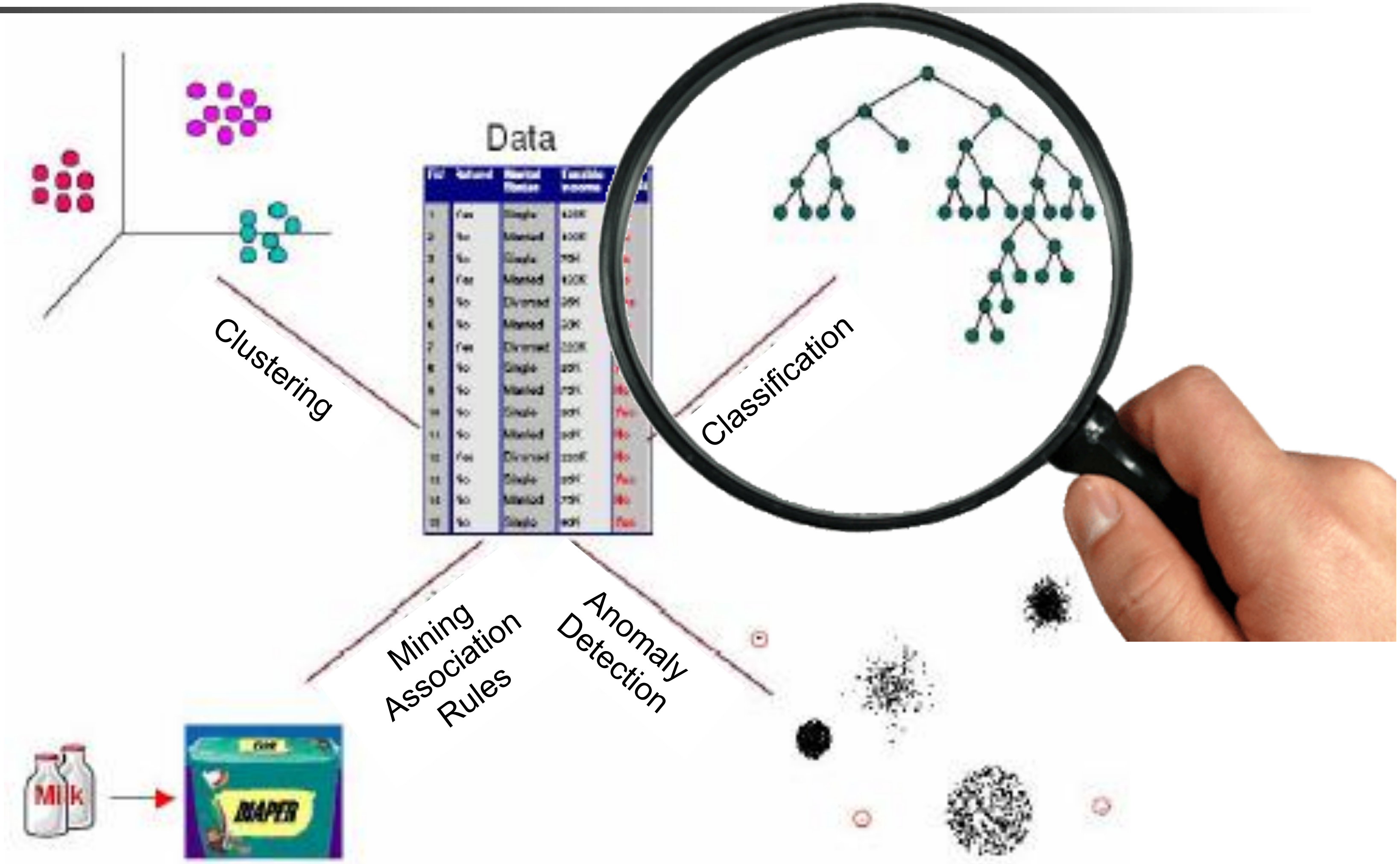
Data Mining Tasks



Classification Algorithms

- Nearest Neighbor
- Naïve Bayes
- **Decision Tree**
- ...

