Data Mining
Classification: Basic Concepts and Techniques

Lecture Notes for Chapter 3

Introduction to Data Mining, 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

Classification

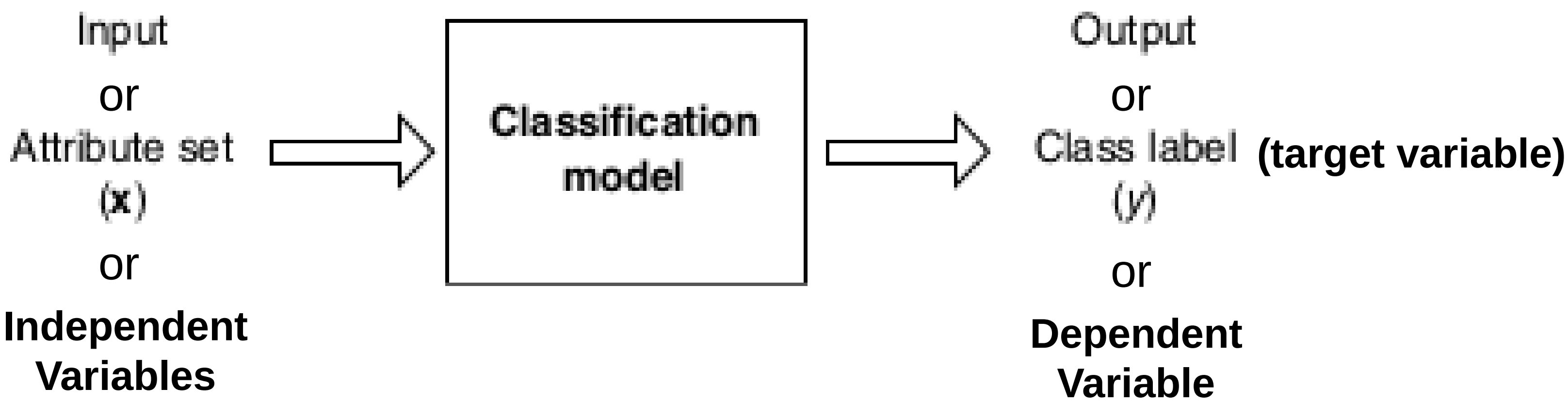
- Predefined Categories or Classes
- Assign objects or records to one of the predefined classes
- Two main types:
  - *Binary classification*
  - *Multi-class classification*
- Example:
  - *Detecting email is a spam or not?*
  - *Categorizing tumors as benign or malignant*
  - *Classify a set of images of fruits*

Classification

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes,* one of the attribute is class
  - characterized by a tuple ($\mathbf{x}$,y), where $\mathbf{x}$ is the ***attribute set*** and y is the ***class label***
    - $\mathbf{x}$: attribute, predictor, independent variable, input
    - y: class, response, dependent variable, output

- 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.

Classification



<A schematic illustration of a classfication task>

Classification

- **X**: attribute set can contain attributes of any type (discrete or continuous)
- **Y**: class label or target or prediction variable must be a categorical – always discrete
- A **Classification model** is an abstract representation of the relationship between the attribute set and the class label.
- The model can be represent in many ways, e.g., as a *tree*, a *probability table*, or simply, a *vector* of real-world parameters
- Classification is the task of learning a target function **f** that maps each attribute set **x** to one of the predefined class labels **y**.

Examples of Classification Task

Task	Attribute set, x	Class label, y
Categorizing email messages	Features extracted from email message header and content	spam or non-spam
Identifying tumor cells	Features extracted from x-rays or MRI scans	malignant or benign cells
Cataloging galaxies	Features extracted from telescope images	Elliptical, spiral, or irregular-shaped galaxies

Classification Example

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
human	warm-blooded	hair	yes	no	no	yes	no	mammal
python	cold-blooded	scales	no	no	no	no	yes	reptile
salmon	cold-blooded	scales	no	yes	no	no	no	fish
whale	warm-blooded	hair	yes	yes	no	no	no	mammal
frog	cold-blooded	none	no	semi	no	yes	yes	amphibian
komodo dragon	cold-blooded	scales	no	no	no	yes	no	reptile
bat	warm-blooded	hair	yes	no	yes	yes	yes	mammal
pigeon	warm-blooded	feathers	no	no	yes	yes	no	bird
cat	warm-blooded	fur	yes	no	no	yes	no	mammal
leopard shark	cold-blooded	scales	yes	yes	no	no	no	fish
turtle	cold-blooded	scales	no	semi	no	yes	no	reptile
penguin	warm-blooded	feathers	no	semi	no	yes	no	bird
porcupine	warm-blooded	quills	yes	no	no	yes	yes	mammal
eel	cold-blooded	scales	no	yes	no	no	no	fish
salamander	cold-blooded	none	no	semi	no	yes	yes	amphibian

< A sample data for the **vertebrate classification problem** >

Classification Example

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

< A sample data for the **loan borrower classification problem** >

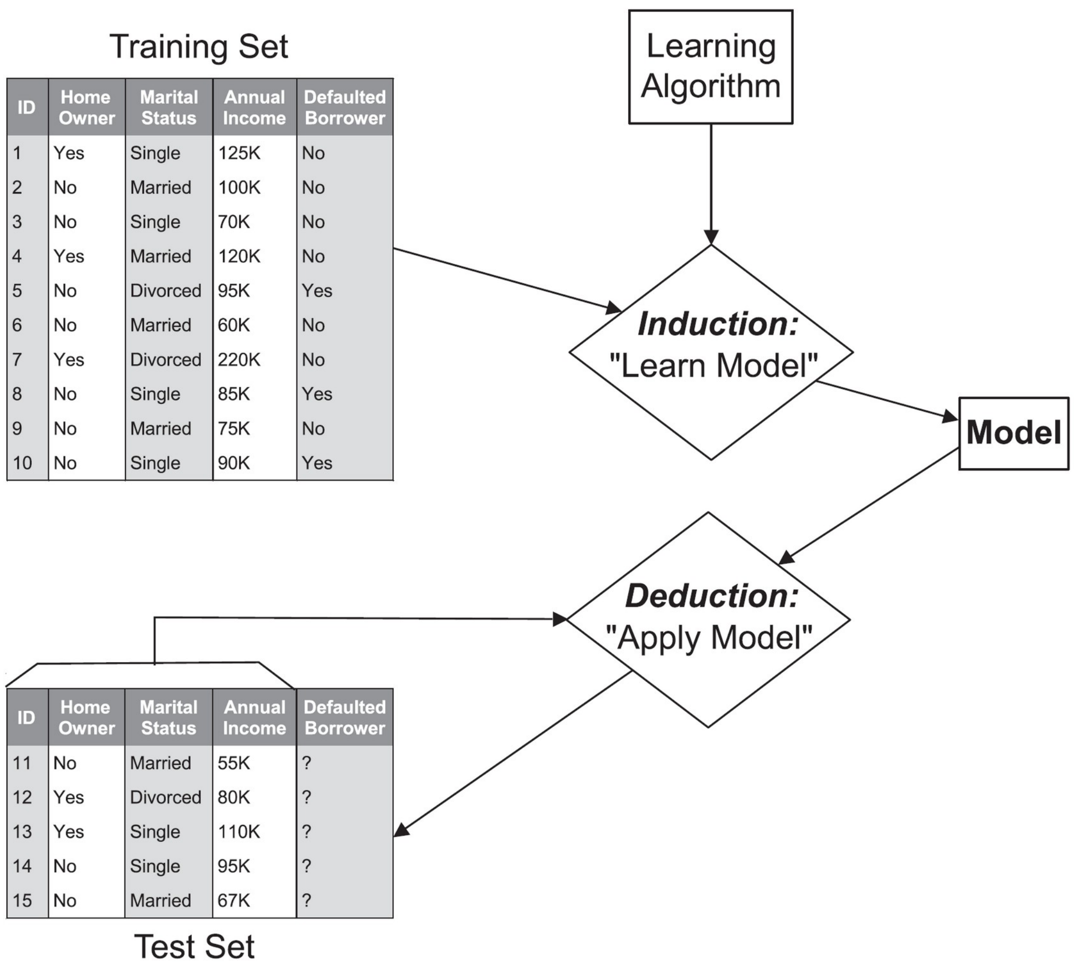
Classification

- **Applications**: A classification model serves two **important roles** in data mining:
  - Descriptive Model:
    - To identify the characteristics that distinguish instances from different classes (*explain or describe objects of a class*)
  - Predictive Model:
    - To classify previously unlabeled instances (*predict class label of unknown records*)

Classification

- **Problem Statement**: Identify target or prediction variable
- **Pre-process data:** cleaning, removing nulls, data standardization, normalization
- **Choose Classification Algorithm**
- **Choose features** to train the model
- **Sample data** by splitting between training and testing sets
- **Train the model** with train set
- **Validate model** with the test set

General Approach for Building Classification Model



Training Set

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
11	No	Married	55K	?
12	Yes	Divorced	80K	?
13	Yes	Single	110K	?
14	No	Single	95K	?
15	No	Married	67K	?

Test Set

Figure 3.3. General framework for building a classification model.

General Approach for Building Classification Model

- Classification is the task of assigning labels to unlabeled data instances and a ***classifier*** is used to perform such task
- ***Training set*** contains attribute values as well as class labels for each instance
- The systematic approach for learning a classification model given a training set is known as a ***learning algorithm***
- The process of using a learning algorithm to build a classification model from the training data is know as ***induction***(also described as “learning a model” or “building a model”)
- The process of applying a classification model on unseen test instances to predict their class labels is known as ***deduction***

General Approach for Building Classification Model

- **Evaluation of the performance of a classification** model is based on <u>the counts of test records correctly and incorrectly predicted by the model.</u>

- Tabulated in a table known as a ***confusion matrix***.

		Predicted Class	
		$Class = 1$	$Class = 0$
Actual Class	$Class = 1$	f_{11}	f_{10}
	$Class = 0$	f_{01}	f_{00}

< confusion matrix for a binary classification problem >

General Approach for Building Classification Model

- Each entry f_{ij} denotes the number of records from class **i** predicted to be of class **j**.
- For instance, f_{01} is the number of records from class **0** incorrectly predicted as class **1**.
- The total number of correct predictions made by the model is:
  - $(f_{11} + f_{00})$
- The total number of incorrect predictions is:
  - $(f_{10} + f_{01})$
- **Performance metric**
  - Accuracy
  - Error rate

General Approach for Building Classification Model

- Accuracy:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}. \quad (4.1)$$

- Error rate:

$$\text{Error rate} = \frac{\text{Number of wrong predictions}}{\text{Total number of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}. \quad (4.2)$$

- Most classification algorithms seek models that attain the **highest accuracy**, or equivalently, the **lowest error rate** when applied to the test set.

