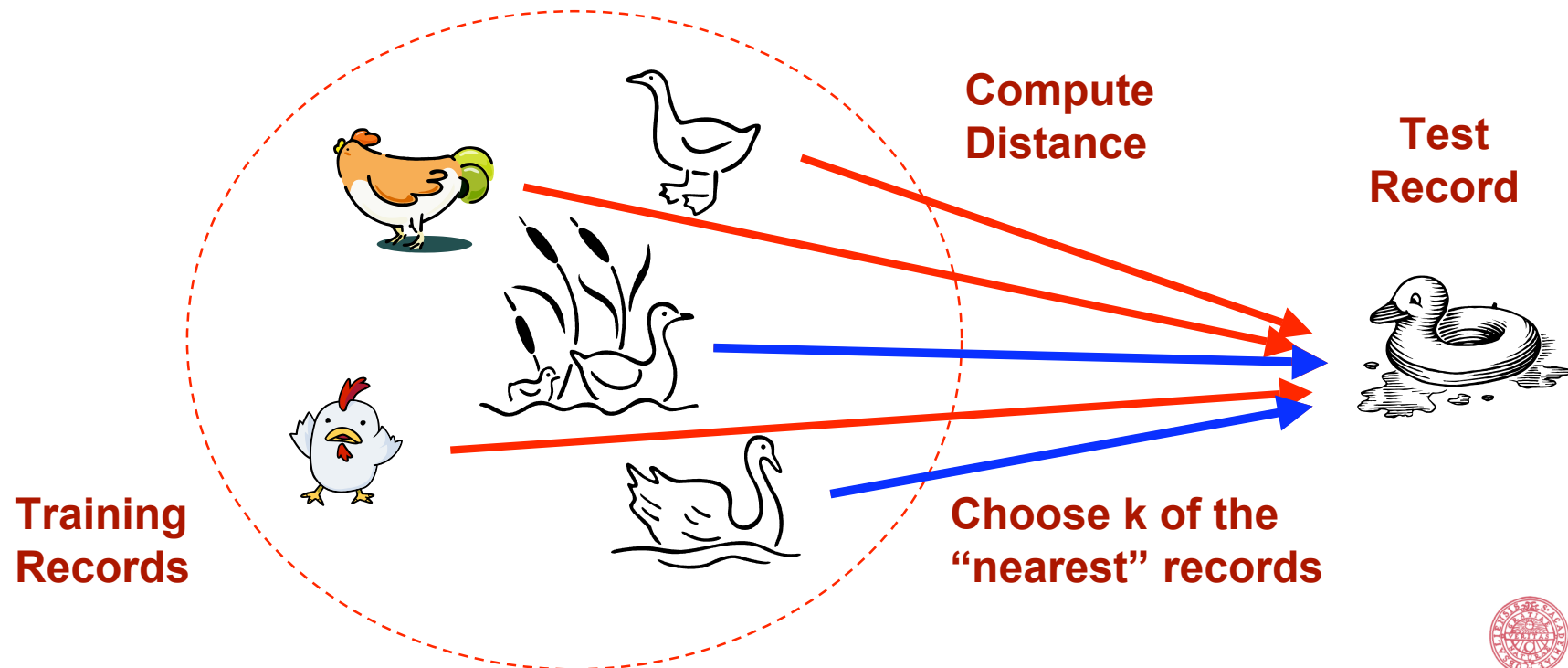


Instance-based classifiers

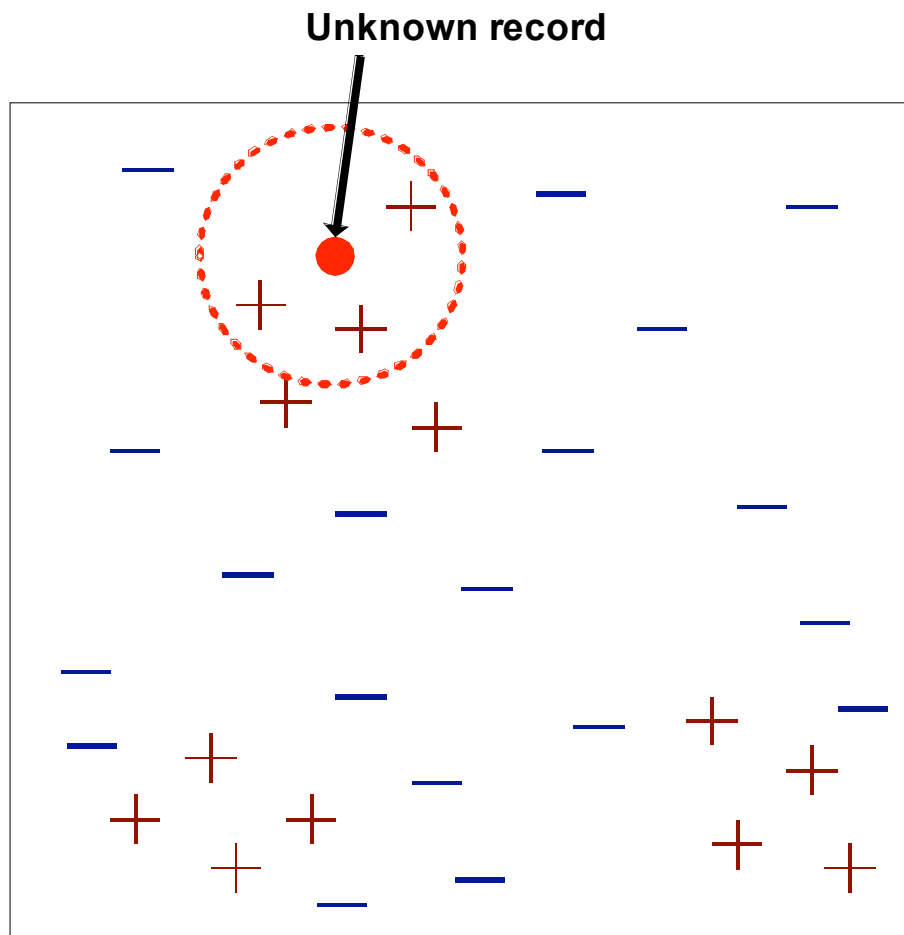
- Examples:
 - Rote-learner
 - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
 - Nearest neighbor
 - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

