



BITS Pilani

Pilani | Dubai | Goa | Hyderabad

S2-20_DSECFZC415

Classification and Prediction



- *The slides presented here are obtained from the authors of the books and from various other contributors. I hereby acknowledge all the contributors for their material and inputs.*
- *I have added and modified a few slides to suit the requirements of the course.*

Classification

Classification

- Classification involves dividing up objects so that each is assigned to one of a number of mutually exhaustive and exclusive categories known as *classes*
- Many practical decision-making tasks can be formulated as classification problems
 - customers who are likely to buy or not buy a particular product in a supermarket
 - people who are at high, medium or low risk of acquiring a certain illness
 - student projects worthy of a distinction, merit, pass or fail grade
 - objects on a radar display which correspond to vehicles, people, buildings or trees
 - people who closely resemble, slightly resemble or do not resemble someone seen committing a crime
 - houses that are likely to rise in value, fall in value or have an unchanged value in 12 months' time
 - people who are at high, medium or low risk of a car accident in the next 12 months
 - people who are likely to vote for each of a number of political parties (or none)
 - the likelihood of rain the next day for a weather forecast (very likely, likely, unlikely, very unlikely).

Classification vs. Prediction

- Classification
 - predicts categorical class labels (discrete or nominal)
 - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Prediction
 - models continuous-valued functions, i.e., predicts unknown or missing values

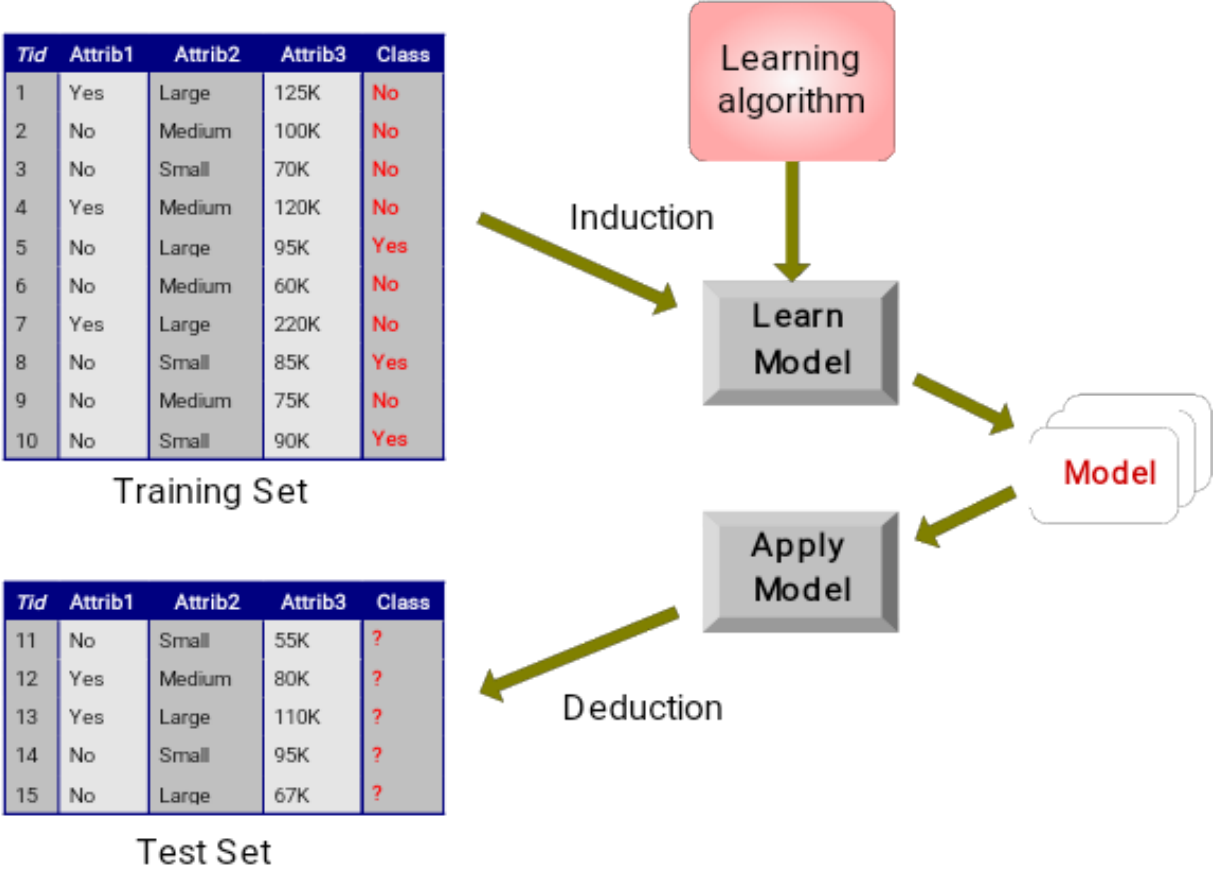
Supervised vs. Unsupervised Learning

- Supervised learning (classification)
 - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
 - New data is classified based on the training set
- Unsupervised learning (clustering)
 - The class labels of training data is unknown
 - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Classification—A Two-Step Process

- Model construction: describing a set of predetermined classes
 - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
 - The set of tuples used for model construction is training set
 - The model is represented as classification rules, decision trees, or mathematical formulae
- Model usage: for classifying future or unknown objects
 - Estimate accuracy of the model
 - The known label of test sample is compared with the classified result from the model
 - Accuracy rate is the percentage of test set samples that are correctly classified by the model
 - Test set is independent of training set, otherwise over-fitting will occur
 - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

Illustrating Classification Task



Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Neural Networks
 - computational networks that simulate the decision process in neurons (networks of nerve cell)
- Naïve Bayes and Bayesian Belief Networks
 - uses the *probability theory* to find the most likely of the possible classifications
- Support Vector Machines
 - fits a boundary to a region of points that are all alike; uses the boundary to classify a new point

