COMP7/8118 M50

Data Mining

Decision Tree: I

Xiaofei Zhang

Slides compiled from Jiawei Han and Raymond C.W. Wong's work

THE UNIVERSITY OF MEMPHIS

Decision Trees

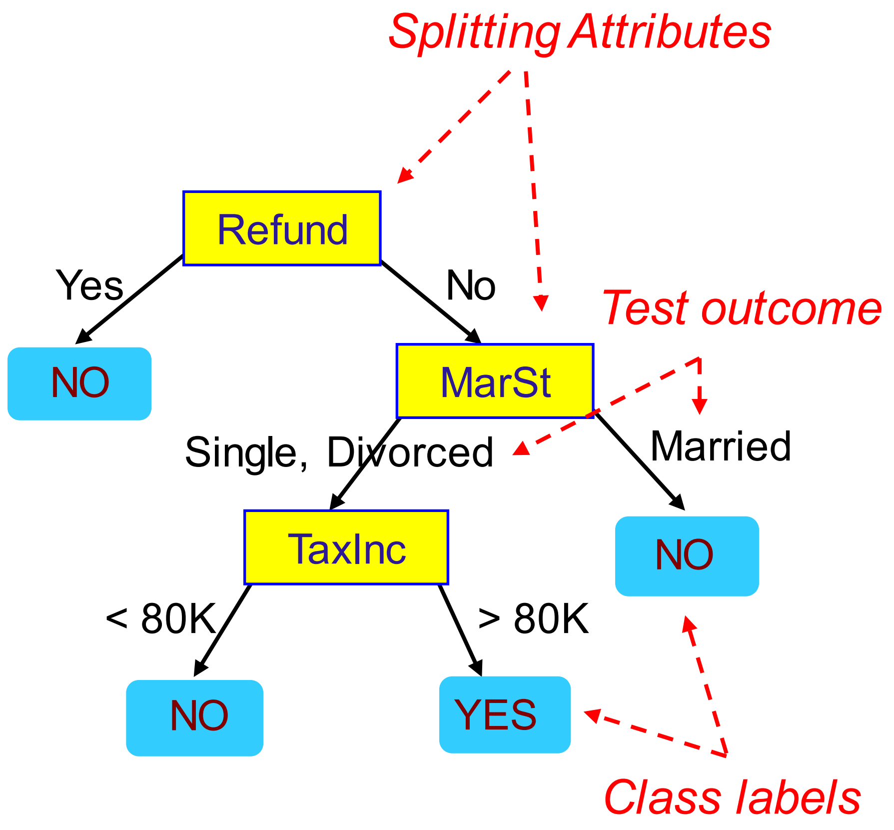
- Decision tree
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution

Example of a Decision Tree

categorical categorical continuous class

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

Training Data

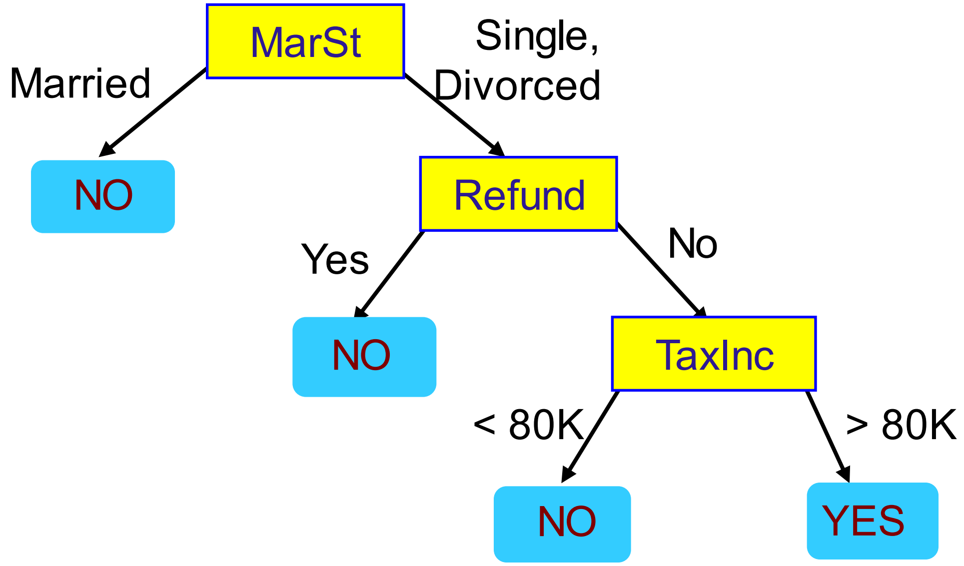


Model: Decision Tree

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

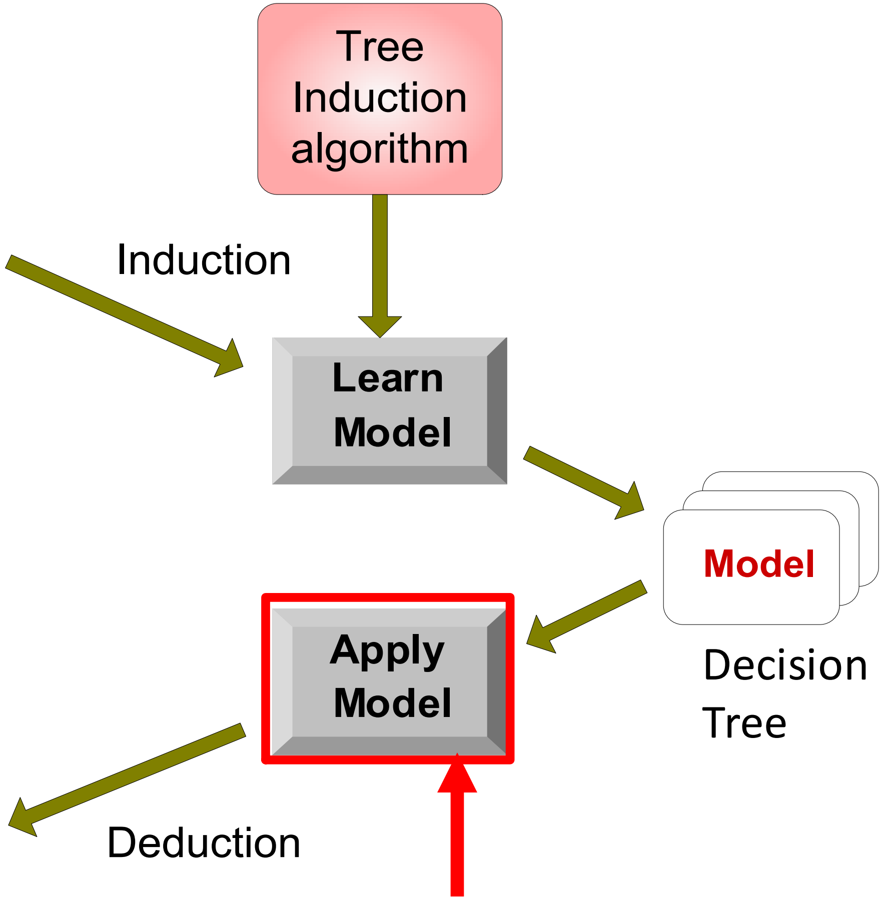


There could be more than one tree that fits the same data!

Decision Tree Classification Task

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

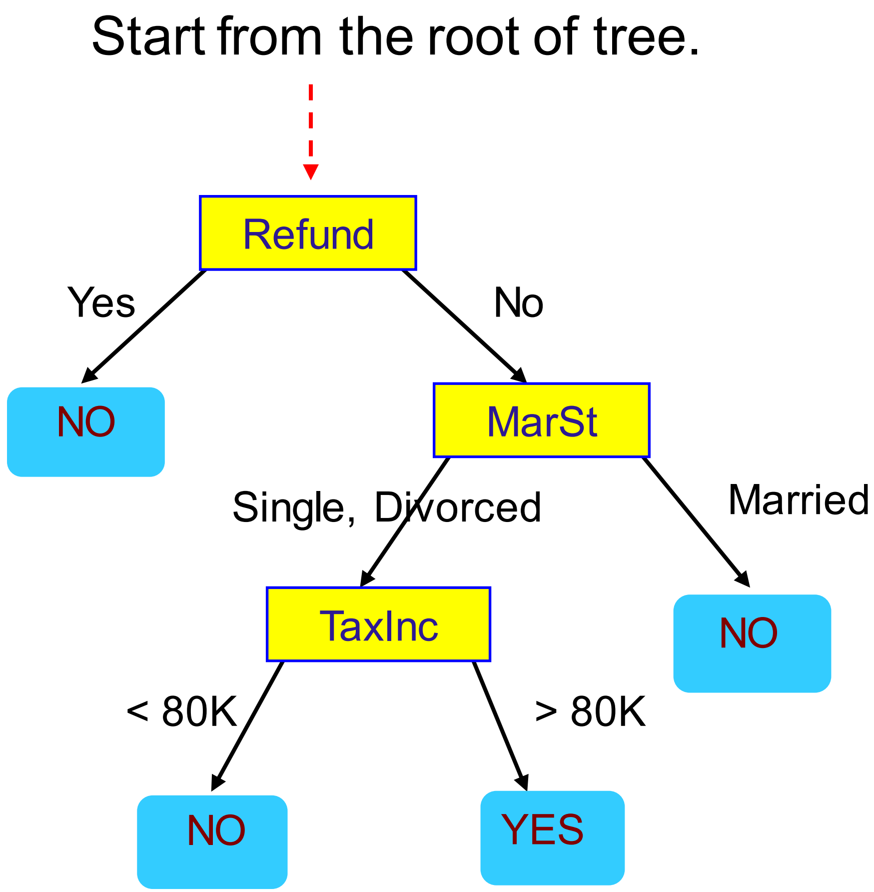
Training Set



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

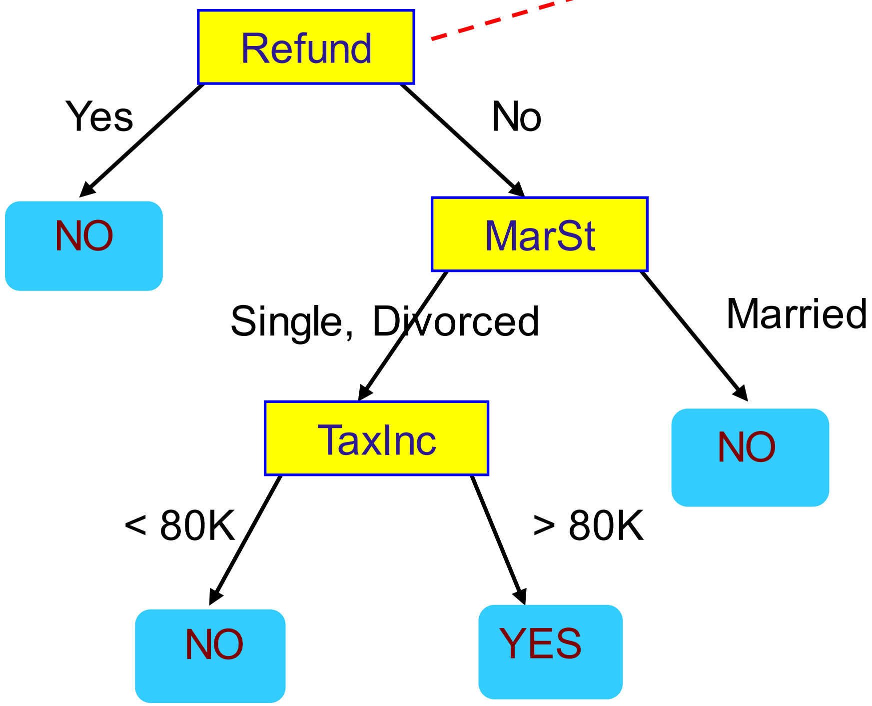
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

