Data Mining

Classification

Basic Concepts

Decision Trees

Model Evaluation

Slides by Tan, Steinbach, Kumar adapted by Pimprapai Thainiam

Topics

- **Introduction**
- Decision Tree Induction
- Model Overfitting
- Evaluating the Performance of a Classifier
  - Metrics for Performance Evaluation
  - Methods for Performance Evaluation

Classification: Definition

- A **classification technique** (or classifier) is a systematic approach used to build **classification models** from an input data set (**training set**) which is a collection of records (aka instance) where each record is characterized by a set of attributes ($\mathbf{x} = \{x_1, x_2, \dots, x_{n-1}\}$) and a special attribute which is the class attribute (aka category or target attribute) ($y = x_n$)

- **Classification Task:** Find a **classification model** using a **classification technique** that maps each attribute set $\mathbf{x}$ to one of the predefined class labels y where each classification technique employs a **learning algorithm** to find a classification model.

- **Goal:** To assign a class to a previously unseen record as accurately as possible using the classification model. 1. training set
2. validation set

- Usually, the given data set is divided into training and test sets where
  1) **A training set** is used to build the classification model.
  2) **A test set** is used to determine the accuracy of the model ~~(validate the model)~~.

Classification: Definition

- A Classification model is useful for the following purposes:

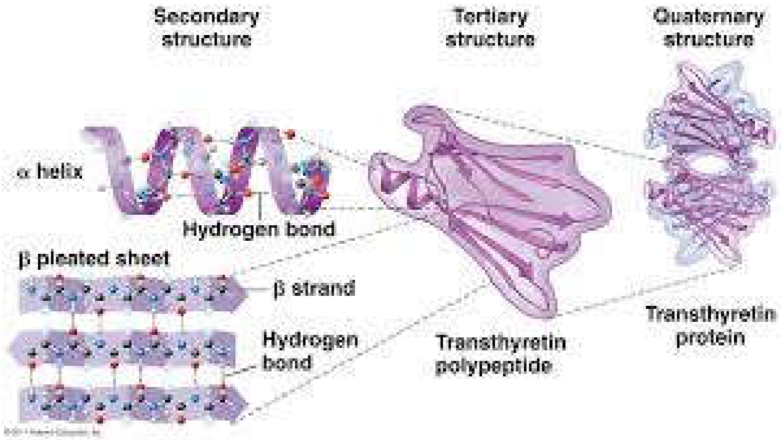
    1. Descriptive Modeling: A classification model can serve as an explanatory tool to distinguish between objects of different classes.

    2. Predictive Modeling: A classification model can also be used to predict the class label of unknown records where a classification model can be treated as a black box that automatically assigns a class label when presented with the attribute set of an unknown record.

- Classification techniques are most suited for predicting or describing data sets with binary or nominal categories.

Classification: Example

- Detecting spam email messages based upon the message header and content.
- Categorizing cells as malignant or benign based upon the results of MRI scans
- Classifying credit card transactions as legitimate or fraudulent based upon usage records
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil based upon protein structure data
- Categorizing news stories as finance, weather, entertainment, sports, based upon words containing in news



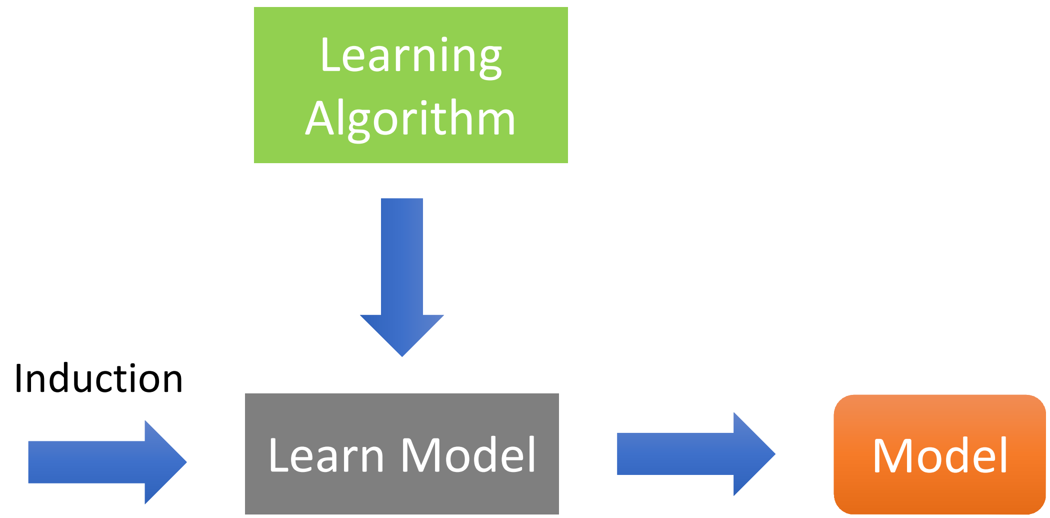
Illustrating Classification Task

Build a classification model

$\underbrace{\hspace{6em}}^{\mathbf{X}}$ $\quad y$

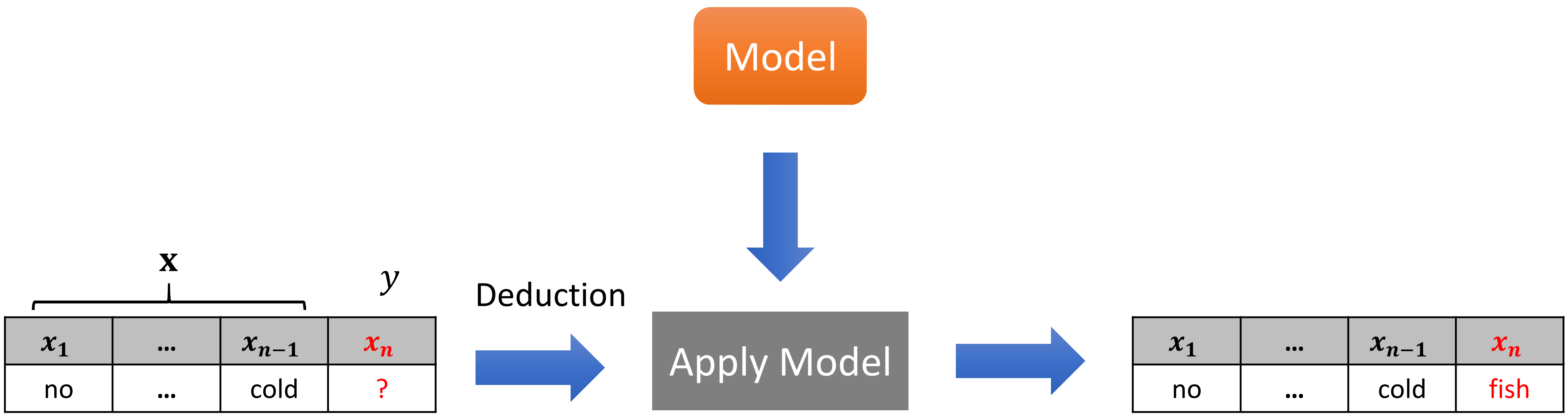
x_1	...	x_{n-1}	x_n
yes	...	warm	fish
⋮	⋮	⋮	⋮
no	...	cold	bird

Training Set

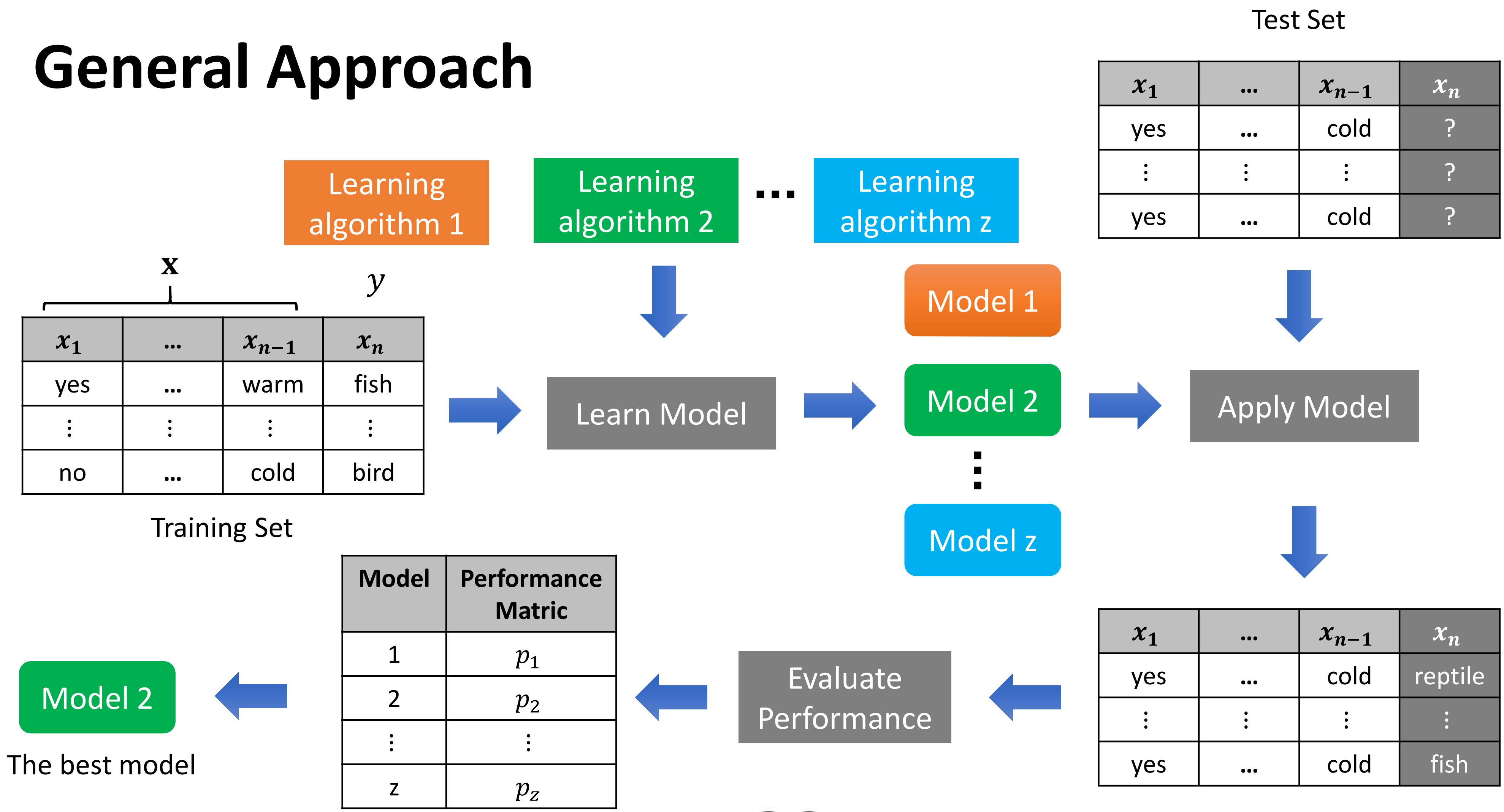


Illustrating Classification Task

Apply a classification model



General Approach



Training Set

x_1	...	x_{n-1}	x_n
yes	...	warm	fish
⋮	⋮	⋮	⋮
no	...	cold	bird

Test Set

x_1	...	x_{n-1}	x_n
yes	...	cold	?
⋮	⋮	⋮	?
yes	...	cold	?

