Classification: Basic Concepts, Decision Trees

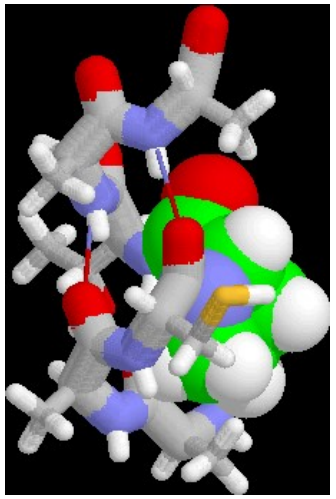
Week 9

Classification Learning: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class label or category*
- Find a *model* for the class attribute as a function of the values of the other attributes
  - So that based on values of other attributes we can determine which class this record belongs to
- Goal: previously unseen records should be assigned a class as accurately as possible
  - Use *test set* to estimate the accuracy of the model

Examples of Classification Task

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc.
- Determining high-risk vs. low-risk patients for addmission to ICU



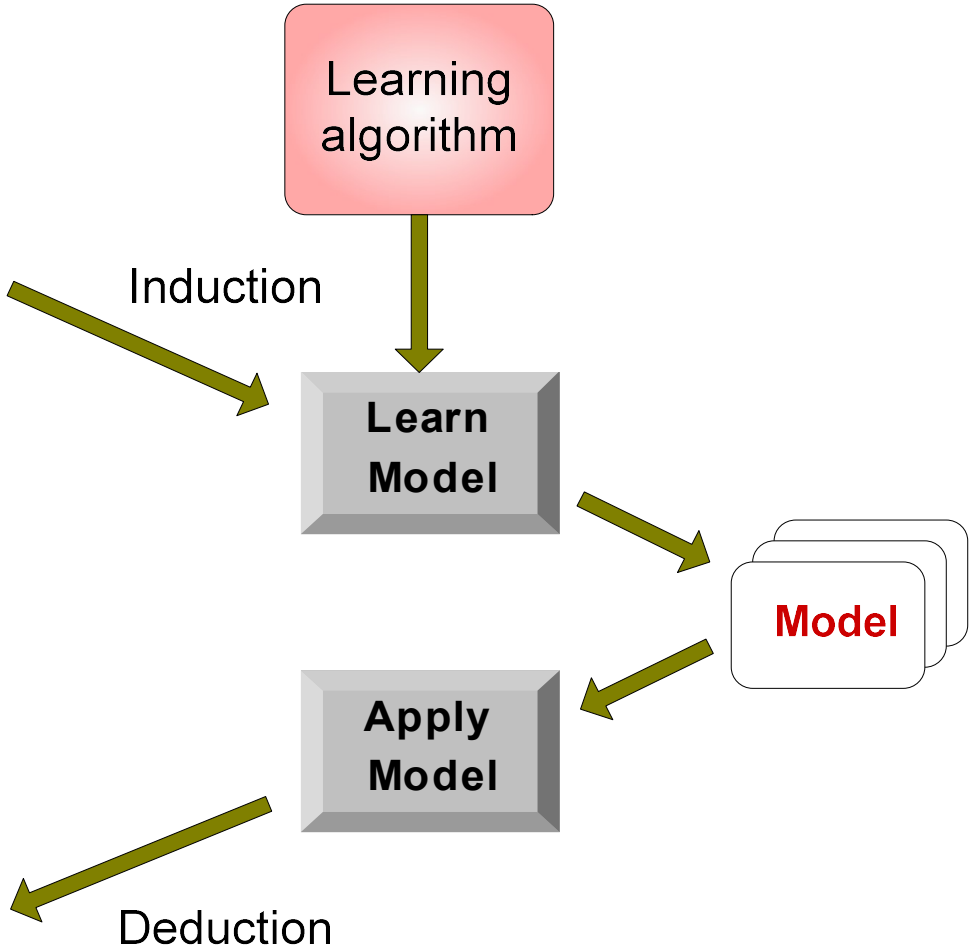
Illustrating Classification Learning

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 Learning Techniques

- Decision tree-based methods
- Rule-based methods
- Instance-based methods
- Probability-based methods
- Neural networks
- Support vector machines
- Logic-based methods

When to Use Decision Tree

- Instances describable by attribute-value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data
- Missing attribute values
- Examples:
  - Medical diagnosis
  - Credit risk analysis
  - Object classification for robot manipulator (Tan 1993)

Decision Trees

- A *decision tree* T encodes d (a classifier or regression function) in form of a tree.
- A node t in T without children is called a *leaf node*.
- Otherwise t is called an *internal node*.

Why Decision Trees

- Relatively fast compared to other classification models
- Obtain similar and sometimes better accuracy compared to other models
- Simple and easy to understand
- Can be converted into simple and easy to understand classification rules
  - Conjunctions of Disjunctions

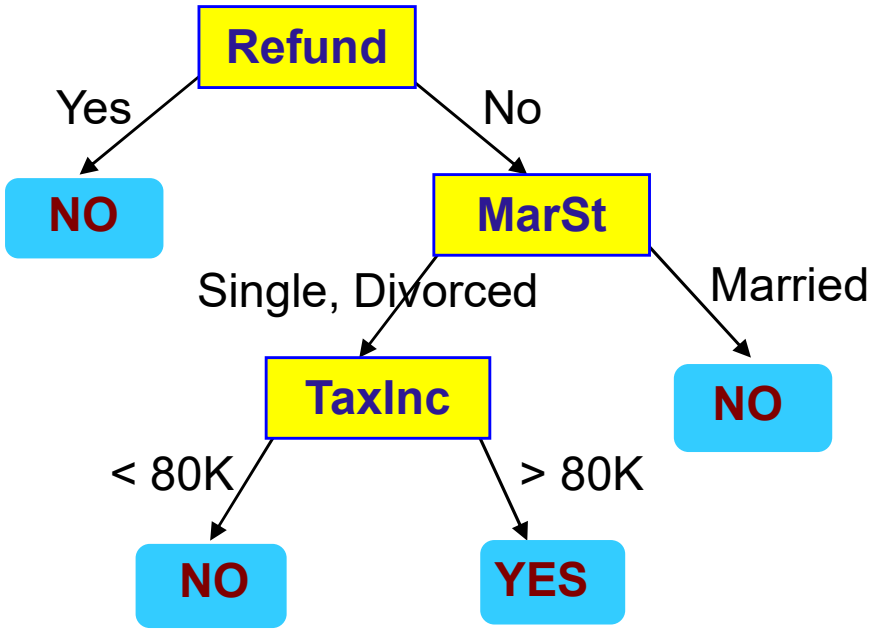
Building a Decision Tree

- A decision tree is created in two phases:
  - Tree Building Phase
    - Repeatedly partition the training data until all the examples in each partition belong to one class or the partition is sufficiently small
  - Tree Pruning Phase
    - Remove dependency on statistical noise or variation that may be particular only to the training set