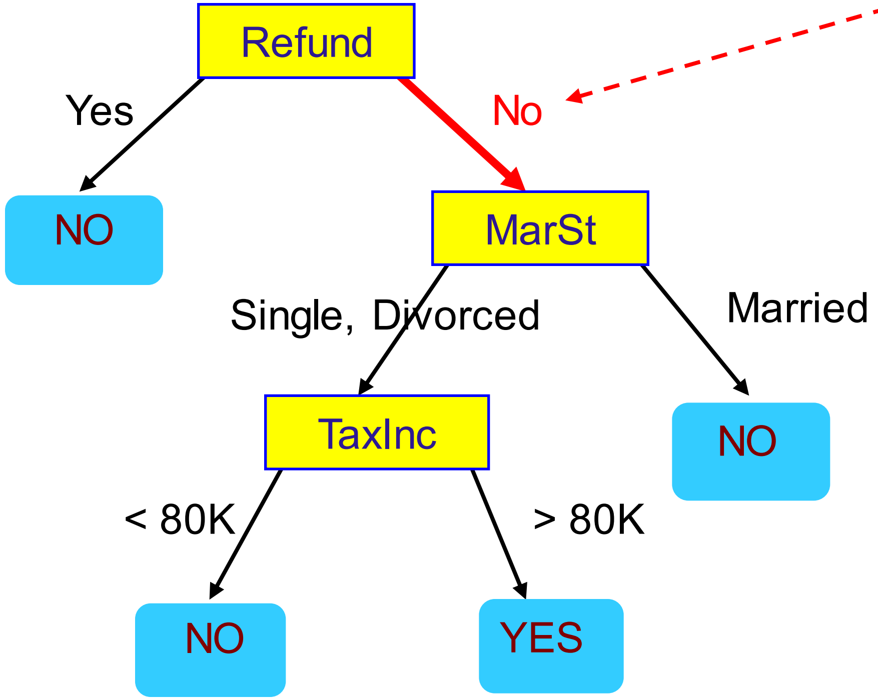
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

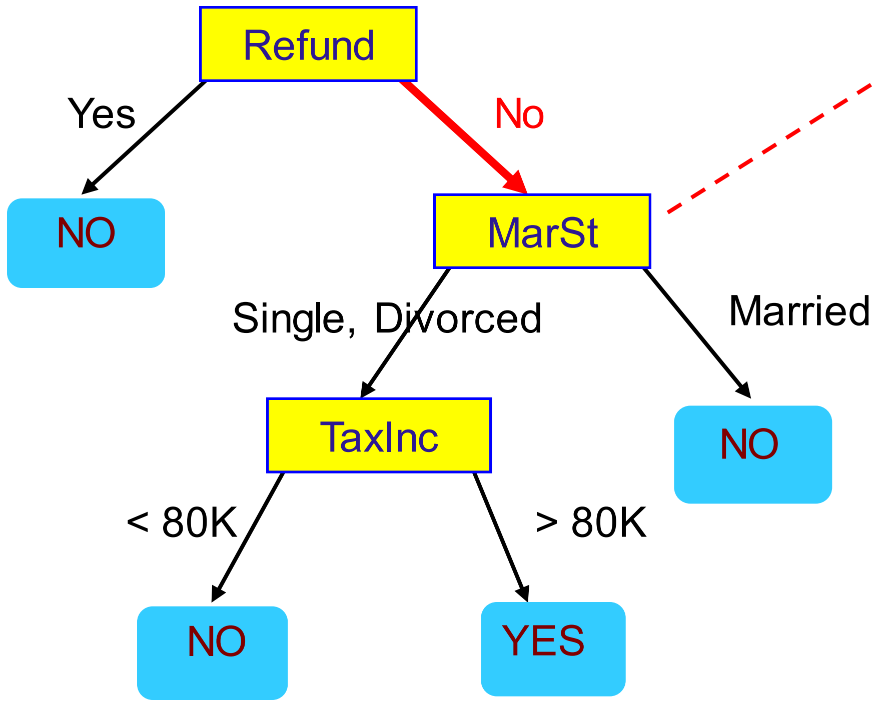
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