x_1	...	x_{n-1}	x_n
yes	...	cold	reptile
⋮	⋮	⋮	⋮
yes	...	cold	fish

Model	Performance Matric
1	p_1
2	p_2
⋮	⋮
z	p_z

Classification Techniques

The well-known classification techniques:

1. Decision Tree based Methods
2. Rule-based Methods
3. Memory based reasoning
4. Neural Networks
5. Naïve Bayes and Bayesian Belief Networks
6. Support Vector Machines

Topics

- Introduction
- **Decision Tree Induction**
- Model Overfitting
- Evaluating the Performance of a Classifier
  - Metrics for Performance Evaluation
  - Methods for Performance Evaluation

How a Decision Tree Works

- One approach to classify a data object to a class is to pose a series of questions related to attributes of the data.
- That means a classification problem can be solved by asking a series of care fully crafted questions about the attributes of the test record.
- The series of questions and their possible answer can be organized in form of a **decision tree**, which is a hierarchical structure consisting of nodes and directed edges.
- The tree has three types of nodes:

  1) Root node → It has no incoming edges and zero or more outgoing edges.

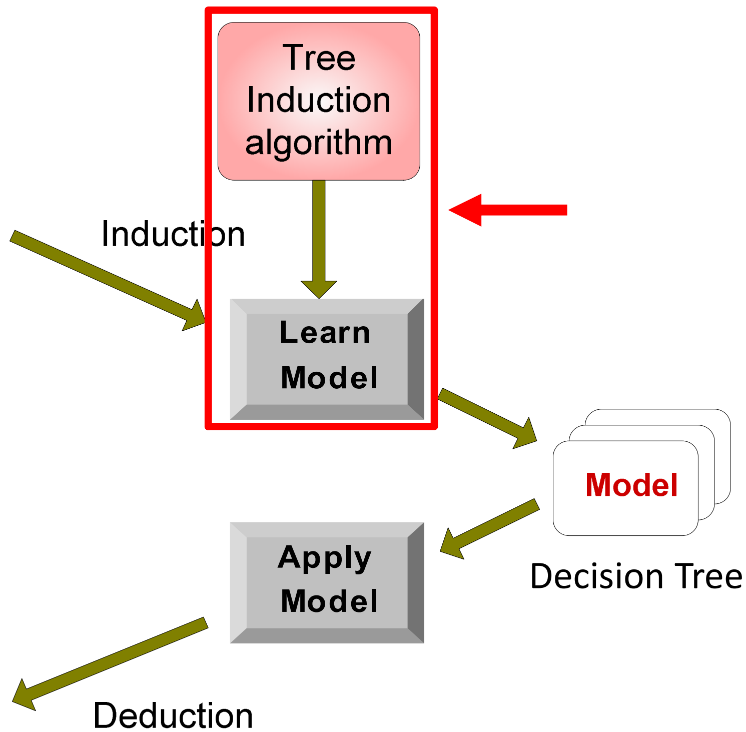
  2) Internal nodes → It has exactly one incoming edge and two or more outgoing edge. This node type contains attribute test condition.

  3) Leaf or terminal nodes → It has exactly one incoming edge and no outgoing edges. This type of nodes is assigned a class label.
- Root and internal nodes contain attribute test conditions to separate records that have different characteristics.

Decision Tree Classification Task: Induction

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

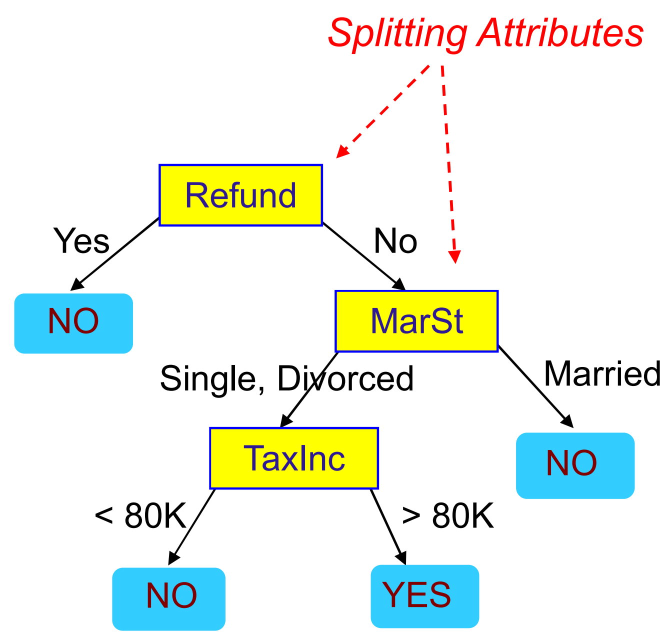
Example of Decision Tree

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

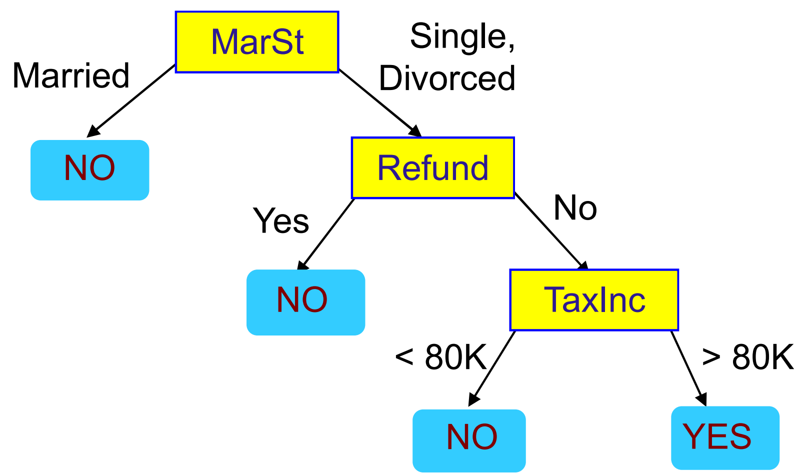
Training Set

Decision Tree Model



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**

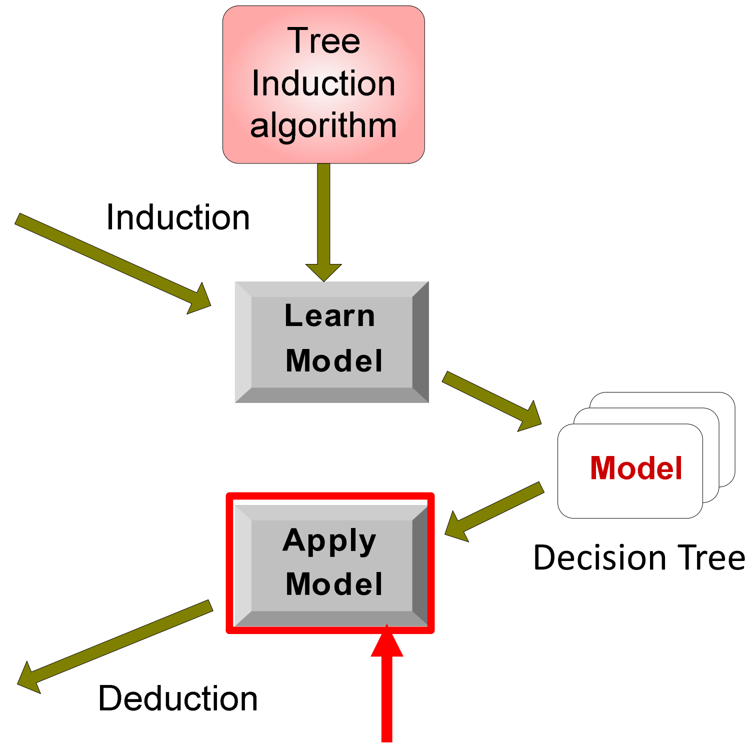


There could be more than one tree that fits the same data!