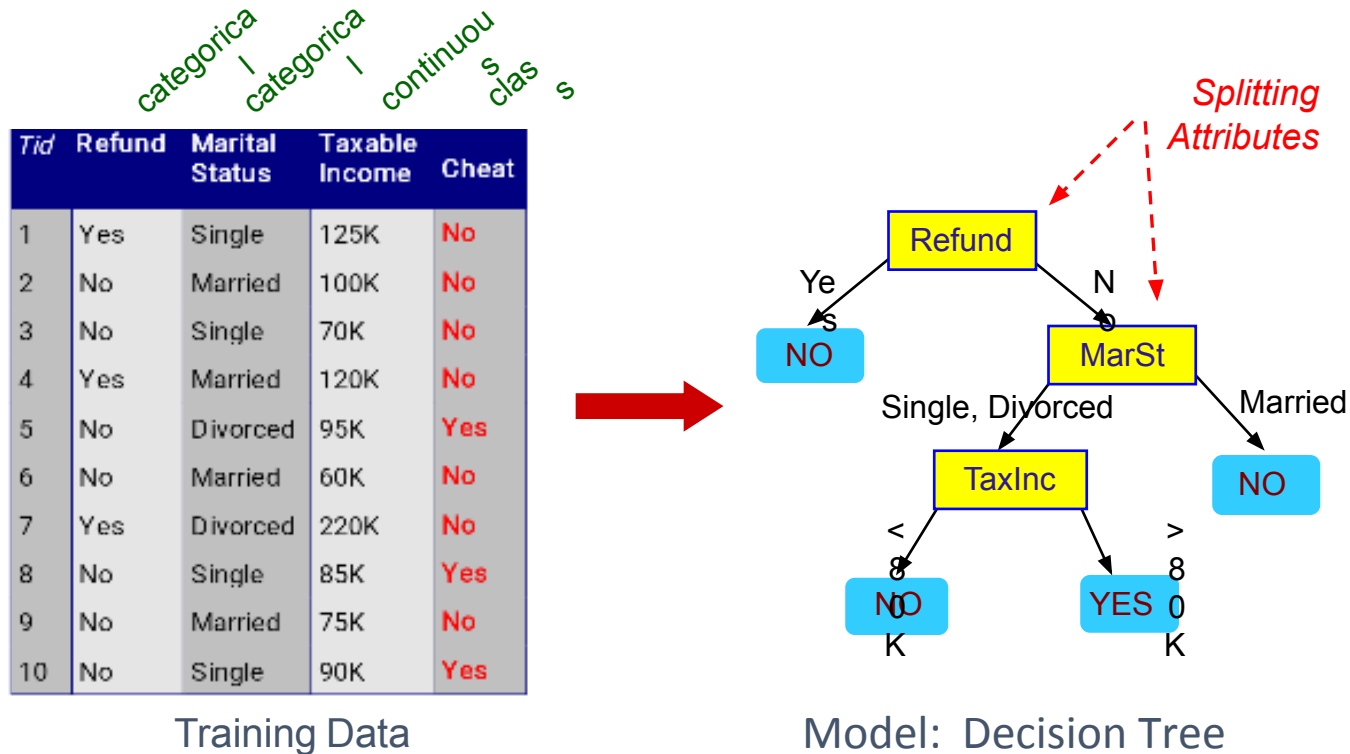
Lazy vs. Eager Learning

- Lazy vs. eager learning
 - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
 - Eager learning (the above discussed methods): Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
 - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
 - Eager: must commit to a single hypothesis that covers the entire instance space

Lazy Learner: Instance-Based Methods

- Instance-based learning:
 - Store training examples and delay the processing (“lazy evaluation”) until a new instance must be classified
- Typical approaches
 - k -nearest neighbor approach
 - Instances represented as points in a Euclidean space.
 - Locally weighted regression
 - Constructs local approximation
 - Case-based reasoning
 - Uses symbolic representations and knowledge-based inference