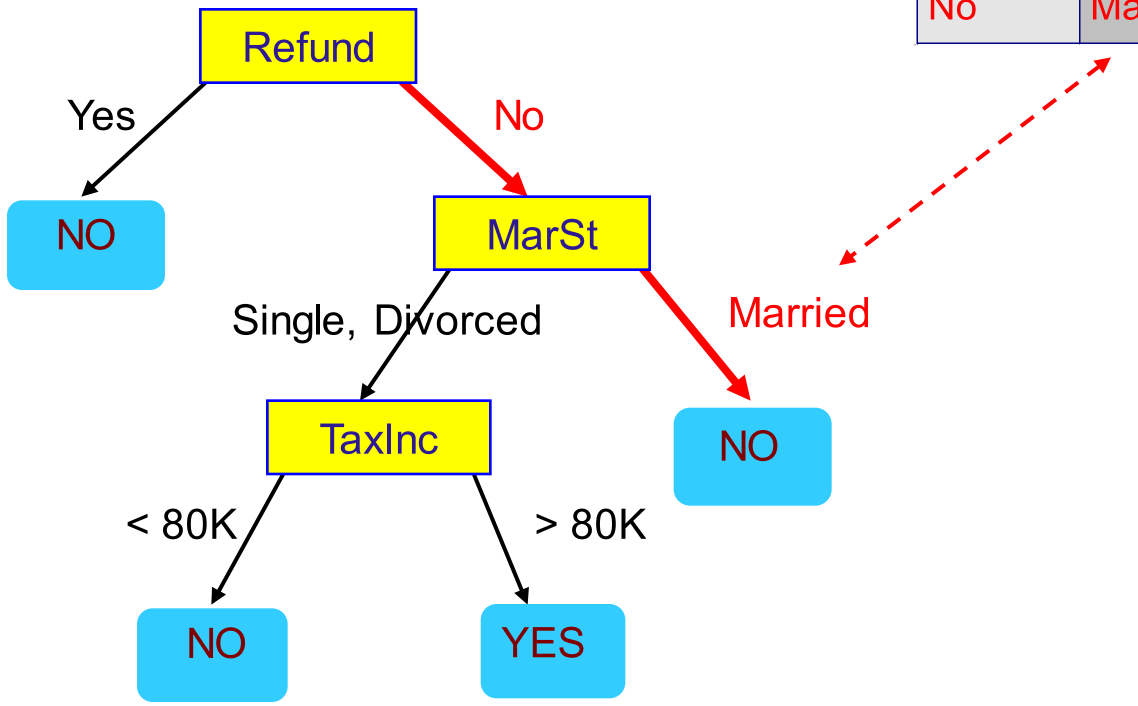
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



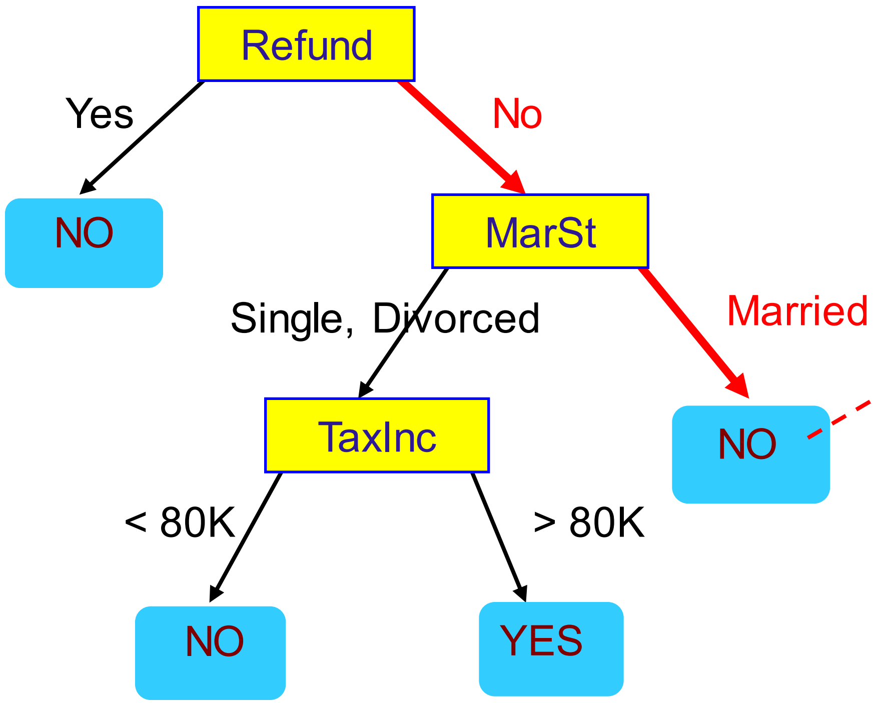
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