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



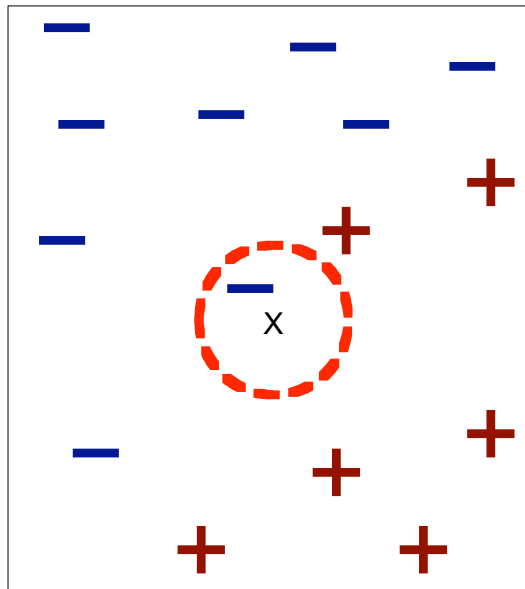
Nearest-Neighbor Classifiers



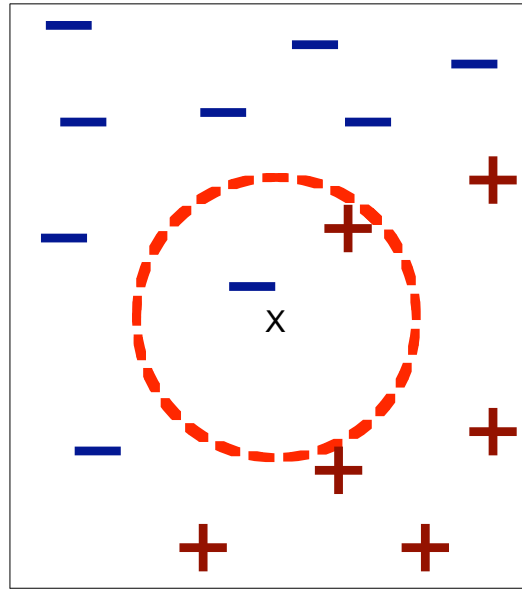
- Requires three things
 - The set of stored records
 - Distance metric to compute distance between records
 - The value of k , the number of nearest neighbors to retrieve
- To classify an unknown record:
 - Compute distance to other training records
 - Identify k nearest neighbors
 - Use class labels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



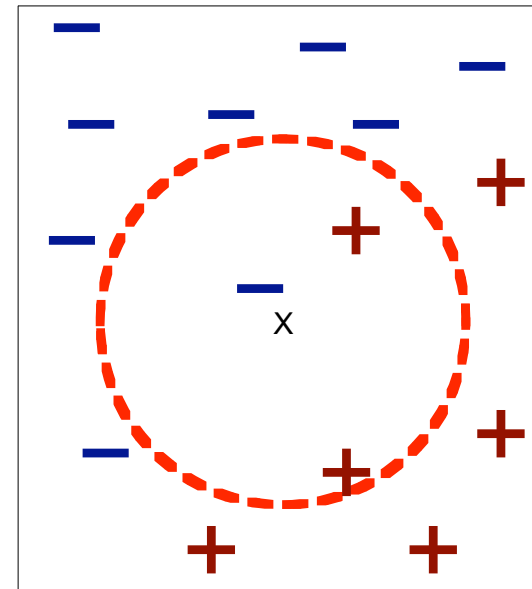
Definition of Nearest Neighbor



(a) 1-nearest neighbor



(b) 2-nearest neighbor



(c) 3-nearest neighbor

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

Nearest Neighbor Classification

- Compute distance between two points:
 - Euclidean distance

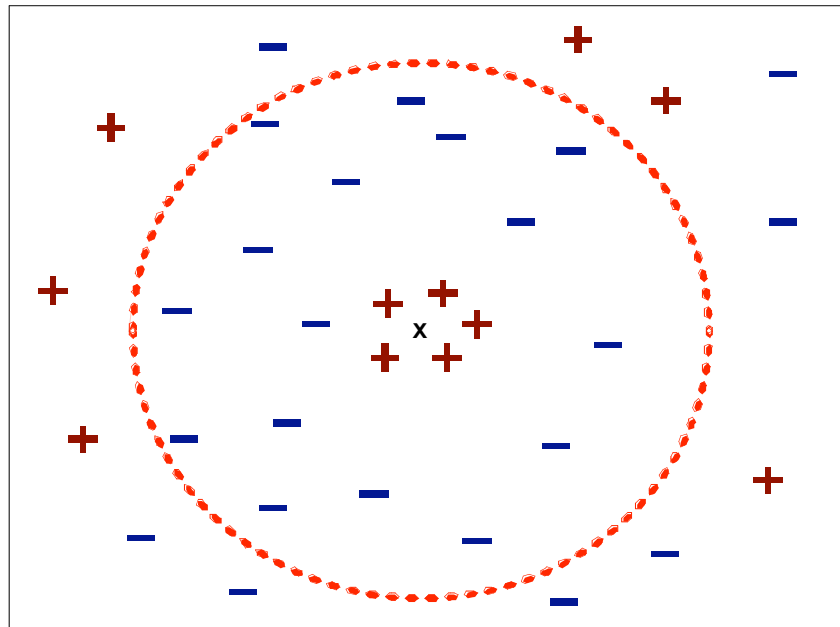
$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- 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 1 0

0 1 1 1 1 1 1 1 1 1 1 1

VS

1 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 1

$d = 1.4142$

$d = 1.4142$

- Solution: Normalize the vectors to unit length

KNN algorithm

Input:

D // Training data
 K // Number of neighbors
 t // Input tuple to classify

Output:

c // Class to which t is assigned

KNN Algorithm:

```
// Algorithm to classify tuple using KNN
 $N = \emptyset$ ;
// Find set of neighbors,  $N$ , for  $t$ 
foreach  $d \in D$  do
  if  $|N| \leq K$  then
     $N = N \cup d$ ;
  else
    if  $\exists u \in N$  such that  $\text{sim}(t, u) \leq \text{sim}(t, d)$  then
      begin
         $N = N - u$ ;
         $N = N \cup d$ ;
      end
    // Find class for classification
 $c = \text{class to which the most } u \in N \text{ are classified};$ 
```

- Training set includes classes.
- Examine K items near item to be classified.
- New item placed in class with the most number of close items.
- Complexity is $O(q)$ for each tuple to be classified. Here q is the size of the training set.
- Rule of thumb: $K \leq \sqrt{\text{no of training examples}}$.
Commercial algorithms use a default of 10.