Classification Techniques

- Base Classifiers
  - **Decision Tree based Methods**
  - **Rule-based Methods**
  - **Nearest-neighbor**
  - **Naïve Bayes and Bayesian Belief Networks**
  - Support Vector Machines
  - Neural Networks, Deep Neural Nets

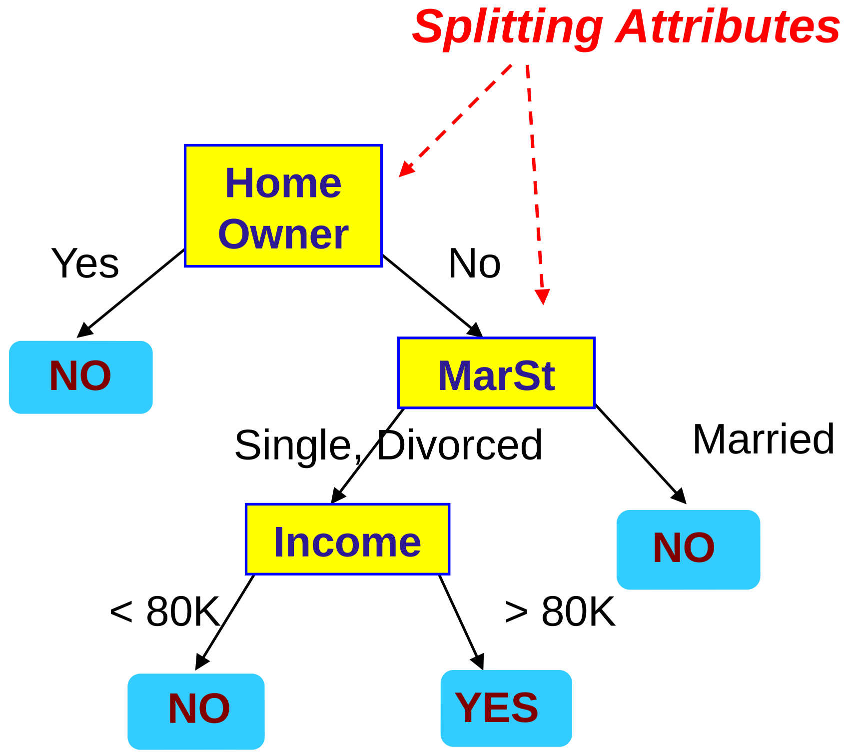
- Ensemble Classifiers
  - Boosting, Bagging, Random Forests

Example of a Decision Tree

categorical categorical continuous class

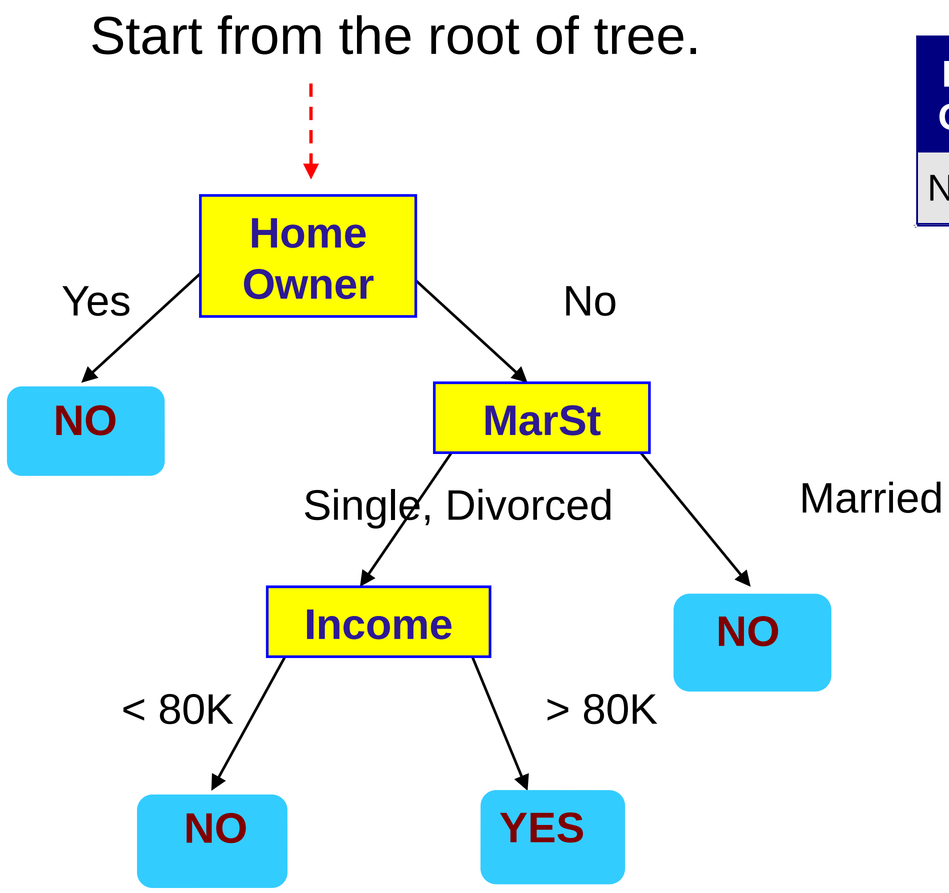
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

Apply Model to Test Data



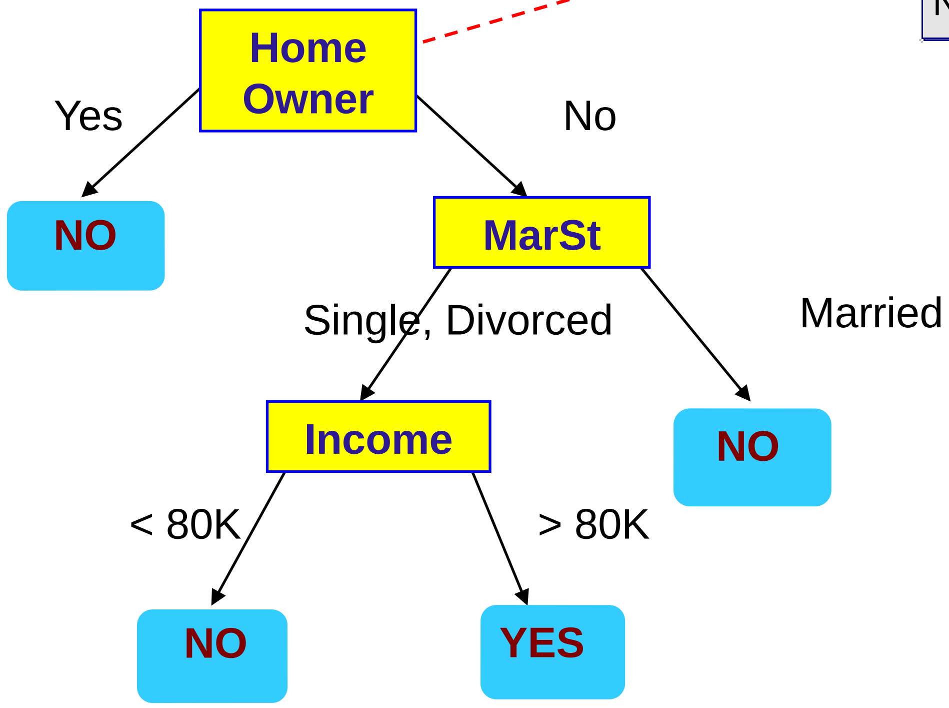
Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