Example of a Decision Tree

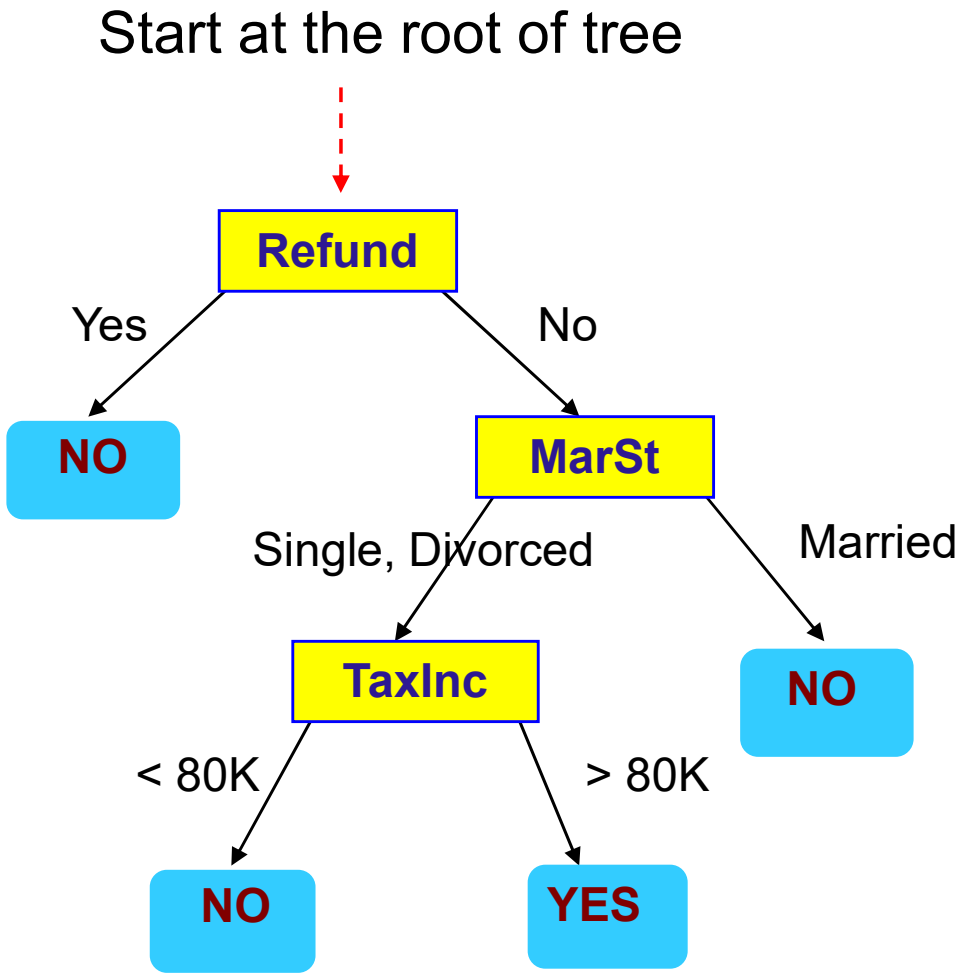
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

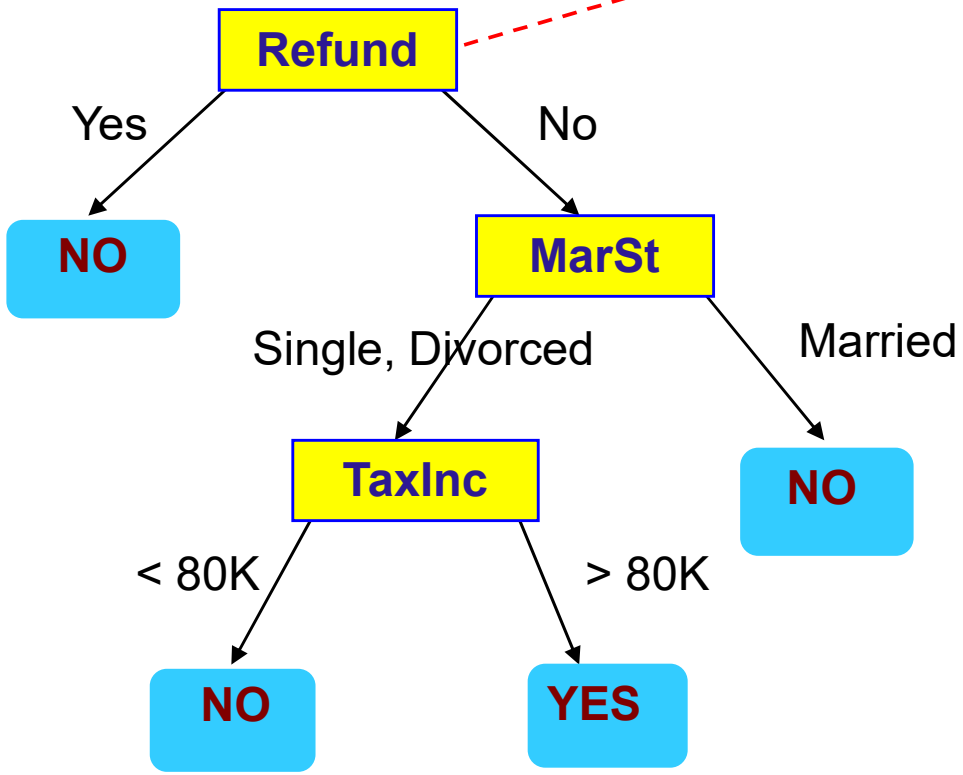
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

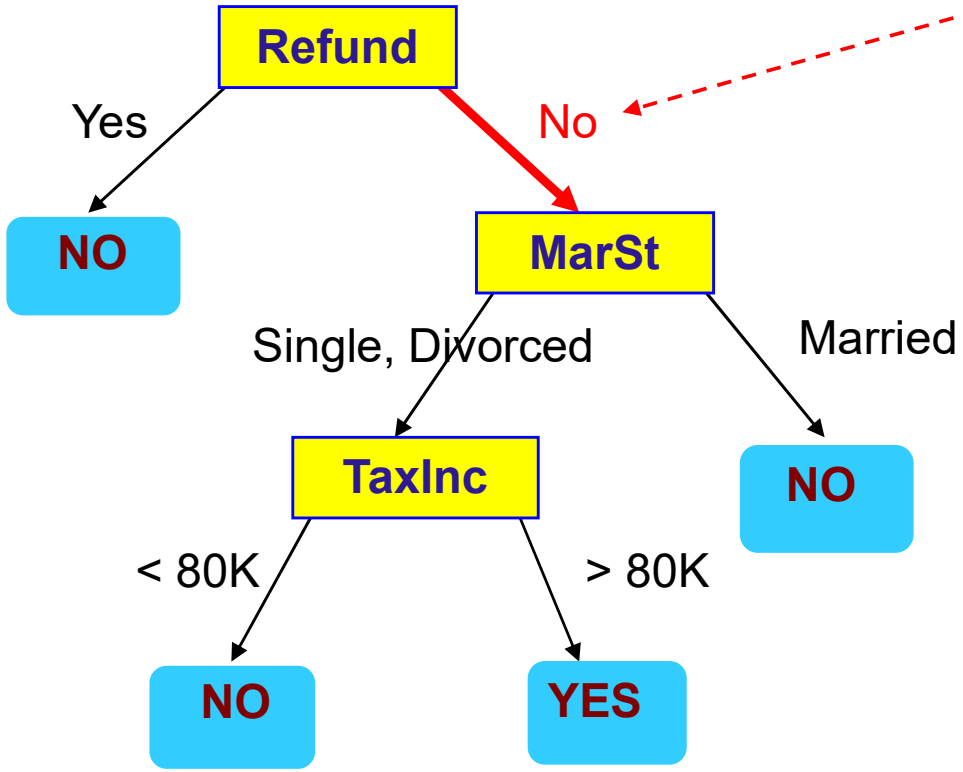
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

