Foundations of Data Science

DS 3001
Data Science Program
Department of Computer Science
Worcester Polytechnic Institute
Instructor: Prof. Kyumin Lee

Project Teams

1. Clay Oshiro-Leavitt, Hunter Caouette, Nick Alescio
2. Danielle Angelini, Elijah Ellis, Ryan Candy, Rob Wondolowski
3. Eva (Yingbing) Lu, Manasi Danke, Erica Lee, Jonathan Dang
4. Arianna Kan, Yihan Lin, Margaret Goodwin, Ken Snoddy
5. Yang Gao, Jose Li, Sarah Burns, Daniel McDonough
6. Noah Puchovsky, Katherine Handy, Alex Tavares, Angelica Puchovsky
7. Armando Zubillaga, Gabriel Rodrigues, Humberto Leon, Joao Omena de Lucena
8. Edward Carlson, Samuel Goldman, Nick Krichevsky, Christopher Myers
9. Jessie White, Lindsay MacInnis, Bao Huynh, Ziqian Zeng
10. Suverino Frith, Nicholas Odell, Fay Whittall, Johvanni Perez
11. Alp Piskin, Robert Scalfani, Jake Barefoot, Mark Bernardo
12. Amanda Chan, Nugzar Chkhaidze, Luke Gebler
13. Daniel Pelaez, Nathan Savard, Kate Sincaglia

Maan Alneami?

HW2

- Implement linear regression
  - https://canvas.wpi.edu/courses/18106/assignments/131989
  - Due date is April 17

Upcoming Schedule

- Exam 1 on April 17
  - I will go over brief review on Tuesday
  - Provide detailed information

- Project Proposal
  - https://canvas.wpi.edu/courses/18106/discussion_topics/101183
  - Due date: April 21

Mining and Analytics: Linear Regression

Some slides are adopted from Andrews Ng

Linear Regression (Big Picture)

Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost function:

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Gradient descent:

Repeat $\{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \ldots, \theta_n)$$

$\}$ (simultaneously update for every $j = 0, \ldots, n$)

Vectorization

- Benefits of vectorization
  - More compact equations
  - Faster code (using optimized matrix libraries)
- Consider our model:

$$h(\boldsymbol{x}) = \sum_{i=0}^{d} \theta_j x_j$$

- Let

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{x}^\mathsf{T} = \begin{bmatrix} 1 & x_1 & \ldots & x_d \end{bmatrix}$$

- Can write the model in vectorized form as $h(\boldsymbol{x}) = \boldsymbol{\theta}^\mathsf{T}\boldsymbol{x}$

Vectorization

- Consider our model for *n* instances:

$$h\left(\boldsymbol{x}^{(i)}\right) = \sum_{j=0}^{d} \theta_j x_j^{(i)}$$

- Let

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(i)} & \dots & x_d^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & \dots & x_d^{(n)} \end{bmatrix}$$

$$\mathbb{R}^{(d+1)\times 1} \qquad \mathbb{R}^{n\times(d+1)}$$

- Can write the model in vectorized form as $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{X}\boldsymbol{\theta}$

Vectorization

- For the linear regression cost function:

$$
\begin{aligned}
J(\boldsymbol{\theta}) &= \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 \\
&= \frac{1}{2n}\sum_{i=1}^{n}\left(\boldsymbol{\theta}^\intercal \boldsymbol{x}^{(i)} - y^{(i)}\right)^2 \\
&= \frac{1}{2n}\underbrace{(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^\intercal}_{\mathbb{R}^{1\times n}}\,(\underbrace{\overset{\mathbb{R}^{n\times(d+1)}}{\boldsymbol{X}}\overset{\mathbb{R}^{(d+1)\times 1}}{\boldsymbol{\theta}}}_{\mathbb{R}^{n\times 1}} - \underbrace{\boldsymbol{y}}_{\mathbb{R}^{n\times 1}})
\end{aligned}
$$

Let:

$$
\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}
$$

Mining and Analytics: Classification +

Decision Trees

Classification: Definition

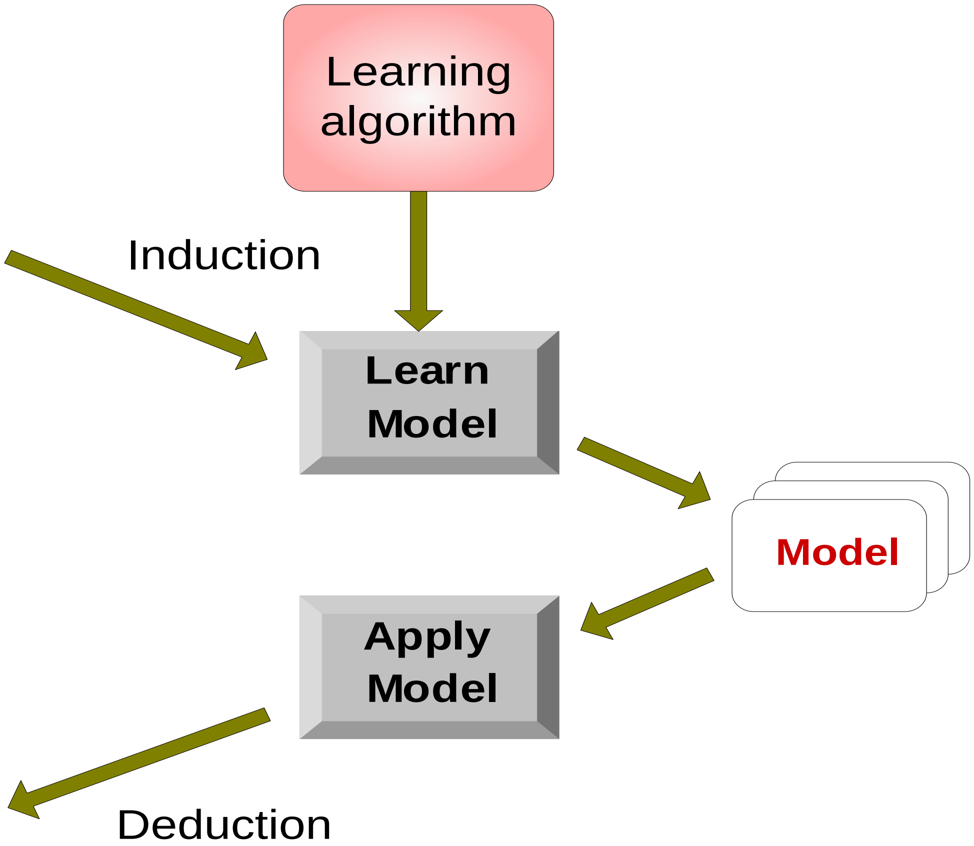
- Given a collection of records (training set)
  - Each record contains a set of attributes, one of the attributes is the class.
- Find a model for class attribute as a function of the values of other attributes.