Nearest neighbor classification...

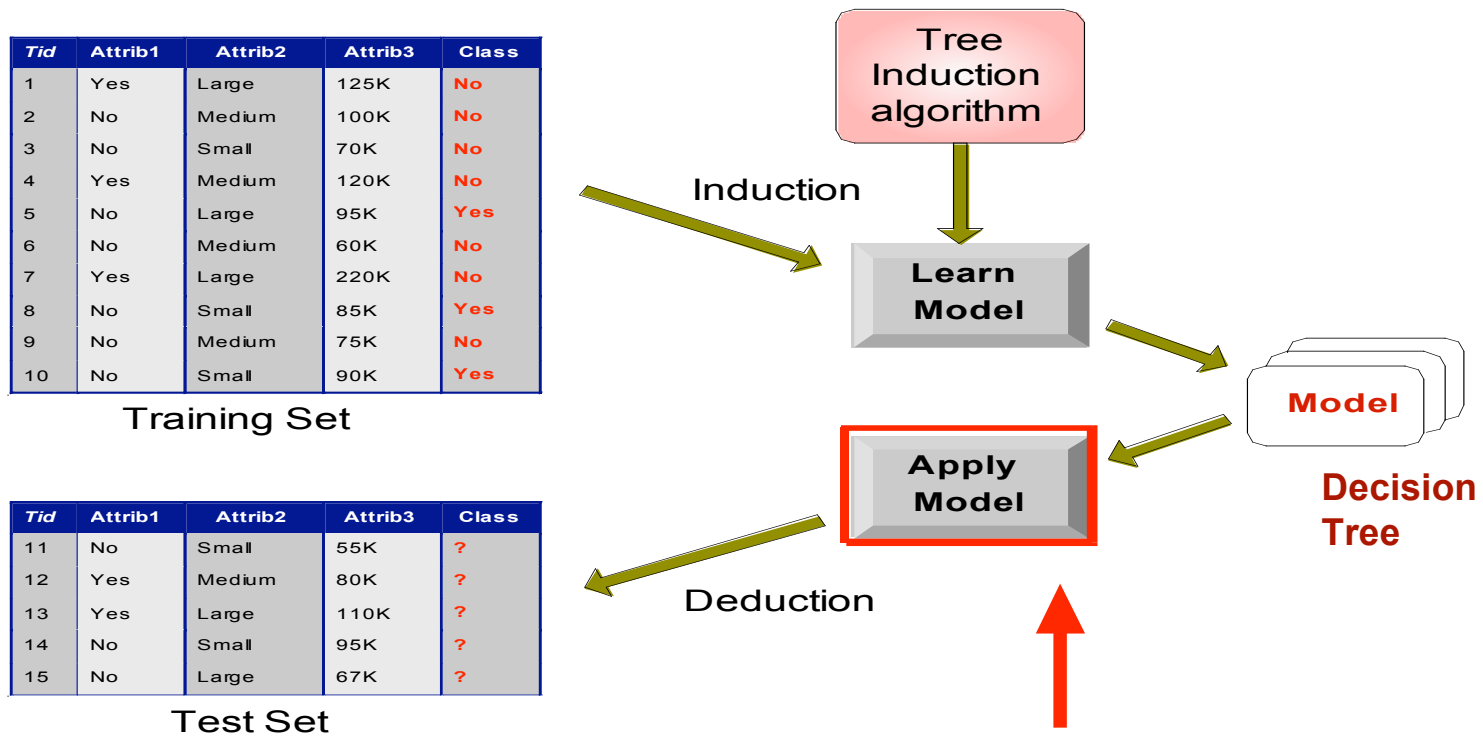
- k-NN classifiers are lazy learners
 - It does not build models explicitly
 - Unlike eager learners such as decision tree induction and rule-based systems
 - Classifying unknown records are relatively expensive

Decision tree induction (ch 4.3)