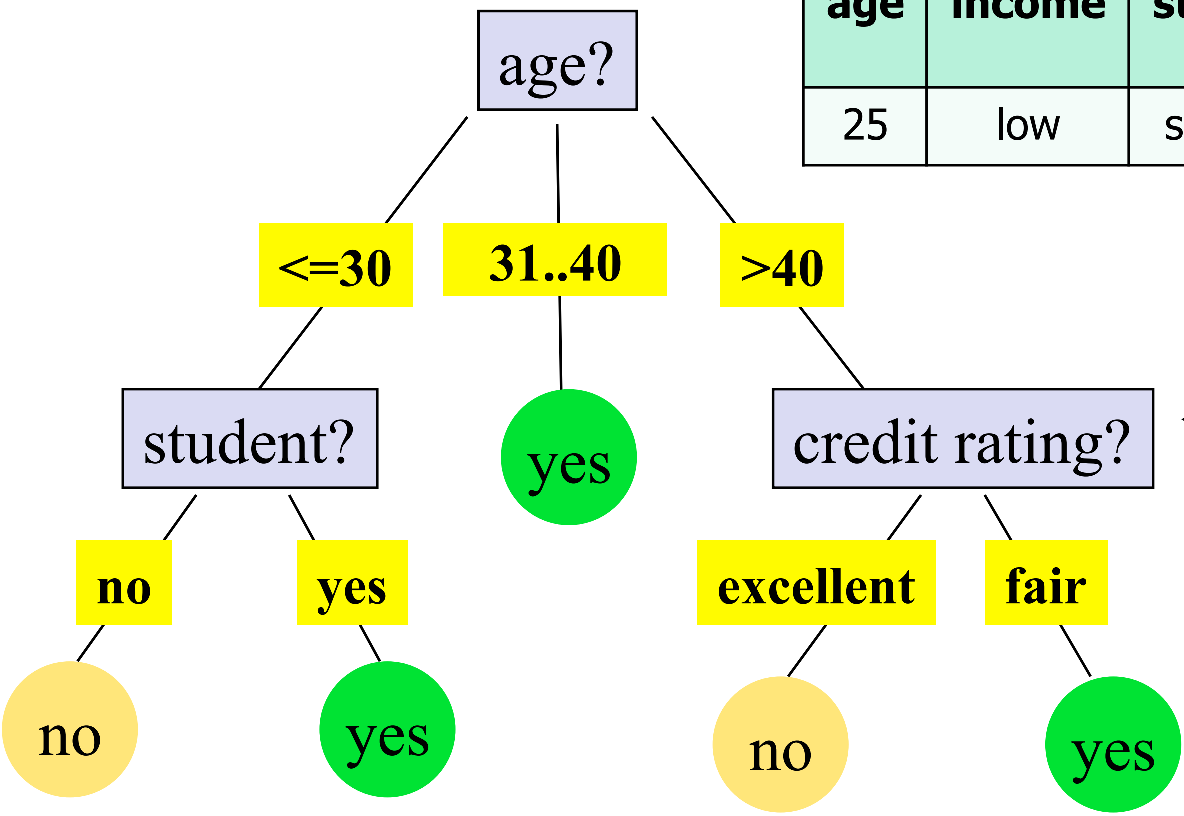
An Example

Whether a customer is likely to purchase a computer

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

A Decision Tree for "Buys Computer"

age	income	student	Credit rating	Buys computer
25	low	student	fair	**?**



Internal node
(a test on an attr)

leaf node
(class label)