- 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.

Illustrating 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

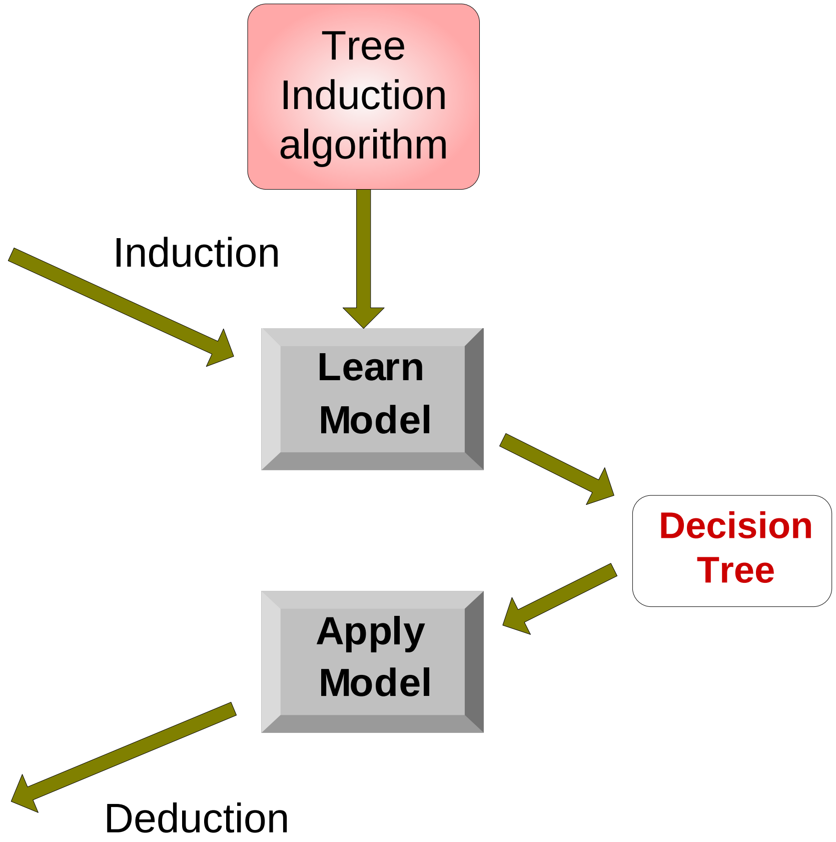
Classification Techniques

- **Decision trees ← today**
- Naive Bayes
- Nearest Neighbors (KNN)
- Support Vector Machines
- Neural Network
- …

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

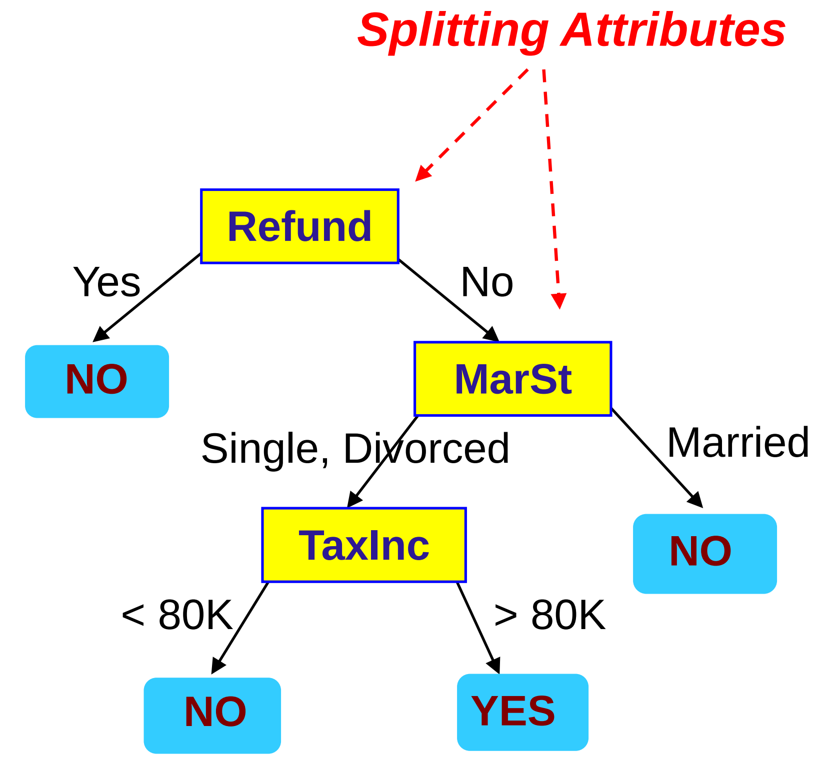
What is a Decision Tree?

