

Data Mining

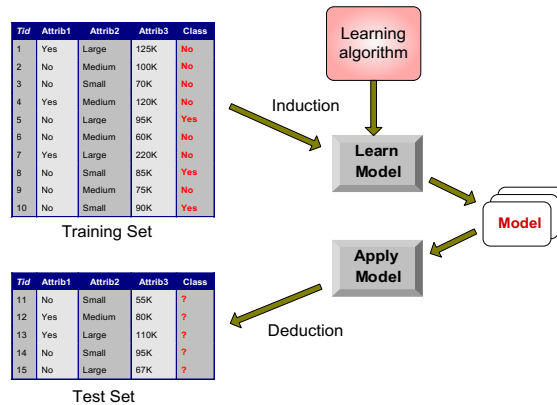
Classification: Basic Concepts, Decision Trees, and Model Evaluation

Thanks to Tan, Steinbach, Kumar

Classification: Definition

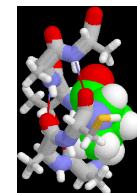
- Given a collection of records (*training set*)
 - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
 - 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



Examples of Classification Task

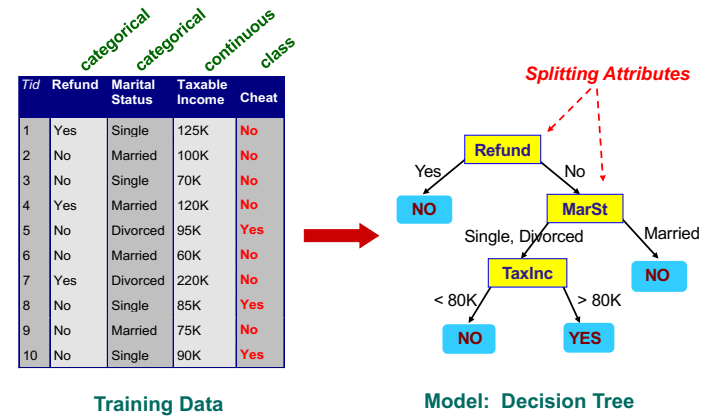
- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc



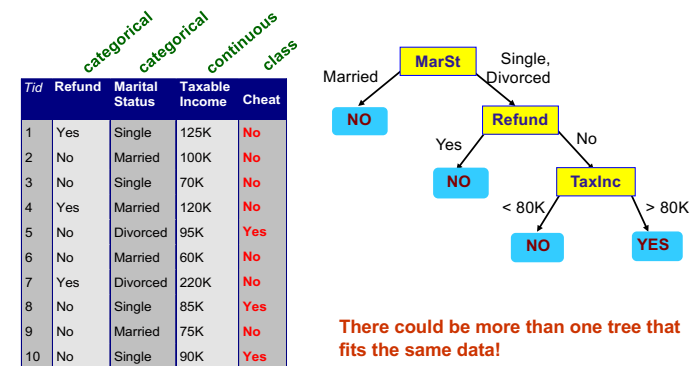
Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning, Instance-Based Learning
- Neural Networks, Deep Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines
- Ensemble Methods
- Concept Lattice based Methods
- ...

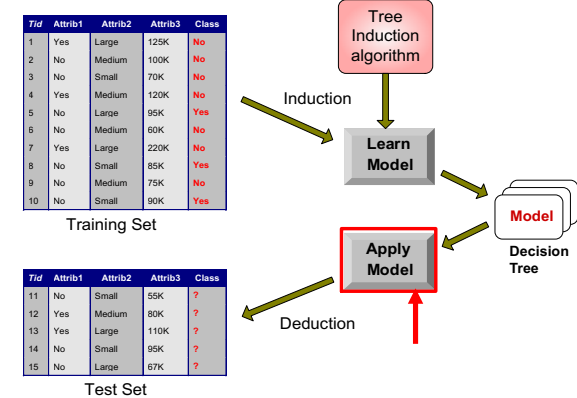
Example of a Decision Tree



Another Example of Decision Tree

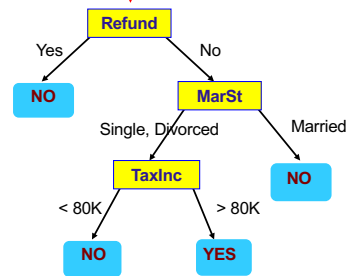


Decision Tree Classification Task



Apply Model to Test Data

Start from the root of tree.