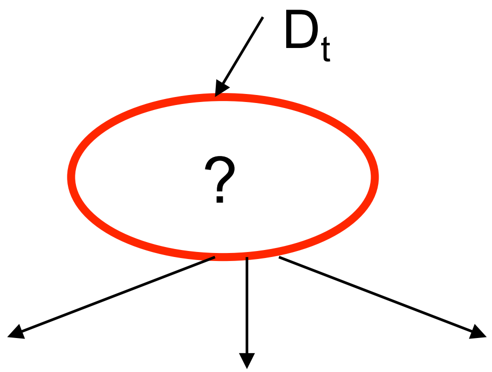
Decision Tree Induction

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

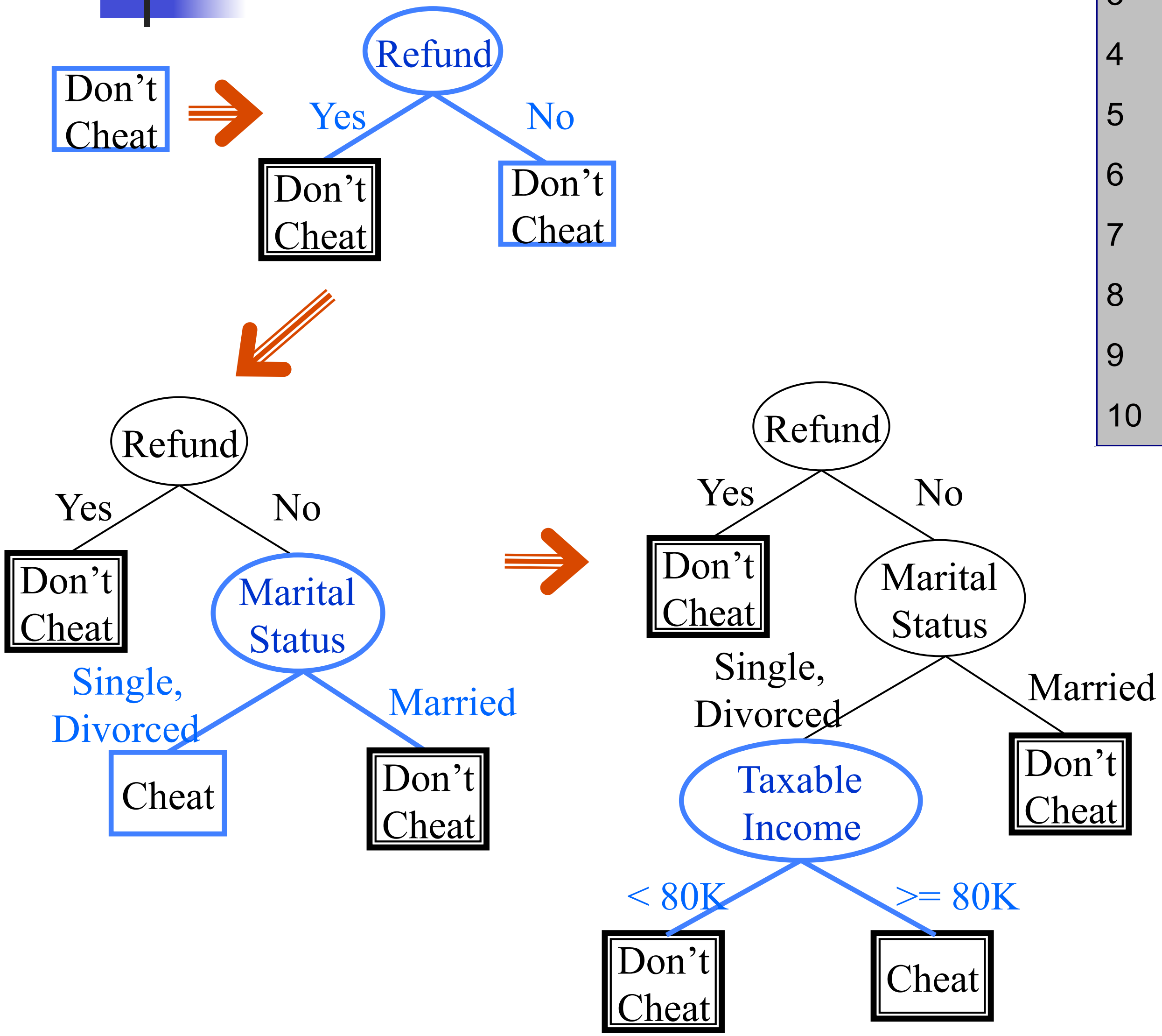
Structure of Hunt's Algorithm

- Let $\mathbf{D_t}$ be the set of training records that reach a node **t**
- **General Procedure:**
  1. If D_t contains records that belong to the same class y_t
     - then t is a **leaf** node labeled as y_t
  2. If D_t is an empty set,
     - then t is a **leaf** node labeled as the default class y_d (e.g., Cheat=No)
  3. If D_t contains records that belong to more than one class,
     - then use an attribute test to split the data into smaller subsets
     - Recursively apply the procedure to each subset.

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



Hunt's Algorithm



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

Tree Induction

- Greedy strategy
  - Split the records based on an **attribute test** that <u>optimizes</u> certain criterion

- Issues
  - Determine how to **split the records**
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to **stop splitting**

Tree Induction

- Greedy strategy
  - Split the records based on an **attribute test** that <u>optimizes</u> 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?