Decision Tree Classification Task: Deduction

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

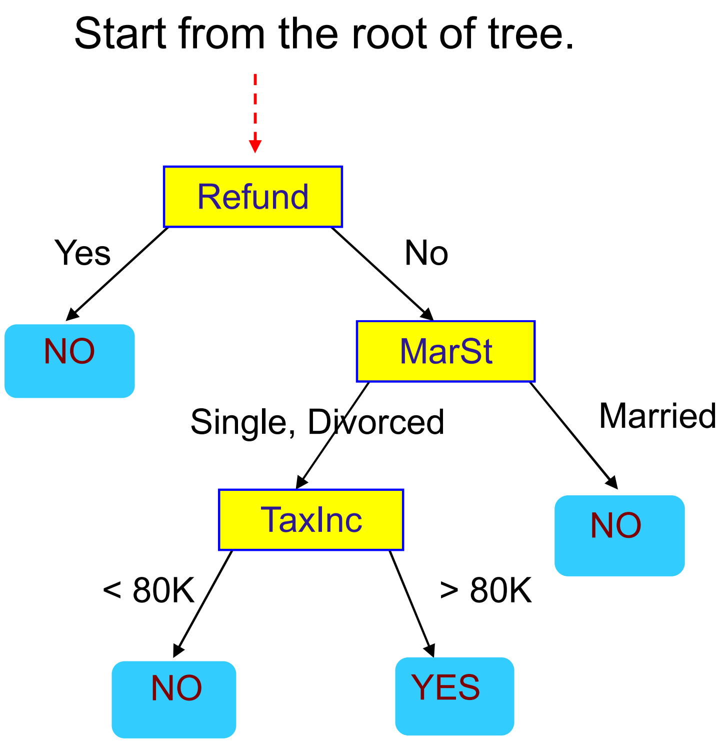
Training Set



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

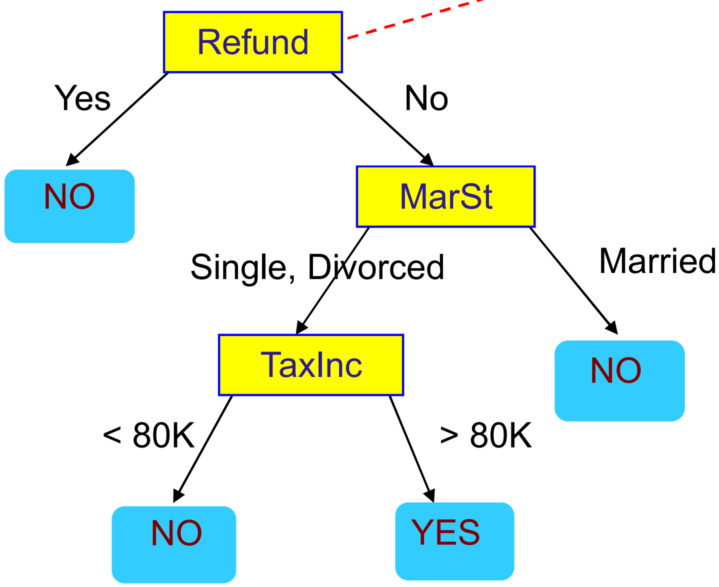
Apply Model to Test Record



Test Record

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Record



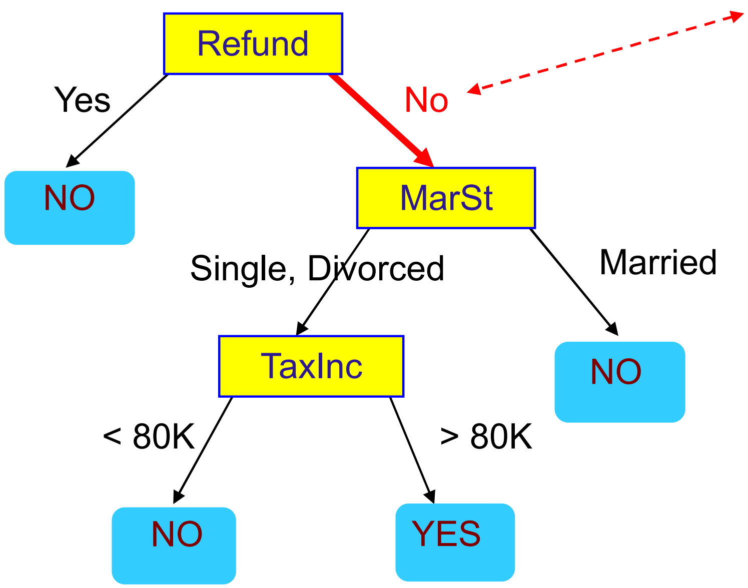
Test Record

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Record

Test Record

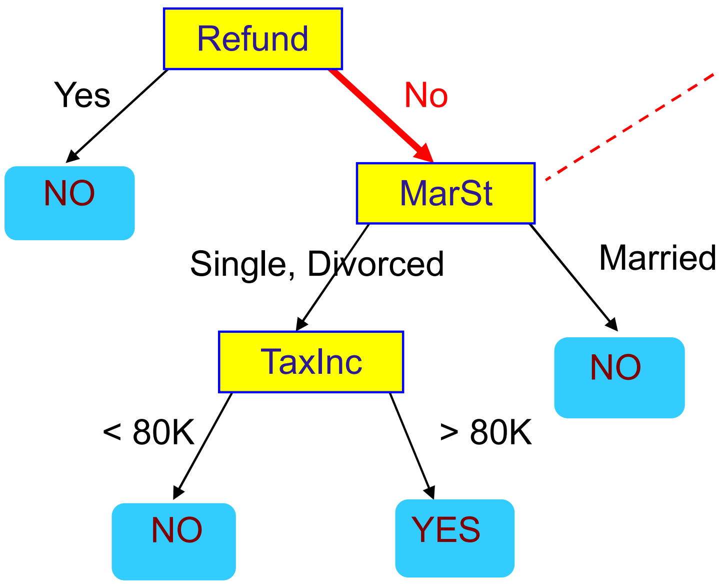
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Record

Test Record

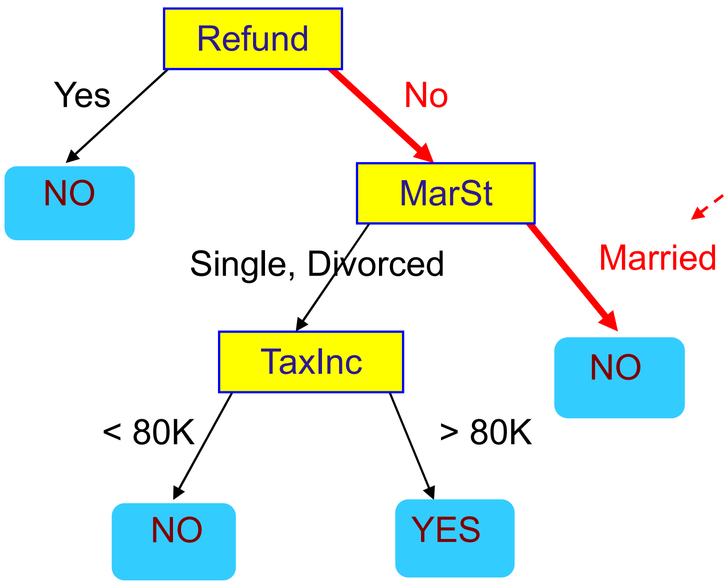
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



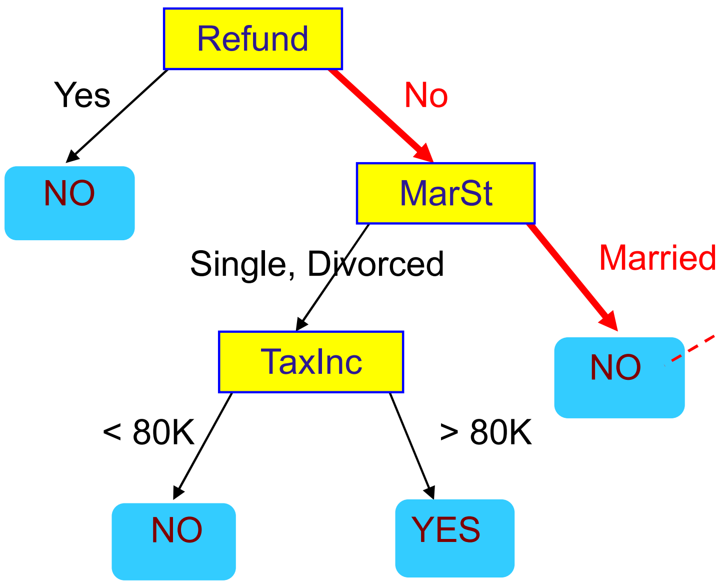
Apply Model to Test Record

Test Record

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Record



Test Record

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

How to Build a Decision Tree

- In principle, there are exponentially many decision trees that can be constructed from a given set of attributes, thus finding the optimal tree is computationally in feasible because of the exponential size of the search space.
- Efficient algorithms have been developed to induce a reasonably accurate, although suboptimal, decision tree in a reasonable amount of time.
- These algorithms usually employ a greedy strategy that grows a decision tree by making a series of locally optimum decision about which attribute to use for partitioning the data.

Knapsack Problem (maximum weight = 8)

item	Value	Weight
1	15	1
2	10	5
3	9	3
4	5	4

Try to maximize total value in your bag

Pick = {1, 2}

15 10 = 25

1 5 = 6

Optimal = {1, 3, 4}

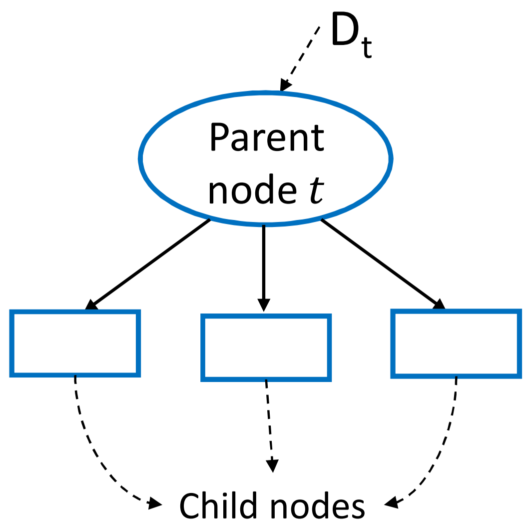
Decision Tree: Induction Algorithms

- **Hunt's Algorithm:** This is one of the earliest and it serves as a basis for some of the more complex algorithms.
- **CART:** Classification and Regression Trees is a non-parametric technique that uses the Gini index to determine which attribute should be split and then the process is continued recursively.
- **ID3, C4.5, C5.0:** They use the entropy of an attribute and picks the attribute with the highest reduction in entropy to determine which attribute should the data be split with first and then through a series of recursive functions that calculate the entropy of the node the process is continued until all the left nodes are pure.
- **CHAID:** Chi-squared Automatic Interaction Detection performs multi-level splits when computing classification trees.
- **MARS:** It extends decision trees to handle numerical data better.
- **SLIQ,SPRINT:** These algorithms are scalable algorithms that have been proposed to deal with the issues the greedy algorithms present.
- **ctree:** Conditional Inference Trees Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid overfitting. This approach results in unbiased predictor selection and does not require pruning.