- Hierarchical structure of **nodes** and **directed edges**
  - **Root node**: no incoming edges; zero or more outgoing edges
  - **Internal node**: one incoming edge; two or more outgoing edges
  - **Leaf node**: one incoming edge; no outgoing edges; **Labeled with a class**

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

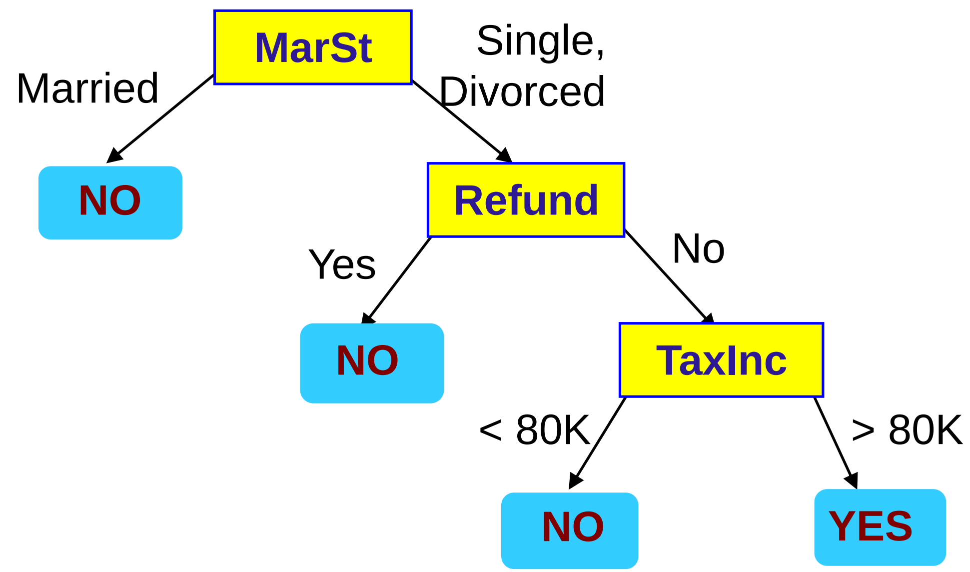


Model: Decision Tree

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

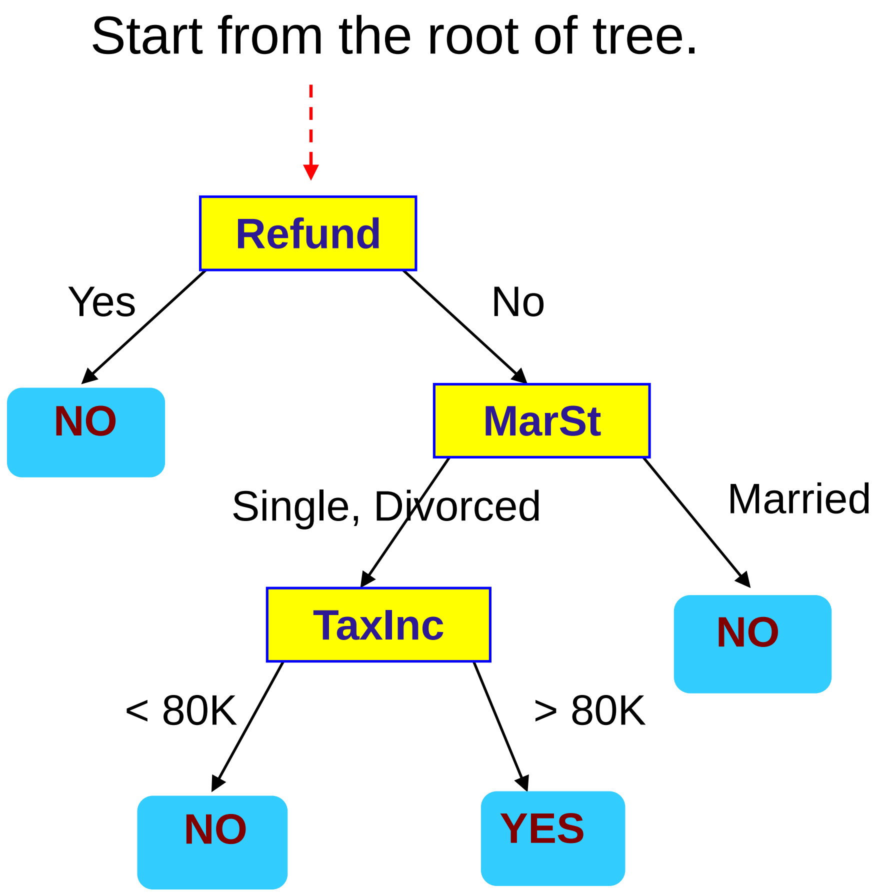


There could be more than one tree that fits the same data!