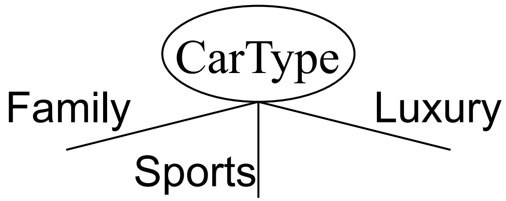
- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

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

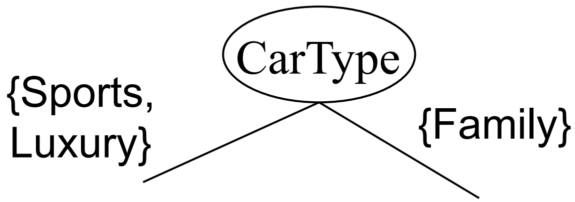
Splitting Nominal Attributes

- Nominal attributes provide enough information to **distinguish** one object from another (e.g., zip codes, ID, gender)

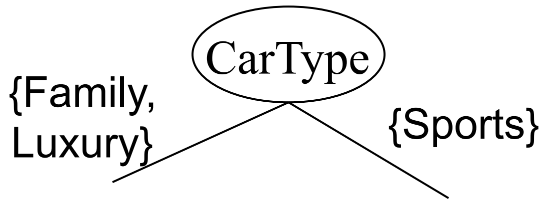
- Multi-way split: Use as many partitions as distinct values.



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

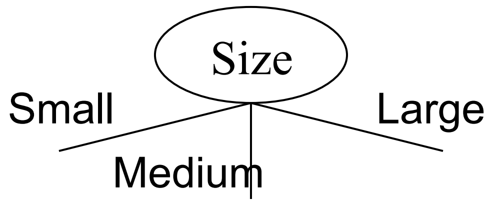


OR

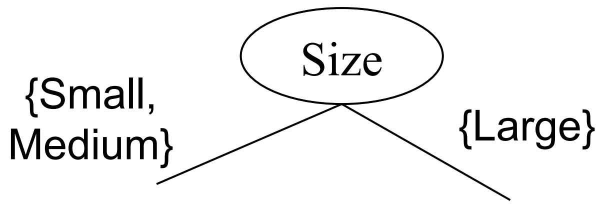


Splitting Ordinal Attributes

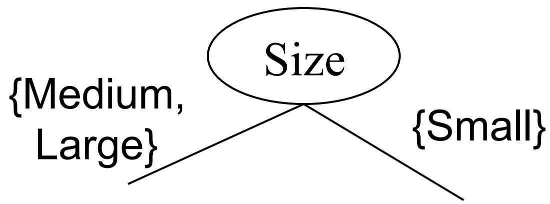
- The values of an ordinal attribute provide enough information to **order** objects (e.g., rankings, grades, height)
- Multi-way split: Use as many partitions as <u>distinct</u> values.



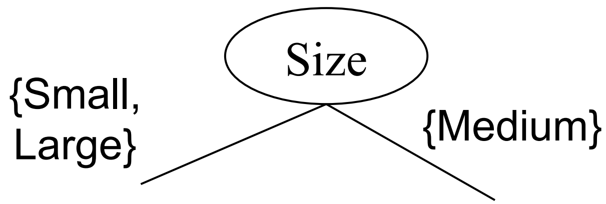
- Binary split: Divides values into two subsets - need to find optimal partitioning



OR



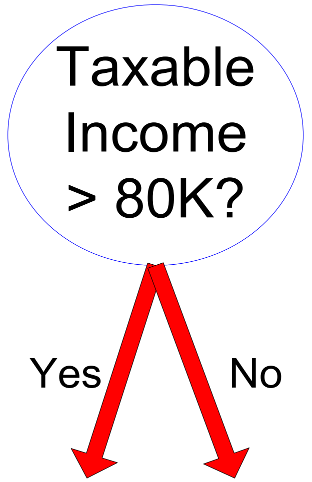
- What about this split?