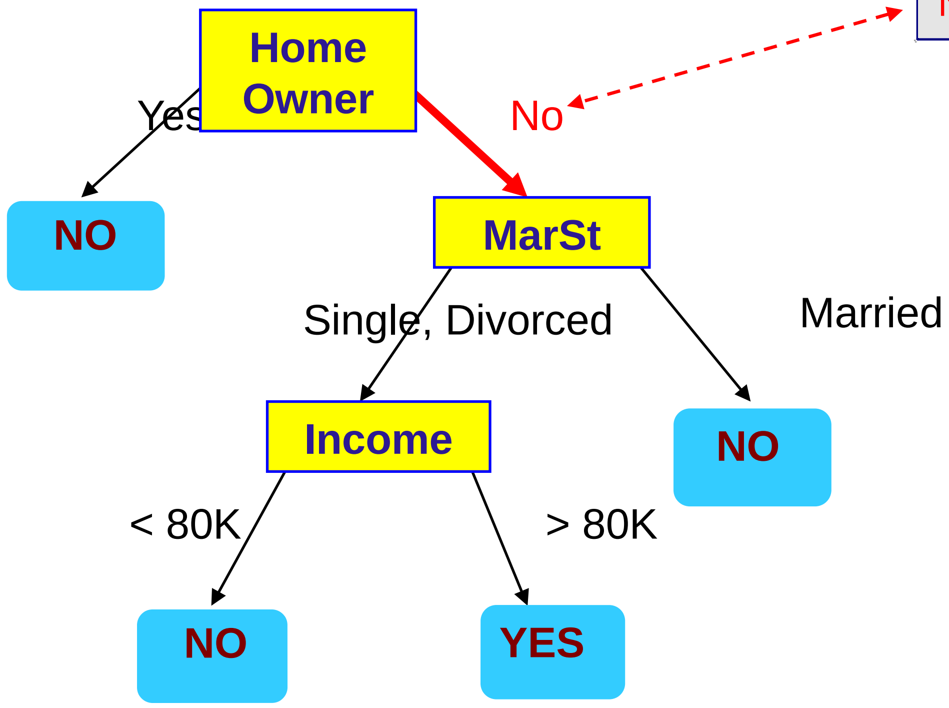
Apply Model to Test Data

Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



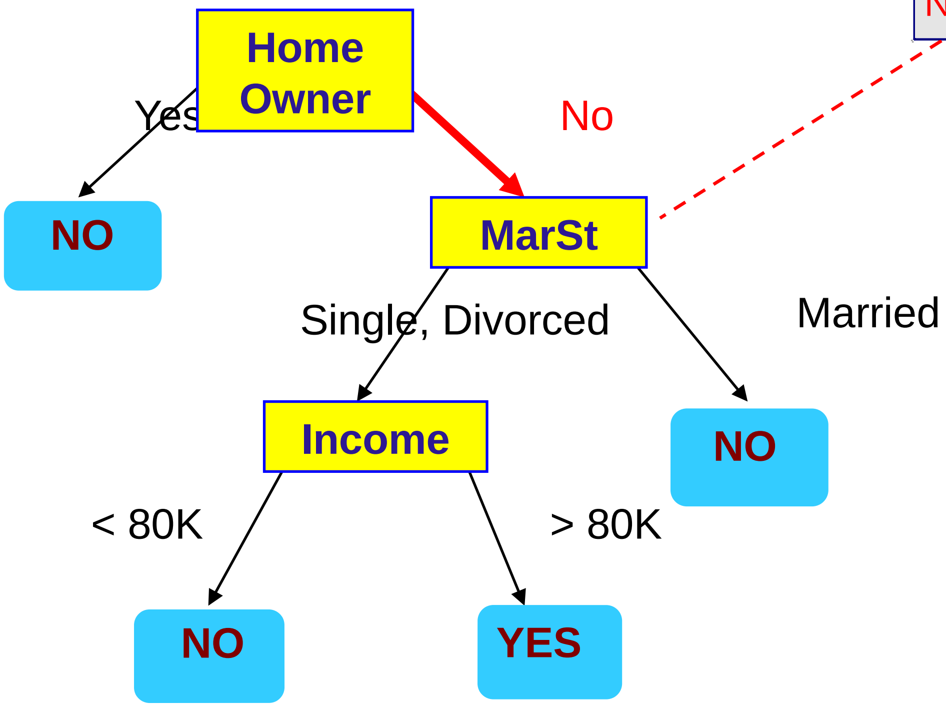
Apply Model to Test Data



Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Apply Model to Test Data



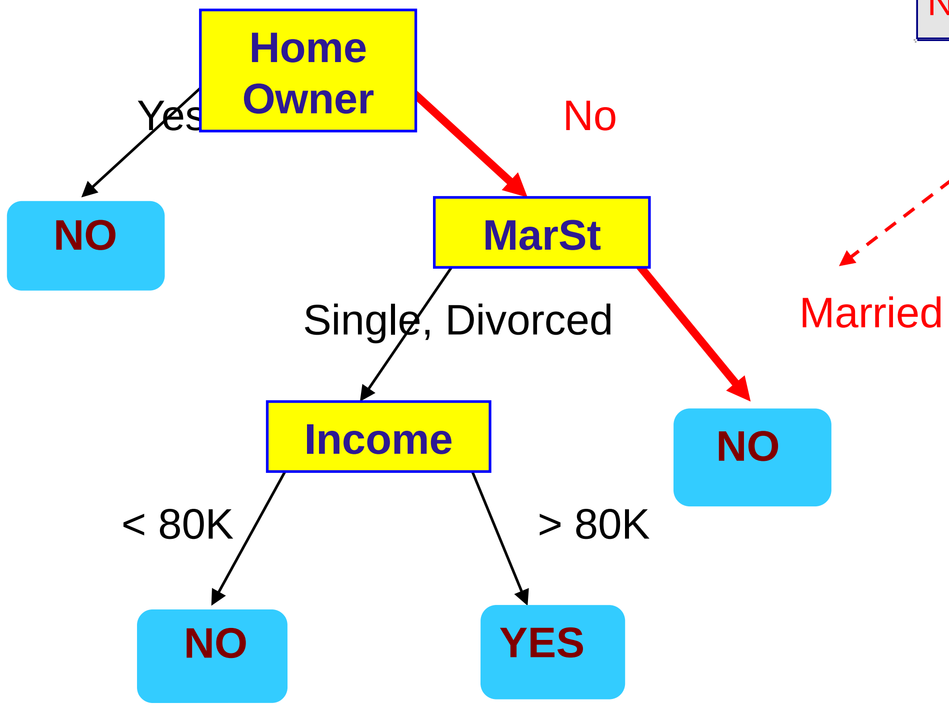
Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

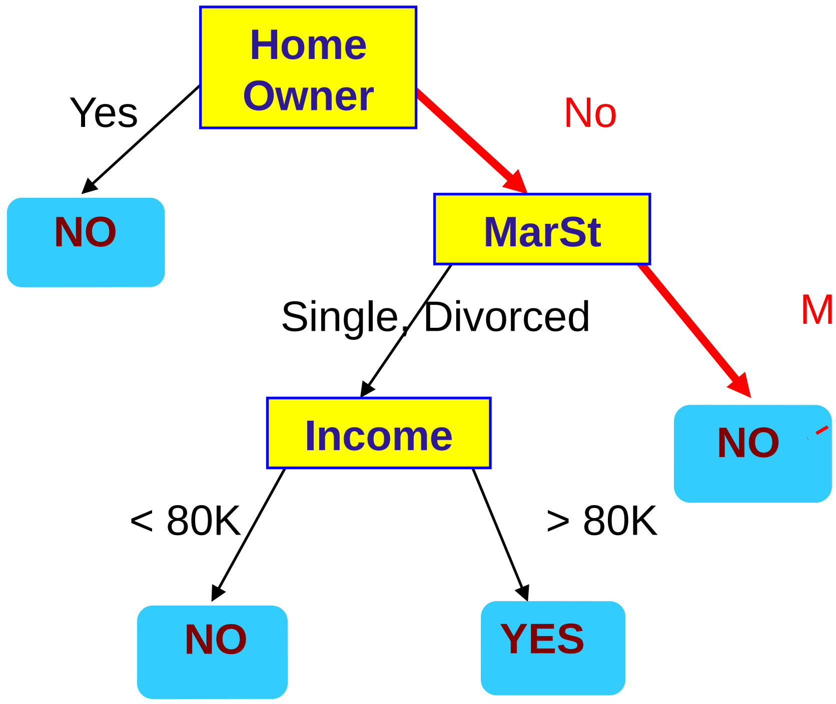
Apply Model to Test Data

Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Apply Model to Test Data



Test Data

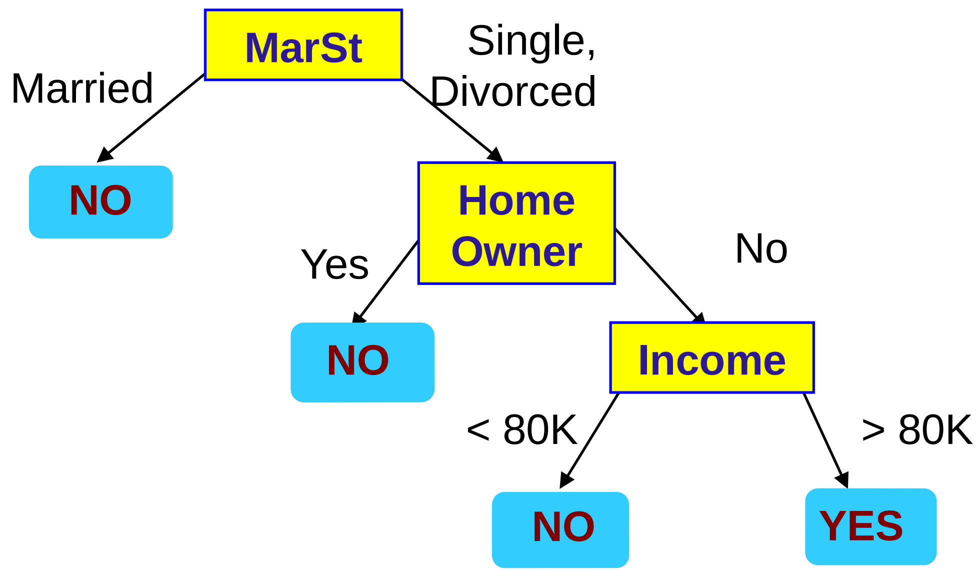
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Assign Defaulted to "No"

Another Example of Decision Tree

categorical categorical continuous class

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

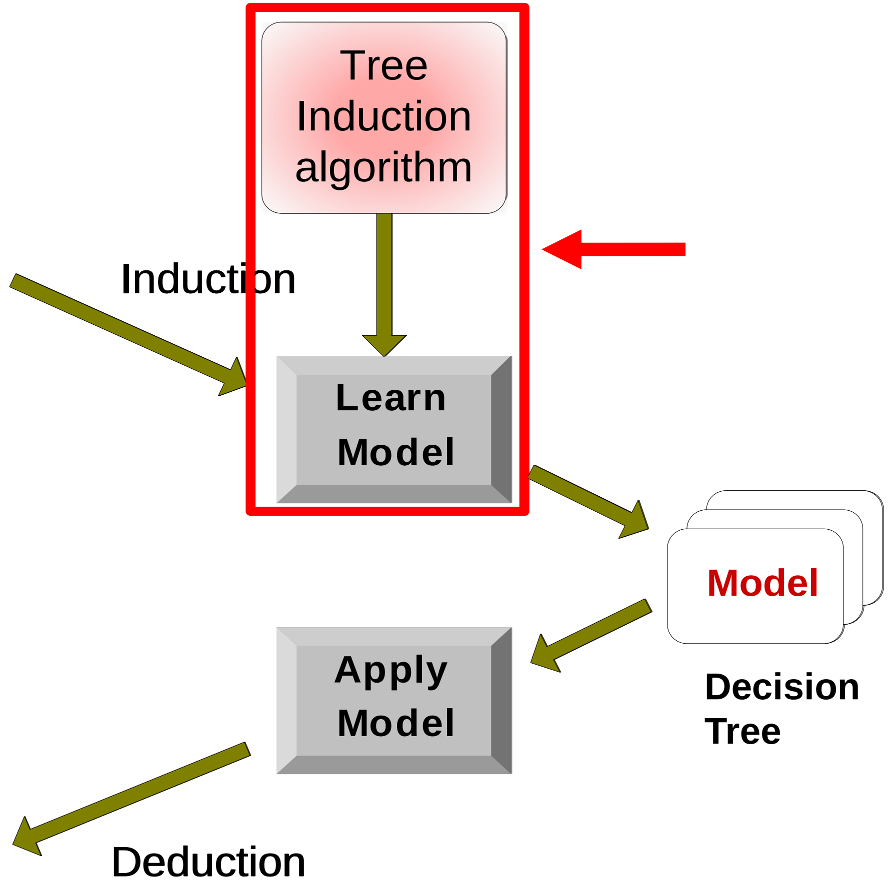


There could be more than one tree that fits the same data!

Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



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

Test Set

Decision Tree Induction

- How a Decision Tree Works:
  - consider a simpler version of the *vertebrate classification problem*
  - Instead of classifying the vertebrates into five distinct groups of species, assign them to two categories: *mammals* and *non-mammals.*
  - For a *new species* discovered by scientists, how can we tell whether it is a mammal or a non-mammal?
  - One approach is to *pose a series of questions about the characteristics of the species*.
  - First Question: whether the species is *cold*- or *warm-blooded?*

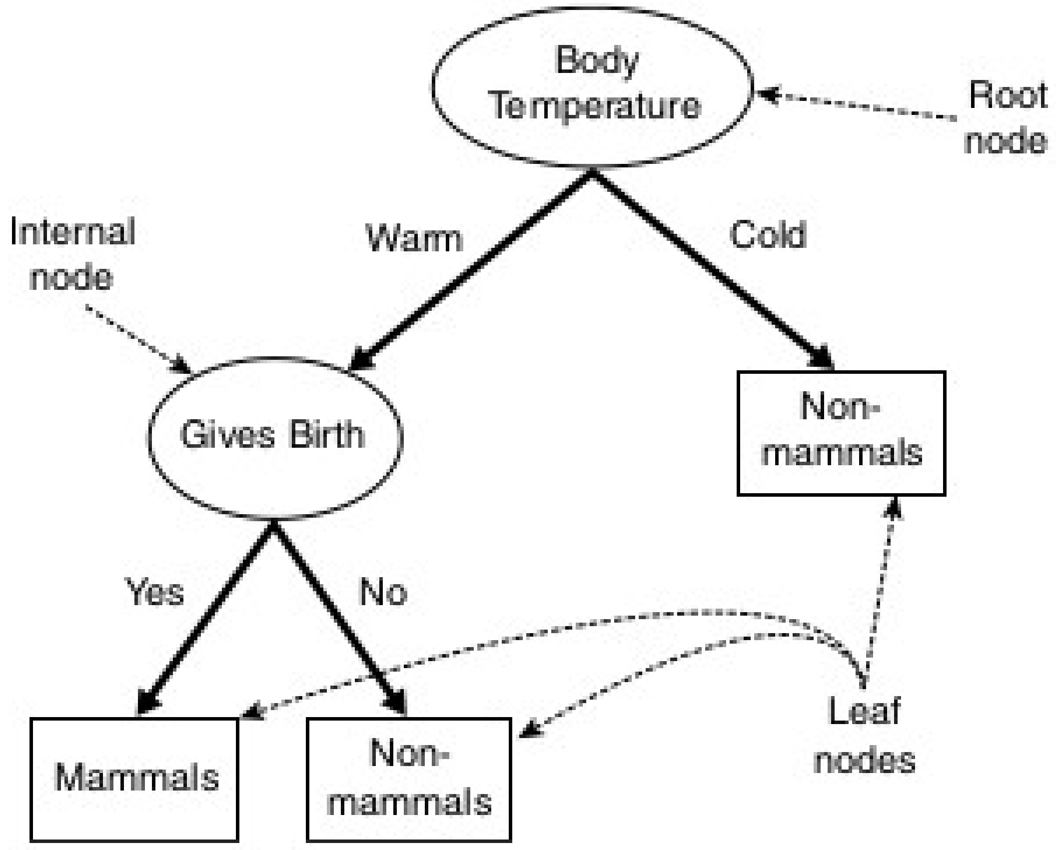
Decision Tree Induction

- How a Decision Tree Works:
  - If it is *cold-blooded*, then it is definitely *not a mammal*. Otherwise, it is either *a bird* or *a mammal*.
  - In the latter case, ask a follow-up question: Do the females of the species *give birth to their young?*
  - Those that *do give birth are definitely mammals*, while those that *do not are likely to be non-mammals* (with the exception of egg-laying mammals such as the platypus and spiny anteater).

Decision Tree Induction

- How a Decision Tree Works:
  - we can solve a classification problem by asking a *series of carefully crafted questions* about the attributes of the test record.
  - Each time we receive an answer, *a follow-up question is asked until we reach a conclusion* about the class label of the record.
  - The series of questions and their possible answers can be *organized in the form of a decision tree*, which is a hierarchical structure consisting of nodes and directed edges.

Decision Tree Induction



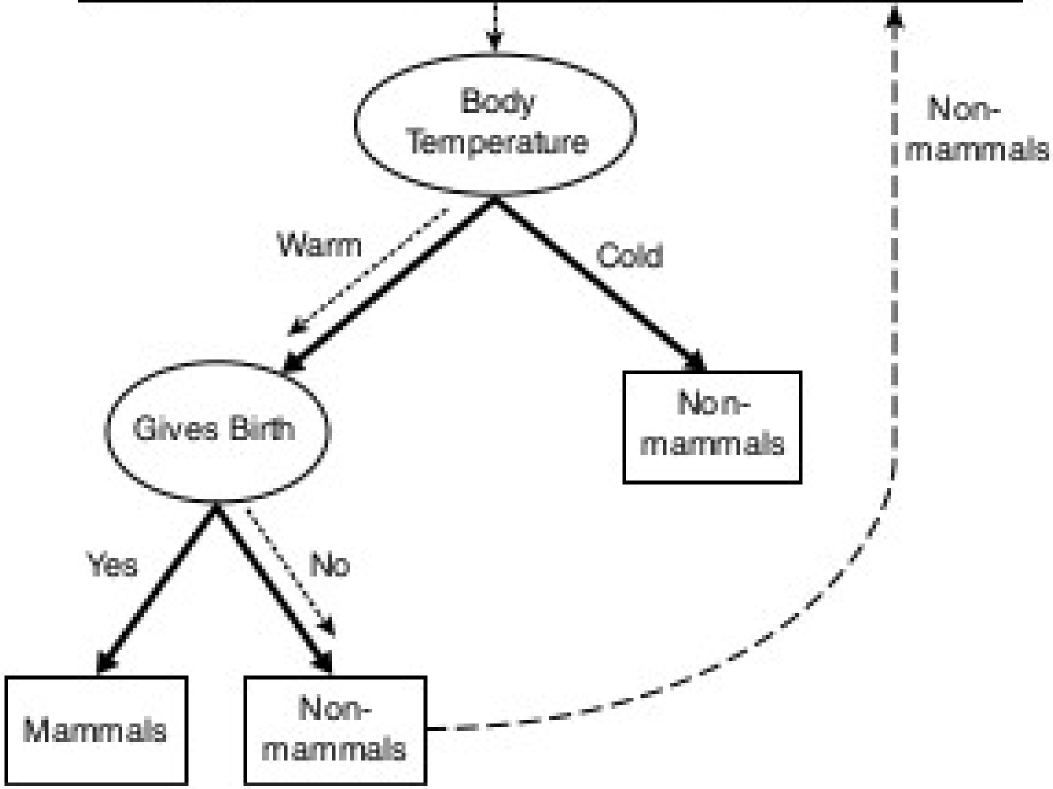
[A decision tree for the mammal classification problem]

Decision Tree Induction

- The decision tree for the mammal classification problem has three types of nodes:
  - A **root node** that has no incoming edges and zero or more outgoing edges,
  - **Internal nodes**, each of which has exactly one incoming edge and two ore more outgoing edges.
  - **Leaf** or **terminal** nodes, each of which has exactly one incoming edge and no outgoing edge

Decision Tree Induction

Unlabeled data	Name	Body temperature	Gives Birth	...	Class
	Flamingo	Warm	No	...	?



[Classifying an unlabeled vertebrate. The dashed lines represent the outcomes of applying various attribute test conditions on the unlabeled vertebrate. The vertebrate is eventually assigned to the Non-mammal class]

Decision Tree Induction

- How to Build a Decision Tree
  - Many possible decision trees can be constructed from a given set of attributes.
  - While some trees are more accurate than others.
  - Finding the optimal tree is computationally expensive because of the exponential size of the search space.
  - Efficient algorithms have been developed to induce a reasonably accurate, albeit suboptimal, decision tree in a reasonable amount of time.
  - These algorithms usually employ a greedy strategy to grow the decision tree in a top-down fashion by making a series of locally optimal decisions about which attribute to use when partitioning the training data.

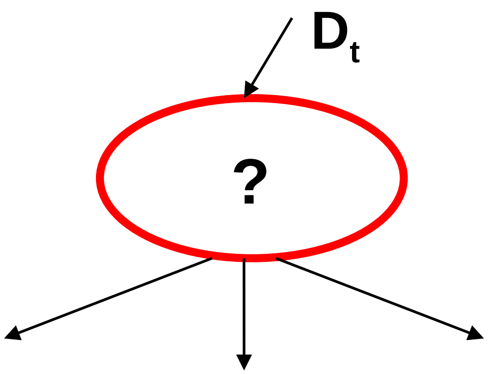
Decision Tree Induction

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

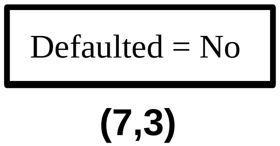
General Structure of Hunt's Algorithm

- 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 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.

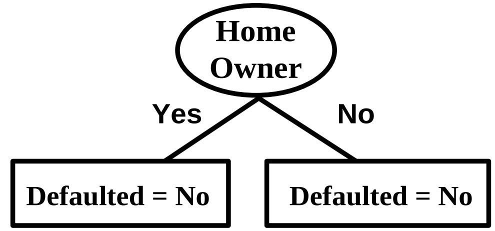
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**



Hunt's Algorithm

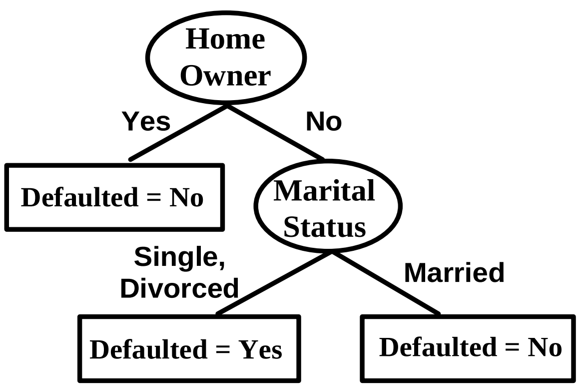


(a)

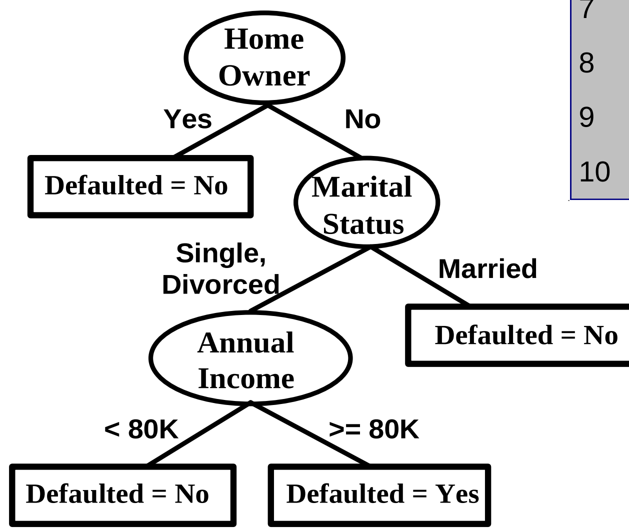


(b)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

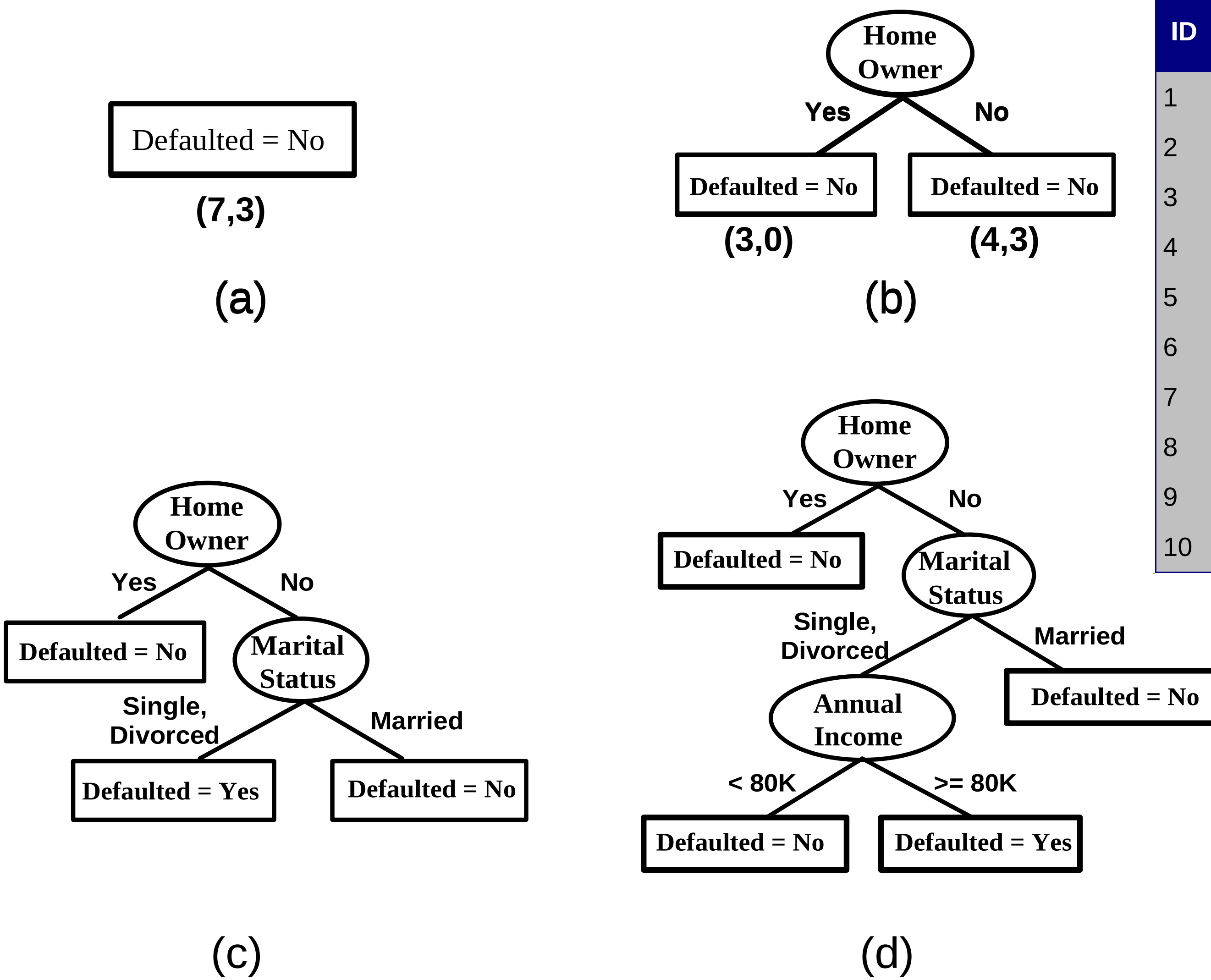


(c)