Example of a Decision Tree



Another Example of Decision Tree

categorical

categorical

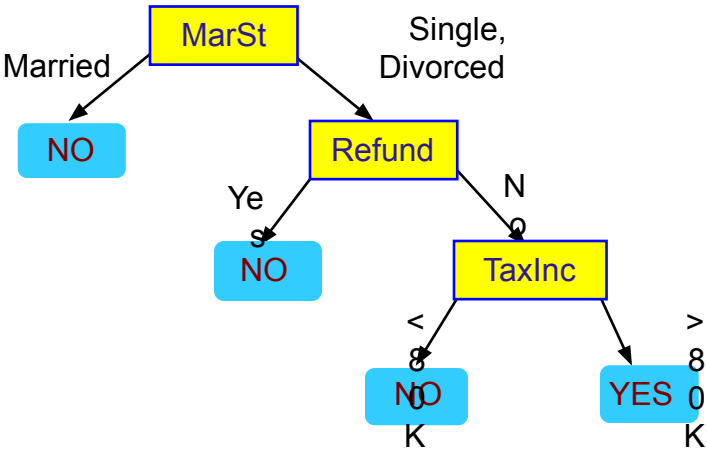
continuous

classes

classes

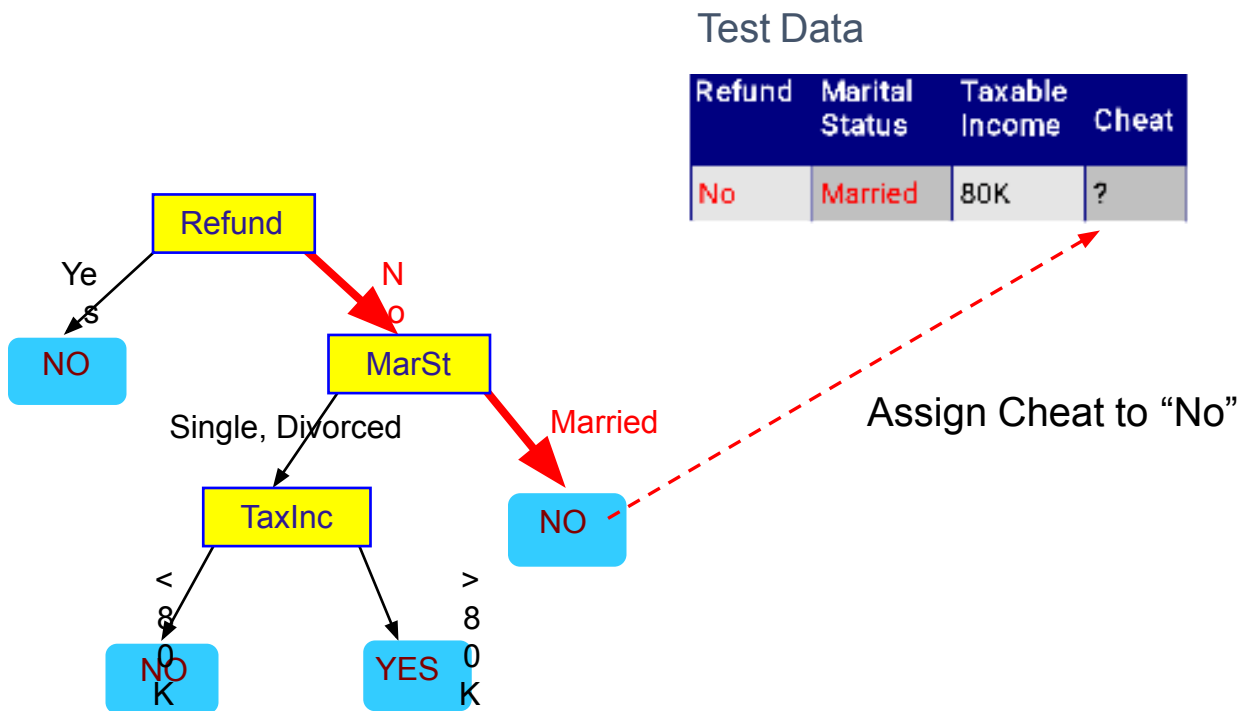
classes

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

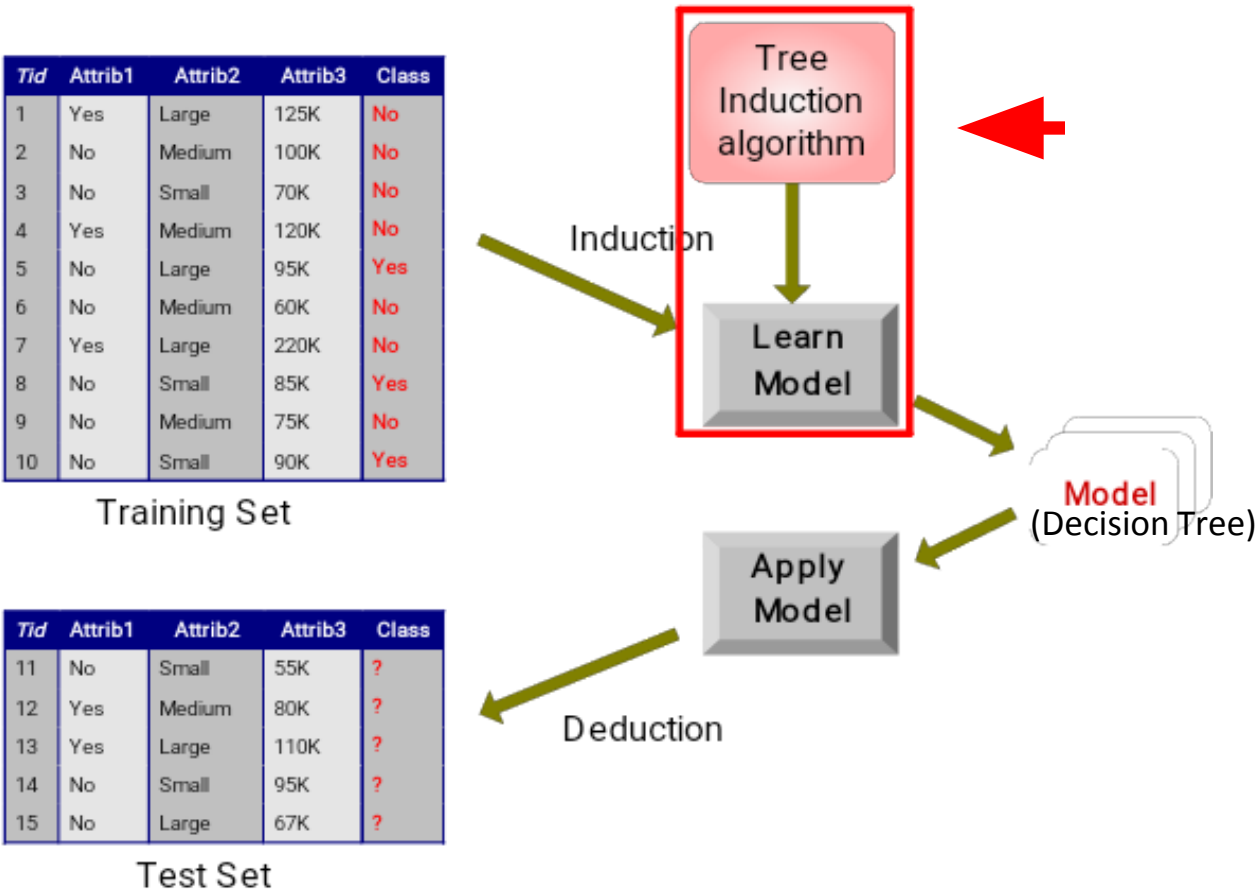


There could be more than one tree that fits the same data!