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



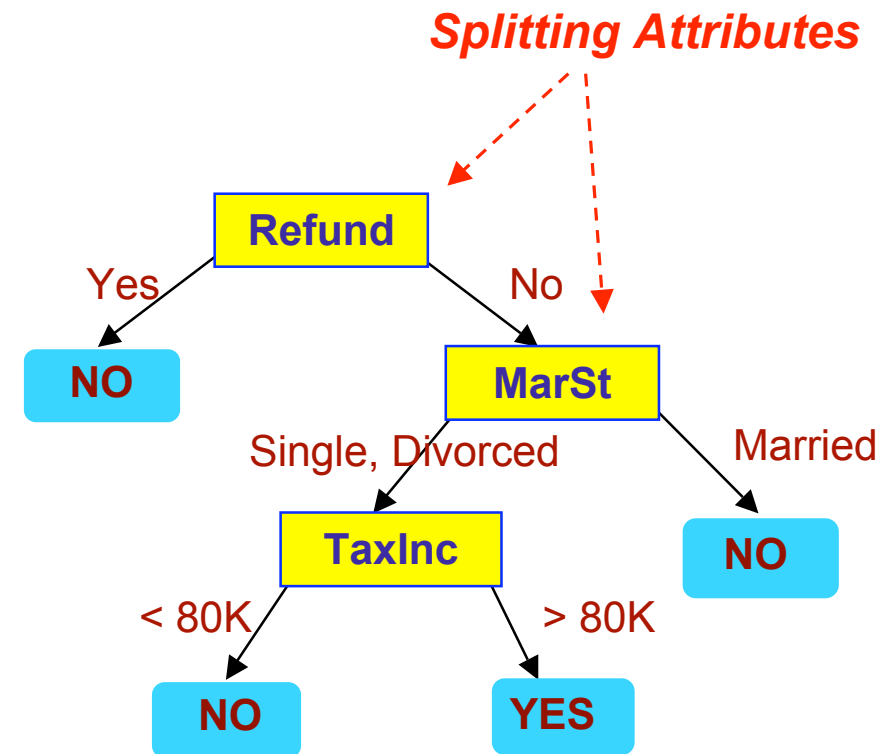
Decision tree classification task



Example of a 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



Training Data

Model: Decision Tree

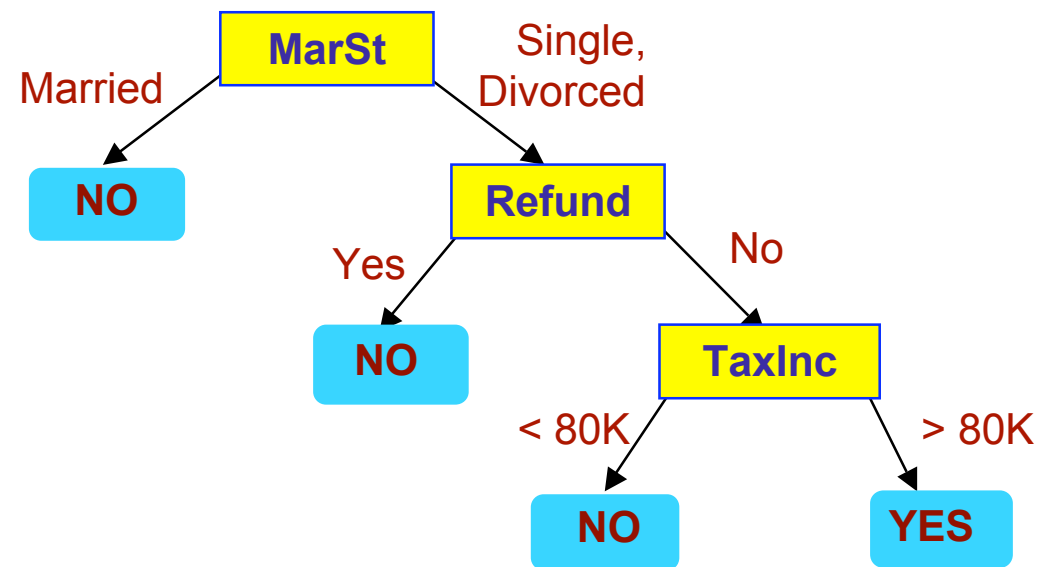


UPPSALA
UNIVERSITET

Another example of decision tree

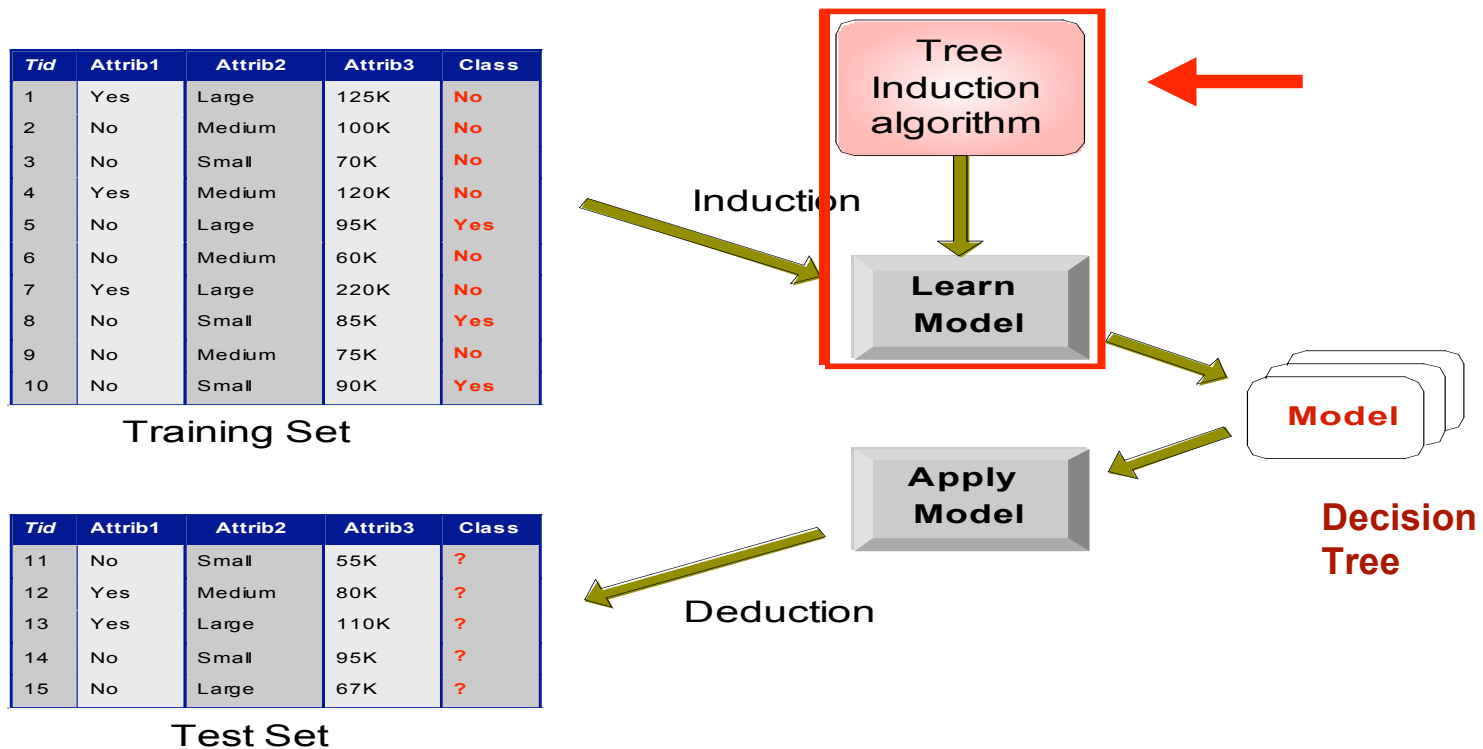
categorical
categorical
continuous
class

<i>Tid</i>	<i>Refund</i>	<i>Marital Status</i>	<i>Taxable Income</i>	<i>Cheat</i>
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!