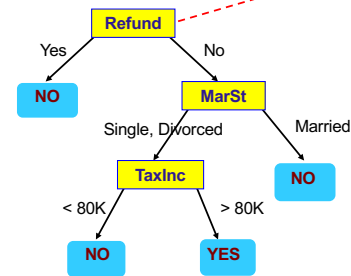
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

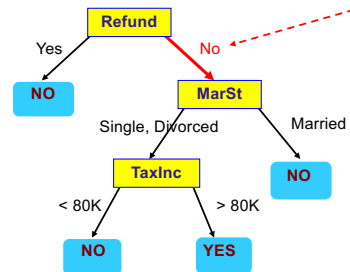
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

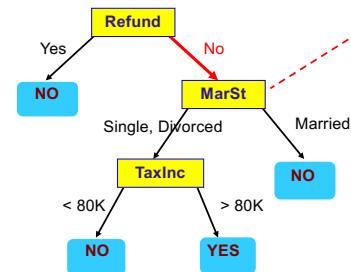
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



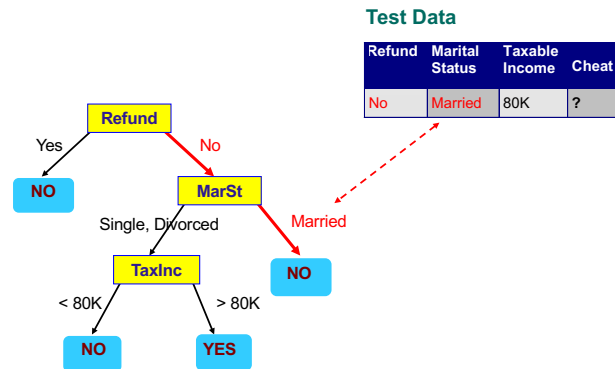
Apply Model to Test Data

Test Data

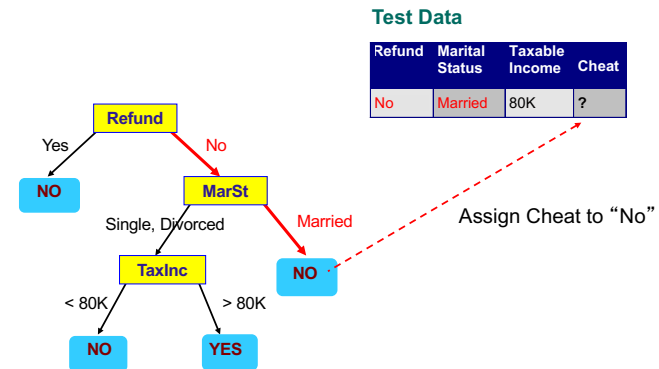
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



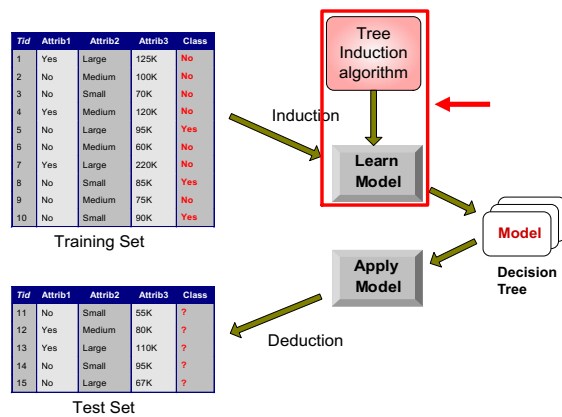
Apply Model to Test Data



Apply Model to Test Data



Decision Tree Classification Task



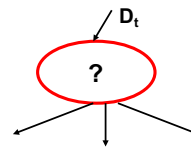
Decision Tree Induction

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

General Structure of Hunt's Algorithm

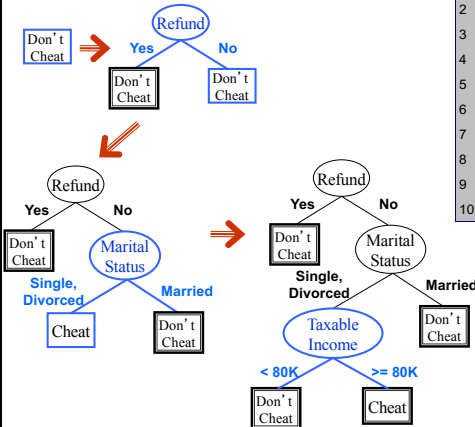
- Let D_t be the set of training records that reach a node t
- General Procedure:
 - If D_t contains records that belong to the same class y_t , then t is a leaf node labeled as y_t
 - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
 - If D_t contains records that belong 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



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?

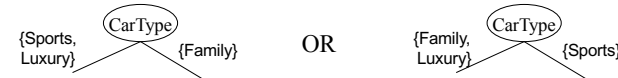
- Depends on attribute types
 - Nominal
 - Ordinal
 - Continuous
- Depends on number of ways to split
 - 2-way split
 - Multi-way split

Splitting Based on Nominal Attributes

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

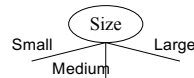


- **Binary split:** Divides values into two subsets. Need to find optimal partitioning.