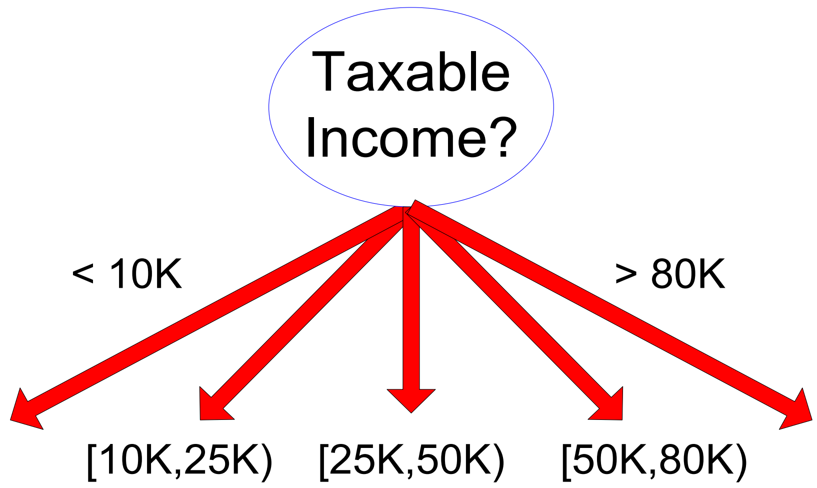
Splitting Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the **best** cut

Splitting Continuous Attributes



(i) Binary split



(ii) Multi-way split

Tree Induction

- Greedy strategy
  - Split the records based on an **attribute test** that <u>optimizes</u> certain criterion

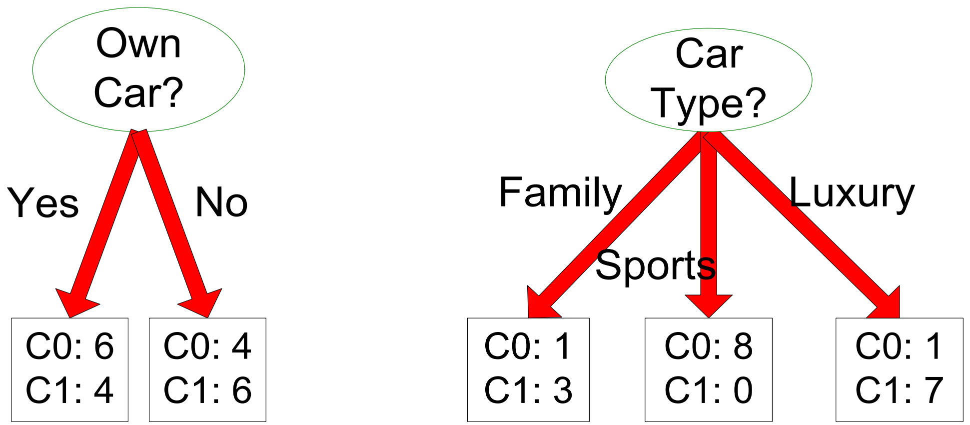
- Issues
  - Determine how to **split the records**
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to **stop splitting**

How to determine the Best Split

Before Splitting:

10 records of class **C0**, and
10 records of class **C1**

Assume **C0**: bad loan (default) and **C1**: good loan (pay)



Which test condition is the best?

How to determine the Best Split

- Greedy approach:
  - Prefers nodes with **homogeneous** class distribution
  - Need a measure of node **impurity**/heterogeneity:

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

Measures of Node Impurity

- Gini Index
- Misclassification error
- Entropy

Measure of Impurity: GINI

- Gini Index for a given node t :

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

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Minimum
  - when all records belong to one class
  - Impurity = 0.0
  - implying most interesting information
- Maximum
  - when records are equally distributed among all classes
  - Impurity = 1 - $1/n_c$ (n_c:number of classes)
  - implying least interesting information

Examples for computing GINI

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

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

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

$\text{Gini} = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$

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

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

$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$

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

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

$\text{Gini} = 1 - (2/6)^2 - (4/6)^2 = 0.444$

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,

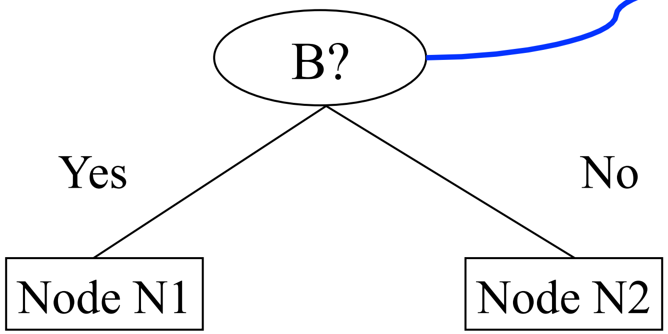
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

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

Goal: **Minimize** this weighted average impurity measure

Binary Attributes: Computing GINI

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



	Parent
C1	6
C2	6
Gini = 0.500	

Gini(N1)
$= 1 - (5/7)^2 - (2/7)^2$
$= 0.408$

Gini(N2)
$= 1 - (1/5)^2 - (4/5)^2$
$= 0.32$

	N1	N2
C1	5	1
C2	2	4
Gini=0.371		

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

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)

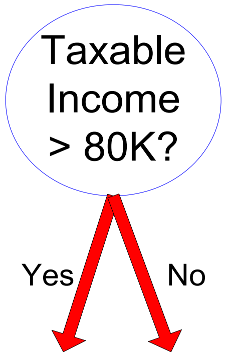
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	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

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...