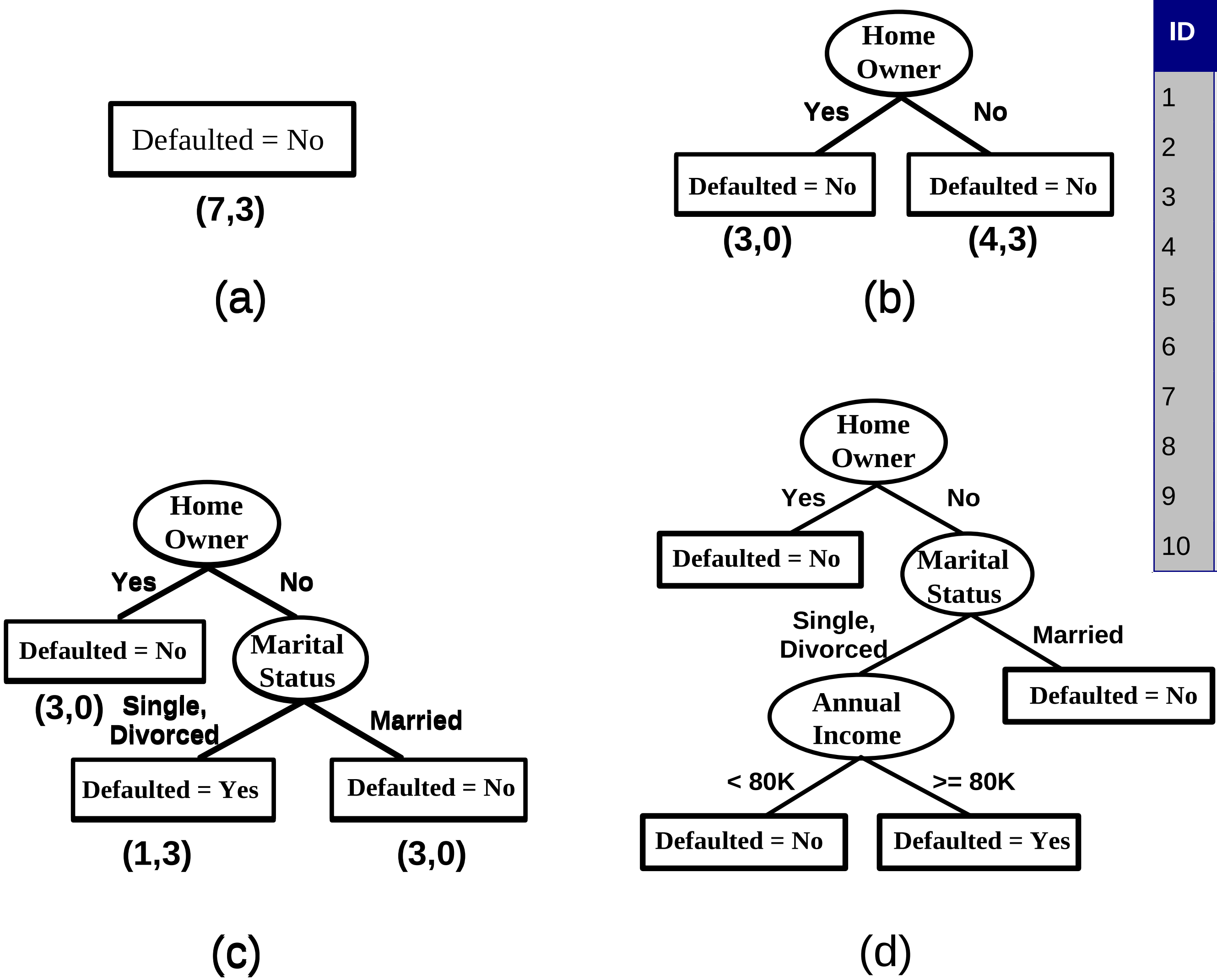
(d)

Hunt's Algorithm



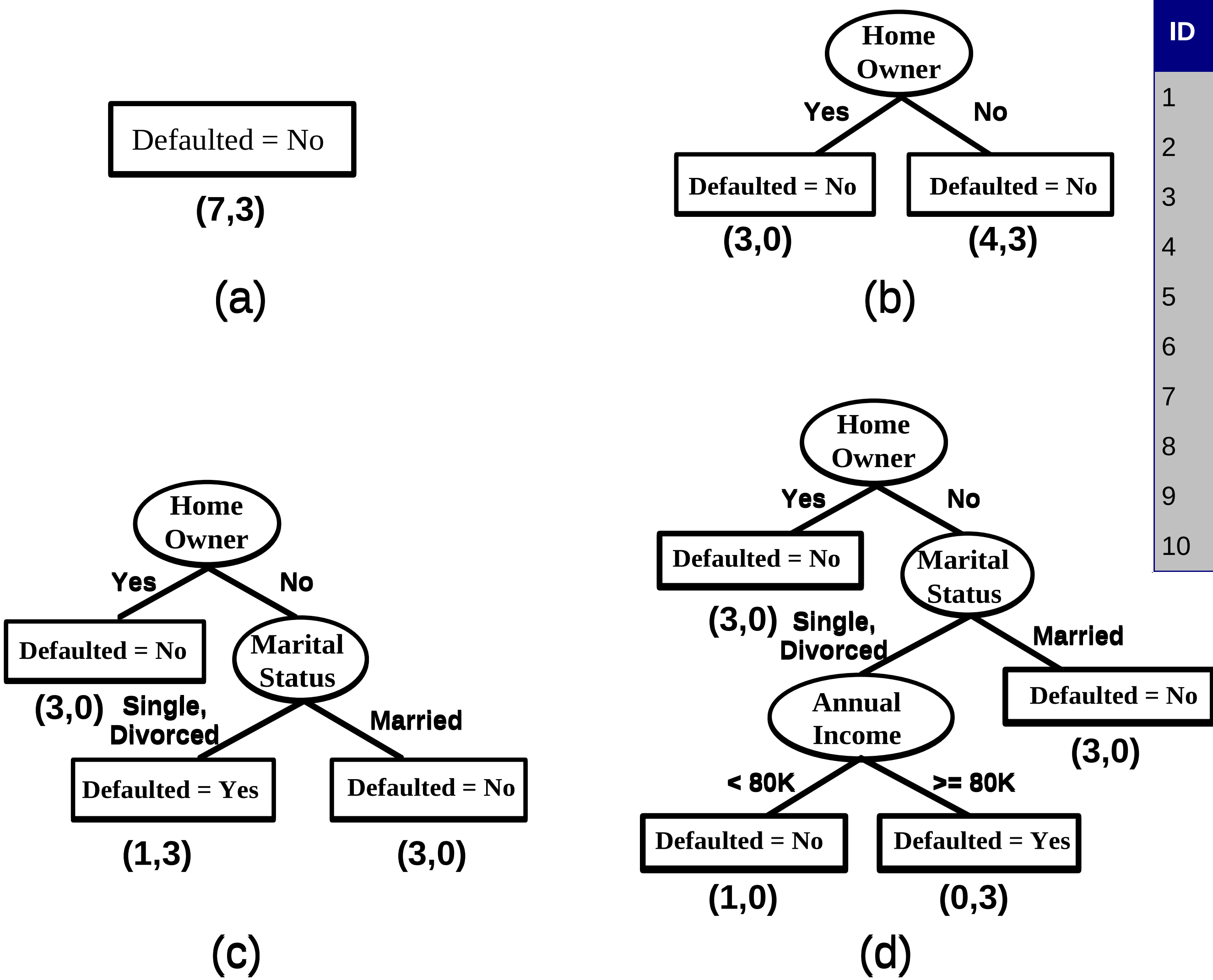
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Hunt's Algorithm



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

Design Issues of Decision Tree Induction

- How should training records be split?
  - Method for expressing test condition
    - depending on attribute types
  - Measure for evaluating the goodness of a test condition

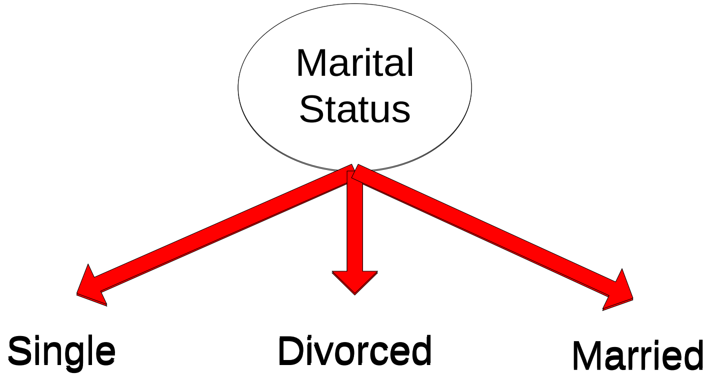
- How should the splitting procedure stop?
  - Stop splitting if all the records belong to the same class or have identical attribute values
  - Early termination

Methods for Expressing Test Conditions

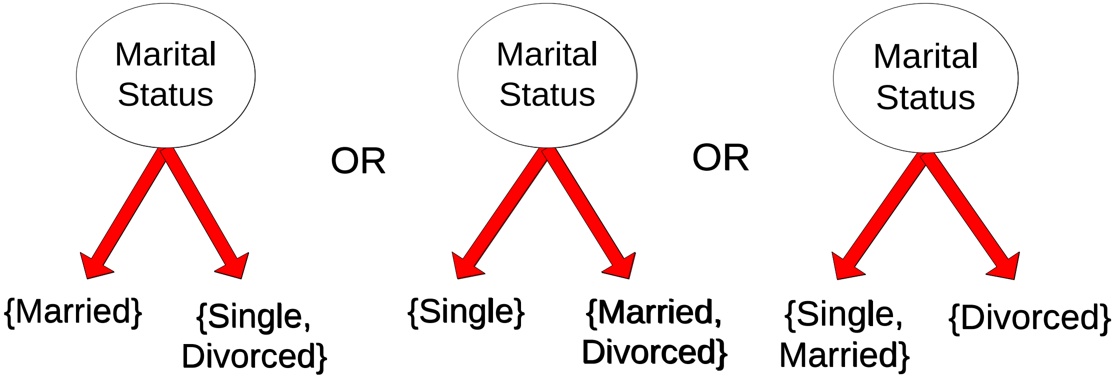
- Depends on attribute types
  - Binary
  - Nominal
  - Ordinal
  - Continuous