Apply Model to Test Data



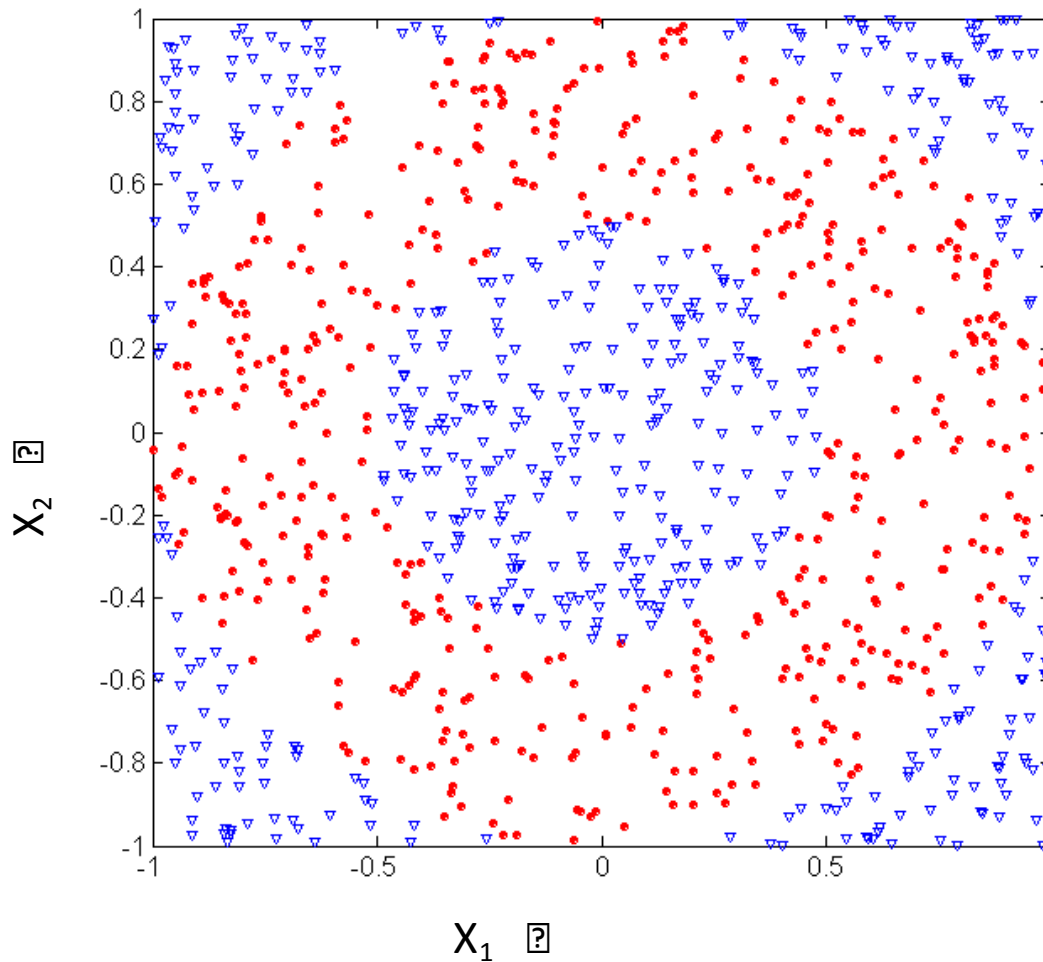
Decision Tree Classification Task



Issues: Evaluating Classification Methods

- Accuracy
 - classifier accuracy: predicting class label
 - predictor accuracy: guessing value of predicted attributes
- Speed
 - time to construct the model (training time)
 - time to use the model (classification/prediction time)
- Robustness: handling noise and missing values
- Scalability: efficiency in disk-resident databases
- Interpretability
 - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Underfitting and Overfitting (Example)



500 circular and 500 triangular data points.

Circular points:

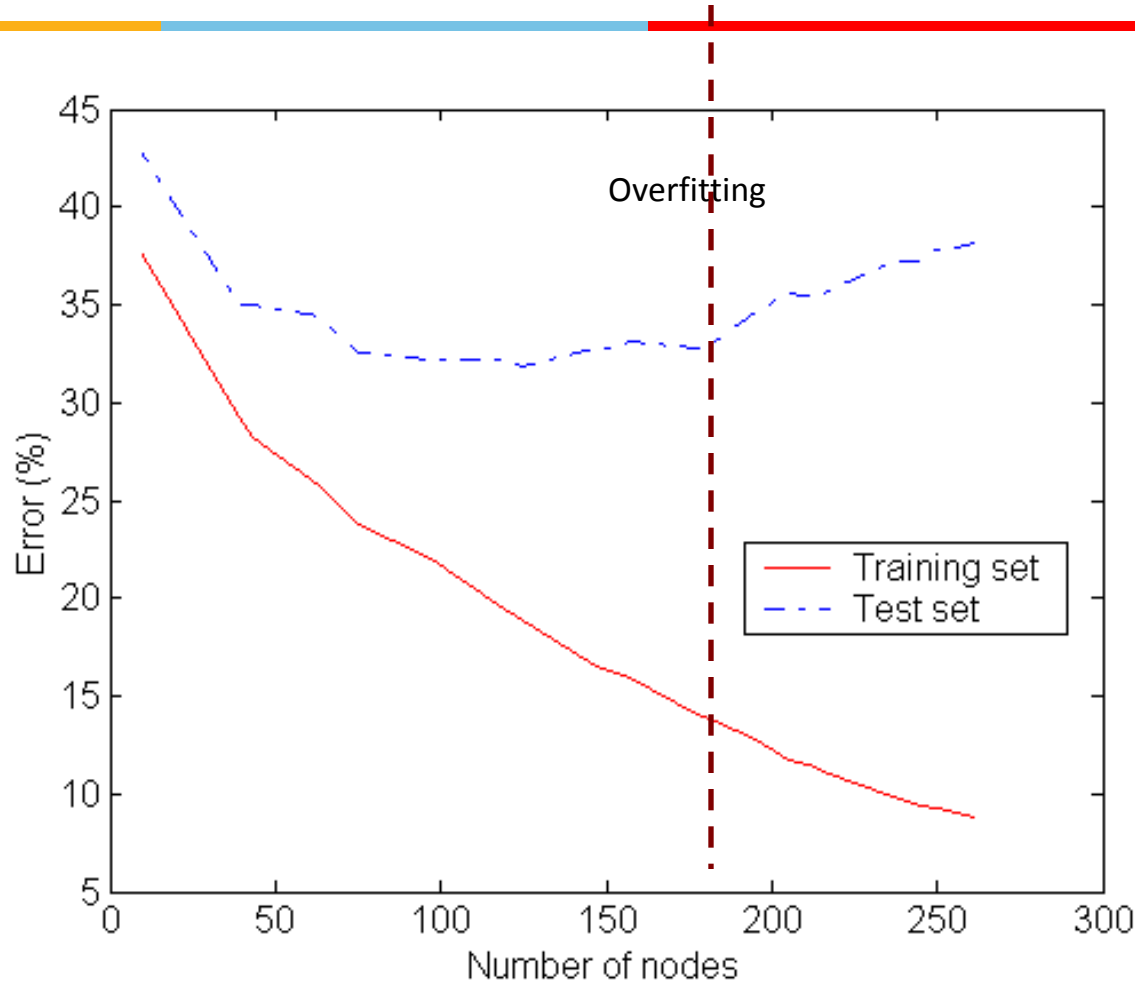
$$0.5 \leq \sqrt{x_1^2 + x_2^2} \leq 1$$

Triangular points:

$$\sqrt{x_1^2 + x_2^2} > 0.5 \text{ or}$$

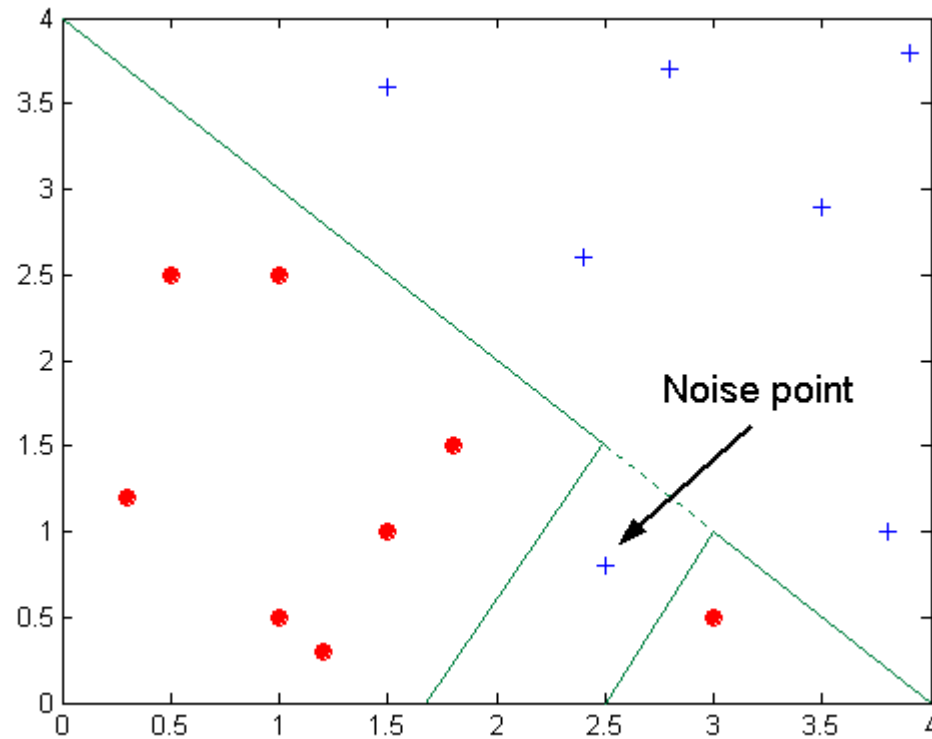
$$\sqrt{x_1^2 + x_2^2} < 1$$

Underfitting and Overfitting



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

Overfitting due to Noise



Decision boundary is distorted by noise point

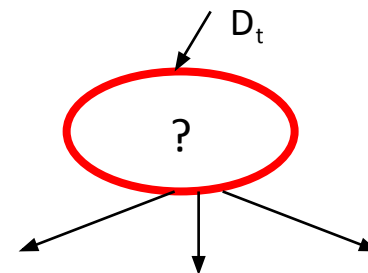
Decision Tree Based Classification

- Decision trees are intuitive and frequently used data mining technique for Classification
- For an analyst, they are easy to set up and for a business user they are easy to interpret.
- A decision tree model is a decision flowchart where an attribute is tested in each node and ends in a leaf node where a prediction is made.
- There are many algorithms for decision tree induction such as Hunt's Algorithm, CART, ID3, C4.5, SLIQ,SPRINT

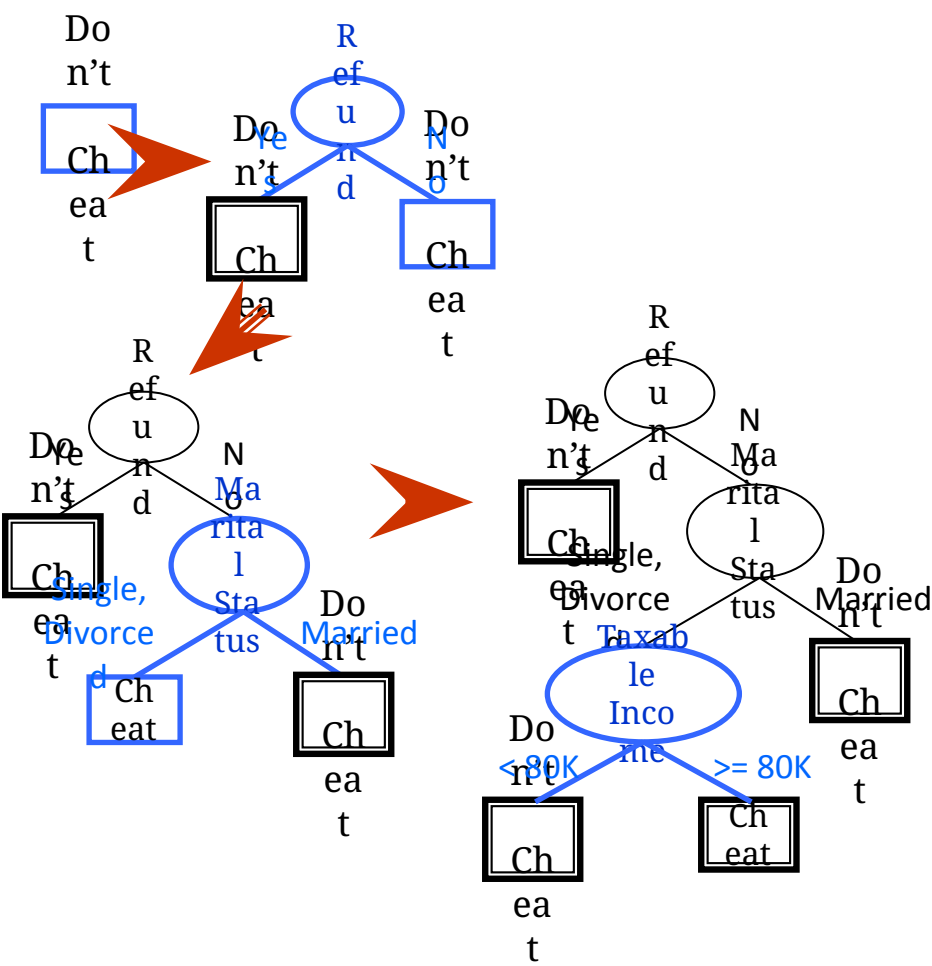
Hunt's Algorithm - Structure

- Hunt's algorithm is among the earliest. More complex algorithms were built upon it.
- It grows a decision tree in a recursive fashion by partitioning the training records into successively purer subsets
- 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 - Example



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

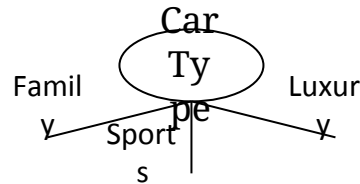
- Greedy strategy.
 - Split the records based on an attribute test that optimizes certain criterion.
- Issues
 - Determine how to split the records
 - How to specify the attribute test condition?
 - How to determine the best split?
 - Determine when to stop splitting

How to Specify Test Condition?