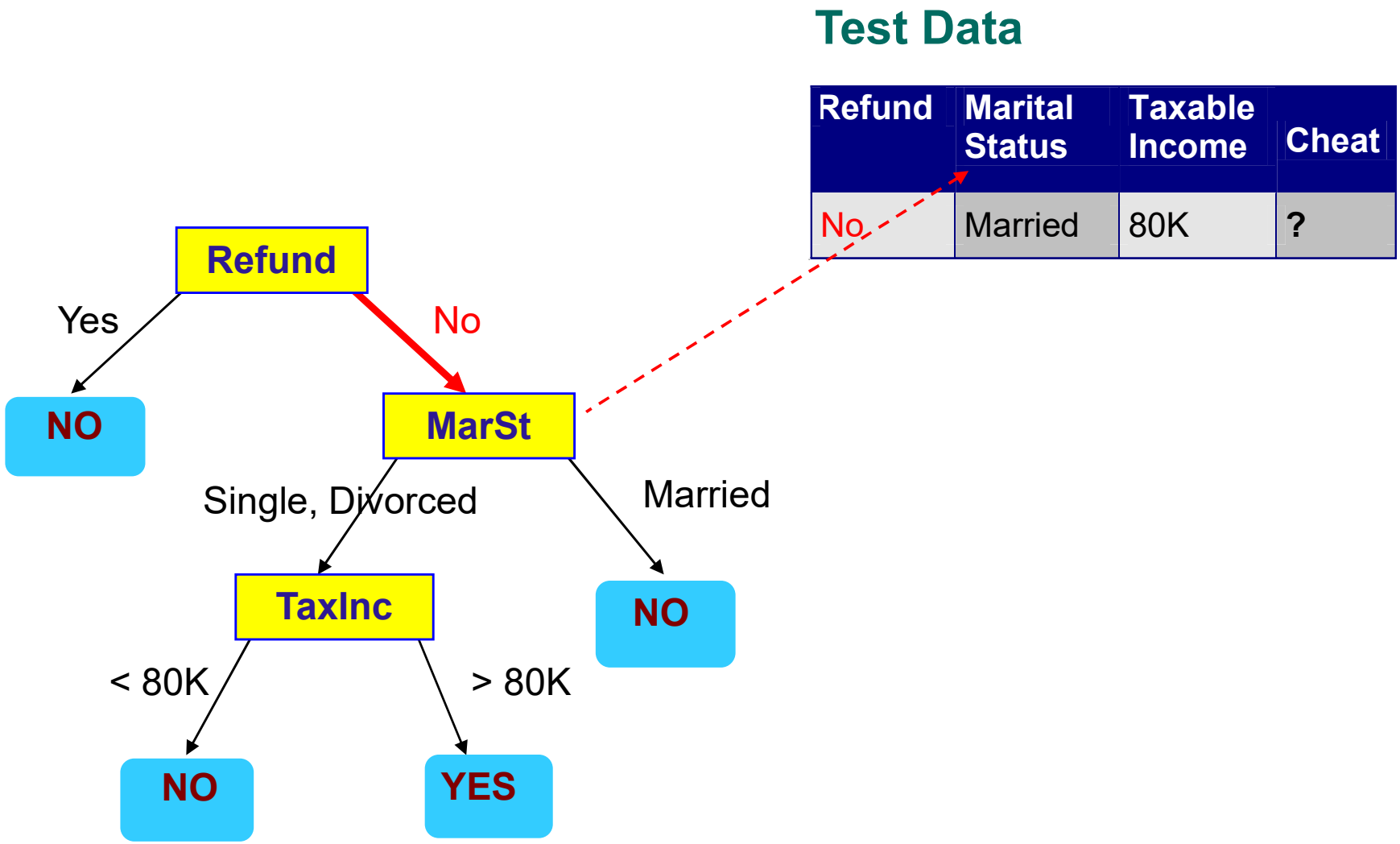
Apply Model to Test Data



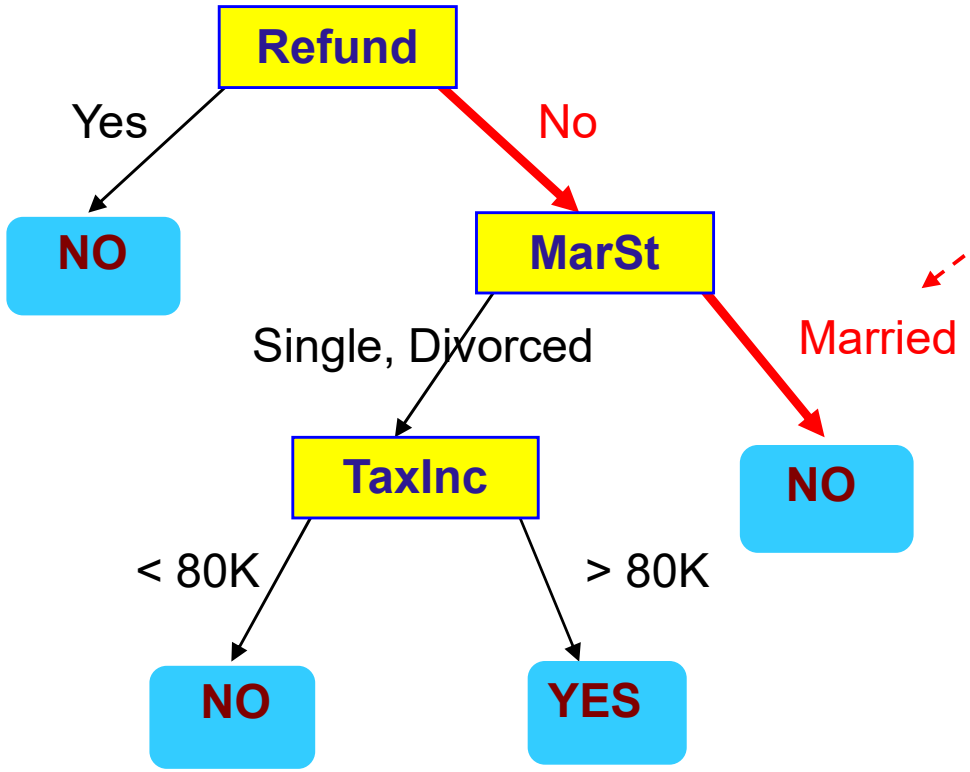
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



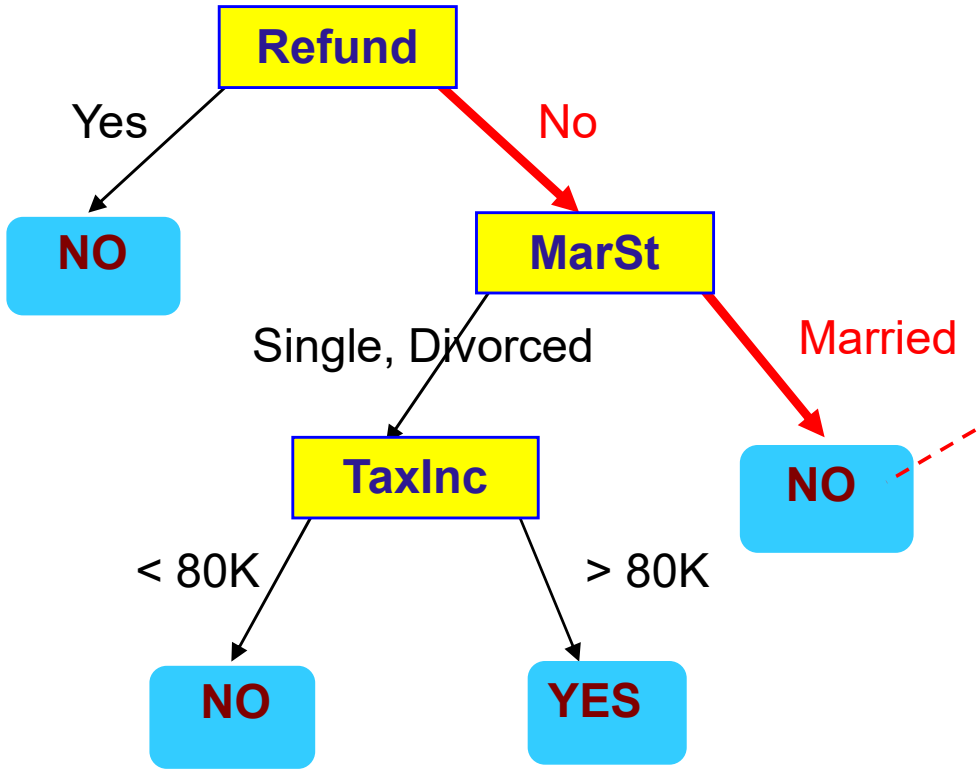
Apply Model to Test Data



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	?

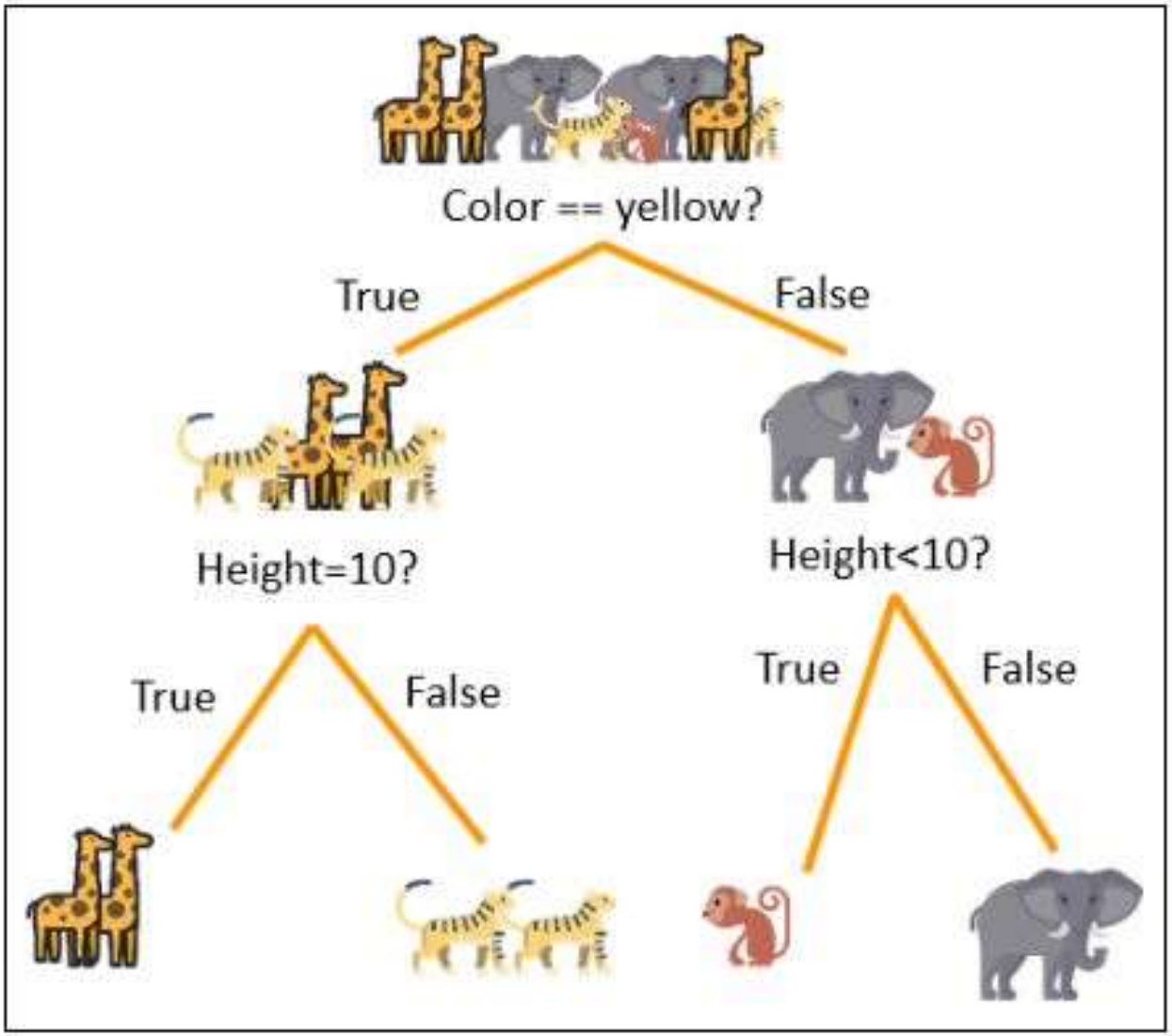
Assign Cheat to “No”

Decision Tree Induction Techniques

- Decision tree induction is a top-down, recursive and divide-and-conquer approach.
- The procedure is to choose an attribute and split it into from a larger training set into smaller training sets.
- Different algorithms have been proposed to take a good control over
  1. Choosing the best attribute to be split, and
  2. Splitting criteria
- Several algorithms have been proposed for the above tasks
  - **ID3**
  - **C 4.5**
  - **CART**

Decision Tree Induction - Intuition

Decision Tree Induction –intuition



Building A Decision Tree

- General tree-growth algorithm (binary tree)

Partition(Data S)

If (all points in S are of the same class) then return;

for each attribute A do

    evaluate splits on attribute A;

Use best split to partition S into S1 and S2;

Partition(S1);

Partition(S2);

Building a DT: What is the best Split/Partition

- Random
- Principled Criteria
  - Entropy
  - Information Gain
  - GINI

What is ID3?

- A mathematical algorithm for building the decision tree.
- Invented by J. Ross Quinlan in 1979.
- Uses Information Theory invented by Shannon in 1948.
- Builds the tree from the top down, with no backtracking.
- Quinlan [1986] introduced the ID3, a popular short form of **I**terative **D**ichotomizer 3 for decision trees from a set of training data.
- In ID3, each node corresponds to a splitting attribute and each arc is a possible value of that attribute.
- At each node, the splitting attribute is selected to be the most informative among the attributes not yet considered in the path starting from the root.
  - Information Gain is used to select the most useful attribute for classification.

Entropy

- A formula to calculate the homogeneity of a sample.
- A completely homogeneous sample has entropy of 0.
- An equally divided sample has entropy of 1.

- **S is a sample of training examples**
- **p_+ is the proportion of positive examples**
- **p_- is the proportion of negative examples**
- **Entropy measures the impurity of S**

$$\textbf{Entropy}(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

Entropy

Suppose S has 25 examples, 15 positive and 10 negatives [15+, 10-]. Then the entropy of S relative to this classification is

$$E(S)=-(15/25)\ \log2(15/25) - (10/25)\ \log2\ (10/25)$$

Information Gain (IG)

- The information gain is based on the decrease in entropy after a dataset is split on an attribute.
- Which attribute creates the most homogeneous branches?
- First the entropy of the total dataset is calculated.
- The dataset is then split on the different attributes.
- The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split.
- The resulting entropy is subtracted from the entropy before the split.
- The result is the Information Gain, or decrease in entropy.
- The attribute that yields the largest IG is chosen for the decision node.

GINI Index

- Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.
- It works with categorical target variable “Success” or “Failure”.
- It performs only Binary splits
- Higher the value of Gini higher the homogeneity.
- CART (Classification and Regression Tree) uses Gini method to create binary splits.

GINI Formula

- Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

- Calculate Gini for split using weighted Gini score of each node of that split

GINI Example

Gini Index For Outlook

GINI(Sunny) $= 1 - P(play)^2 - P(stay)^2$

$= 1 - (4/5)^2 - (1/5)^2$

$=$ **0.32**

GINI(Rain) $= 1 - (1/5)^2 - (4/5)^2$

$=$ **0.32**

GINI(Over..) $= 1 - (4/4)^2 - (0/4)^2$

$=$ **0**

GINI-Split(O'look) $= 0.320 * (5/14) + 0.32 * (5/14) + 0 * (4/14) =$ **0.229**

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	Strong	Hot	Sunny	High	Play
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	**Weak**	**Mid**	**Overcast**	**High**	**Play**
Day 5	Strong	Cold	Rain	Normal	Stay
Day 6	**Weak**	**Cold**	**Overcast**	**Normal**	**Play**
Day 7	Strong	Cold	Rain	Normal	Stay
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	Weak	Cold	Sunny	Normal	Play
Day 10	**Strong**	**Mid**	**Overcast**	**Normal**	**Play**
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	Strong	Mid	Rain	High	Stay
Day 13	**Weak**	**Hot**	**Overcast**	**Normal**	**Play**
Day 14	Weak	Cold	Rain	High	Play

Algorithm ID3

- In ID3, entropy is used to measure how informative a node is.
  - It is observed that splitting on any attribute has the property that average entropy of the resulting training subsets will be less than or equal to that of the previous training set.
- ID3 algorithm defines a measurement of a splitting called **Information Gain** to determine the goodness of a split.
  - The attribute with the largest value of information gain is chosen as the splitting attribute and
  - it partitions into a number of smaller training sets based on the distinct values of attribute under split.

Information Gain (cont'd)

- A branch set with entropy of 0 is a leaf node.
- Otherwise, the branch needs further splitting to classify its dataset.
- The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

Baseball Data

Where	When	Fred Starts	Joe offense	Joe defense	Opp C	*OutCome*
Home	7pm	Yes	Center	Forward	Tall	*Won*
Home	7pm	Yes	Forward	Center	Short	*Won*
Away	7pm	Yes	Forward	Forward	Tall	*Won*
Home	5pm	No	Forward	Center	Tall	*Lost*
Away	9pm	Yes	Forward	Forward	Short	*Lost*
Away	7pm	No	Center	Forward	Tall	*Won*
Home	7pm	No	Forward	Center	Tall	*Lost*
Home	7pm	Yes	Center	Center	Talls	*Won*
Away	7pm	Yes	Center	Center	Short	*Won*
Home	9pm	No	Forward	Center	Short	*Lost*

●
●
●

Where	When	Fred Starts	Joe offense	Joe defense	Opp C	*Outcome*
Away	9pm	No	Center	Forward	Tall	??

Solution

Entropy (Outcome) = 0.6*0.73 + 0.4*1.32 = 0.966

For the attribute where

Where			
	+	-	
Home	3	3	6
Away	3	1	4

Entropy(Outcome, Where=Home) = -3/6log2(3/6)- 3/6(log2(3/6) = 1

Entropy(Outcome, Where=Away) = -3/4LOG2(3/4)- 1/4log2(1/4) = 0.75*0.415+0.25*2 = 0.81125

Entropy(outcome, where) = (6/10)*1 +(4/10)*0.81125 = 0.6+0.3245 = 0.9245

I.G. = 0.966 – 0.925 = 0.041

Solution

When			
	+	-	
5pm	0	1	1
7pm	6	1	7
9pm	0	2	2

Entropy(Outcome, When=5p.m) = -0/1(log2)(0/1) – 1/1log2(1/2) = 0

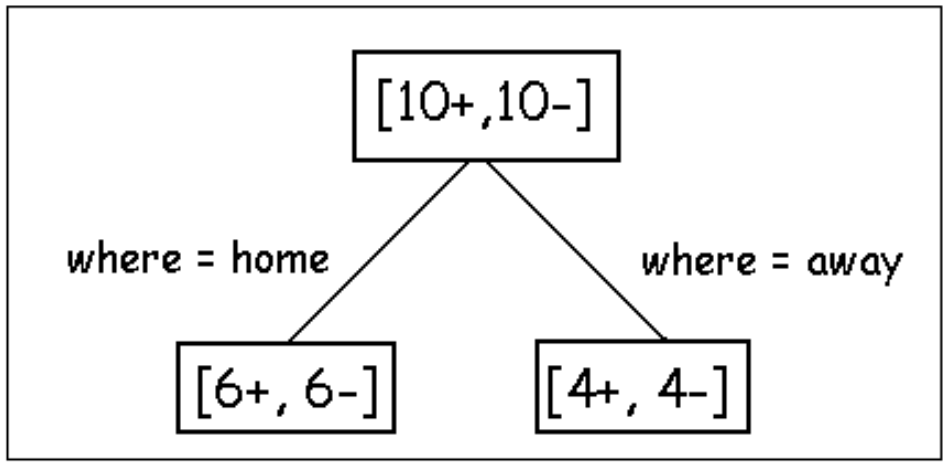
Entropy(Outcome, When=7p.m) = -6/7log2(6/7) – 1/7(log2)(1/7) =0.857*0.223 + 0.142*2.816 = 0.191 + 0.399 = 0.591