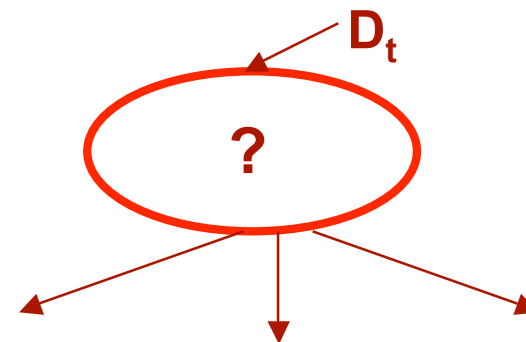
Decision tree classification task



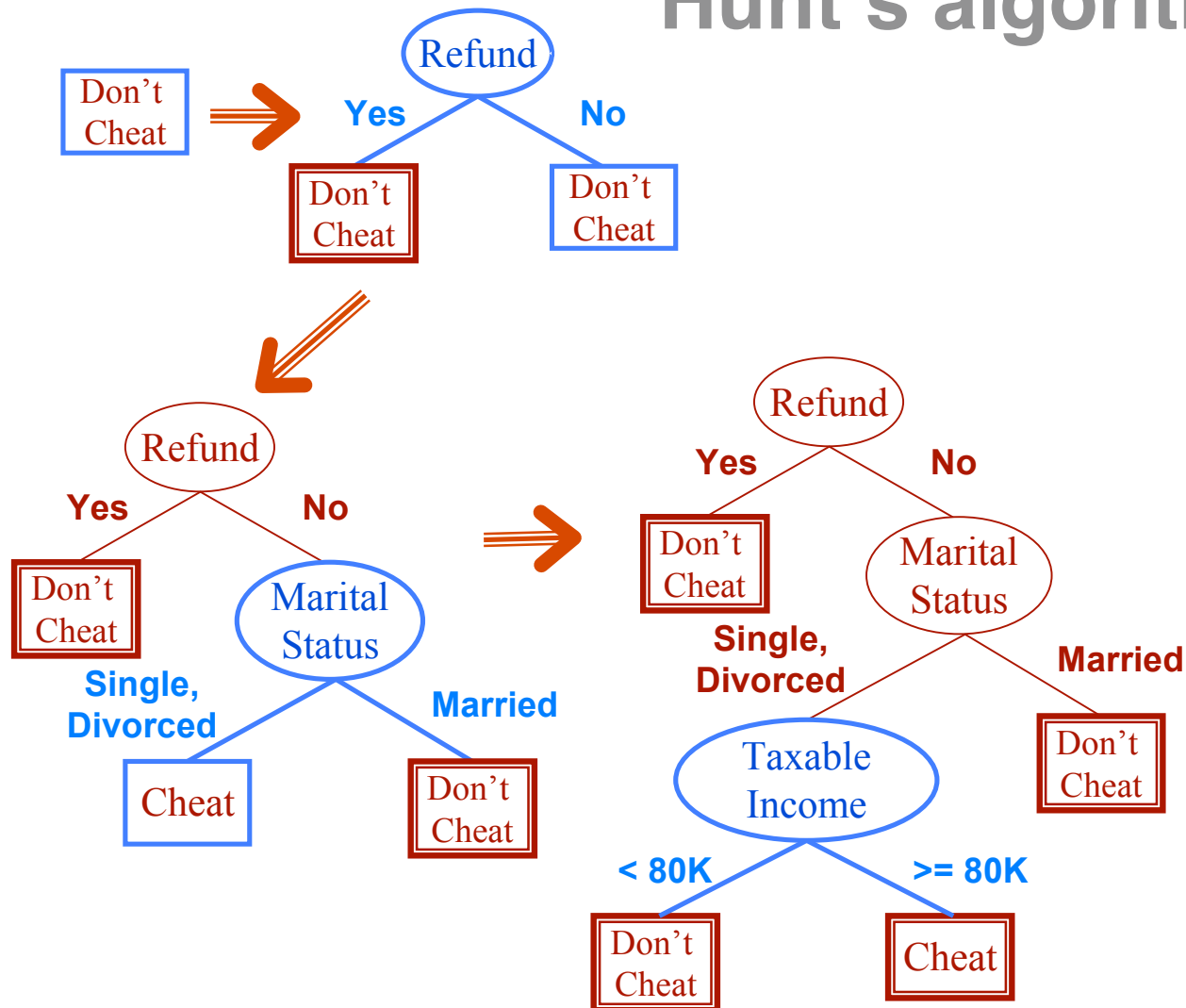
General structure of Hunt's algorithm

- 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.

<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



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

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



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



Splitting based on nominal attributes

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

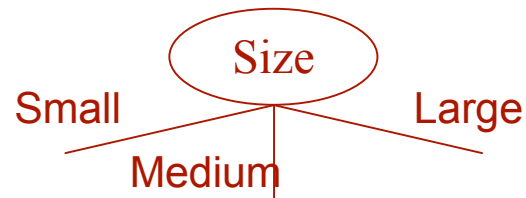


- Binary split: Divides values into two subsets.
Need to find optimal partitioning.

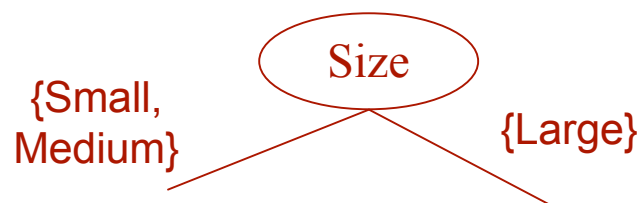


Splitting based on ordinal attributes

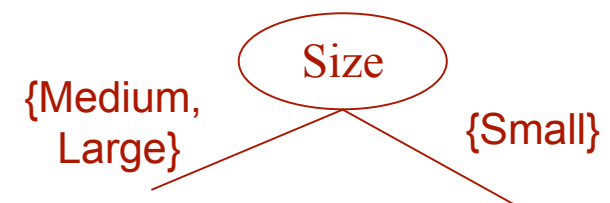
- Multi-way split: Use as many partitions as distinct values.