General Structure of Hunt's Algorithm

- **Hunt's Algorithm** 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 are associated with node t (reach node t) and $y = \{y_1, y_2, \dots, y_c\}$ be the class labels

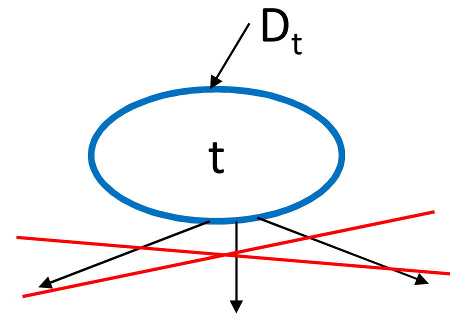
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



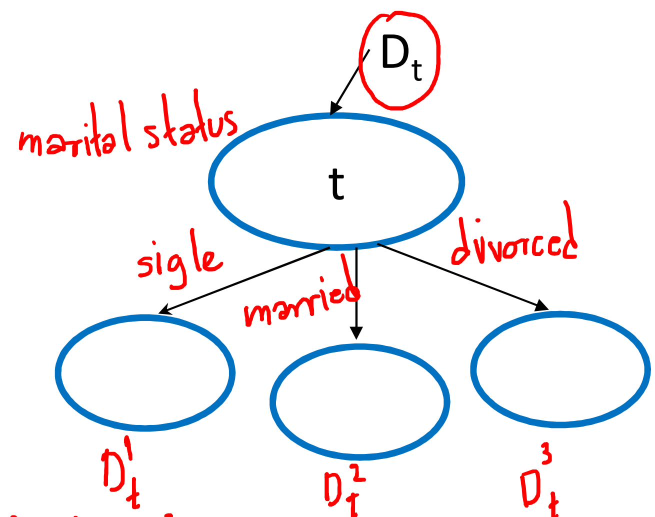
General Structure of Hunt's Algorithm

A recursive definition of Hunt's Algorithm:

Step 1: If D_t contains records that belong the same class y_t, then t is a **leaf node** labeled as y_t



Step 2: If D_t contains records that belong to more than one class, an attribute test condition is selected to partition the records D_t into smaller subsets. A child node is created for each outcome of the test condition and the record in D_t are distributed to the children based on the outcomes. Recursively apply the procedure to each child node.



$D_t = \{ D_t^1, D_t^2, D_t^3 \}$

binary ← 3 attributes + a class attribute

$2^3 = 8$ → unique class label X

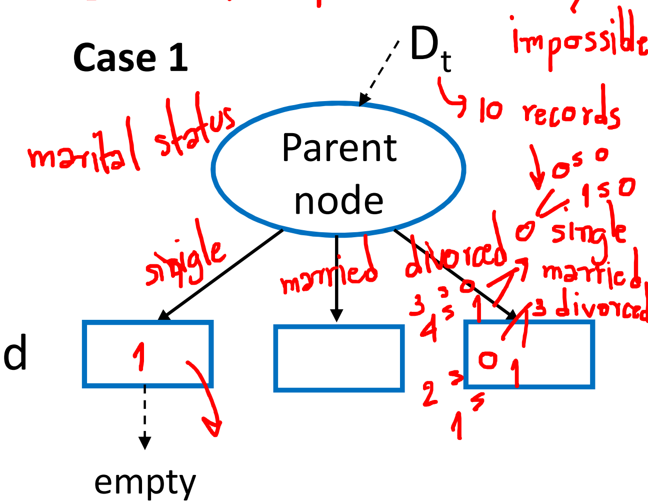
General Structure of Hunt's Algorithm

- **Hunt's Algorithm** will work if every combination of attribute values is present in the training data and each combination has a unique class label. → too stringent
- Additional conditions are needed to handle the following cases:

Case 1: If a child node is an empty set → The node is declared a leaf node with the same class label as the majority class of training records associated with its parent node. (majority class of D_t)

Case 2: If all the record associated with D_t have identical attribute values (except for the class attribute) → It is impossible to split further, the node is declare a leaf node with the same class label as the majority class of training records associated with this node. (majority class of D_t)

Case 1

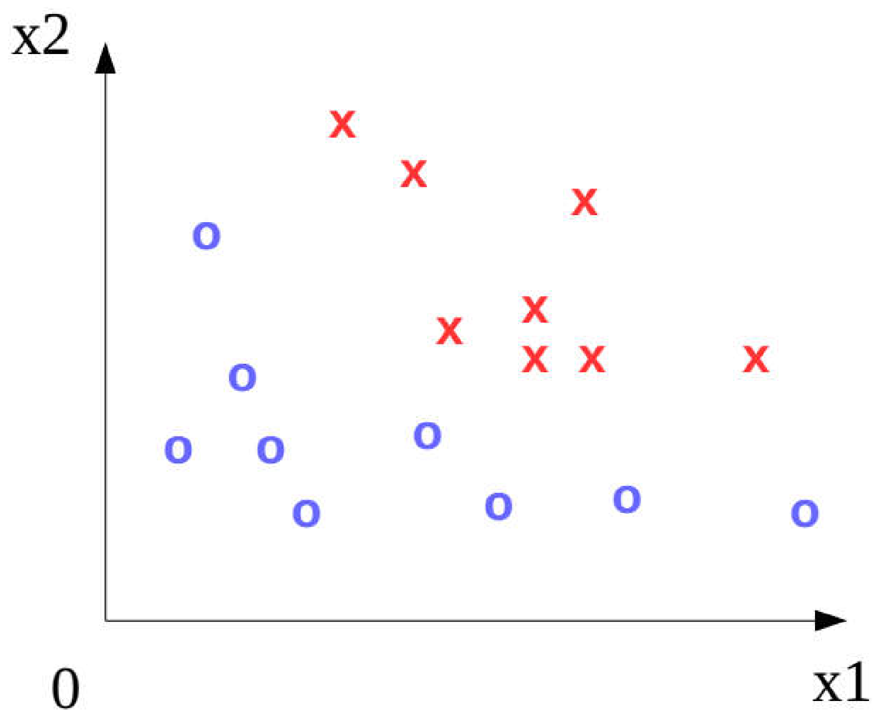


Case 2:

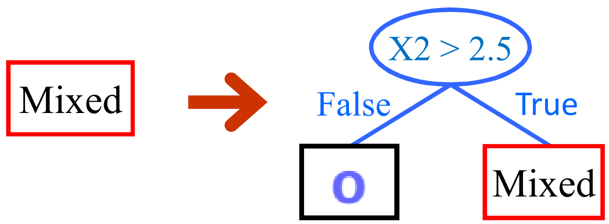
$D_t = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$

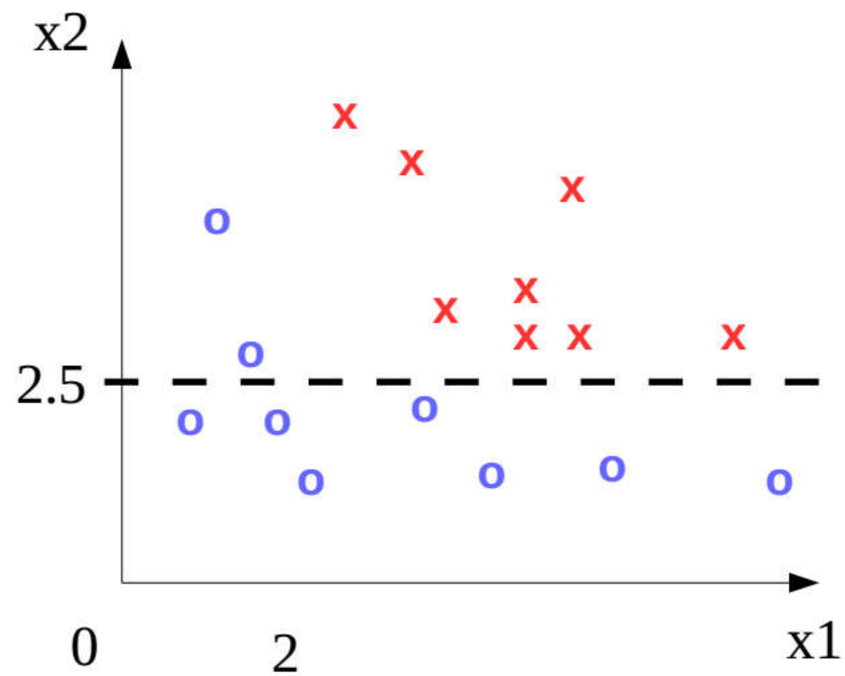
where $\mathbf{x} = \{x_1, x_2, \dots, x_{n-1}\}$