The Hope

- The decision tree (or whatever classifier we use) **generalizes** to new data!!
  - So we can have confidence in it

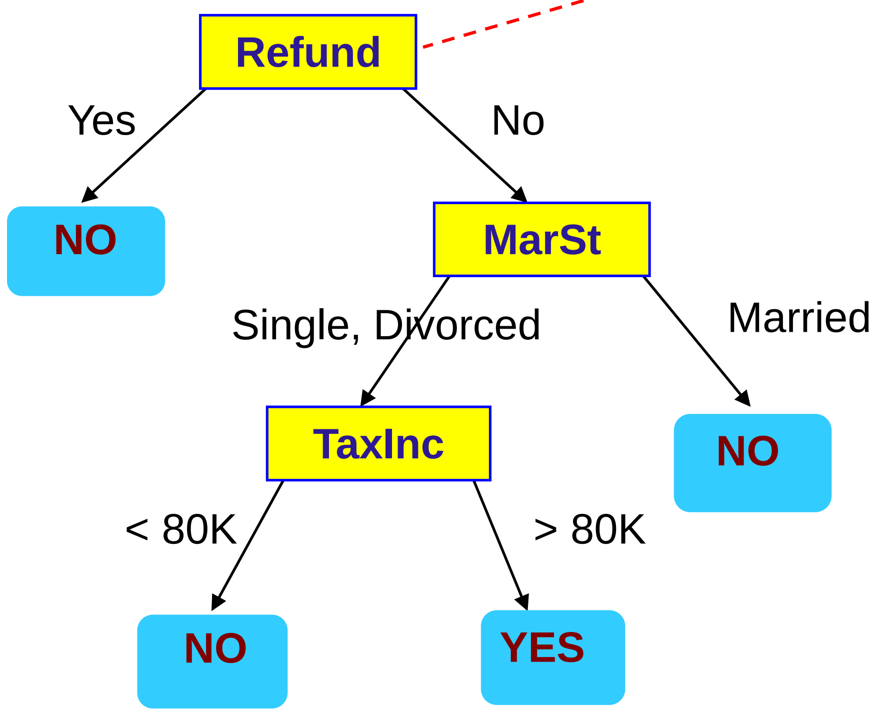
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



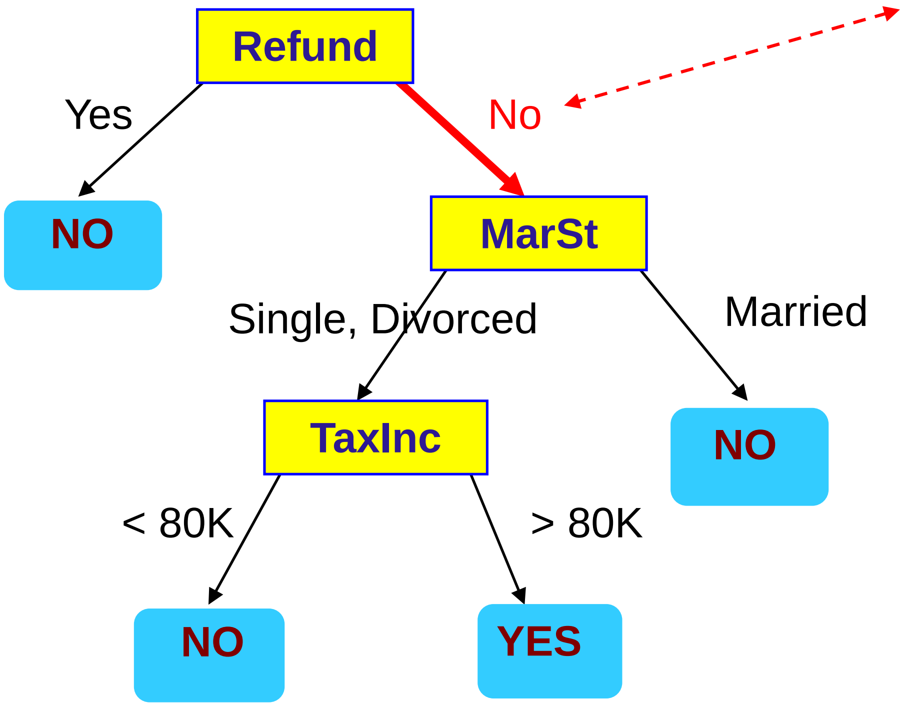
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

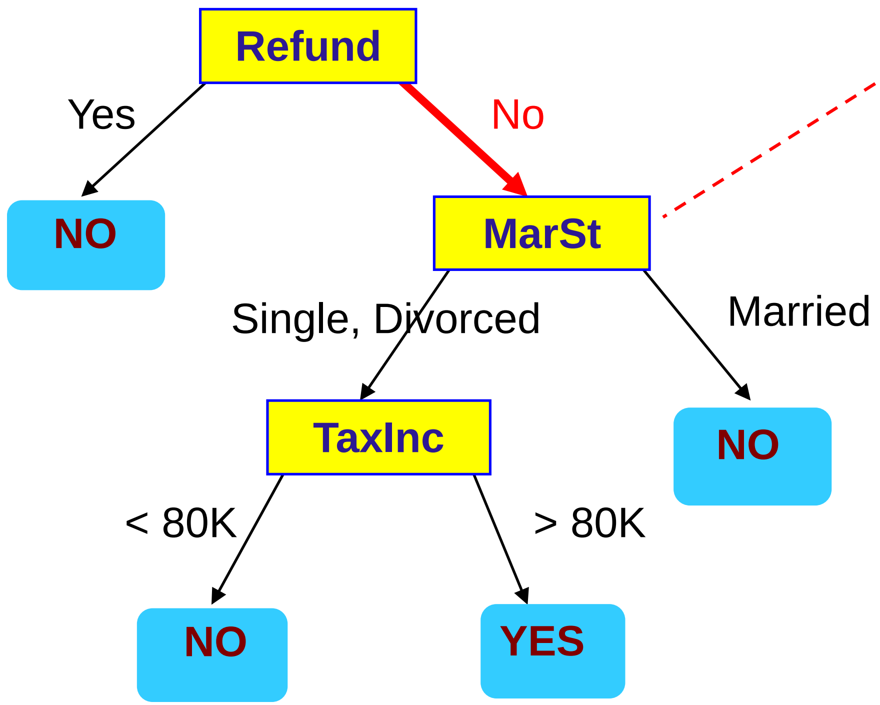
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