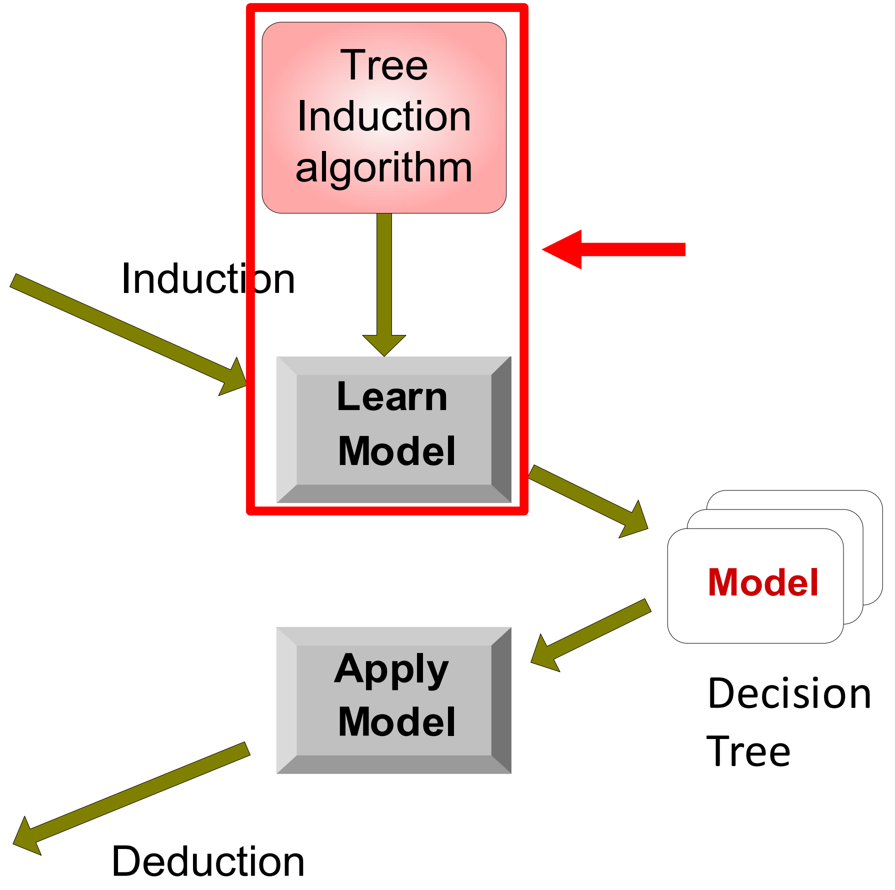


Assign Cheat to “No”

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

Tree Induction

- Finding the best decision tree (lowest training error) is NP-hard

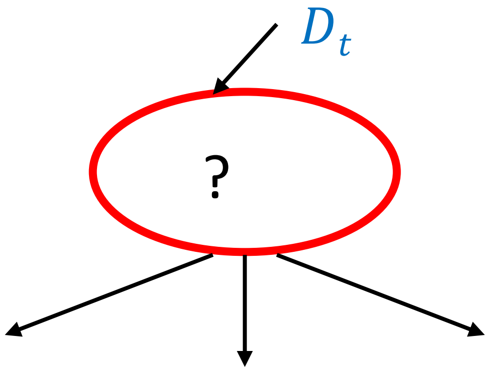
- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

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

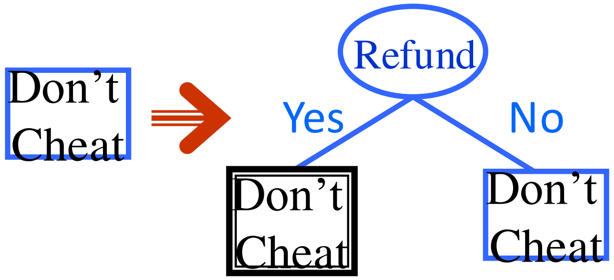
General Structure of Hunt's Algorithm

- 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 with the same attribute values, then t is a leaf node labeled with the majority class y_t
  - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets.
- Recursively apply the procedure to each subset.

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

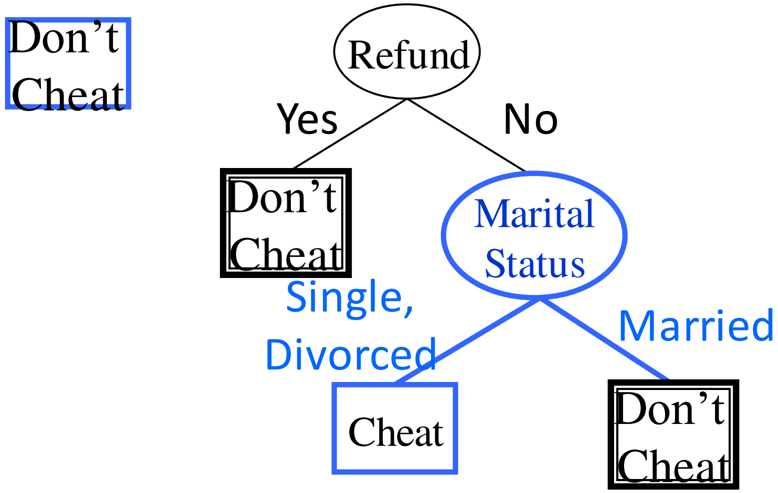


Hunt’s Algorithm



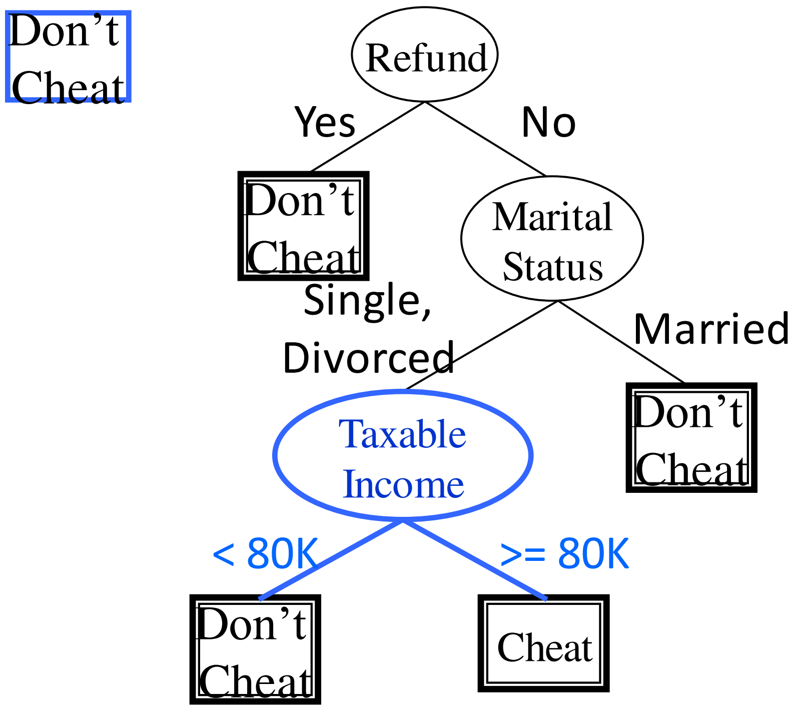
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	**No**
4	Yes	Married	120K	**No**
7	Yes	Divorced	220K	**No**
2	No	Married	100K	**No**
6	No	Married	60K	**No**
9	No	Married	75K	**No**
3	No	Single	70K	**No**
5	No	Divorced	95K	**Yes**
8	No	Single	85K	**Yes**
10	No	Single	90K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