Example 1: Hunt's Algorithm

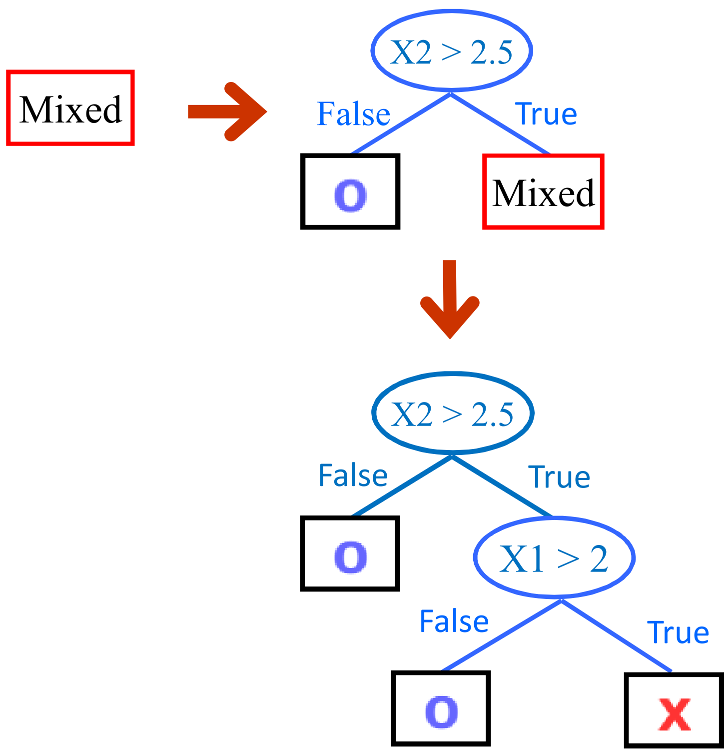


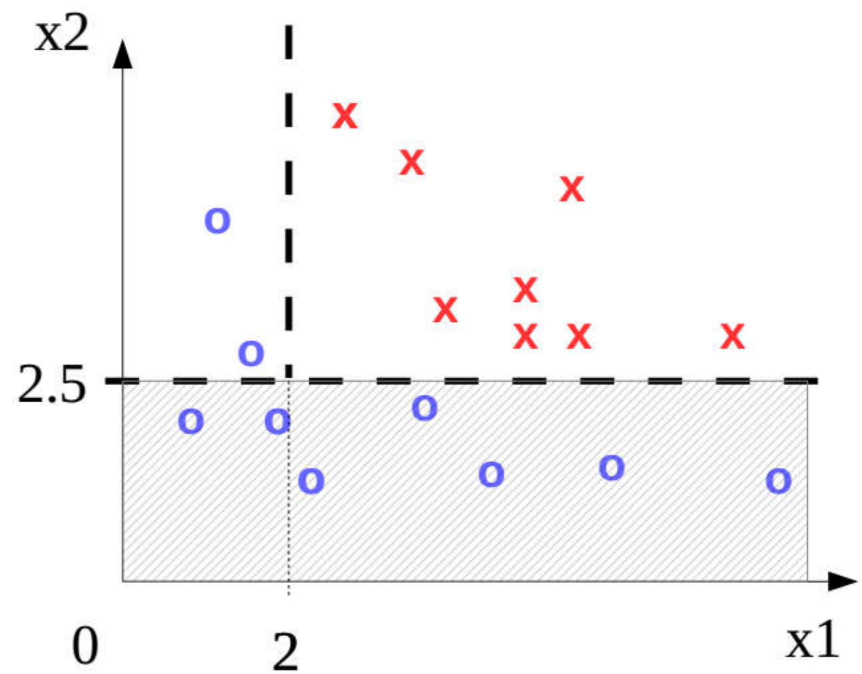
Example 1: Hunt's Algorithm



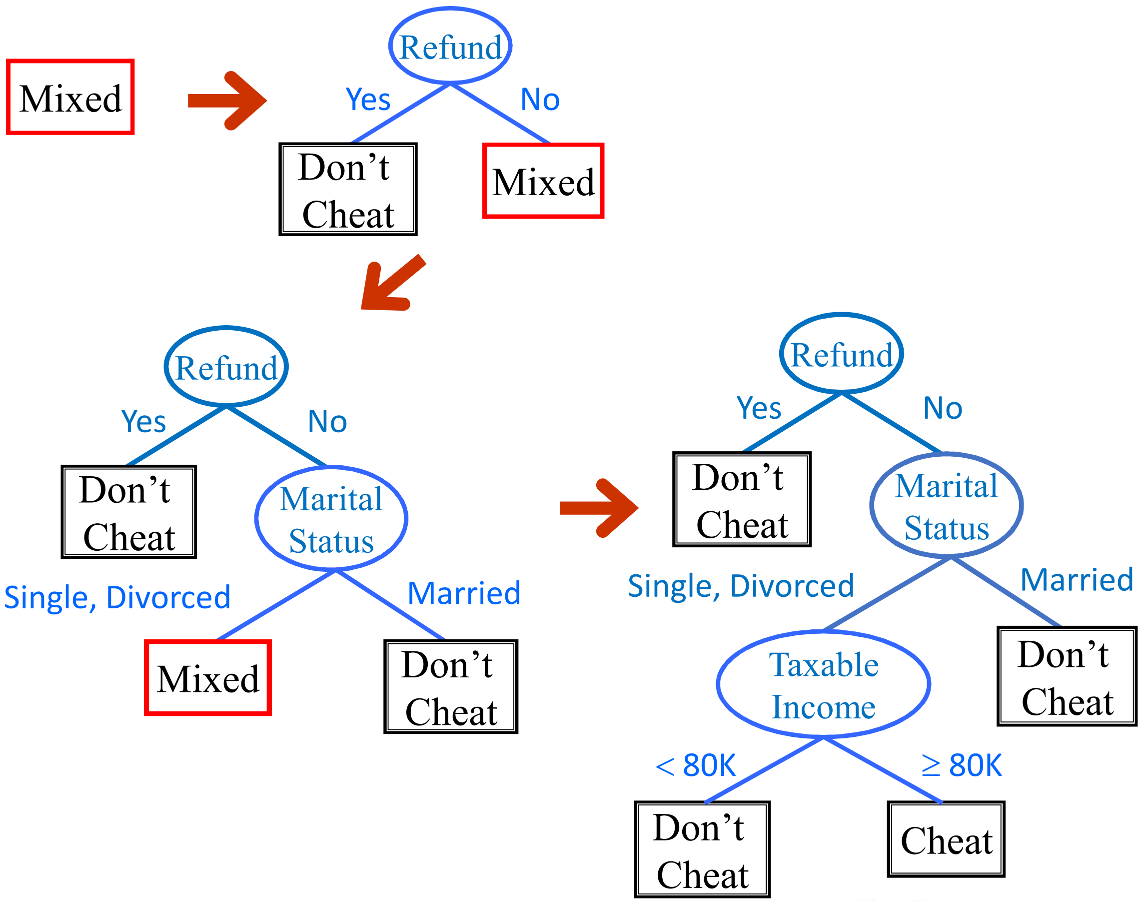


Example 1: Hunt’s Algorithm





Example 2: 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

Decision Tree Induction

Two design issues of decision tree induction:

1. Determine how to split the training records

- How to specify the attribute test condition?
- How to determine the best split of each test condition?

2. Determine when to stop splitting

Decision Tree Induction

Two design issues of decision tree induction:

1. Determine how to split the training records

- How to specify the attribute test condition?
- How to determine the best split of each test condition?

2. Determine when to stop splitting

How to specify the attribute test condition?

Methods for expressing attribute test conditions are depending on two components:

1. Depends on attribute types

- Binary
- Nominal
- Ordinal
- Continuous

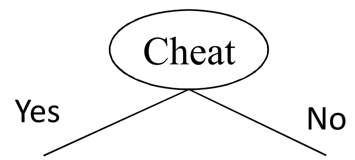
2. Depends on number of ways to split

- 2-way split
- Multi-way split

Splitting Based on Binary Attributes

Binary Attributes

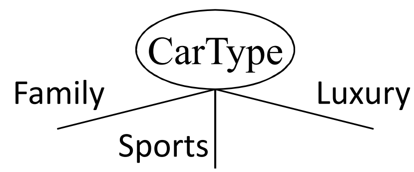
- The records can only be divided into two subsets.



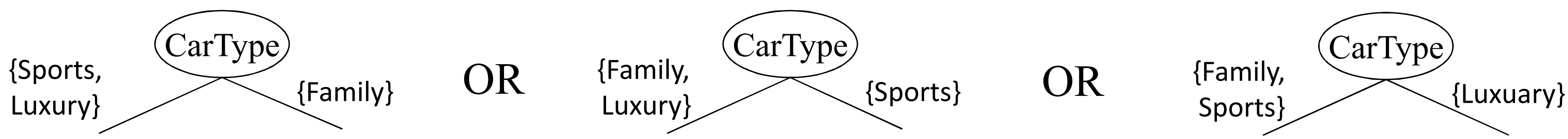
Splitting Based on Nominal Attributes

Nominal Attributes

- **Multi-way split:** The number of outcomes (child node) depends on the number of distinct values for the corresponding attribute.



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

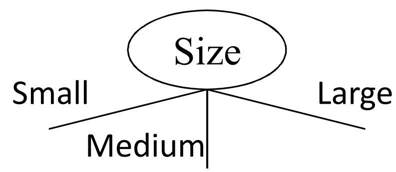


Number of ways to split s attribute values into 2 subsets = $C_2^s = \frac{s!}{(s-2)!2!}$

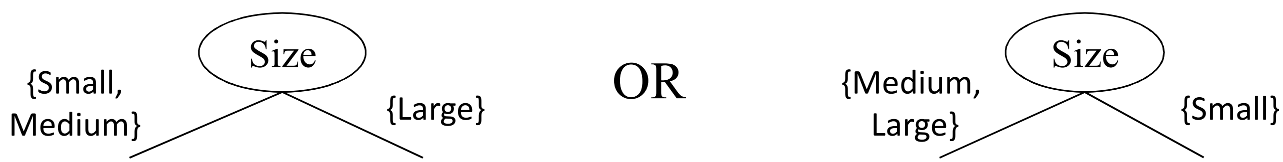
Splitting Based on Ordinal Attributes

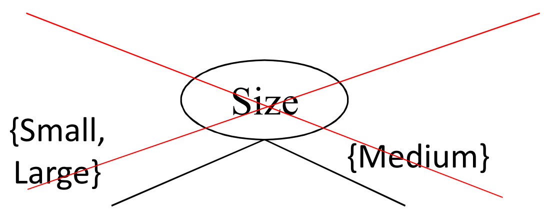
Ordinal Attributes

- **Multi-way split:** The number of outcomes (child node) depends on the number of distinct values for the corresponding attribute.



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





Ordinal attribute values can be grouped as long as the group does not violate the order property of the attribute values.

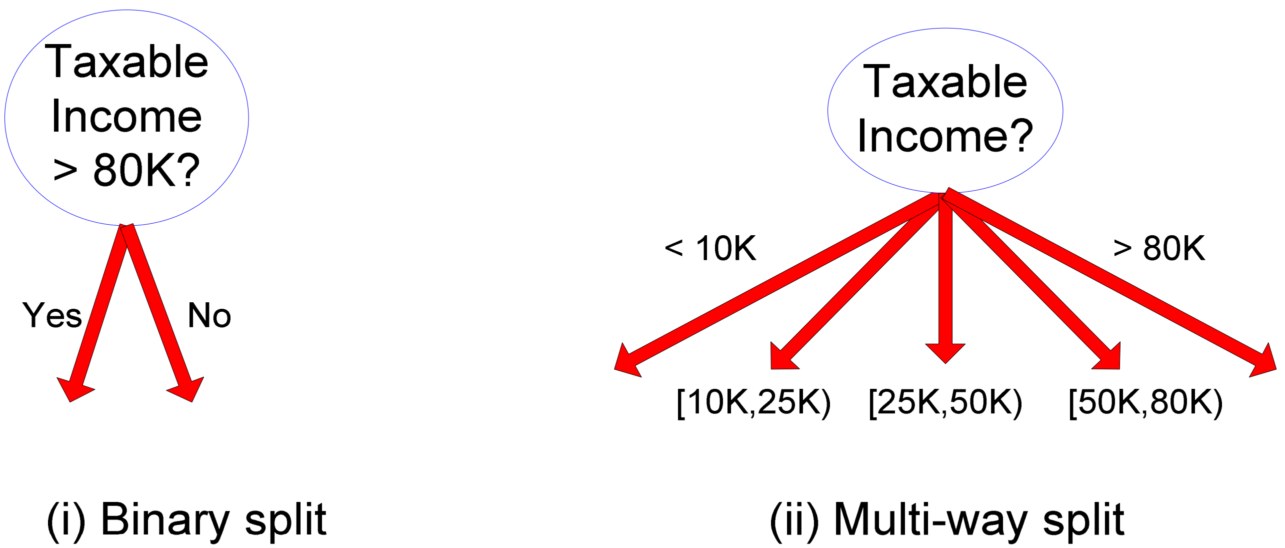
Splitting Based on Continuous Attributes

Continuous Attributes

- **Binary split**: $A < v$ and $v \leq A$
  - The decision tree must consider all possible splits positions (v), and selects the best split position.
  - It can be more compute intensive.

- **Multi-way split:** $v_i \leq A < v_{i+1}$, for $i = 1, \dots, k$
  - The decision tree algorithm must consider all possible ranges of continuous values.
  - One approach is to apply discretization methods to form ordinal categorical attributes.
  - Adjacent intervals can also be aggregated into wider ranges as long as the order property is preserved

Splitting Based on Continuous Attributes



Decision Tree Induction

Two design issues of decision tree induction:

1. Determine how to split the training records

- How to specify the attribute test condition?

- How to determine the best split of each test condition?

2. Determine when to stop splitting

How to determine the best split?

Definition:

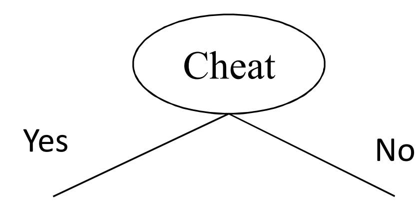
Given that the data set consists of L classes $\{c_1, c_2, \ldots, c_L\}$:

p_i denotes the fraction of the records belonging to class i at a given node

$$p_{c_1} + p_{c_2} + \cdots + p_{c_L} = 1$$

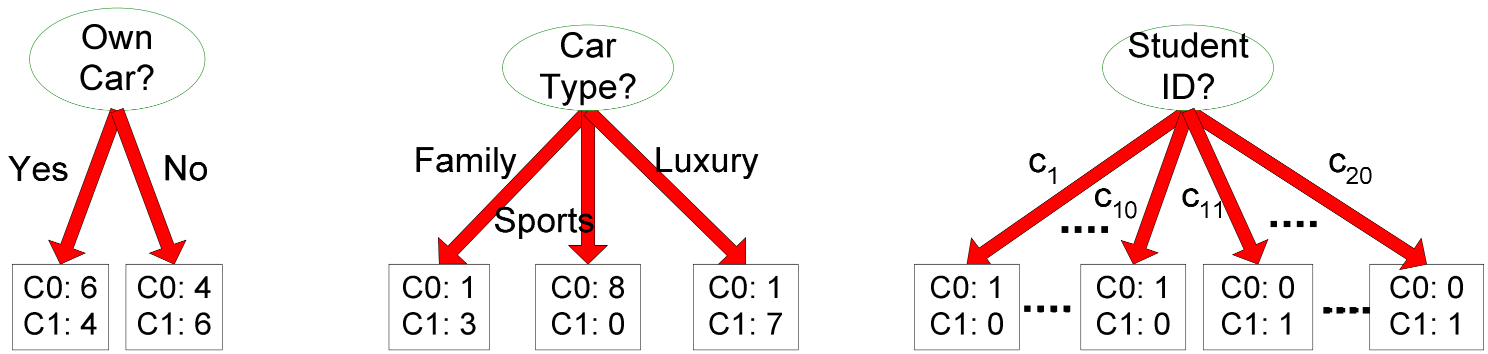
Example: - The data set consists of two classes {0, 1}

- At node t (cheat attribute), there are 10 records consisting of 2 class 0 records and 8 class 1 records.
- Thus, $p_0 = 0.2$ and $p_1 = 0.8$



How to determine the best split?

20 records
- 10 records of class 0 (C0)
- 10 records of class 1 (C1)



Which test attribute is the best?

How to determine the best split?

- Nodes with homogeneous class distribution are preferred which means that we want to have the degree of impurity at any child node as low as possible.

- The measures developed for selecting the best split are often based on the degree of impurity.

- Examples of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous $(p_{C0}, p_{C1}) = (0.5, 0.5)$ Highest degree of impurity	Homogeneous $(p_{C0}, p_{C1}) = (0.9, 0.1)$ Low degree of impurity

How to determine the best split?

General steps to determine the best split:

Step 1 : Calculate the degree of impurity of the parent node (before splitting) using the selected impurity measure

Step 2 : Calculate the degree of impurity of the child nodes (after splitting) for each test condition using the selected impurity measure

Step 3 : Calculate the quality of split (the weighted average of impurity measures) for each test condition

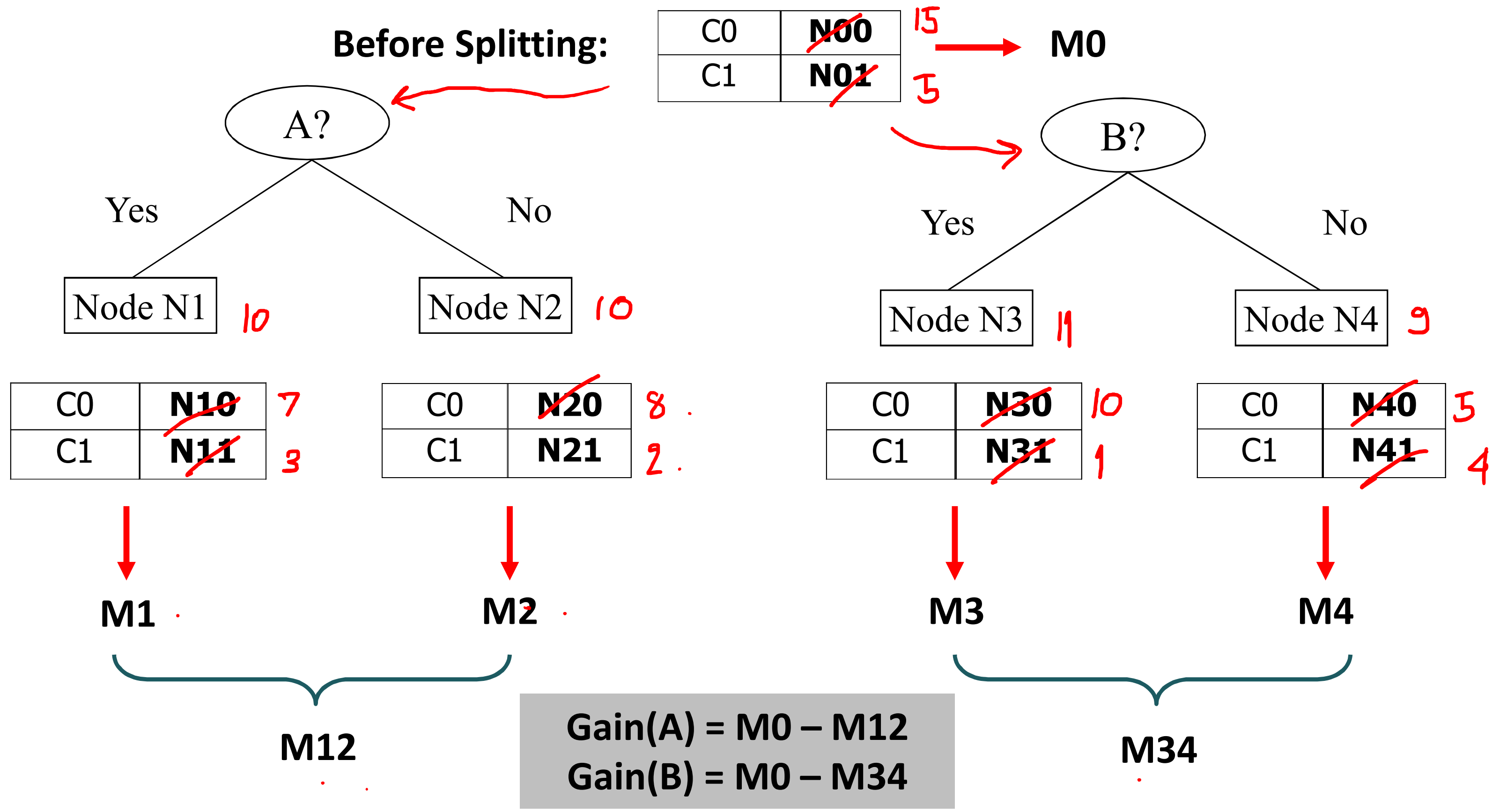
Step 4 : Calculate the gain (the difference between values obtained in step 1 and step 3) for each split in order to determine the goodness of split

Step 5 : Select the split that has the maximum gain as the best split.

Note : The degree of impurity of the parent node is the same for all test conditions because D_t is the same, thus maximizing the gain (Δ) is equivalent to minimizing the weighted average impurity measures of the child nodes (I_{split}).

How to determine the best split?

Assume we have an **impurity measure M** that tells us how "impure" a node is.

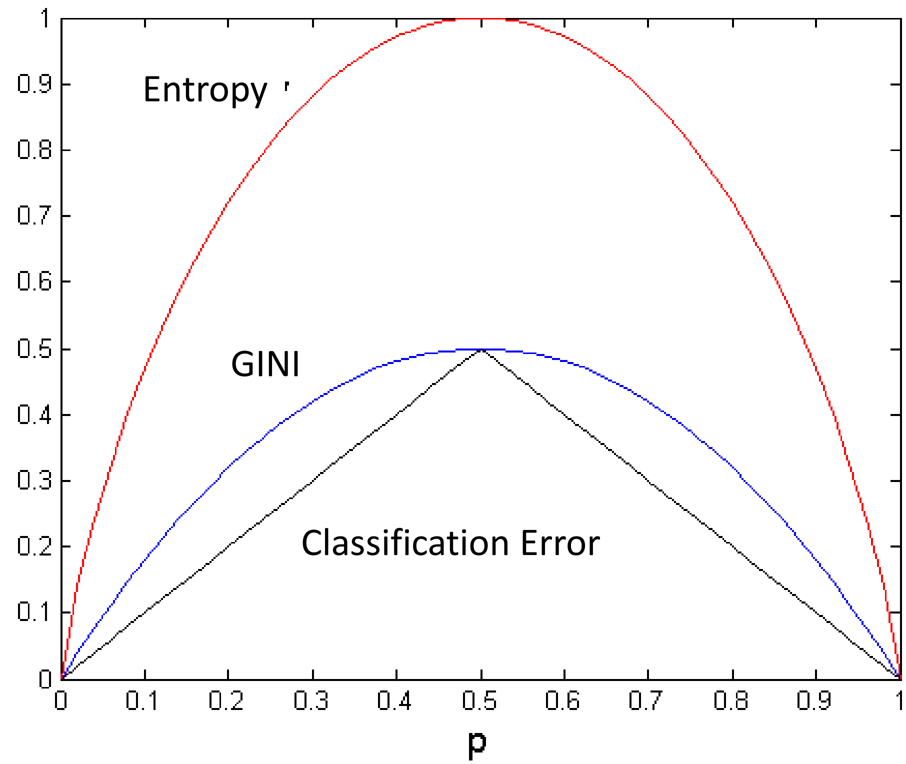


How to determine the best split?