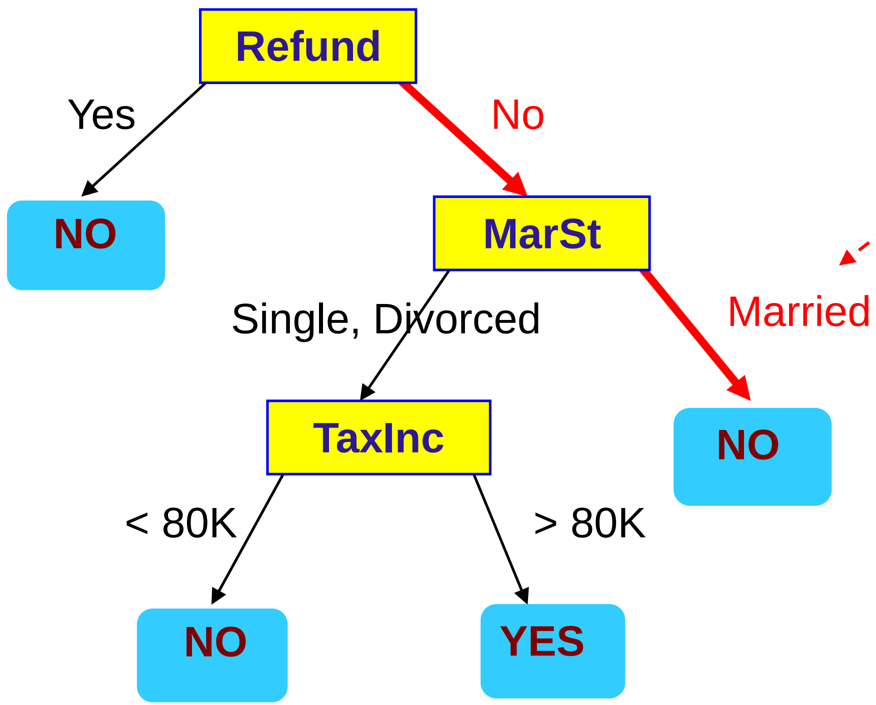
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data



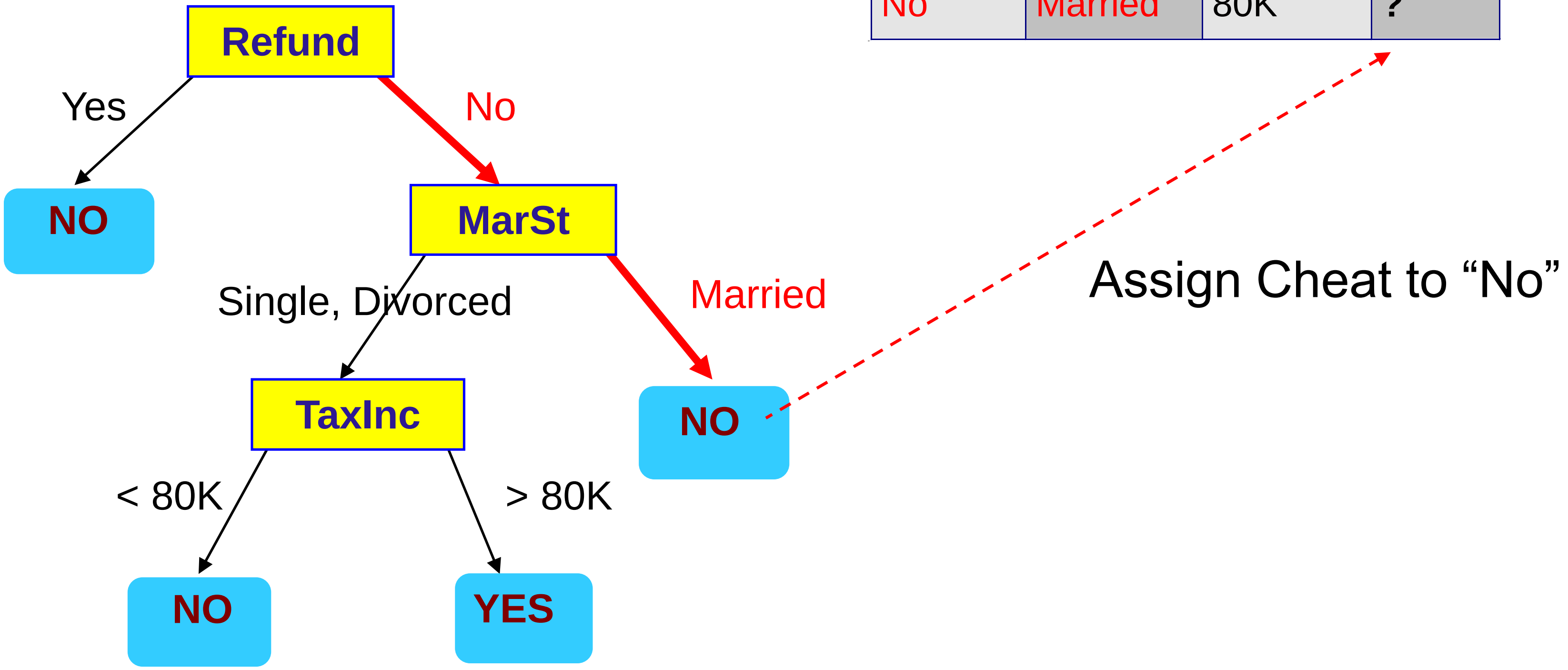
Test Data

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	?