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



OR



- What about this split?

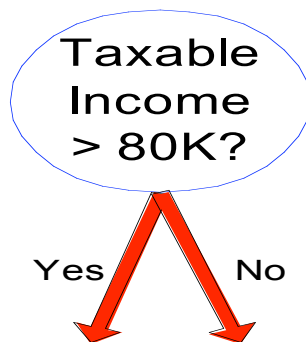


Splitting based on continuous attributes

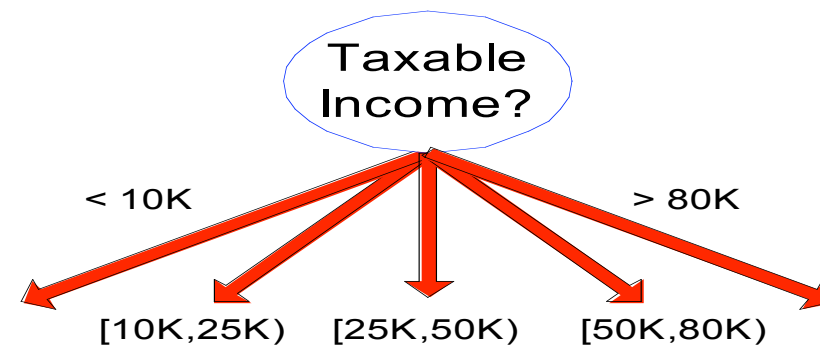
- Different ways of handling
 - Discretization to form an ordinal categorical attribute
 - Static – discretize once at the beginning
 - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
 - Binary Decision: ($A < v$) or ($A \geq v$)
 - consider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

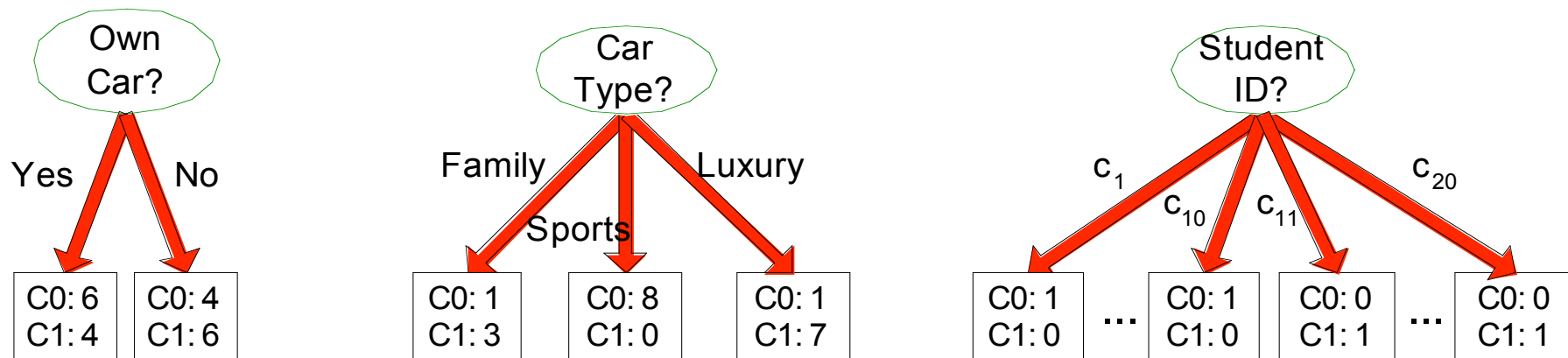
Tree induction

- 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



How to determine the best split

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



Which test condition is the best?

How to determine the best split

- Greedy approach:
 - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5
C1: 5

**Non-homogeneous,
High degree of impurity**

C0: 9
C1: 1

**Homogeneous,
Low degree of impurity**

Measures of node impurity

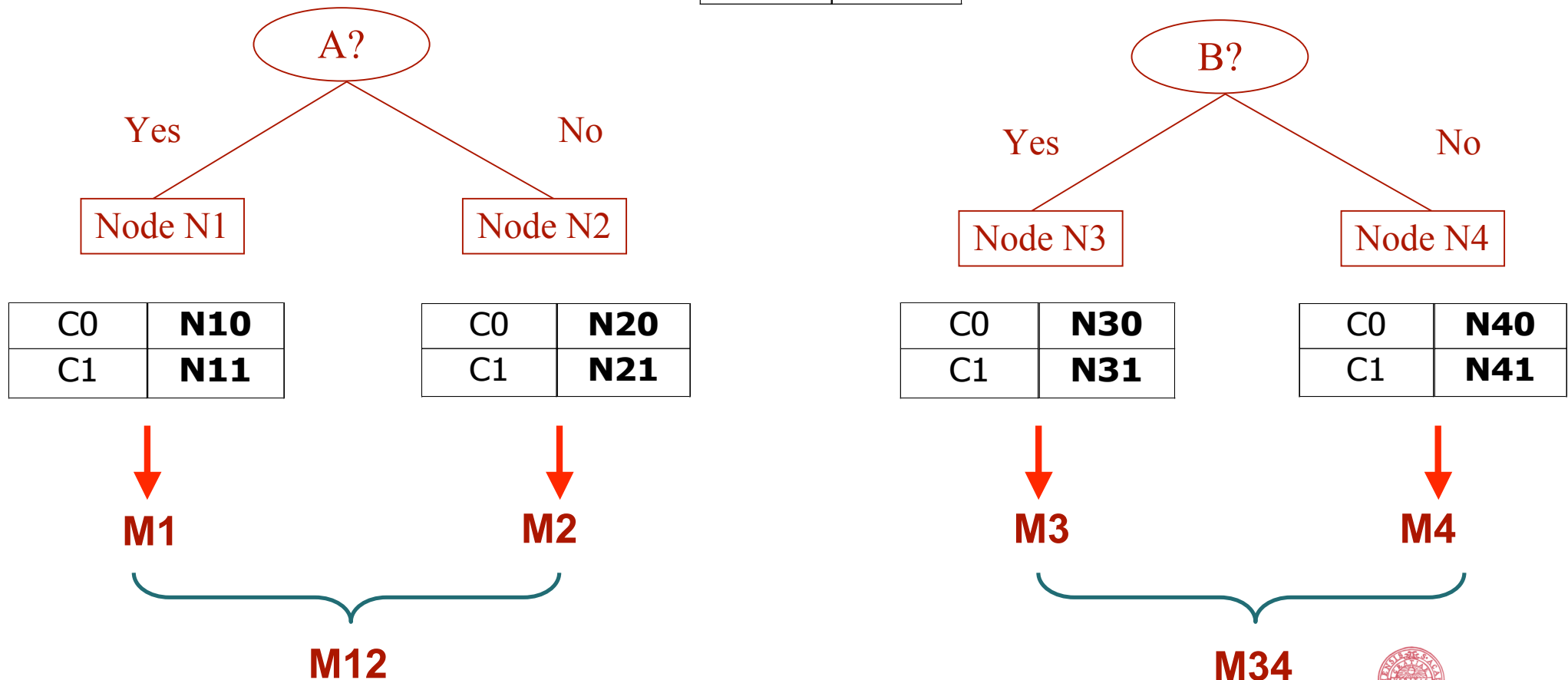
- Gini index
- Entropy
- Misclassification error