- **Node Impurity Measures :**
  1) Gini Index
  2) Entropy
  3) Classification Error

- All three measures attain their
  - Maximum value when $p = 0.5$; this happens when records are equally distributed among all classes, implying least interesting information.
  - Minimum value when $p = 0 \ or \ 1$; this happens when all records belong to one class, implying most interesting information

Binary class attribute



Measure of Node Impurity: GINI Index

- **Gini Index** for a given node t:

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

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

- Maximum = $1 - \frac{1}{L}$ where L denotes number of classes
- Minimum = 0
- This measure is mostly used in Classification And Regression Trees (CART), SLIQ algorithm, SPRINT algorithm.

Measure of Node Impurity: Entropy

- **Entropy** at a given node t:

$$Entropy(t) = -\sum_{j} p(j|t) \log_2 p(j|t)$$

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

- Maximum = $\log L$ where L denotes number of classes
- Minimum = 0
- This measure is mostly used in ID3 and C4.5.

Measure of Node Impurity: Classification Error

- **Classification error** at a given node t :

$$\boxed{Error(t) = 1 - \max_i p(i|t)}$$

  where $p(i|t)$ is the relative frequency of class i at node t

- Maximum = $1 - \frac{1}{L}$ where L denotes number of classes
- Minimum = 0

Examples for Computing GINI Index

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

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

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 ($\log_2 0$) – 1 ($\log_2 1$) = – 0 – 0 = 0

C1	1
C2	5

P(C1) = 1/6

P(C2) = 5/6

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

Examples for Computing Classification Error

$n = 6$

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

n_{c_1}

C1	0
C2	6

n_{c_2}

P(C1) = 0/6 = 0

P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0 $\rightarrow I(\cdot)$

C1	1
C2	5

P(C1) = 1/6

P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 0.167 $\rightarrow I(\cdot)$

C1	2
C2	4

P(C1) = 2/6

P(C2) = 4/6

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 0.33 $\rightarrow I(\cdot)$

Splitting Based on Impurity Measures

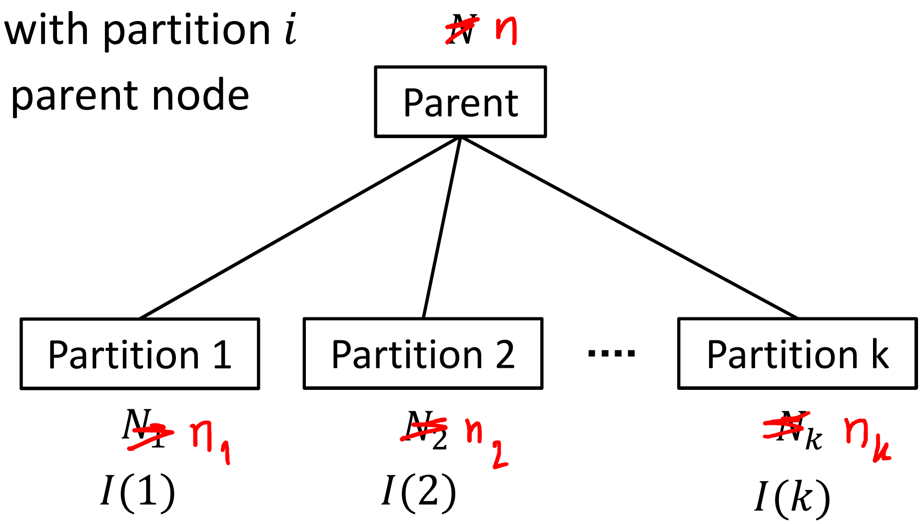
- When a node p is split into k partitions, the **quality of split** (the weighted average of impurity measures) is computed as,

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

where $I(\cdot)$ = The impurity measure (GINI index, Entropy, Classification Error) of a given node

n_i: $\cancel{N_i}$ = The number of records associated with partition i

n $\cancel{N}$ = The total number of records at the parent node



Splitting Based on Information Gain

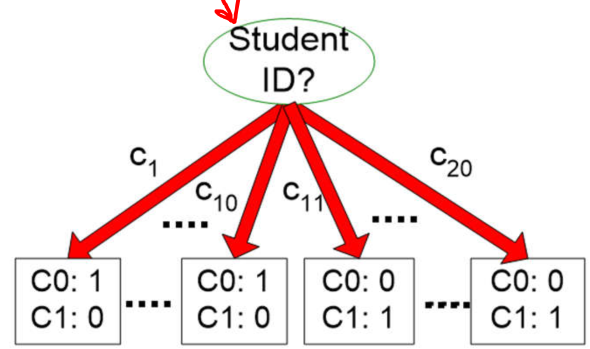
- **Information Gain**: Parent Node is split into k partitions

$$\boxed{GAIN(\Delta) = I(parent) - I_{split}}$$

where $I(parent)$ = The impurity measure of a parent node

- Since $I(parent)$ is the same for all test conditions, maximizing the gain is equivalent to minimizing the weighted average impurity measure of the child nodes (I_{split}).

- **Disadvantage**: GINI Index and Entropy tend to prefer splits that result in large number of partitions, each being small but pure.



Splitting Based on Gain Ratio

- There are two strategies for overcoming the disadvantage of information gain:
  1. To restrict the test conditions to binary splits only.
  2. To modify the splitting criterion to take into account the number of outcomes produced by the attribute test condition. This can be done by computing gain ratio instead of information gain. This method adjusts Information Gain (Δ) by the entropy of the partitioning ($SplitINFO$). Higher entropy partitioning (large number of small partitions) is penalized.

$$GainRATIO_{split} = \frac{GAIN}{SplitINFO}$$

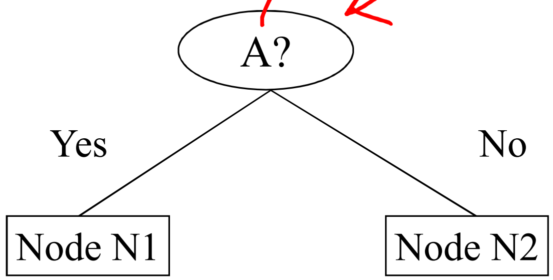
$$SplitINFO = -\sum_{i=1}^{k} \frac{N_i}{N} \log_2 \frac{N_i}{N}$$

where N_i = The number of records associated with partition i

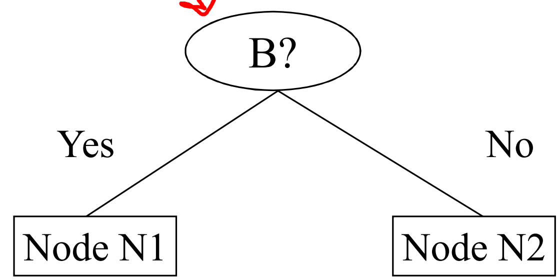
N = The total number of records at the parent node

Splitting of Binary Attributes: Computing GINI Index

test condition



	Parent
C1	**6**
C2	**6**
Gini =	**0.500**



Attribute A:

	N1	N2
C1	**4**	**2**
C2	**3**	**3**

Attribute B:

	N1	N2
C1	**1**	**5**
C2	**4**	**2**

Gini(N1) $= 1 - [(4/7)^2 + (3/7)^2]$
$= 0.49$

Gini(N2) $= 1 - [(2/5)^2 + (3/5)^2]$
$= 0.48$

Gini(Split) $= (7/12)(0.49) + (5/12)(0.48)$
$= 0.486$

The subsets of attribute B have a smaller Gini index, thus choose B as a split attribute.

Gini(N1) $= 1 - [(1/5)^2 + (4/5)^2]$
$= 0.32$

Gini(N2) $= 1 - [(5/7)^2 + (2/7)^2]$
$= 0.408$

Gini(Split) $= (5/12)(0.32) + (7/12)(0.408)$
$= 0.371$

Splitting of Categorical Attributes: Computing GINI Index

- **Multi-way split:** The GINI index is computed for every attribute value.

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

- **Binary split:** The same approach as the approach used for binary attributes can be used.

	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	

	CarType	
	{Luxury}	{Family, Sports}
C1	1	3
C2	1	5
Gini	0.475	

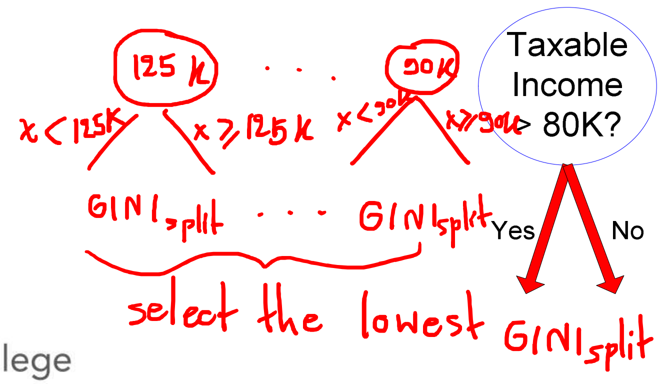
The first group has a lower Gini index because its corresponding subsets are much purer.

Splitting of Continuous Attributes: Computing GINI Index

Binary Split

- There are several choices for the splitting value where number of possible splitting values = number of distinct values
- Each splitting value (v) has a count matrix associated with it that is the class counts in each of the partitions, $\boxed{A < v}$ and $\boxed{v \leq A}$.
- **Simple method to identify the best v :**
  1) For each candidate v, scan the database to count the number of records associated with $A < v$ and $v \leq A$
  2) Compute $\text{GINI}_{\text{split}}$ for split point v
  3) Repeat 1) and 2) for all N distinct records
  4) Select the one that gives the lowest $\text{GINI}_{\text{split}}$
- This method is computationally expensive!

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



Splitting of Continuous Attributes: Computing GINI Index

- **For efficient computation:**
  1) Sort the attribute on values
  2) Split positions are the midpoints between two adjacent sorted values
  3) Scan N records, each time updating the count matrix
  4) Compute $\text{GINI}_{\text{split}}$ for each split point
  5) Choose the split position that has the least $\text{GINI}_{\text{split}}$

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	

Decision Tree Induction

Two design issues of decision tree induction:

1. Determine how to split the training records

- How to specify the attribute test condition?
- How to determine the best split of each test condition?

2. Determine when to stop splitting

Stopping Criteria for Decision Tree Induction

Two main stopping criteria:

1) Stop expanding a node when all the records belong to the same class.