Why Decision Trees?

- Popular!
- Relatively inexpensive to build
- Fast to classify new data
- **Easy to interpret**

But first, we must “Learn the model”

– (i.e., build the right decision tree)

Lots of approaches

- Hunt’s Algorithm
- CART
- ID3, C4.5
- SLIQ,SPRINT

- **Main ideas:**
  - Tree induction + tree pruning

General Structure of Hunt's Algorithm

- ***[Recursively apply]*** Let D_t be the set of training records (i.e., instances) that are associated with node t and $y = \{y_1, y_2, \dots y_c\}$ be the set of class labels
  - If D_t contains records that belong the same class y_t, then its decision tree consists of a leaf node labeled as y_t
  - If D_t is an empty set, then its decision tree is a leaf node whose class label is determined from other information such as the majority class of the records
  - If D_t contains records that belong to several classes, then a **test condition** based on one of the attributes of D_t is applied to split the data into more homogenous subsets

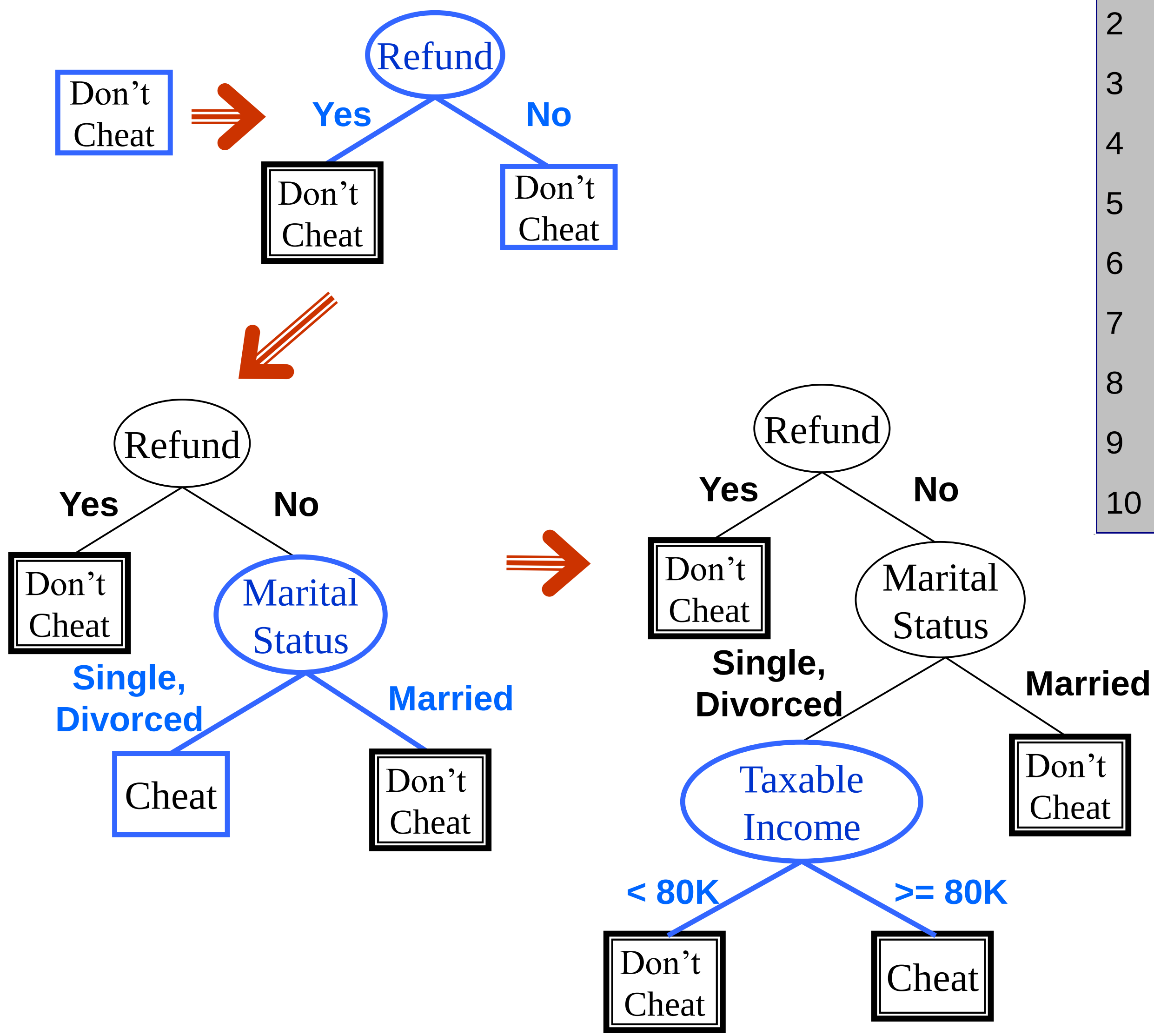
Example

- Attributes:
  - Refund (Yes, No)
  - Martial Status (Single, Divorced, Married)
  - Taxable Income (quantitative)

- Class:
  - Cheat, Don’t Cheat

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

- Determine how to split the records
  - Use greedy heuristics to make a series of locally optimum decision about which attribute to use for partitioning the data
  - At each step of the greedy algorithm, a test condition is applied to split the data in to subsets with a more homogenous class distribution
    - How to specify test condition for each attribute
    - How to determine the best split