Hunt’s Algorithm



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

Hunt’s Algorithm



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

Constructing decision-trees (pseudocode)

GenDecTree(Sample S, Features F)

1. **If stopping_condition(S,F)** = true **then**
    a. **leaf = createNode()**
    b. **leaf.label= Classify(S)**
    c. **return leaf**
2. **root = createNode()**
3. **root.test_condition = findBestSplit(S,F)**
4. V = {v| v a possible outcome of **root.test_condition**}
5. **for** ***each*** value **v**ϵ**V**:
    a. S_v: = {s | **root.test_condition(s)** = v and s ϵ S};
    b. **child = GenDecTree(**S_v ,F) ;
    c. Add **child** as a descent of **root** and label the edge **(root→child) as** v
6. **return root**

Tree Induction

- Issues
  - How to **Classify** a leaf node
    - Assign the majority class
    - If leaf is empty, assign the default class – the class that has the highest popularity.
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

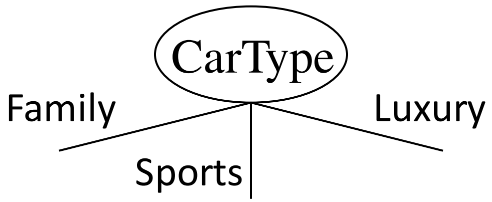
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

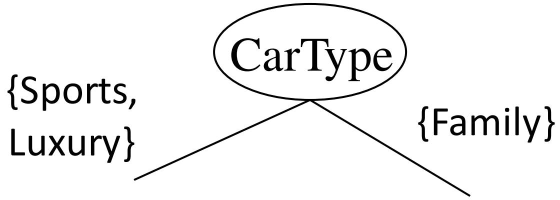
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

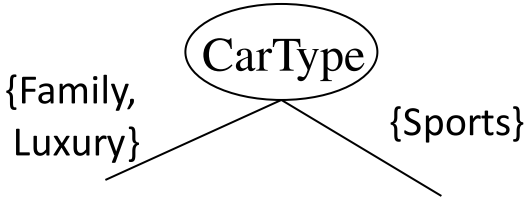
- Multi-way split: Use as many partitions as distinct values.



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

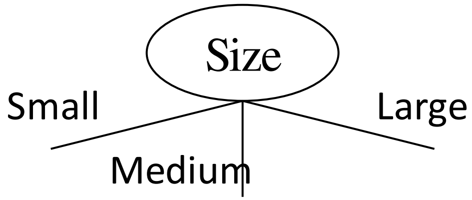


OR

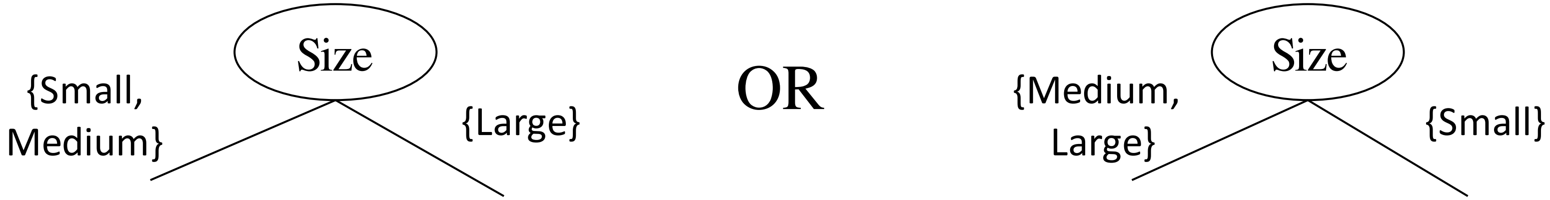


Splitting Based on Ordinal Attributes

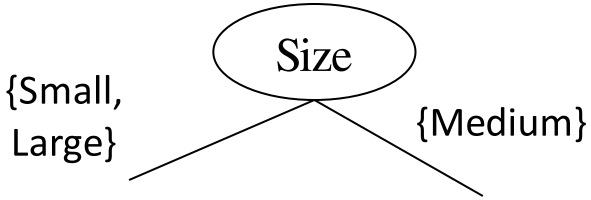
- Multi-way split: Use as many partitions as distinct values.



- Binary split: Divides values into two subsets – respects the order. Need to find optimal partitioning.



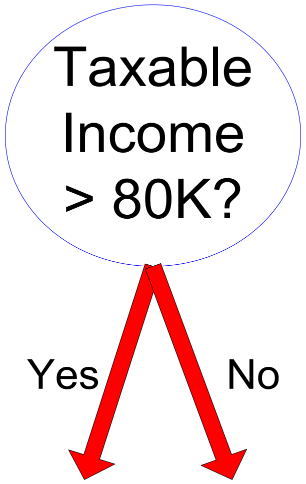
- What about this split?