Test Condition for Nominal Attributes

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

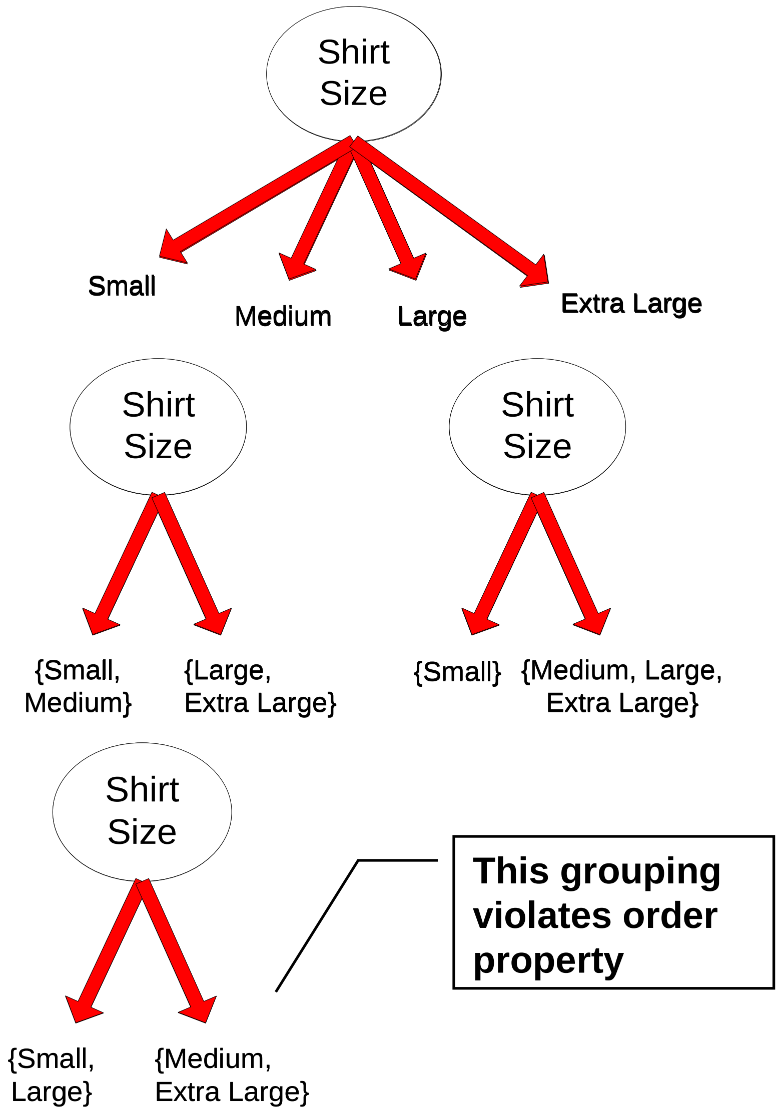


- Binary split:
  – Divides values into two subsets

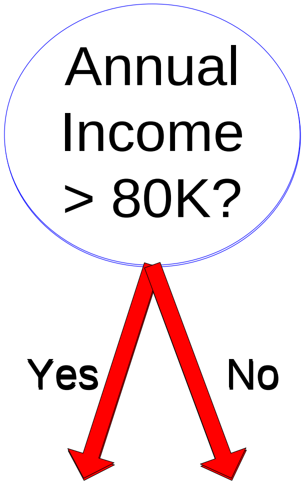


Test Condition for Ordinal Attributes

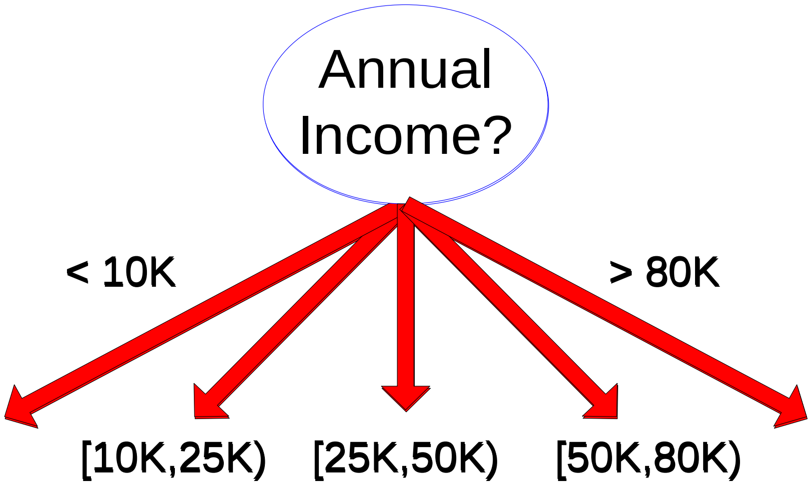
- Multi-way split:
  - Use as many partitions as distinct values
- Binary split:
  - Divides values into two subsets
  - Preserve order property among attribute values



Test Condition for Continuous Attributes



(i) Binary split

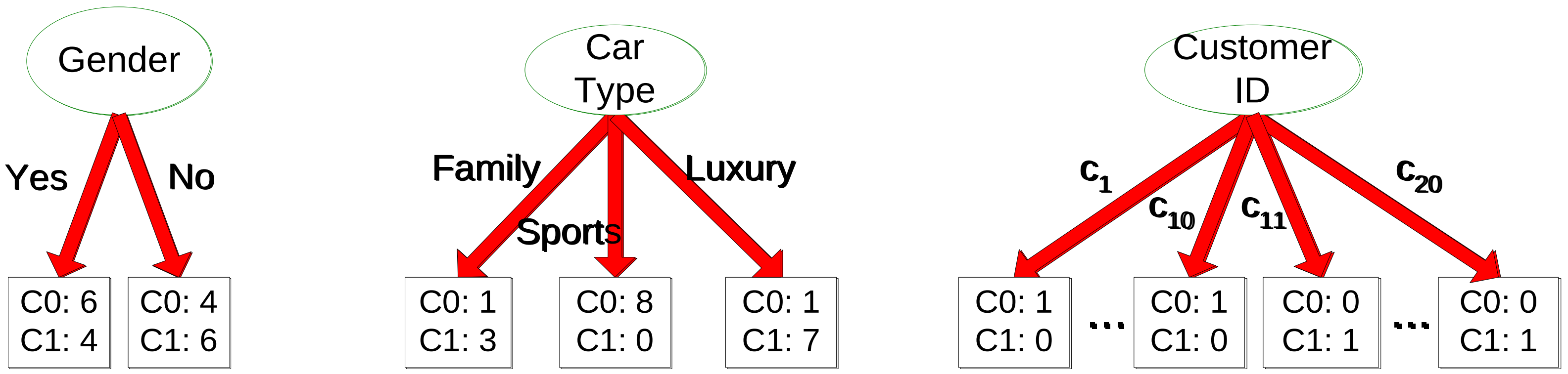


(ii) Multi-way split

How to determine the Best Split

Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

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 purer class distribution are preferred
- Need a measure of node impurity:

C0: 5	C0: 9
C1: 5	C1: 1
High degree of impurity	**Low degree of impurity**

Measures of Node Impurity

- Gini Index

$$Gini\,Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

Where is the frequency of class at node **t**, and is the total number of classes

- Entropy

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

- Misclassification error

$$Classification\,error = 1 - max\,[p|i(t)]$$

Finding the Best Split

1. Compute impurity measure (P) before splitting
2. Compute impurity measure (M) after splitting
   - Compute impurity measure of each child node
   - M is the weighted impurity of child nodes
3. Choose the attribute test condition that produces the highest gain

Gain = P - M

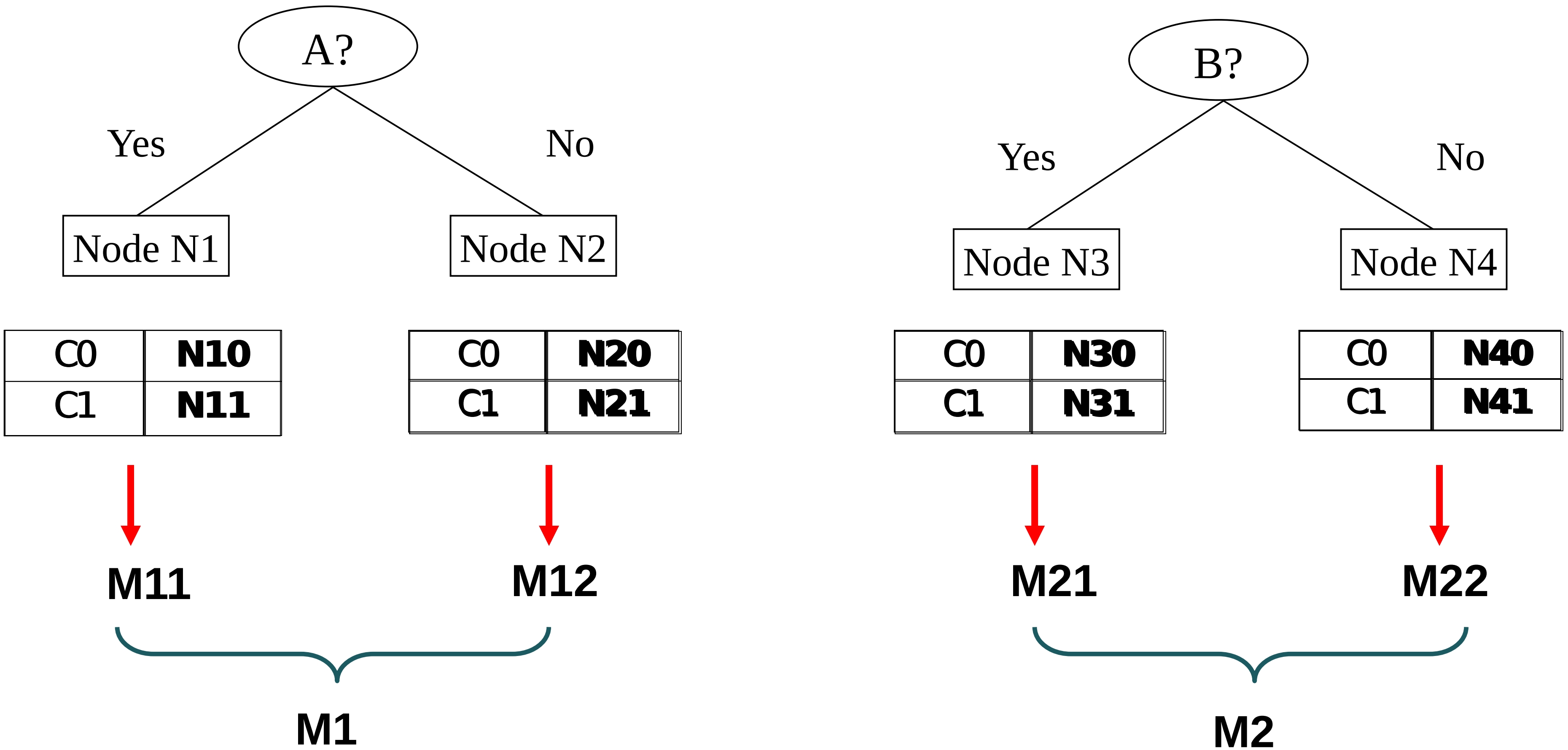
or equivalently, lowest impurity measure after splitting (M)

Finding the Best Split

Before Splitting:

C0	**N00**
C1	**N01**

→ **P**



Gain = P – M1 vs P – M2

Measure of Impurity: GINI

- Gini Index for a given node

$$Gini\,Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

Where is the frequency of class at node , and is the total number of classes

  - Maximum of when records are equally distributed among all classes, implying the least beneficial situation for classification
  - Minimum of 0 when all records belong to one class, implying the most beneficial situation for classification
  - Gini index is used in decision tree algorithms such as CART, SLIQ, SPRINT

Measure of Impurity: GINI

- Gini Index for a given node t :

$$Gini\,Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

  - For 2-class problem (p, 1 – p):
    - GINI = 1 – p^2 – $(1 – p)^2$ = 2p (1-p)

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	

Computing Gini Index of a Single Node

$$Gini\,Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

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

Computing Gini Index for a Collection of Nodes

- When a node is split into partitions (children)

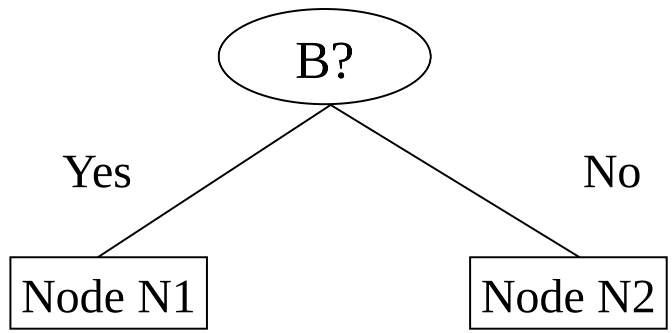
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, = number of records at child ,

 = number of records at parent node .