- Determine when to stop splitting
  - A stopping condition is needed to terminate tree growing process. Stop expanding a node
    - if all the instances belong to the same class
    - if all the instances have similar attribute values

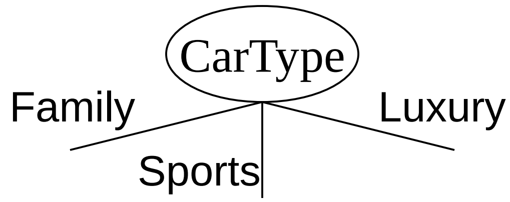
Splitting?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

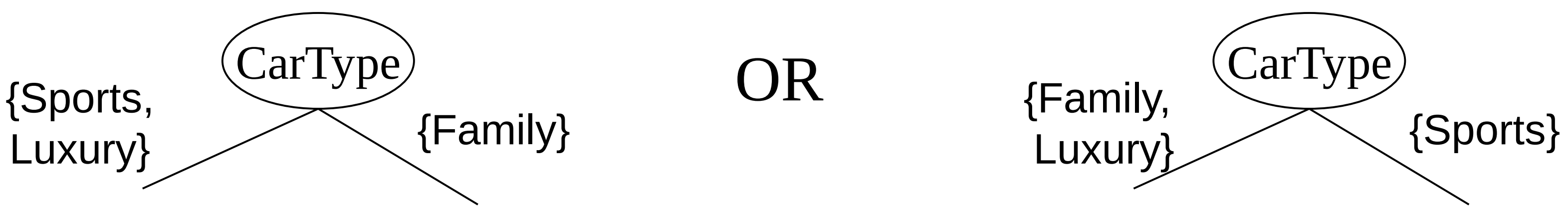
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

For Nominal Attributes

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

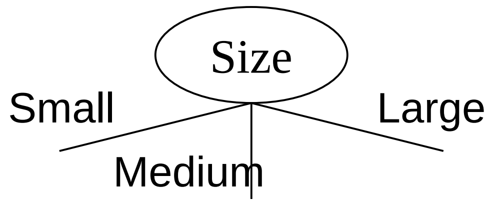


- Binary split: Divides values into two subsets. Need to find optimal partitioning.

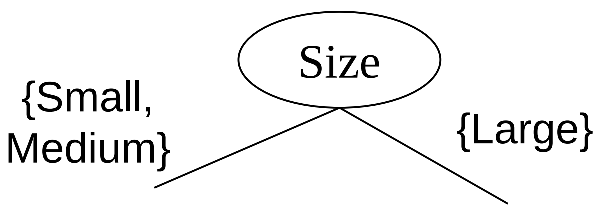


For Ordinal Attributes

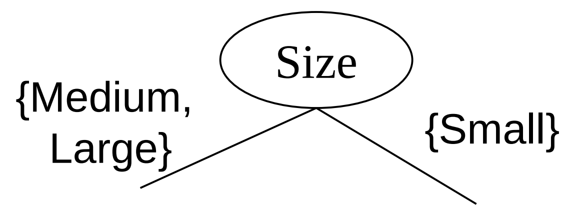
- Multi-way split: Use as many partitions as distinct values.



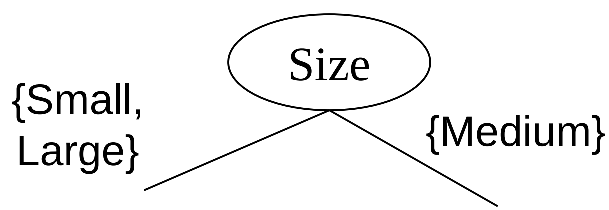
- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.



OR

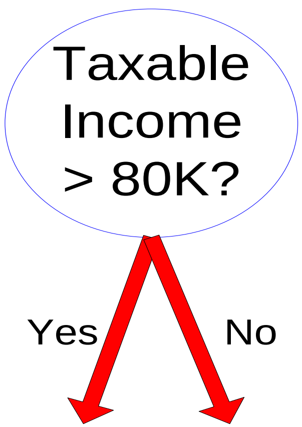


- What about this split?