- Sort the attribute on values
- Split positions are identified by taking midpoints between two adjacent values 55, 65, 72, …
- Scan these values, each time updating the count matrix and computing gini index
- Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
	Taxable Income									
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 Criteria based on Classification Error

- Classification error at a node t :

$$\boxed{Error(t) = 1 - \max_i P(i \mid t)}$$

- Measures misclassification error made by a node
  - Minimum
    - when <u>all records belong to one class</u>
    - Impurity = 0.0
    - implying most interesting information

Examples for Computing Error

$$Error(t) = 1 - \max_i P(i \mid t)$$

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

Tree Induction

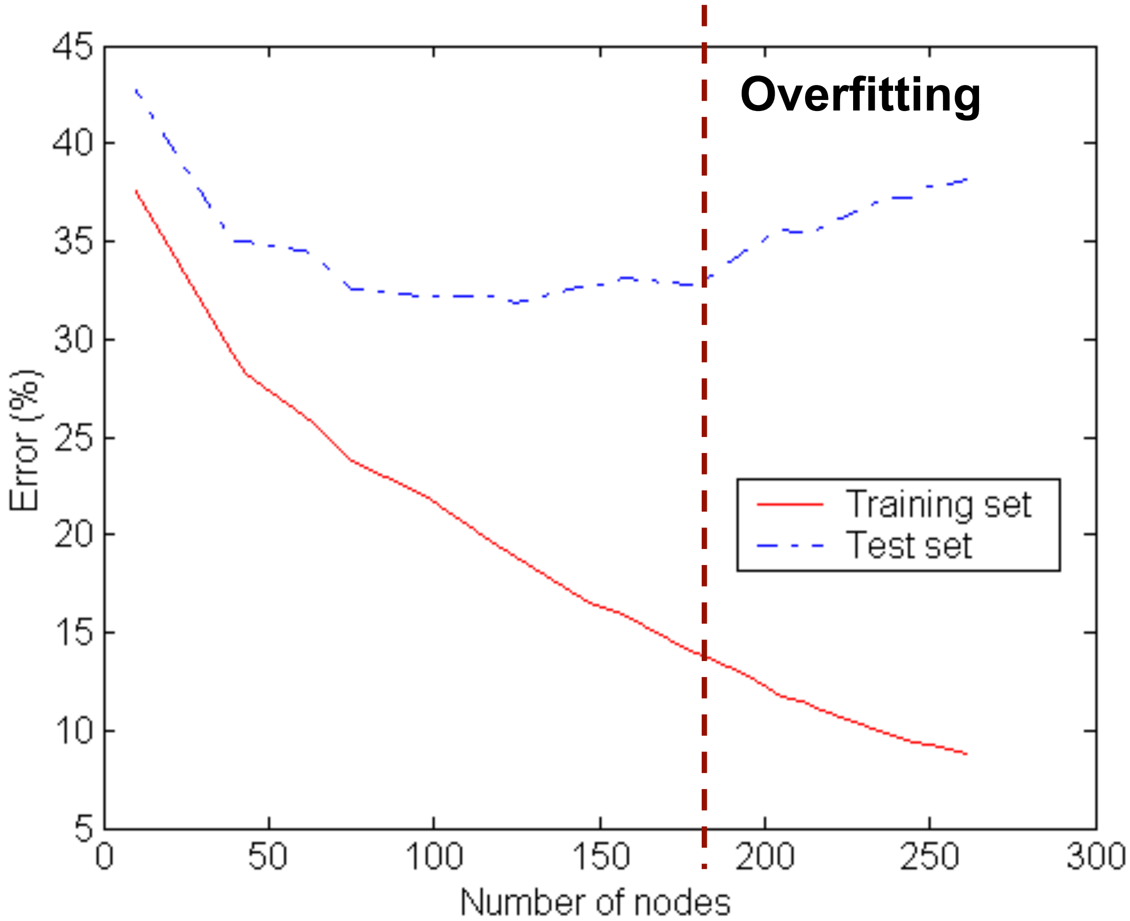
- Greedy strategy
  - Split the records based on an **attribute test** that <u>optimizes</u> certain criterion