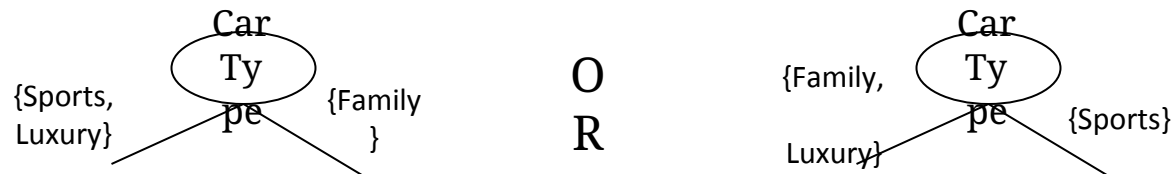
- Depends on attribute types
 - Nominal
 - Ordinal
 - Continuous
- Depends on number of ways to split
 - 2-way split
 - Multi-way split

Splitting Based on Nominal Attributes

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

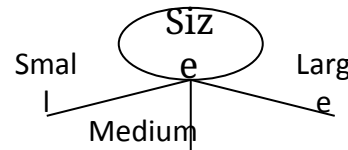


- **Binary split:** Divides values into two subsets.
Need to find optimal partitioning.

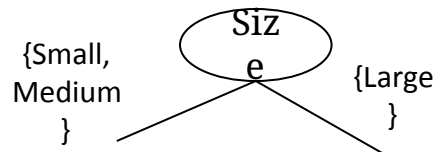


Splitting Based on Ordinal Attributes

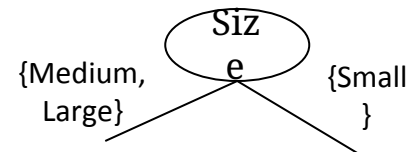
- **Multi-way split:** Use as many partitions as distinct values.



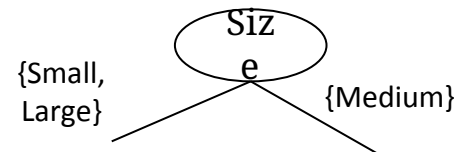
- **Binary split:** Divides values into two subsets.
Need to find optimal partitioning.



OR



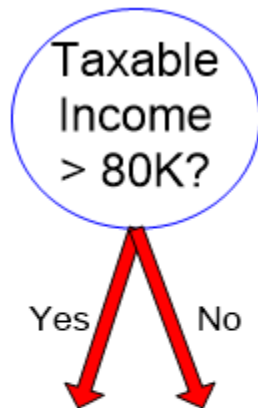
- What about this split?



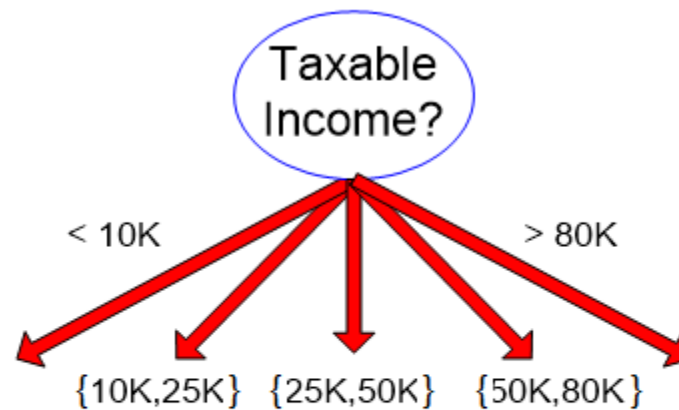
Splitting Based on Continuous Attributes