How to find the best split

Before Splitting:

C0	N00
C1	N01

→ **M0**



Gain = M0 – M12 vs M0 – M34

Measure of impurity: GINI

- Gini index for a given node t :

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

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

- Maximum $(1 - 1/nc)$ 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

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

C1	0
C2	6

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

$$Gini = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$$

C1	1
C2	5

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

$$Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278$$

C1	2
C2	4

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

$$Gini = 1 - (2/6)^2 - (4/6)^2 = 0.444$$

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,

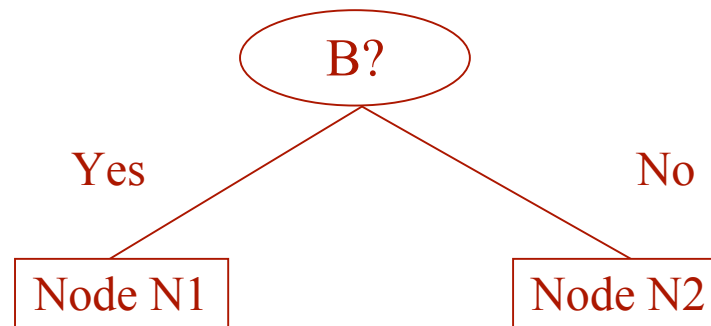
$$GINI_{split} = \sum_{i=1}^k \frac{n_i}{n} GINI(i)$$

where,

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

Binary attributes: computing GINI index

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



Gini(N1)

$$= 1 - (5/6)^2 - (2/6)^2$$

$$= 0.194$$

Gini(N2)

$$= 1 - (1/6)^2 - (4/6)^2$$

$$= 0.528$$

	N1	N2
C1	5	1
C2	2	4
Gini=0.333		

	Parent
C1	6
C2	6
Gini = 0.500	

Gini(Children)

$$= 7/12 * 0.194 +$$

$$5/12 * 0.528$$

$$= 0.333$$



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)

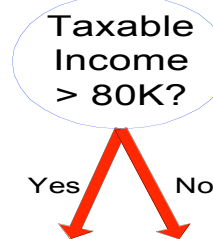
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

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

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 $A \geq v$
- 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.

<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



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

Sorted Values Split Positions		Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No		
		Taxable Income																					
		60		70		75		85		90		95		100		120		125		220			
		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		<u>0.300</u>		0.343		0.375		0.400		0.420	



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

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

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

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:

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

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

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

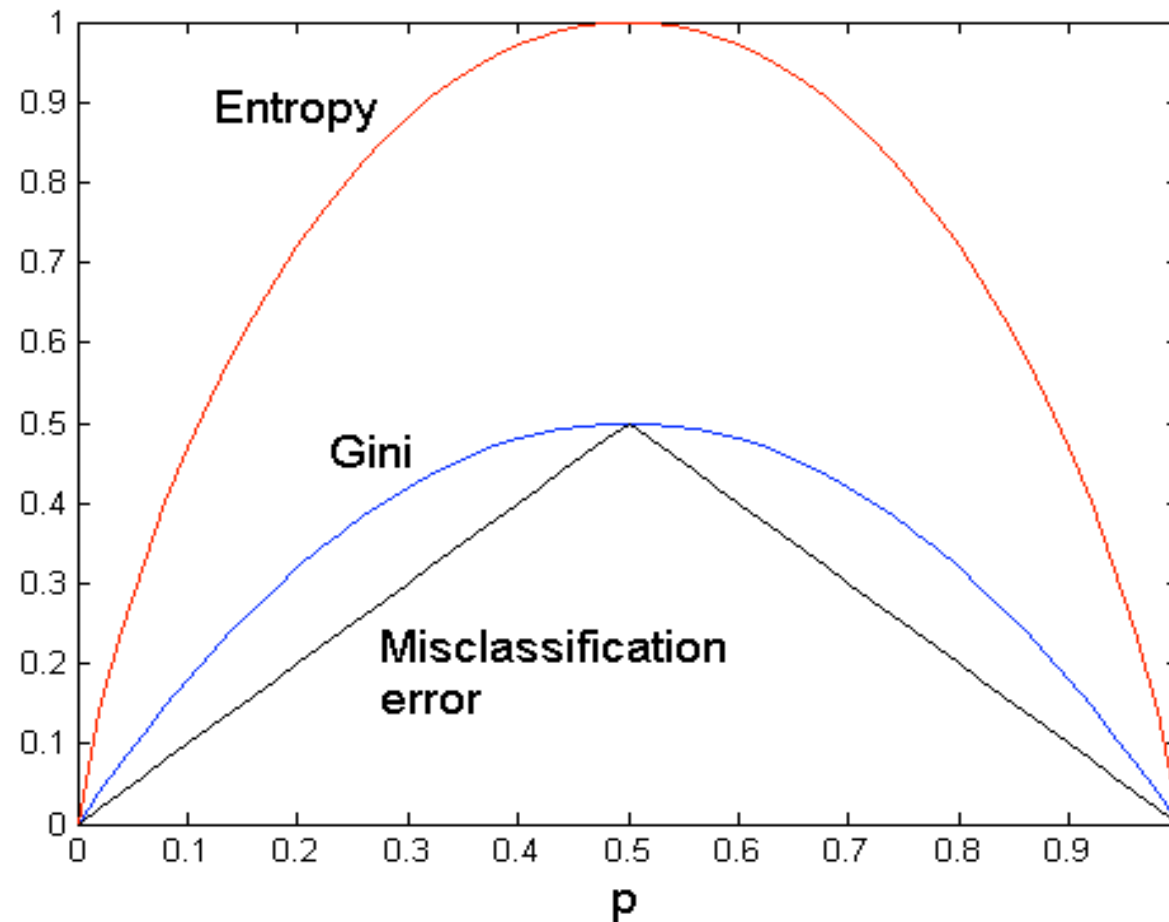
$$Error = 1 - \max(1/6, 5/6) = 1 - 5/6 = 1/6$$