Binary Attributes: Computing GINI Index

- Splits into two partitions (child nodes)
- Effect of Weighing partitions:
  - Larger and purer partitions are sought



	Parent
C1	**7**
C2	**5**
Gini = 0.486	

Gini(N1)
$= 1 - (5/6)^2 - (1/6)^2$
$= 0.278$

Gini(N2)
$= 1 - (2/6)^2 - (4/6)^2$
$= 0.444$

	N1	N2
C1	**5**	**2**
C2	**1**	**4**
Gini=0.361		

Weighted Gini of N1 N2
$= 6/12 * 0.278 +$
$6/12 * 0.444$
$= 0.361$

Gain = 0.486 – 0.361 = 0.125

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	8	1
C2	3	0	7
Gini	0.163		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C1	9	1
C2	7	3
Gini	0.468	

	CarType	
	{Sports}	{Family, Luxury}
C1	8	2
C2	0	10
Gini	0.167	

Which of these is the best?

Continuous Attributes: Computing Gini Index

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

ID	Home Owner	Marital Status	Annual Income	Defaulted
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

Annual Income ?

	≤ 80	> 80
Defaulted Yes	0	3
Defaulted No	3	4

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

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
Annual Income	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	

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
	Annual 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	

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
	Annual 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	

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
	Annual 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	

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	
		Annual 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	

Measure of Impurity: Entropy

- Entropy at a given node

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

Where is the frequency of class at node , and is the total number of classes

  - Maximum of when records are equally distributed among all classes, implying the least beneficial situation for classification
  - Minimum of 0 when all records belong to one class, implying most beneficial situation for classification

– Entropy based computations are quite similar to the GINI index computations

Computing Entropy of a Single Node

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

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

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

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

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

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

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

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

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

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Computing Information Gain After Splitting

- Information Gain:

$$Gain_{split} = Entropy(p) - \sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)$$

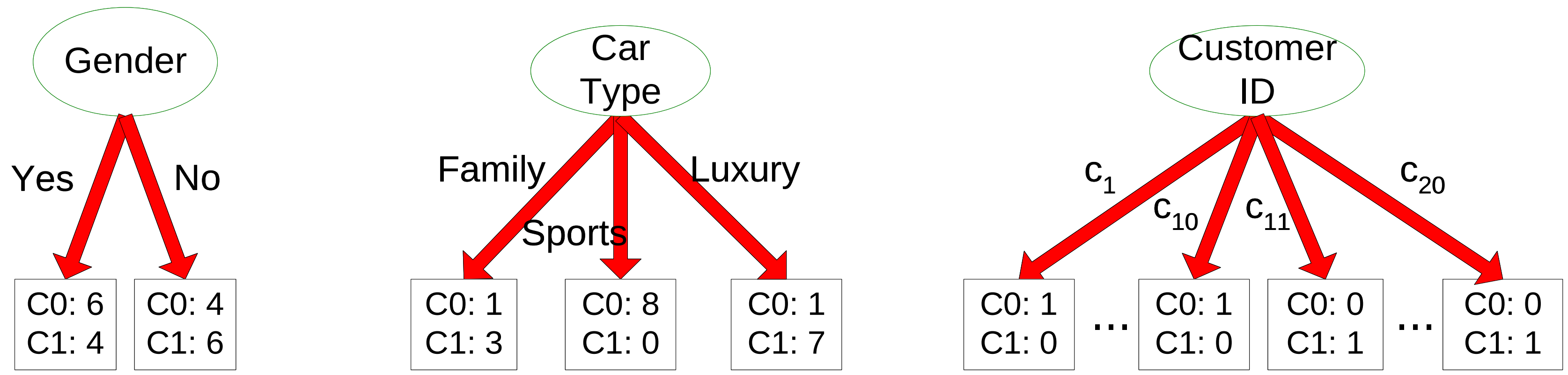
Parent Node, is split into partitions (children)

is number of records in child node

  - Choose the split that achieves most reduction (maximizes GAIN)
  - Used in the ID3 and C4.5 decision tree algorithms
  - Information gain is the mutual information between the class variable and the splitting variable

Problem with large number of partitions

- Node impurity measures tend to prefer splits that result in large number of partitions, each being small but pure



  – Customer ID has highest information gain because entropy for all the children is zero

Gain Ratio

- Gain Ratio:

$$Gain\,Ratio = \frac{Gain_{split}}{Split\,Info} \quad Split\,Info = -\sum_{i=1}^{k} \frac{n_i}{n} \log_2 \frac{n_i}{n}$$

Parent Node, is split into partitions (children)

 is number of records in child node

  - Adjusts Information Gain by the entropy of the partitioning ().
    - Higher entropy partitioning (large number of small partitions) is penalized!
  - Used in C4.5 algorithm
  - Designed to overcome the disadvantage of Information Gain

Gain Ratio

- Gain Ratio:

$$Gain\ Ratio = \frac{Gain_{split}}{Split\ Info} \quad Split\ Info = \sum_{i=1}^{k} \frac{n_i}{n} \log_2 \frac{n_i}{n}$$

Parent Node, is split into partitions (children)

 is number of records in child node

	CarType		
	Family	Sports	Luxury
C1	1	8	1
C2	3	0	7
Gini	0.163		

SplitINFO = 1.52

	CarType	
	{Sports, Luxury}	{Family}
C1	9	1
C2	7	3
Gini	0.468	

SplitINFO = 0.72

	CarType	
	{Sports}	{Family, Luxury}
C1	8	2
C2	0	10
Gini	0.167	

SplitINFO = 0.97

Measure of Impurity: Classification Error

- Classification error at a node

$$Error(t) = 1 - \max_i [p_i(t)]$$

  – Maximum of when records are equally distributed among all classes, implying the least interesting situation
  – Minimum of 0 when all records belong to one class, implying the most interesting situation

Computing Error of a Single Node

$$Error(t) = 1 - \max_i [p_i(t)]$$

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

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

Error = 1 – max (0, 1) = 1 – 1 = 0

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

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

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

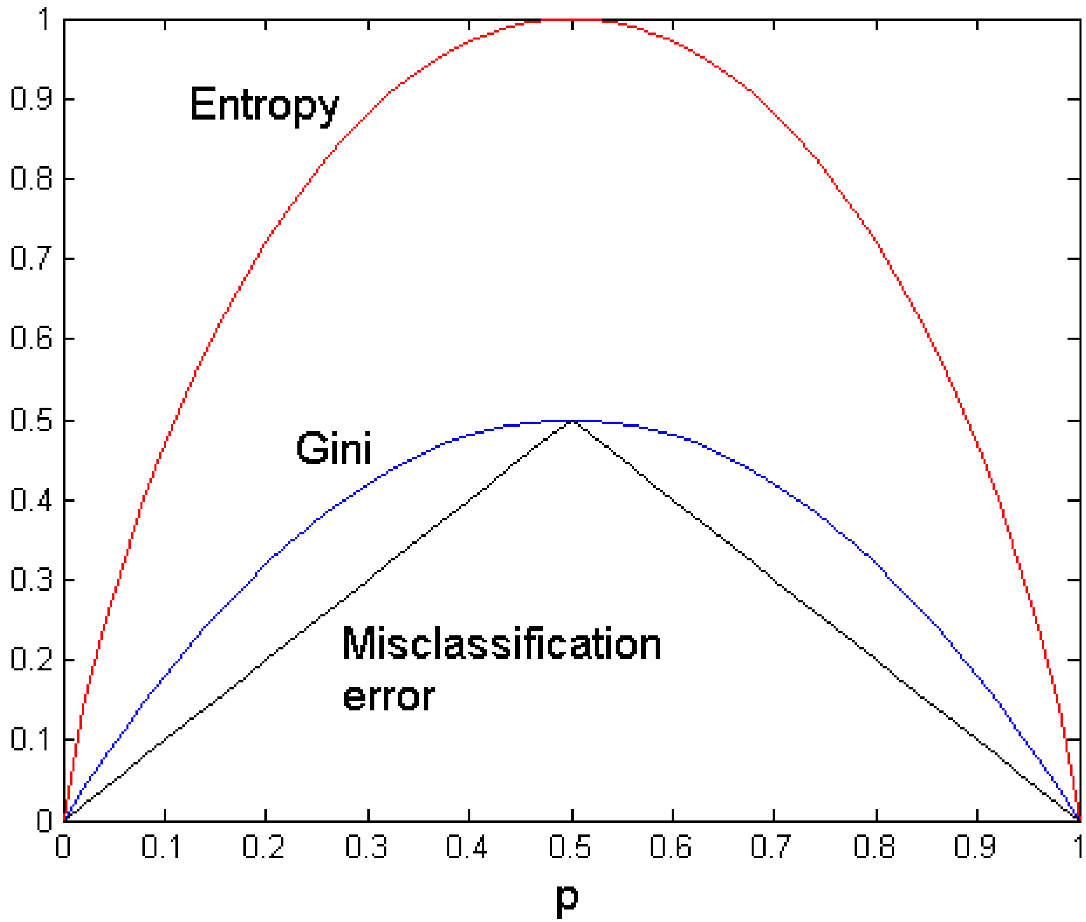
C1	**2**
C2	**4**

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

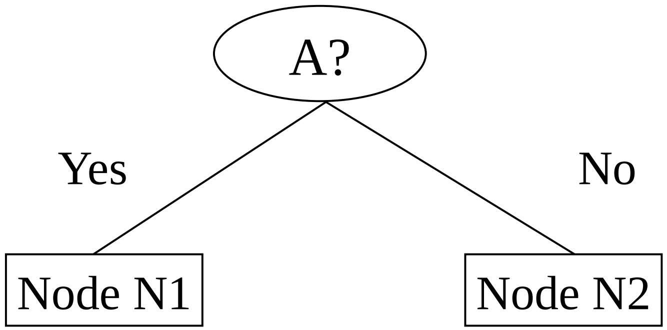
Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Comparison among Impurity Measures

For a 2-class problem:



Misclassification Error vs Gini Index



	Parent
C1	**7**
C2	**3**
Gini = 0.42	

Gini(N1)
$= 1 - (3/3)^2 - (0/3)^2$
$= 0$

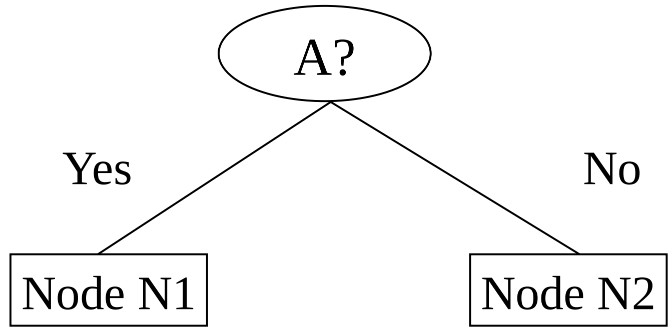
Gini(N2)
$= 1 - (4/7)^2 - (3/7)^2$
$= 0.489$

	N1	N2
C1	**3**	**4**
C2	**0**	**3**
Gini=0.342		

Gini(Children)
$= 3/10 * 0$
$+ 7/10 * 0.489$
$= 0.342$

Gini improves but error remains the same!!