- 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



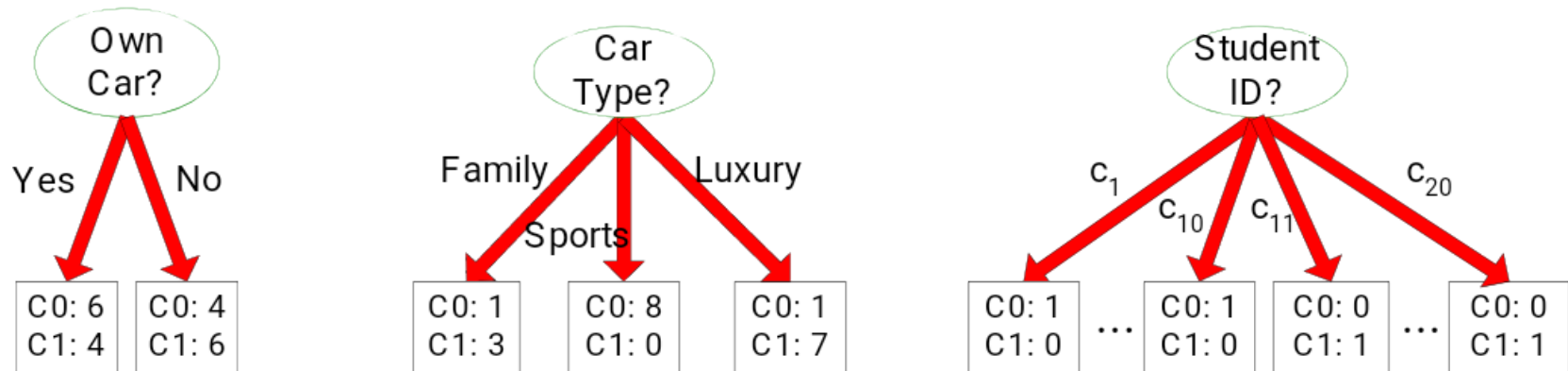
(i) Binary split



(ii) Multi-way split

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

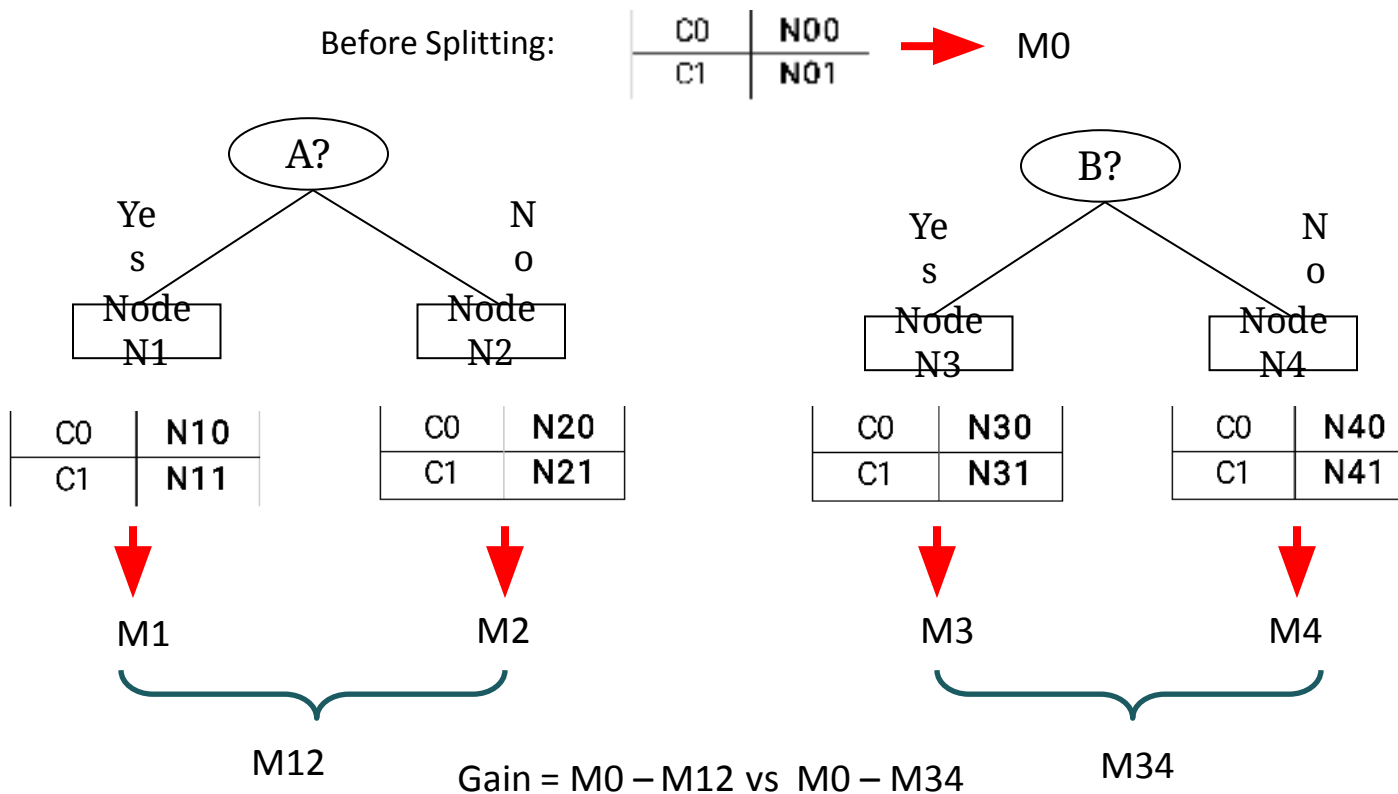
C0: 9
C1: 1

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	C1	1	C1	2	C1	3
C2	6	C2	5	C2	4	C2	3
Gini= 0.000		Gini= 0.278		Gini= 0.444		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

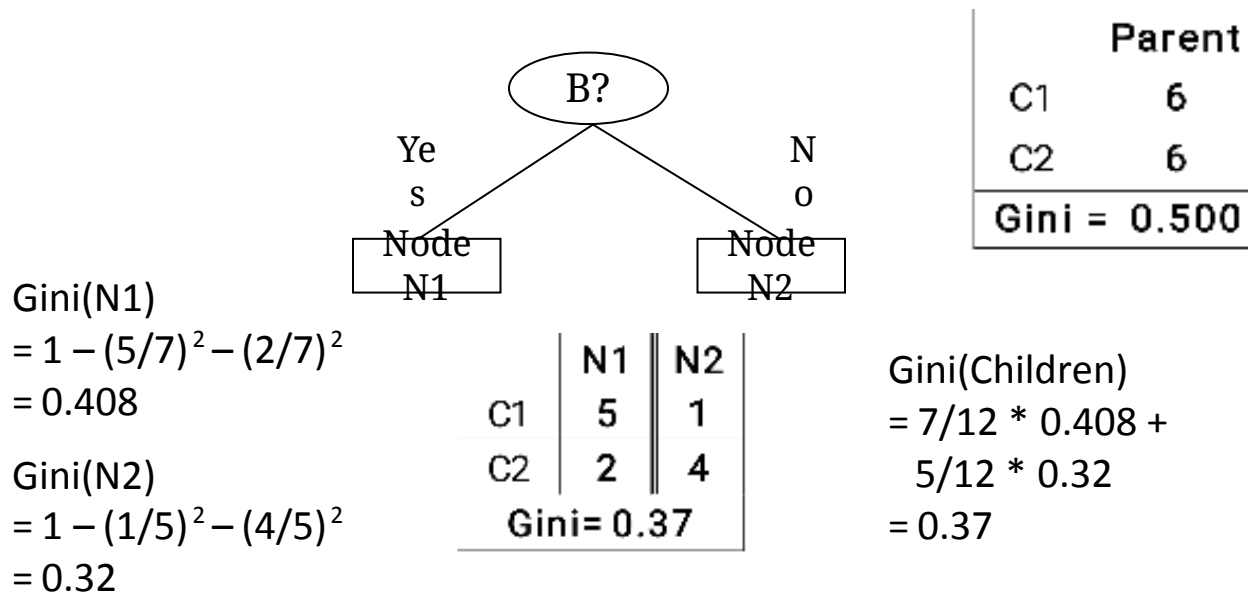
- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

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

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.



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)

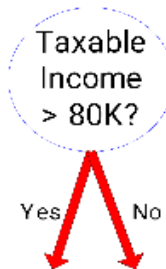
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	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

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

- 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 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

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

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

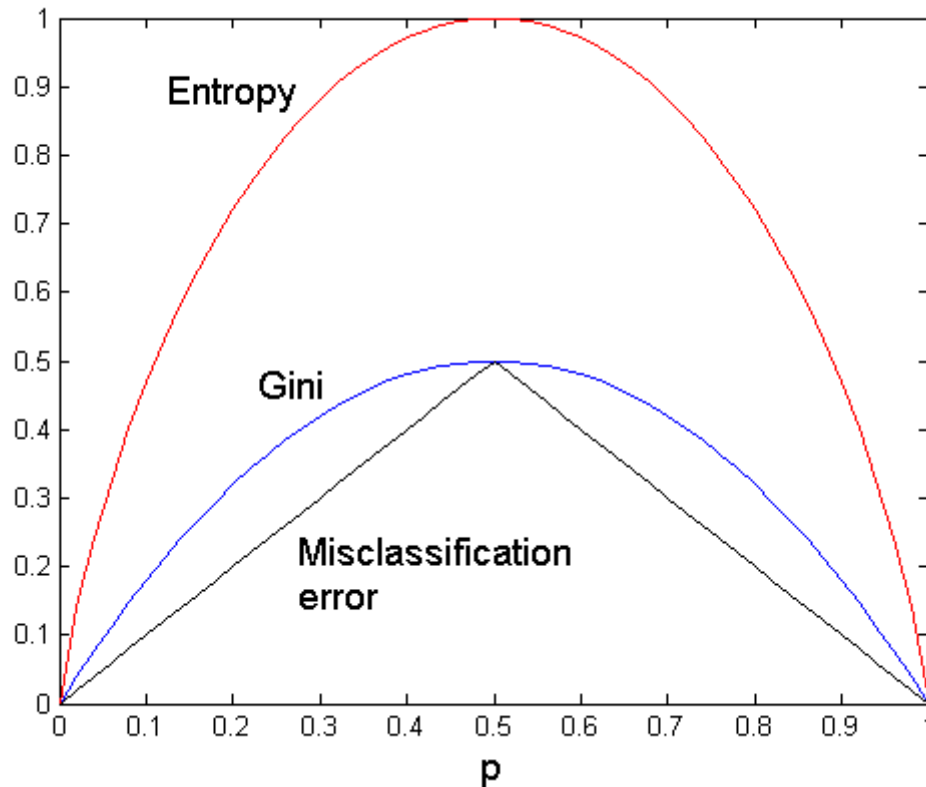
C1	2
C2	4

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

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

Comparison among Splitting Criteria

For a 2-class problem:



Gain Ratio

Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i , estimated by $|C_{i,D}|/|D|$
- **Expected information** (entropy) needed to classify a tuple in D :

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i)$$

- **Information** needed (after using A to split D into v partitions) to classify D :

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j)$$

- **Information gained** by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute Selection: Information Gain

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31... 40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31... 40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31... 40	medium	no	excellent	yes
31... 40	high	yes	fair	yes
>40	medium	no	excellent	no

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

Attribute Selection: Information Gain

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2)$$

$$= 0.694$$

age	p_i	n_i	$I(p_i, n_i)$
≤ 30	2	3	0.971
31... 40	4	0	0
> 40	3	2	0.971

$\frac{5}{14}I(2,3)$ means “age ≤ 30 ” has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit_rating) = 0.048$$

Gain Ratio for Attribute Selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

$$SplitInfo_A(D) = - \sum_{j=1}^v \frac{|D_j|}{|D|} \times \log_2 \left(\frac{|D_j|}{|D|} \right)$$

- GainRatio(A) = Gain(A)/SplitInfo(A)

- Ex.

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2 \left(\frac{4}{14} \right) - \frac{6}{14} \times \log_2 \left(\frac{6}{14} \right) - \frac{4}{14} \times \log_2 \left(\frac{4}{14} \right) = 1.557$$

- gain_ratio(income) = 0.029/1.557 = 0.019

- The attribute with the maximum gain ratio is selected as the splitting attribute

Refining Decision Tree Model

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

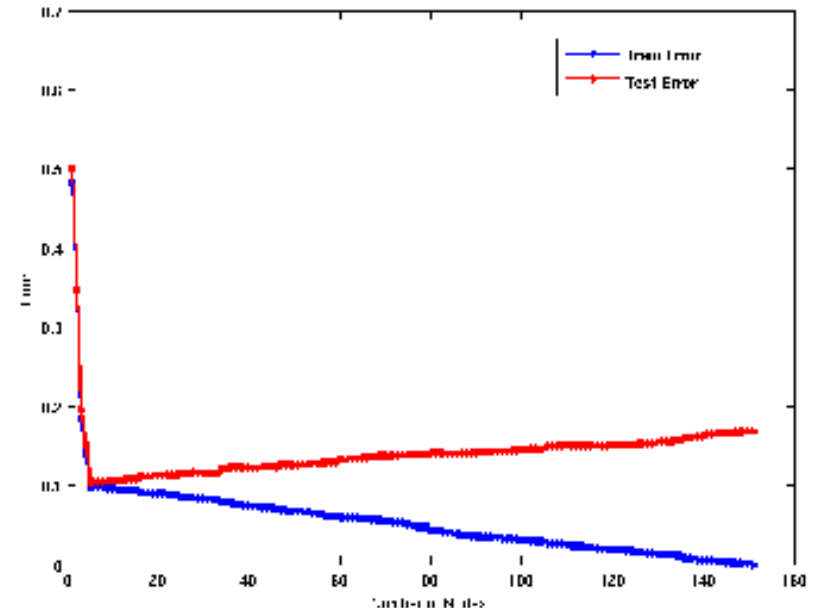
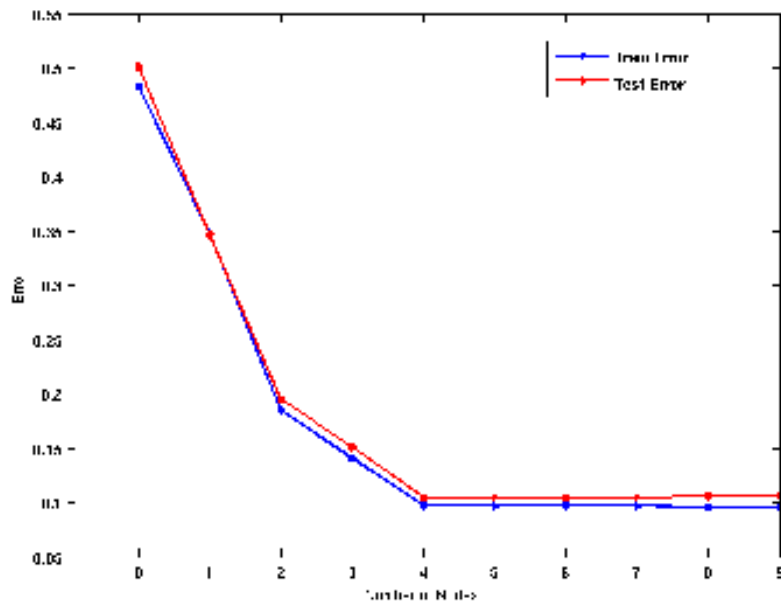
Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

Underfitting vs. Overfitting

- Underfitting results in decision trees that are too simple to solve the problem. They may offer superior interpretability.
- 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

Model Overfitting



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

Overfitting: when model is too complex, training error is small but test error is large

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
 - Stop the algorithm before it becomes a fully-grown tree
 - General 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 (for pre-pruning) :
 - 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(i.e. expected error of the model on previously unseen records) 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

Prescribed Text Books

	Author(s), Title, Edition, Publishing House
T1	Tan P. N., Steinbach M & Kumar V. "Introduction to Data Mining" Pearson Education
T2	Data Mining: Concepts and Techniques, Third Edition by Jiawei Han, Micheline Kamber and Jian Pei Morgan Kaufmann Publishers
R2	Principles of Data Mining, Second Edition by Max Bramer Springer © 2013
R1	Predictive Analytics and Data Mining: Concepts and Practice with RapidMiner by Vijay Kotu and Bala Deshpande Morgan Kaufmann Publishers

Thank You