2) Stop expanding a node when all the records have identical attribute values.

$$\mathbf{x}_1 = \mathbf{x}_2 = \mathbf{x}_3 = \ldots = \mathbf{x}_m$$

where $\mathbf{x} = \{x_1, x_2, \ldots, x_{n-1}\}$

Decision Tree Based Classification Techniques

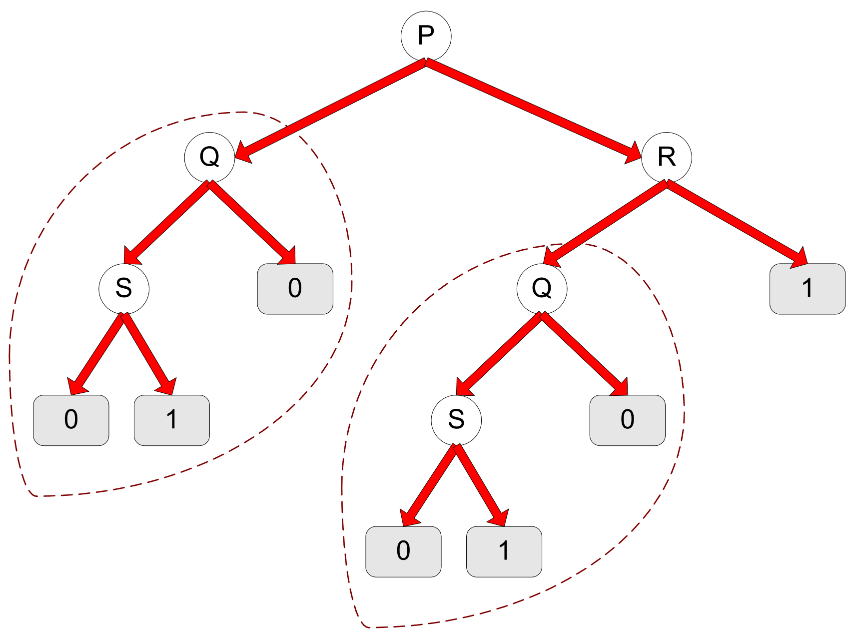
Advantages:

1) They are inexpensive to construct.
2) They are extremely fast at classifying unknown records.
3) They are quite robust to the presence of noises.
4) The models are easy to interpret for small-sized trees.
5) The presence of redundant attributes does not adversely affect the accuracy of decision trees.
6) Accuracy is comparable to other classification techniques for many simple data sets

Decision Tree Based Classification Techniques

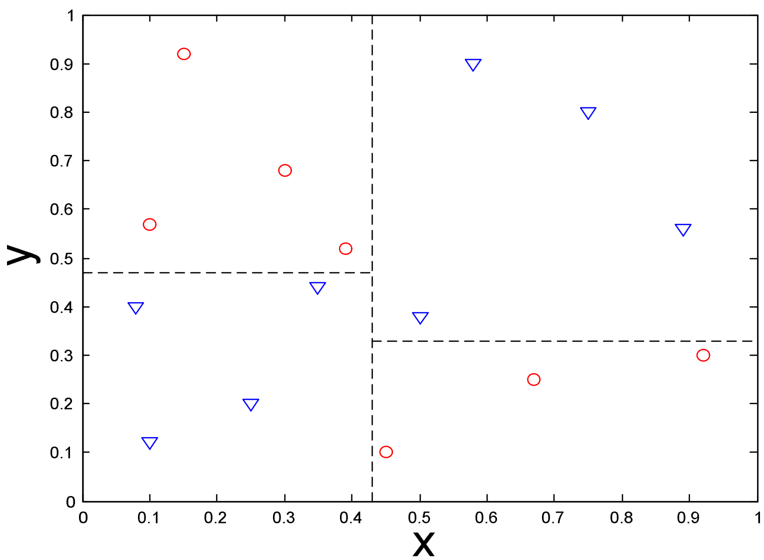
Disadvantages:

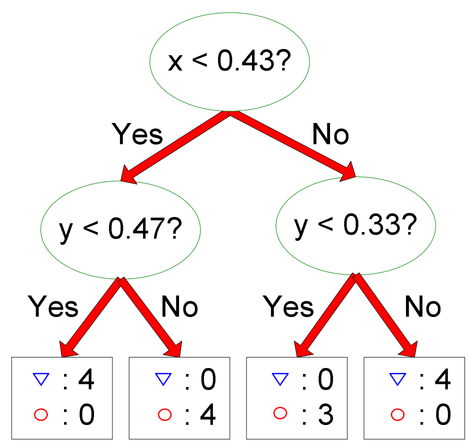
1) A subtree can be replicated multiple times in a decision tree which makes the decision tree more complex than necessary and perhaps more difficult to interpret.

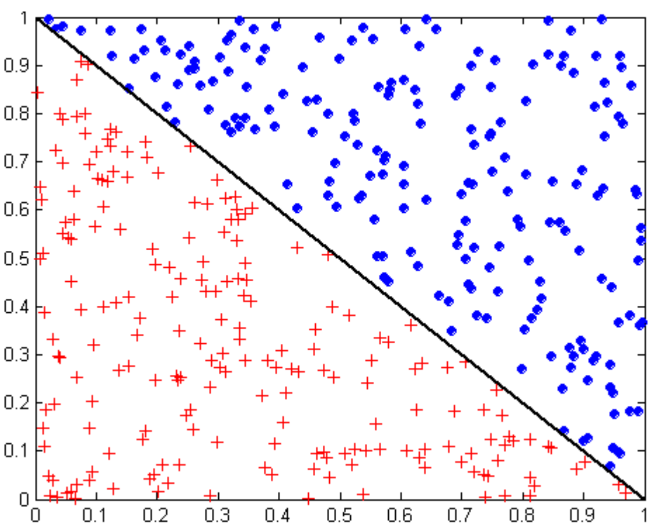


Decision Tree Based Classification Techniques

2) Decision Boundaries (border lines between two neighboring regions) created by a decision tree are usually parallel to axes because they are constructed from the test condition that involves only a single attribute.







Example: Building a decision tree using Hunt's Algorithm

Using Gini index as an impurity measure, Binary split

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

Mixed $\begin{cases} No = 7 \\ Yes = 3 \end{cases}$

Refund = { Yes, No }.

Refund

Yes → N1   No → N2

N1: No = 3, Yes = 0

N2: No = 4, Yes = 3

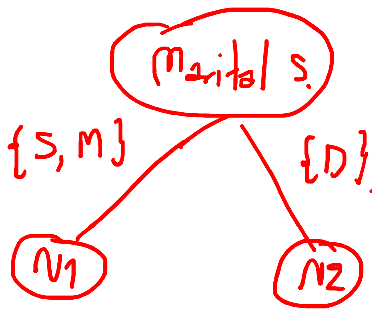
$$Gini(N1) = 1 - \left[\left(\frac{0}{3}\right)^2 + \left(\frac{3}{3}\right)^2\right] = 0$$

$$Gini(N2) = 1 - \left[\left(\frac{4}{7}\right)^2 + \left(\frac{3}{7}\right)^2\right] = 0.489$$

$$Gini_{split} = \left(\frac{3}{10}\right)(0) + \left(\frac{7}{10}\right)(0.489) = 0.342$$

Example: Building a decision tree using Hunt's Algorithm

Marital Status = {Single, Married, Divorced}



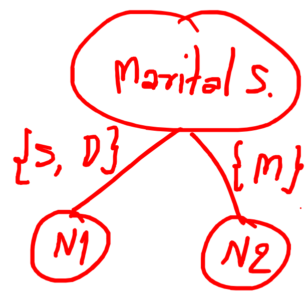
Yes = 2 | Yes = 1

No = 6 | No = 1

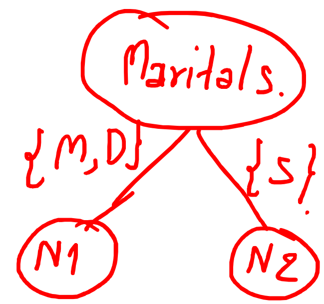
$$GINI(N1) = 1 - \left[\left(\frac{2}{8}\right)^2 + \left(\frac{6}{8}\right)^2\right] = 0.375$$

$$GINI(N2) = 1 - \left[\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2\right] = 0.875$$

$$GINI_{split} = \left(\frac{8}{10}\right)(0.375) + \left(\frac{2}{10}\right)(0.875) = 0.475$$



$GINI_{split} = A$

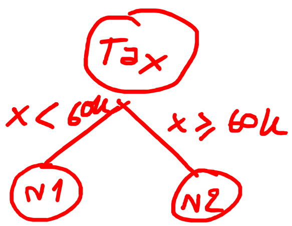


$GINI_{split} = B$

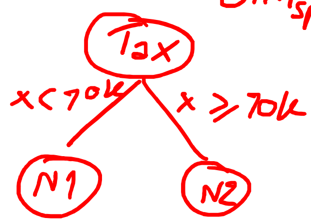
Example: Building a decision tree using Hunt's Algorithm

Taxble Income = {60k, 70k, 75k, 85k, 90k, 95k, 100k, 120k, 125k, 220k}

$GINI_{split}$ = C D E F G H I J

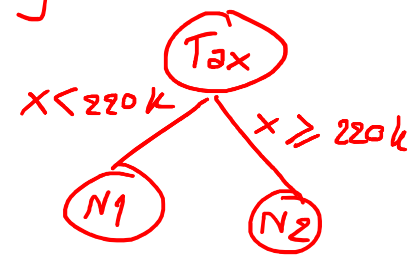


$GINI_{split} = 0.42$



$GINI_{split} = 0.405$

. . .

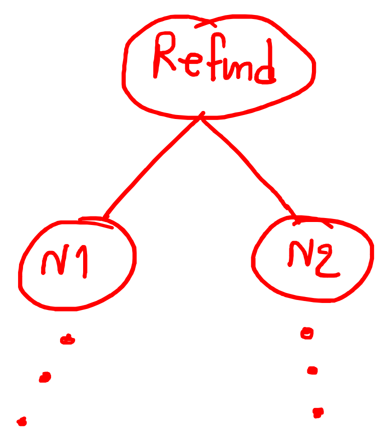


$GINI_{split} = J$

Example: Building a decision tree using Hunt's Algorithm

Select minimum $GINI_{split}$ $\{0.342, 0.475, A, B, 0.42, 0.405, C, D, E, F, G, H, I, J\}$

Assume that 0.342 is the lowest $GINI_{split}$, thus



Need to split further, until a prespecified stopping criterion is met.

Example: Building a decision tree using Hunt's Algorithm