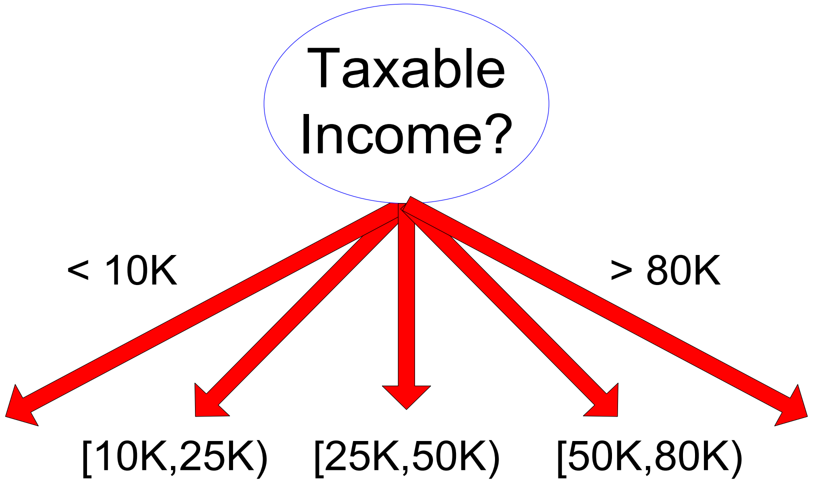
Splitting Based on Continuous Attributes

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

Splitting Based on Continuous Attributes



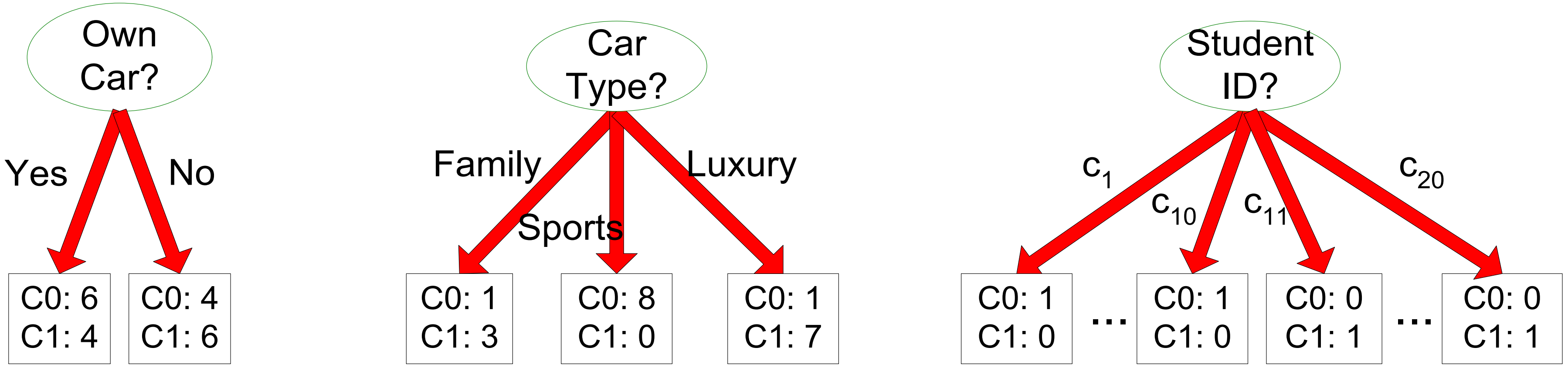
(i) Binary split



(ii) Multi-way split

How to determine the Best Split

Before Splitting: 10 records of class C0, 10 records of class C1



Which test condition is the best?

How to determine the Best Split

- Greedy approach:
  - Creation of nodes with homogeneous class distribution is preferred
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, High degree of impurity	Homogeneous, Low degree of impurity

Measuring Node Impurity

- p(i|t): fraction of records associated with node **t** belonging to class **i**
  - Used in ID3 and C4.5

$$\text{Entropy}(t) = -\sum_{i=1}^{c} p(i|t)\log p(i|t)$$

  - Used in CART, SLIQ, SPRINT.

$$\text{Gini}(t) = 1 - \sum_{i=1}^{c} [p(i|t)]^2$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)]$$

Gain

- ***Gain* of an attribute split:** compare the impurity of the parent node with the average impurity of the child nodes

$$\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

- Maximizing the gain $\Leftrightarrow$ Minimizing the weighted average impurity measure of children nodes
- If **I() = Entropy(),** then Δ_{info} is called **information gain**