Entropy(Outcome, When = 9p.m.) = 0

Entropy(outcome, when) = 0.1*0 + 0.7*0.591 + 0.2*0 = 0.4137

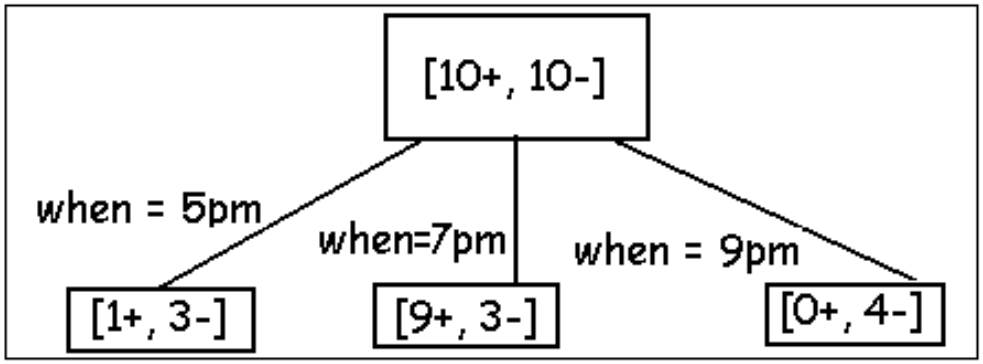
I.G. = 0.966 – 0.4137 = 0.5523

Partitioning on where



- Before partitioning, the entropy is
  - *H(10/20, 10/20) = - 10/20 log(10/20) - 10/20 log(10/20) = 1*
- Using the ``where'' attribute, divide into 2 subsets
  - Entropy of the first set H(home) = - 6/12 log(6/12) - 6/12 log(6/12) = 1
  - Entropy of the second set H(away) = - 4/8 log(6/8) - 4/8 log(4/8) = 1
- Expected entropy after partitioning
  - 12/20 * H(home) + 8/20 * H(away) = 1

Partitioning on when



- Using the ``when'' attribute, divide into 3 subsets
  - Entropy of the first set H(5pm) = - 1/4 log(1/4) - 3/4 log(3/4);
  - Entropy of the second set H(7pm) = - 9/12 log(9/12) - 3/12 log(3/12);
  - Entropy of the second set H(9pm) = - 0/4 log(0/4) - 4/4 log(4/4) = 0
- Expected entropy after partitioning
  - 4/20 * H(1/4, 3/4) + 12/20 * H(9/12, 3/12) + 4/20 * H(0/4, 4/4) = 0.65
- Information gain 1-0.65 = 0.35

Decision

- Knowing the ``when'' attribute values provides larger information gain than ``where''.
- Therefore the ``when'' attribute should be chosen for testing prior to the ``where'' attribute.
- Similarly, we can compute the information gain for other attributes.
- At each node, choose the attribute with the largest information gain.

Decision Tree Learning: ID3

- Function ID3(*Training-set*, *Attributes*)
  - If all elements in *Training-set* are in same class, then return leaf node labeled with that class
  - Else if *Attributes* is empty, then return leaf node labeled with majority class in *Training-set*
  - Else if *Training-Set* is empty, then return leaf node labeled with default majority class
  - Else
    - Select and remove *A* from *Attributes*
    - Make *A* the root of the current tree
    - For each value *V* of *A*
      - Create a branch of the current tree labeled by *V*
      - *Partition_V* ← Elements of *Training-set* with value *V* for *A*
      - Induce-Tree(*Partition_V*, *Attributes*)
      - Attach result to branch *V*

Illustrative Training Set

Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

Solution

- ENTROPY(RISK) = -(9/14)log2(9/14) – (5/14)log2(5/14)

Solution – taking Credit History for split

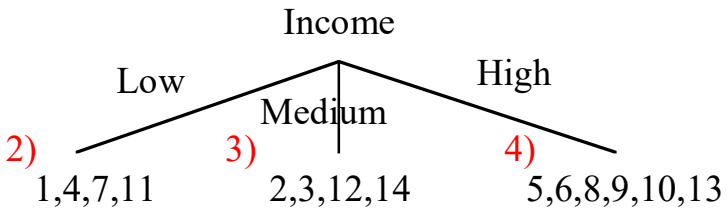
Credit History			
	+	-	
Good	2	3	5
Bad	4	0	4
Unknown	3	2	5

Entropy(outcome, credit-history=good) = - 2/5 log2(2/5) – 3/5 log2(3/5)
Entropy(outcome, credit-history = bad) = 0
Entropy(outcome, credit-history = unknown) = -3/5 log2(3/5) – 2/5 log2(2/5)

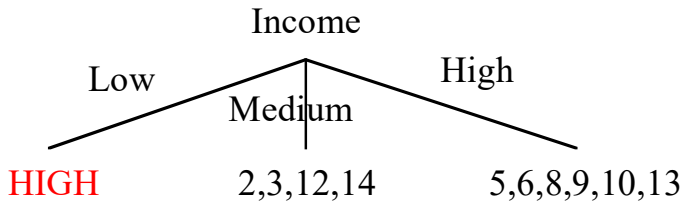
Entropy (come, credit-history) = 0.677

ID3 Example (I)

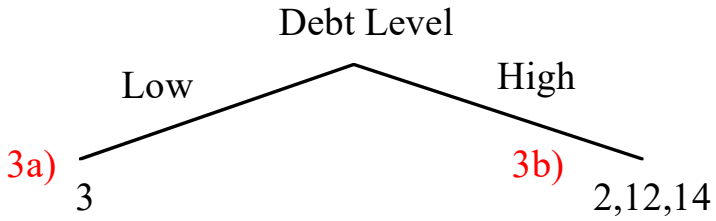
1) Choose Income as root of tree.



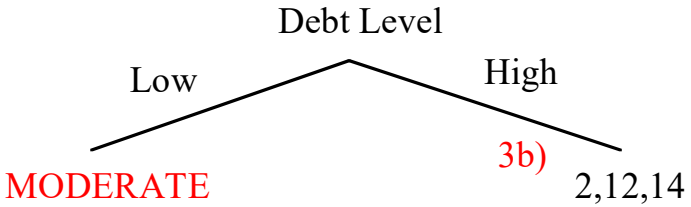
2) All examples are in the same class, HIGH.
Return Leaf Node.



3) Choose Debt Level as root of subtree.

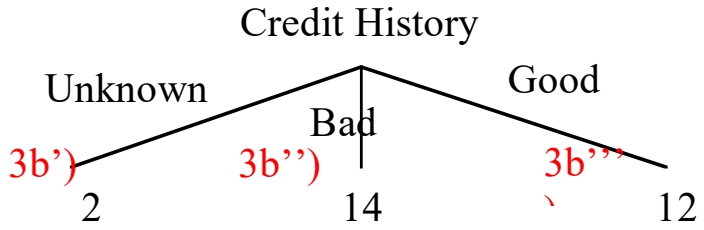


3a) All examples are in the same class, MODERATE.
Return Leaf node.

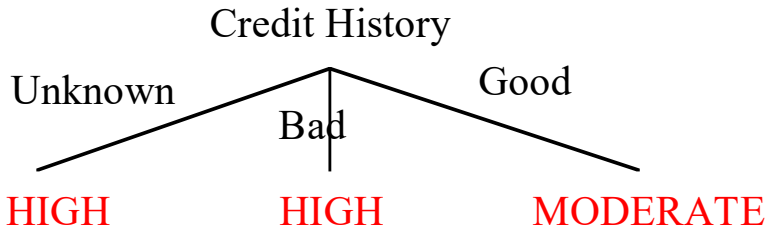


ID3 Example (II)

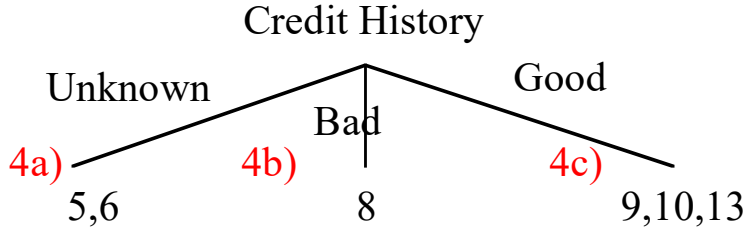
3b) Choose Credit History as root of subtree.