Misclassification Error vs Gini Index



	Parent
C1	7
C2	3
Gini = 0.42	

	N1	N2
C1	3	4
C2	0	3
Gini=0.342		

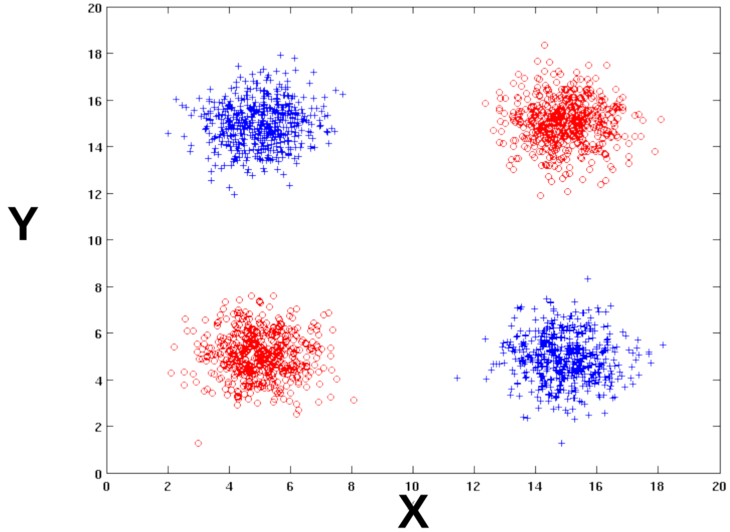
	N1	N2
C1	3	4
C2	1	2
Gini=0.416		

Misclassification error for all three cases = 0.3 !

Decision Tree Based Classification

- Advantages:
  - Relatively inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Robust to noise (especially when methods to avoid overfitting are employed)
  - Can easily handle redundant attributes
  - Can easily handle irrelevant attributes (unless the attributes are interacting)
- Disadvantages: .
  - Due to the greedy nature of splitting criterion, interacting attributes (that can distinguish between classes together but not individually) may be passed over in favor of other attributed that are less discriminating.
  - Each decision boundary involves only a single attribute

Handling interactions

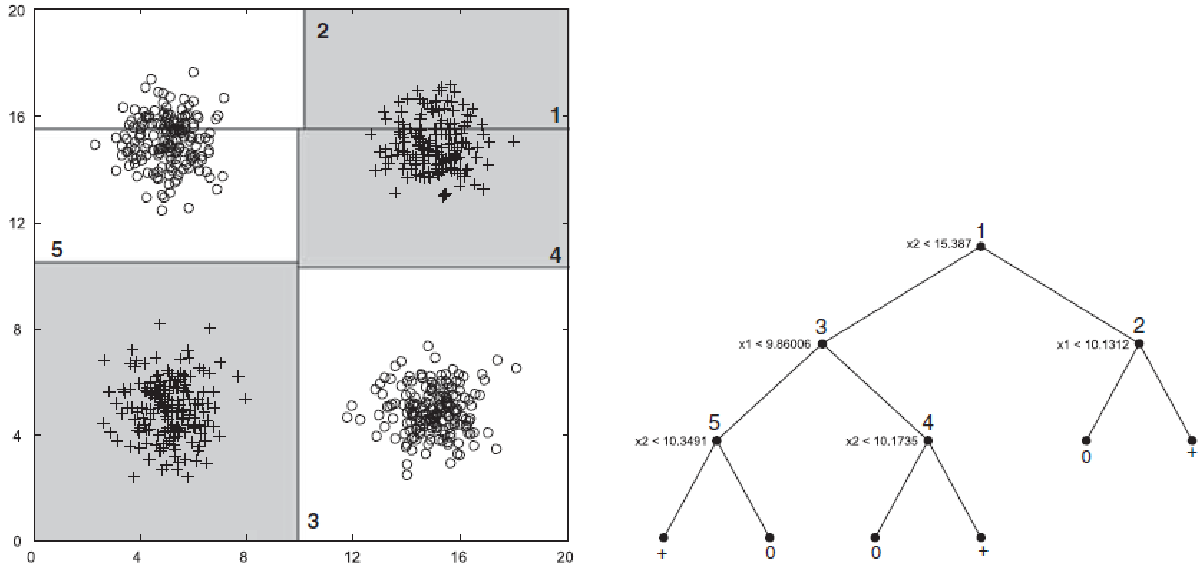


+ : 1000 instances

o : 1000 instances

Entropy (X) : 0.99
Entropy (Y) : 0.99

Handling interactions



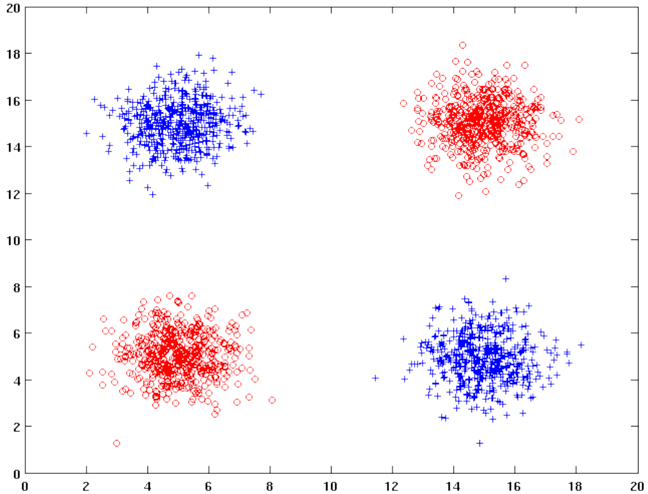
(a) Decision boundary for tree with 6 leaf nodes.

(b) Decision tree with 6 leaf nodes.

Figure 3.28. Decision tree with 6 leaf nodes using X and Y as attributes. Splits have been numbered from 1 to 5 in order of other occurrence in the tree.

Handling interactions given irrelevant attributes

Y



X

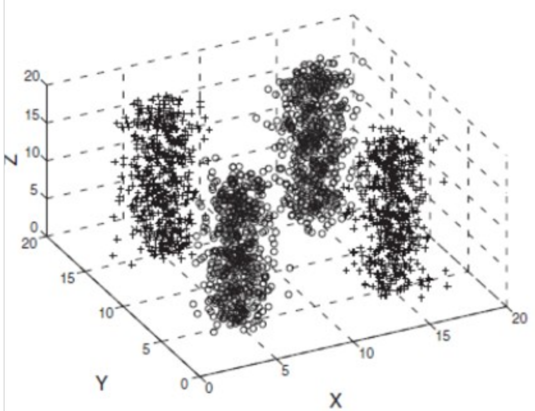
+ : 1000 instances

o : 1000 instances

Adding Z as a noisy attribute generated from a uniform distribution

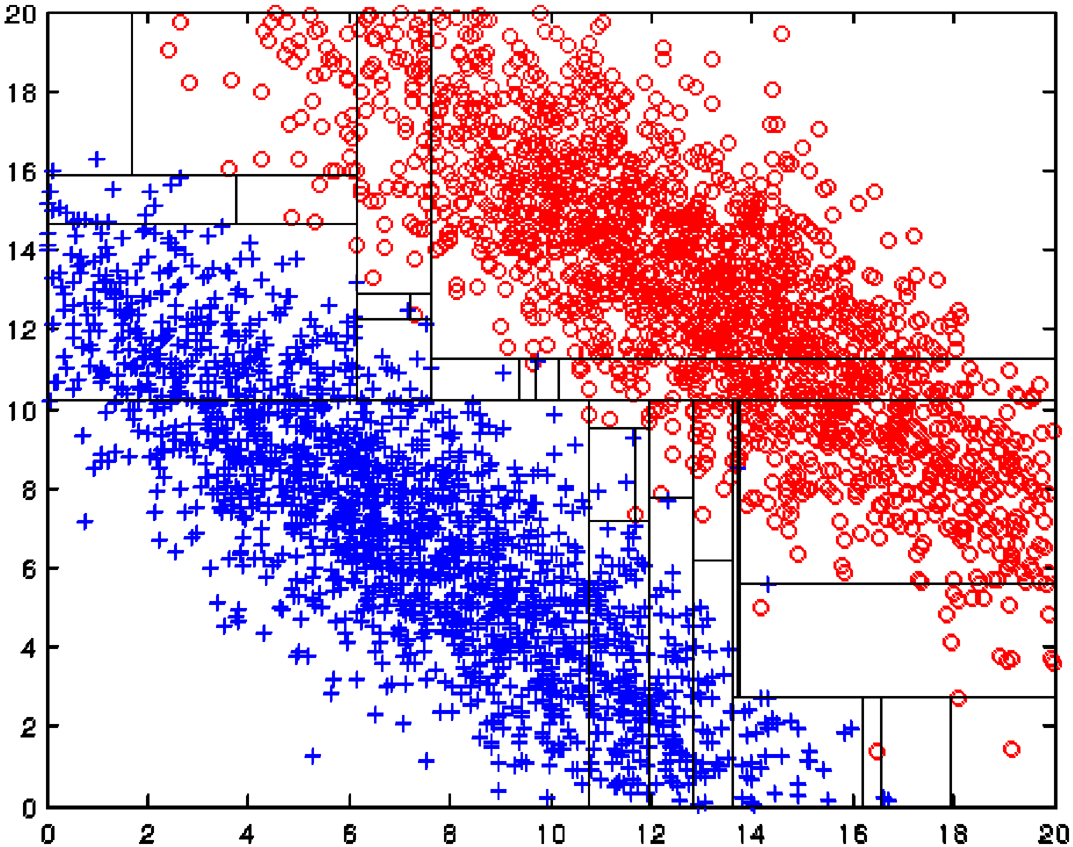
Entropy (X) : 0.99
Entropy (Y) : 0.99
Entropy (Z) : 0.98

Attribute Z will be chosen for splitting!



(a) Three-dimensional data with attributes X, Y, and Z.

Limitations of single attribute-based decision boundaries



Both positive (+) and negative (o) classes generated from skewed Gaussians with centers at (8,8) and (12,12) respectively.