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 (e.g. to small resulting class set)

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

Model evaluation (ch 4.5)

- Metrics for Performance Evaluation
 - How to evaluate the performance of a model?
- Methods for Performance Evaluation
 - How to obtain reliable estimates?



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		
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...

ACTUAL CLASS	PREDICTED CLASS	
	Class=Yes	Class=No
	Class=Yes	Class=No
	a (TP)	b (FN)
	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
	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		
ACTUAL CLASS	C(ilj)	+	-
	+	-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	
ACTUAL CLASS		Class=Yes
	Class=Yes	a
	Class=No	c

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

Accuracy is proportional to cost if

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

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

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

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



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 $C(\text{Yes}|\text{Yes})$ & $C(\text{Yes}|\text{No})$
- Recall is biased towards $C(\text{Yes}|\text{Yes})$ & $C(\text{No}|\text{Yes})$
- F-measure is biased towards all except $C(\text{No}|\text{No})$

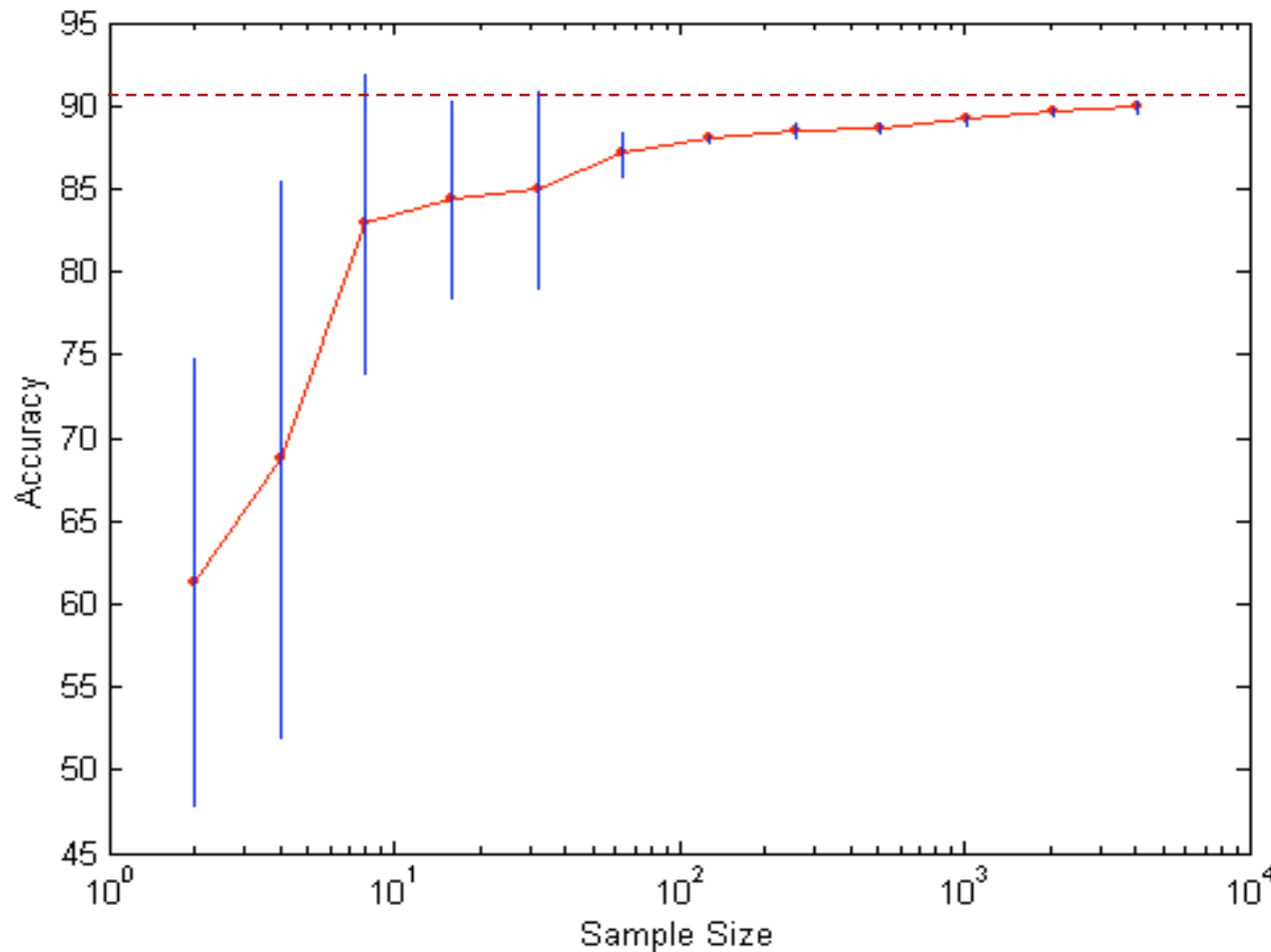
$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

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