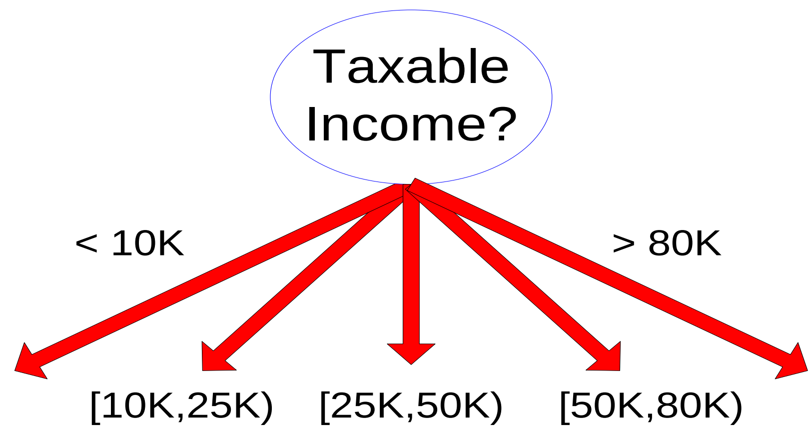


For Quantitative

- 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



(i) Binary split

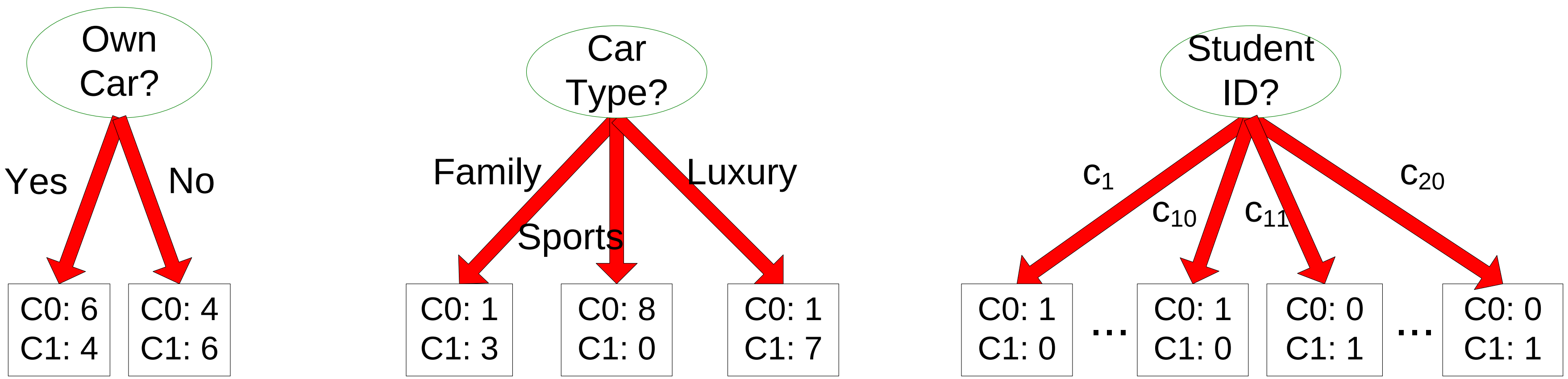


(ii) Multi-way split

But How do we split?

- Splitting Criterion
  - Given the data associated with a particular node in the tree, what **test condition** do we apply?

Before Splitting: 10 records of class 0,
10 records of class 1



Which test condition is the best?

Splitting Criterion

- Ideas?
- Intuition: Prefer nodes with *homogeneous* class distribution

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

- Typical methods (i.e., measuring impurity)
  - Gini Index
  - Entropy / Information Gain
  - Classification error

Splitting Criterion: GINI

- Gini Index for a given node t :

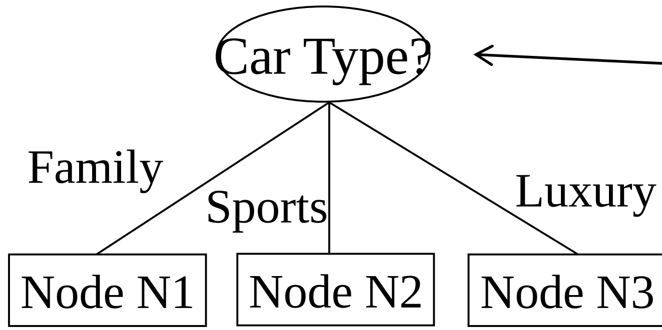
$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

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

- Measure the impurity of a node
  - Maximum $(1 - 1/n_c)$ when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

GINI Example

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



	Parent
C1	**6**
C2	**6**

Node N1

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

Node N1

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

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

Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278

Node N1

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

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

Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444

Splitting Based on GINI

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

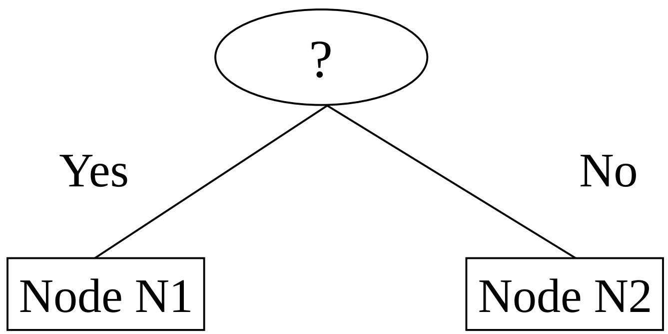
$$\boxed{GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)}$$

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

Also called collective impurity of child nodes

GINI for Binary Attributes

- 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)
$= 1 - (5/7)^2 - (2/7)^2$
$= 0.409$

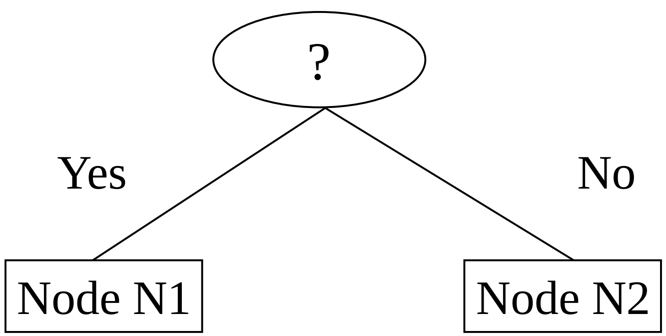
Gini(N2)
$= 1 - (1/5)^2 - (4/5)^2$
$= 0.32$

	N1	N2
C1	**5**	**1**
C2	**2**	**4**
Gini=0.371		

Gini(Children)
$= 7/12 * 0.409 +$
$\quad 5/12 * 0.32$
$= 0.371$

GINI for Binary Attributes

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



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

Attribute A

	N1	N2
C1	**0**	**6**
C2	**6**	**0**
Gini=0.000		

Attribute B

	N1	N2
C1	**5**	**1**
C2	**1**	**5**
Gini=0.278		

Attribute C

	N1	N2
C1	**4**	**2**
C2	**3**	**3**
Gini=0.486		

Attribute D

	N1	N2
C1	**3**	**3**
C2	**3**	**3**
Gini=0.500		