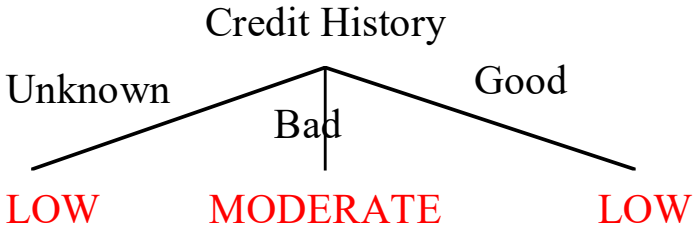
3b'-3b''') All examples are in the same class.
Return Leaf nodes.



4) Choose Credit History as root of subtree.

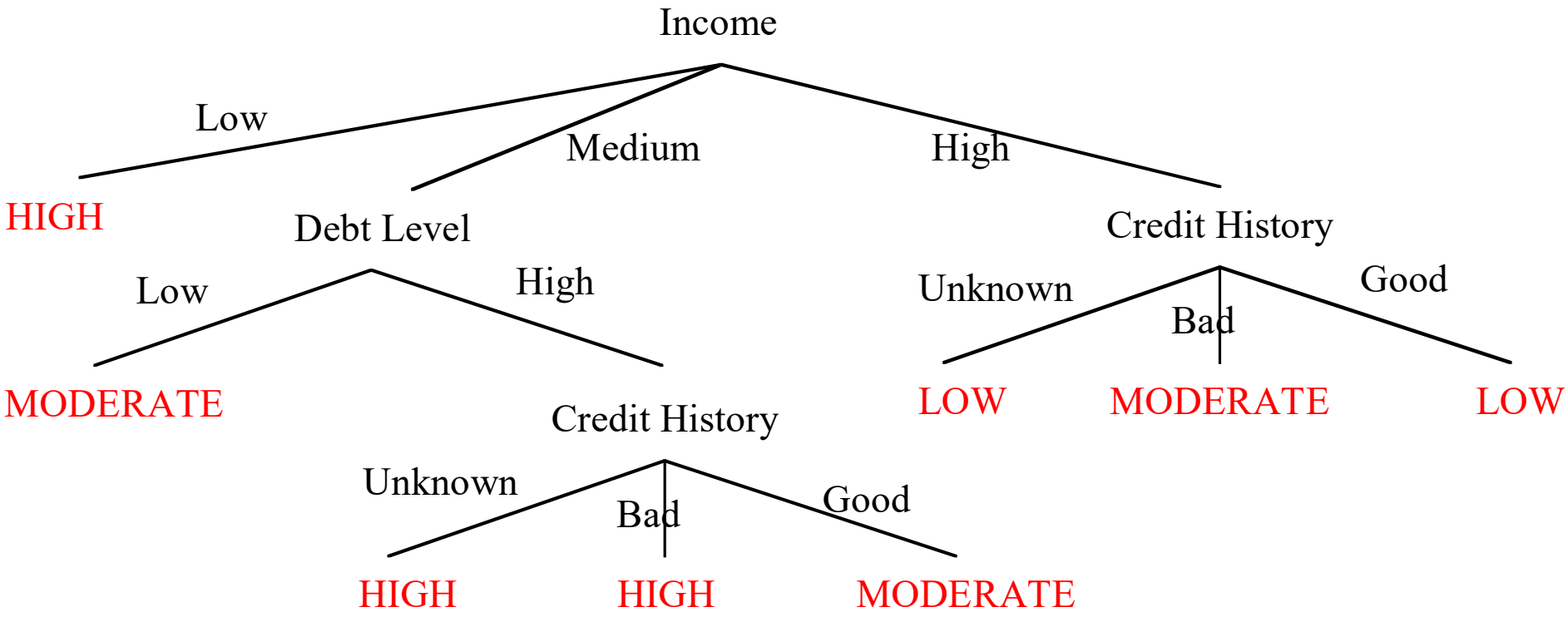


4a-4c) All examples are in the same class.
Return Leaf nodes.



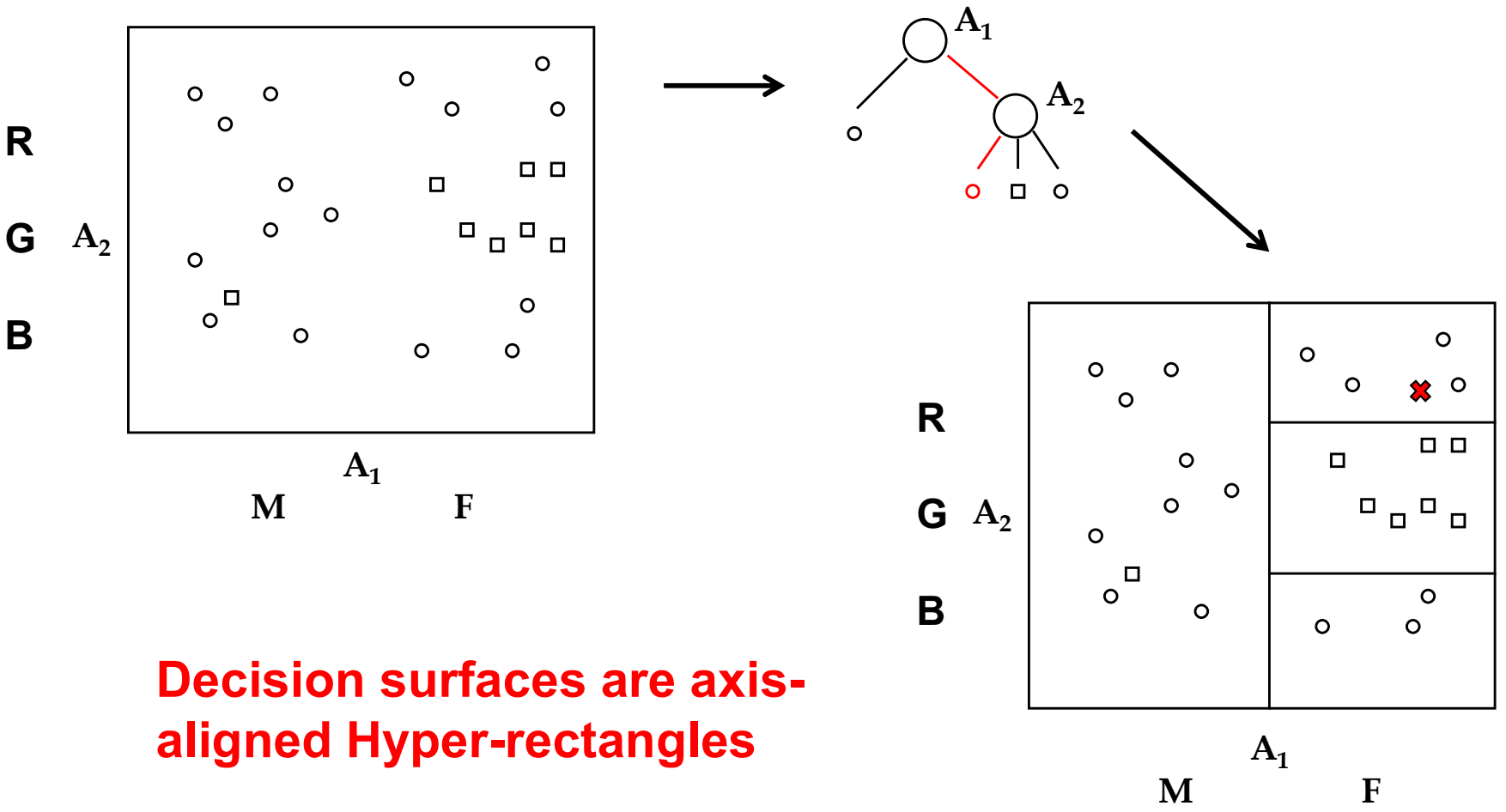
ID3 Example (III)