- Issues
  - Determine how to **split the records**
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to **stop splitting**

Stopping Criteria

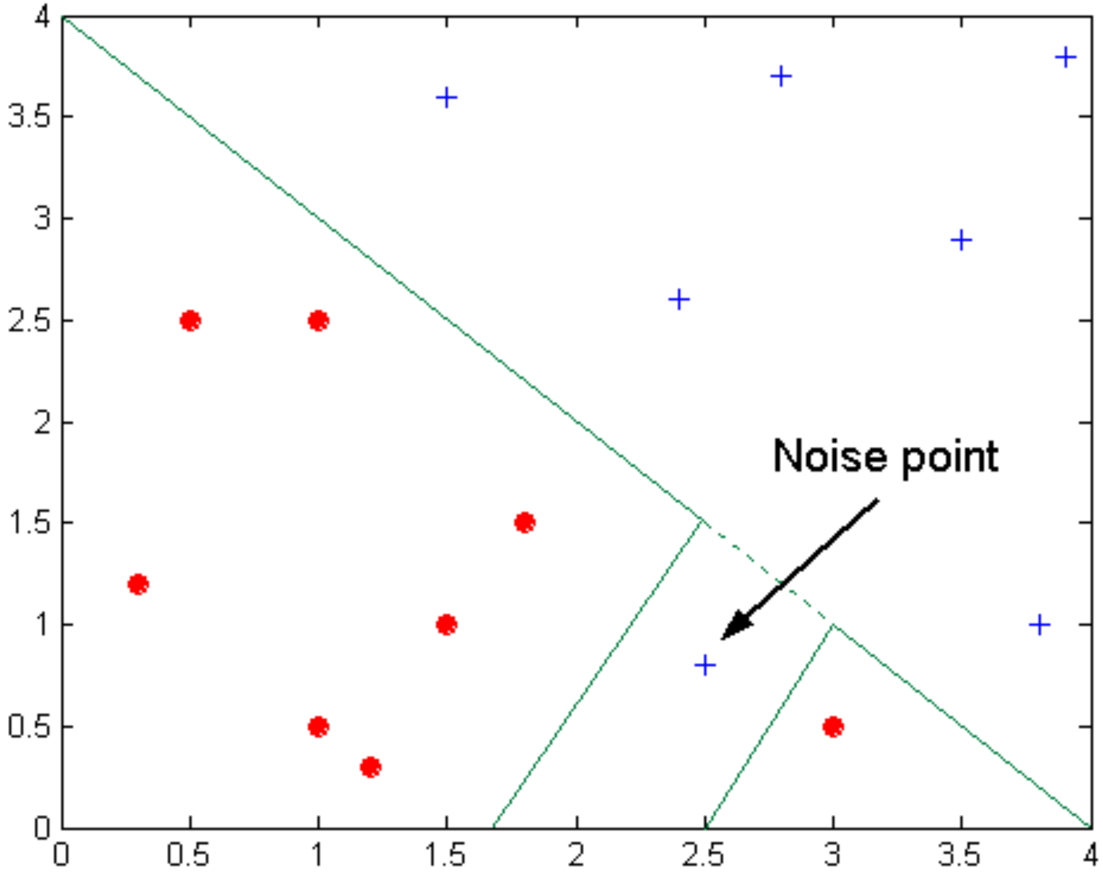
- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- But sometimes, **early termination!**
  - To avoid **overfitting**!

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

Notes on Overfitting

- An induced tree may overfit the training data
  - Too many branches (complex tree)
    - Some may reflect noise or outliers
  - Poor accuracy for unseen samples
    - Training error no longer provides a good estimate

How to Address Overfitting

- **Stop** the algorithm before creating fully-grown tree!
  - Typical stopping conditions for a node:
    1. Stop if all instances belong to the same class
    2. Stop if all the attribute values are the same
  - More restrictive conditions:
    1. Stop if number of instances is less than **some user-specified threshold**
    2. Stop if expanding the current node does not improve **impurity measures** (e.g., Gini).

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