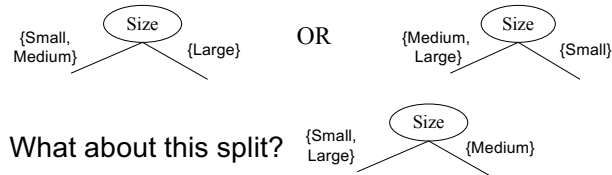


Splitting Based on Ordinal Attributes

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



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

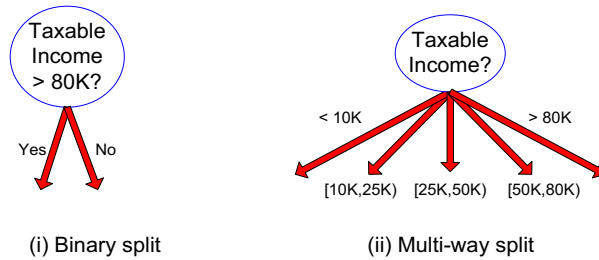


- What about this split?

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

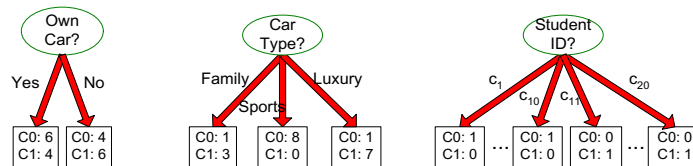


Tree Induction

- 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

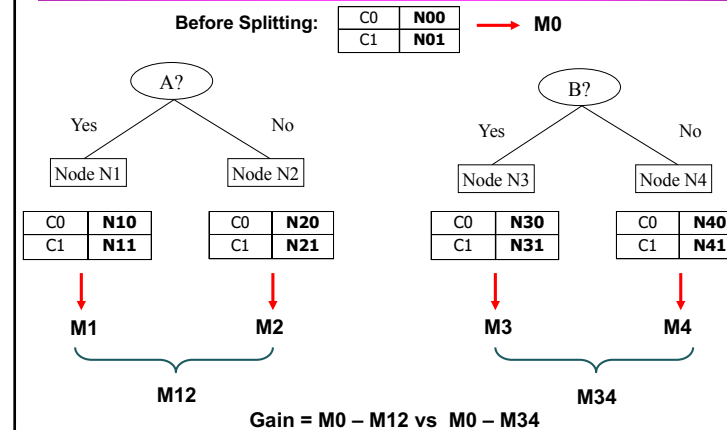
C0: 9
C1: 1

Homogeneous,
Low degree of impurity

Measures of Node Impurity

- Gini Index
- Entropy
- Misclassification error

How to Find the Best Split



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/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

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

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

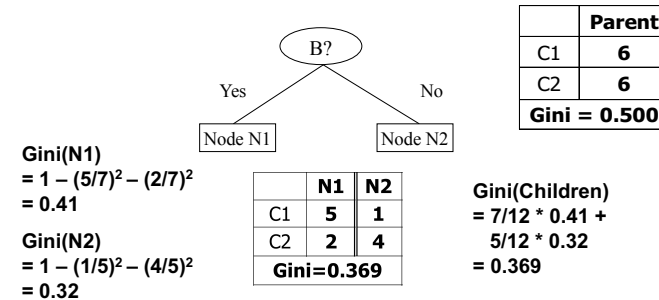
- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

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

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

Binary Attributes: Computing GINI Index

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



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)

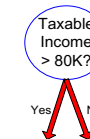
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	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.

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																		
		60	70	75	85	90	95	100	120	125	220									
Sorted Values	→	55	65	72	80	87	92	97	110	122	172	230								
Split Positions	→	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>							
	Yes	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0			
	No	0	7	1	6	2	5	3	4	3	4	3	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								

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	P(C1) = 0/6 = 0	P(C2) = 6/6 = 1
C2	6	Entropy = - 0 log 0 - 1 log 1 = - 0 - 0 = 0	
C1	1	P(C1) = 1/6	P(C2) = 5/6
C2	5	Entropy = - (1/6) log ₂ (1/6) - (5/6) log ₂ (5/6) = 0.65	
C1	2	P(C1) = 2/6	P(C2) = 4/6
C2	4	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

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

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

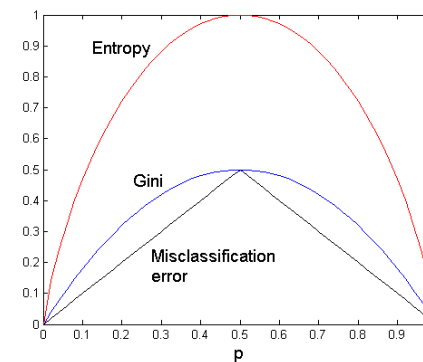
C1	2
C2	4

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

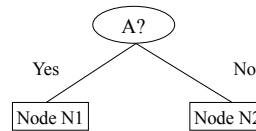
$$Error = 1 - \max(2/6, 4/6) = 1 - 4/6 = 1/3$$