Example

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

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

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

Entropy $= – 0 \log 0 – 1 \log 1 = – 0 – 0 = 0$

Error $= 1 – \max (0, 1) = 1 – 1 = 0$

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

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

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

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

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

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

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

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

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

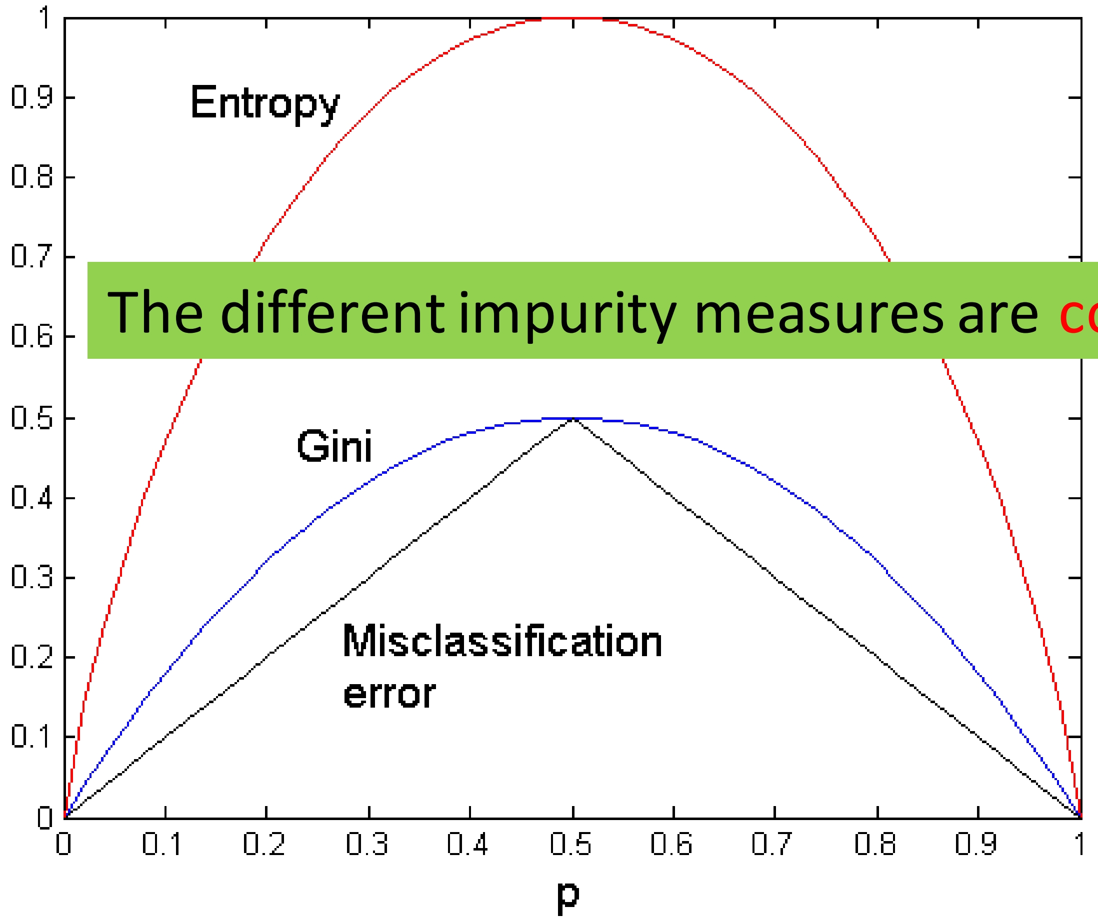
Error $= 1 – \max (2/6, 4/6) = 1 – 4/6 = 1/3$

Impurity measures

- All of the impurity measures take value zero (minimum) for the case of a pure node where a single value has probability 1
- All of the impurity measures take maximum value when the class distribution in a node is uniform.

Comparison among Splitting Criteria

For a 2-class problem:



The different impurity measures are consistent

Categorical Attributes

- For binary values split in two
- For multivalued attributes, 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)

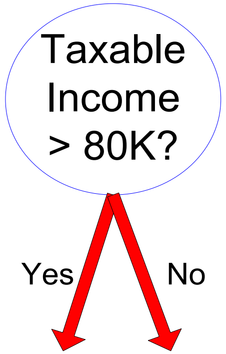
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

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

Continuous Attributes

- 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 $A \geq v$
- Exhaustive method to choose best v
  - For each v, scan the database to gather count matrix and compute the impurity 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

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing impurity
  - Choose the split position that has the least impurity

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	

Splitting based on impurity

- Impurity measures favor attributes with large number of values
- A test condition with large number of outcomes may not be desirable
  - # of records in each partition is too small to make predictions

Splitting based on INFO

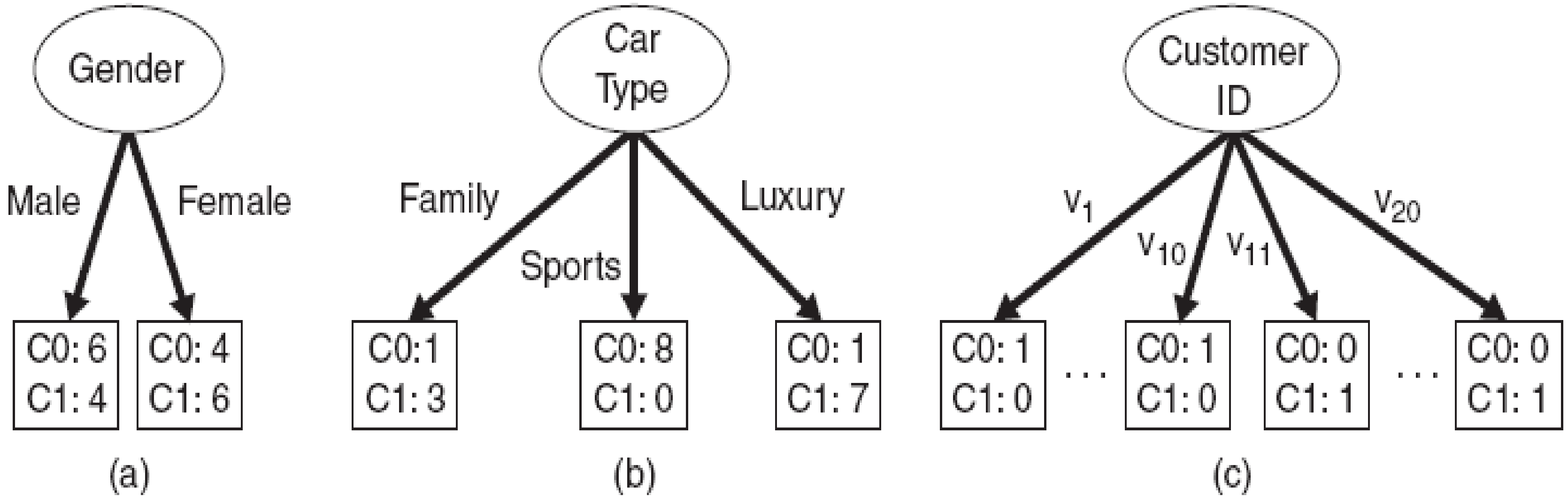


Figure 4.12. Multiway versus binary splits.

Gain Ratio

- Splitting using information gain

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

Parent Node, p is split into k partitions, n_i is the number of records in partition i

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

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