Attach subtrees at appropriate places.



Another Example

- Assume A_1 is binary feature (Gender: M/F)
- Assume A_2 is nominal feature (Color: R/G/B)

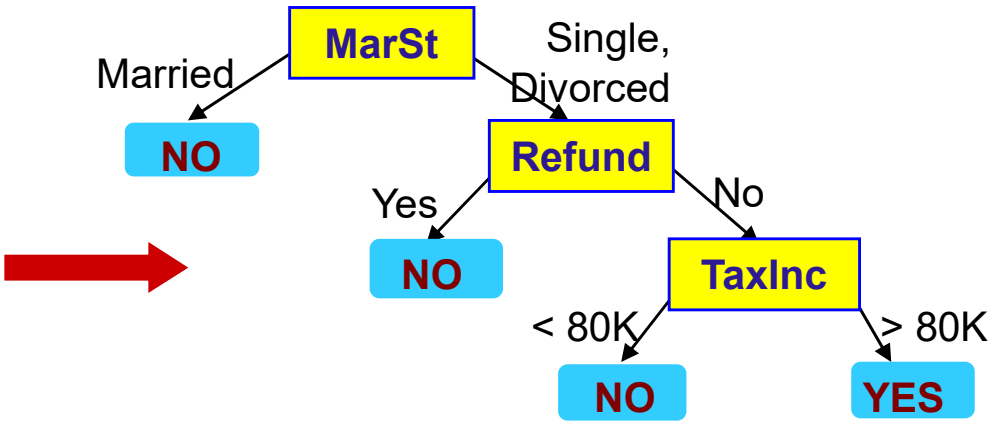


Decision surfaces are axis-aligned Hyper-rectangles

Non-Uniqueness

- Decision trees are not unique:
  - Given a set of training instances *T*, there generally exists a number of decision trees that are consistent with (or fit) *T*

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



Examples of Computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

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

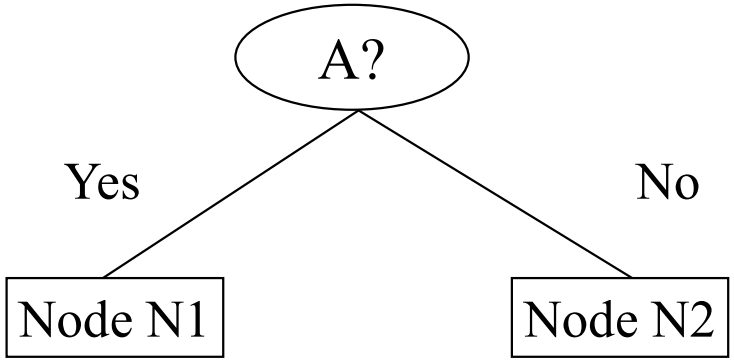
Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

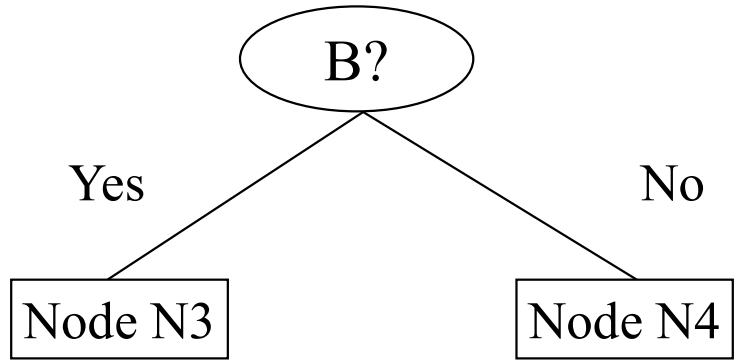
Computing Gain

Before Splitting:

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

→ **E0**





C0	**N10**
C1	**N11**

C0	**N20**
C1	**N21**

C0	**N30**
C1	**N31**

C0	**N40**
C1	**N41**

E1 **E2** **E3** **E4**

E12 **E34**

Gain = E0 – E12 vs. E0 – E34

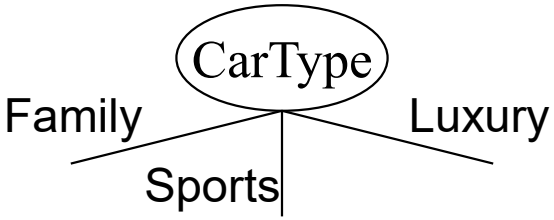
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

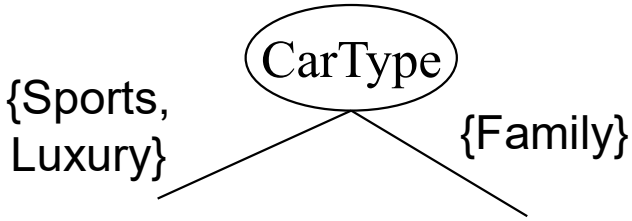
- Depends on number of ways to split
  - Binary split
  - Multi-way split

Splitting Based on Nominal Attributes

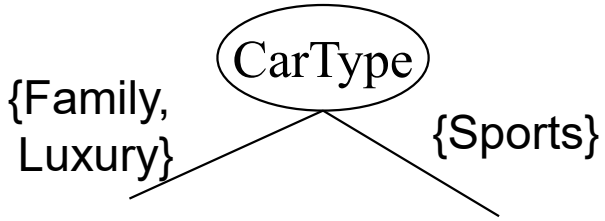
- Multi-way split: Use as many partitions as values



- Binary split: Divide values into two subsets



OR



Need to find optimal partitioning!

Splitting Based on Continuous Attributes

- Different ways of handling
  - Multi-way split: form ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – repeat on each new partition
  - Binary split: $(A < v)$ or $(A \geq v)$
    - How to choose v?

Need to find optimal partitioning!

Can use GAIN or GINI !

Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Good accuracy
- Disadvantages:
  - Redundancy
  - Need data to fit in memory
  - Need to retrain with new data

Practice – Find Out the best attribute for root

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no