Comparison among Splitting Criteria

For a 2-class problem:



Misclassification Error vs Gini



	Parent
C1	7
C2	3
Gini = 0.42	

$$\begin{aligned} \text{Gini}(N1) &= 1 - (3/3)^2 - (0/3)^2 \\ &= 0 \end{aligned}$$

$$\begin{aligned} \text{Gini}(N2) &= 1 - (4/7)^2 - (3/7)^2 \\ &= 0.489 \end{aligned}$$

	N1	N2
C1	3	4
C2	0	3
Gini=0.342		

$$\begin{aligned} \text{Gini(Children)} &= 3/10 * 0 \\ &+ 7/10 * 0.489 \\ &= 0.342 \end{aligned}$$

Gini improves !!

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

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.

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.
- 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...

	PREDICTED CLASS	
	Class=Yes	Class=No
	Class=Yes	Class=No
ACTUAL CLASS	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		
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

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
	Class=Yes	Class=No
ACTUAL CLASS	a	b
	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
	Class=Yes	Class=No
ACTUAL CLASS	p	q
	q	p

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

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 $C(\text{No}|\text{No})$

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

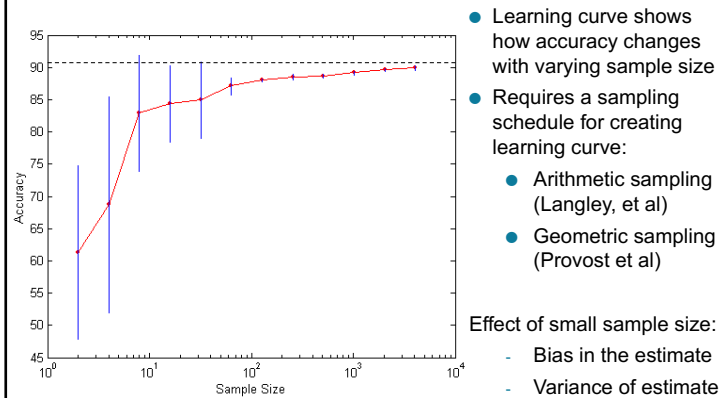
Model Evaluation

- 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



Methods of Estimation

- Holdout
 - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
 - Repeated holdout
- Cross validation
 - 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
 - Sampling with replacement

Model Evaluation

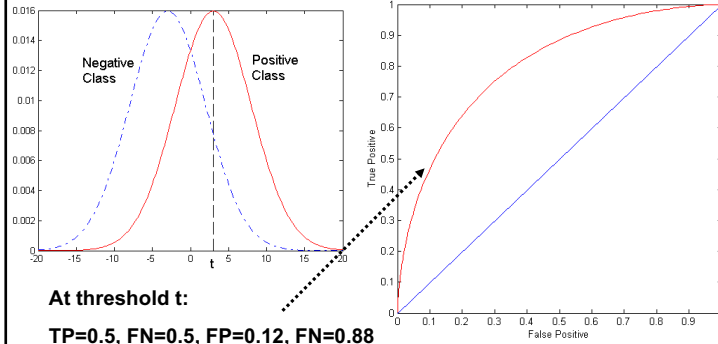
- Metrics for Performance Evaluation
 - How to evaluate the performance of a model?
- Methods for Performance Evaluation
 - How to obtain reliable estimates?
- **Methods for Model Comparison**
 - How to compare the relative performance among competing models?

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

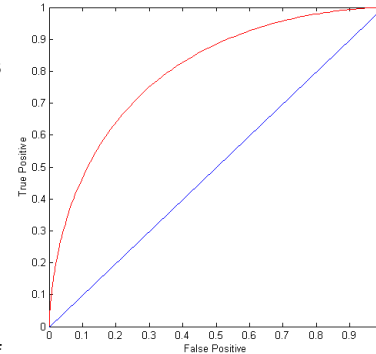
- 1-dimensional data set containing 2 classes (positive and negative)
- any points located at $x > t$ is classified as positive



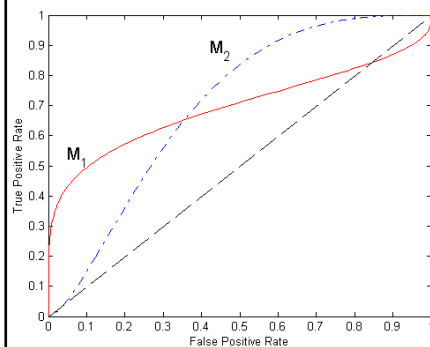
ROC Curve

(TP,FP):

- (0,0): declare everything to be negative class
- (1,1): declare everything 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 \geq	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:

