CLASSIFICATION

Slides by:
Shree Jaswal

TOPICS TO BE COVERED

- Basic Concepts;
- **Classification methods**:

1. Decision Tree Induction: Attribute Selection Measures, Tree pruning.

2. Bayesian Classification: Naïve Bayes' classifier

- **Prediction**: Structure of regression models; Simple linear regression, Multiple linear regression.
- **Model Evaluation & Selection:** Accuracy and Error measures, Holdout, Random Sampling, Cross Validation, Bootstrap; Comparing Classifier performance using ROC Curves.
- **Combining Classifiers**: Bagging, Boosting, Random Forests.

WHICH CHAPTER FROM WHICH TEXT BOOK ?

- Chapter 8: Classification: Basic Concepts from Han, Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann 3nd Edition
- Chapter 4: Classification: Basic Concepts, decision trees and model evaluation from P. N. Tan, M. Steinbach, Vipin Kumar, “Introduction to Data Mining”, Pearson Education

COURSE OUTCOME ADDRESSED

- **TEITC604.3**: Implement the appropriate data mining methods like classification, clustering or association mining on large data sets.
- **TEITC604.4**: Define and apply metrics to measure the performance of various data mining algorithms.
- **TEITC604.5**: Implement Prediction using Regression technique

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.

CLASSIFICATION

- Maps data into predefined groups or classes.
- Two step process
  - Training set
    - A model built describing a predetermined set of data classes
    - Supervised learning
  - Use model for classification
    - Accuracy of the model is first estimated.
    - Then classify/ predict the data.

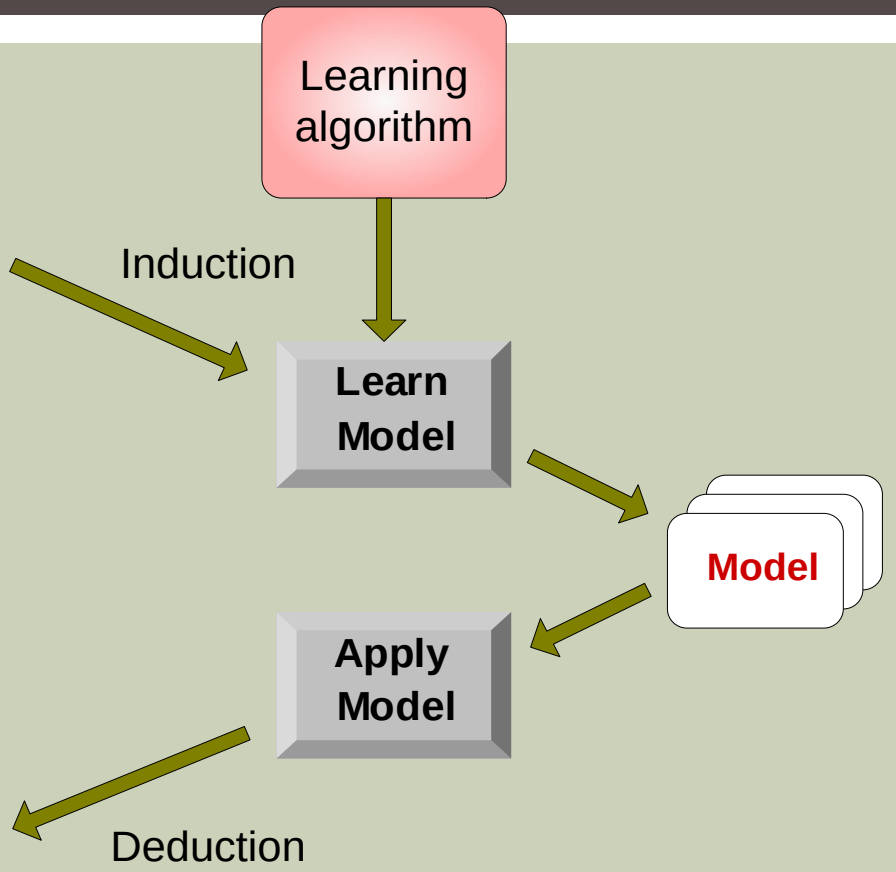
ILLUSTRATING CLASSIFICATION TASK

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

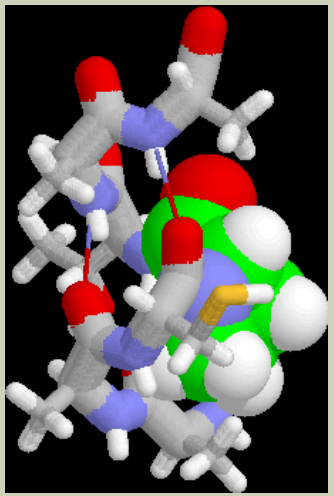
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



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



CLASSIFICATION V/S PREDICTION

- Prediction: assess the class of an unlabeled sample.
  - Classification: predict discrete or nominal values
  - Regression: predict continuous or ordered values.

  Commonly *"classification"* is used to predict class labels, while *"prediction"* is used to predict continuous values as in regression.
- Regression is a data mining function that predicts a number.
- A regression task begins with a data set in which the target values are known.
- Profit, sales, mortgage rates, house values, square footage, temperature, or distance could all be predicted using regression techniques.
- For example, a regression model could be used to predict the value of a house based on location, number of rooms, lot size, and other factors.

ISSUES IN CLASSIFICATION

- Missing data
  - Values missing in the training data
  - Attribute values missing in the samples.
  - Handling missing data
    - Ignore
    - Assume a mean or average value.
    - Assume a special value
- Measuring performance
  - Classifier accuracy

CLASSIFIER ACCURACY

- Estimate classifier accuracy
  - Confusion Matrix
  - Holdout method
  - Cross-validation method
- Increase classifier accuracy
  - Tree pruning, in case of decision trees.
  - Bagging
  - Boosting

CLASSIFICATION TECHNIQUES

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines

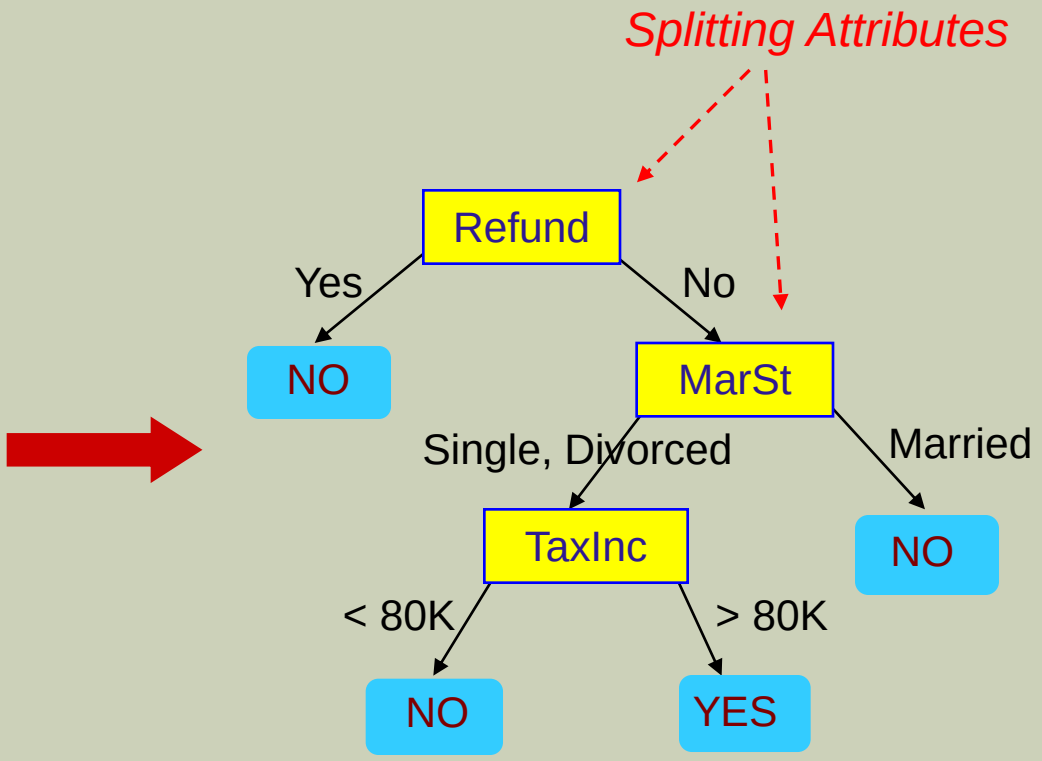
BY DECISION TREE

- A decision tree is a flow chart like tree structure, where
  - Each *internal node* denotes a *test* on an attribute
  - Each *branch* denotes an *outcome* of the test
  - Each *leaf node* represent a *class*
- In order to classify an unknown sample, the attribute values of the sample are tested against the decision tree.

EXAMPLE OF A DECISION TREE

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
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

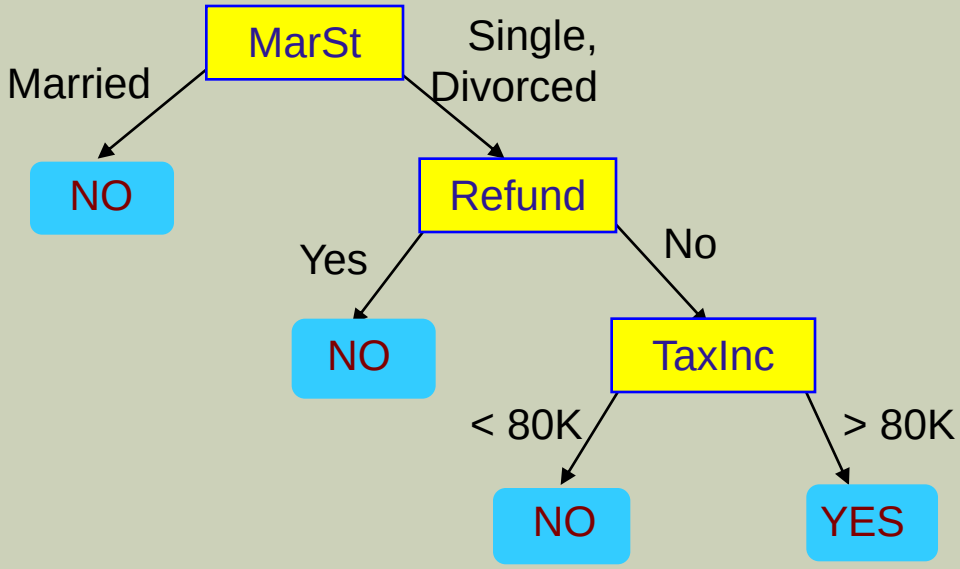
Training Data



Model: Decision Tree

ANOTHER EXAMPLE OF DECISION TREE

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
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!

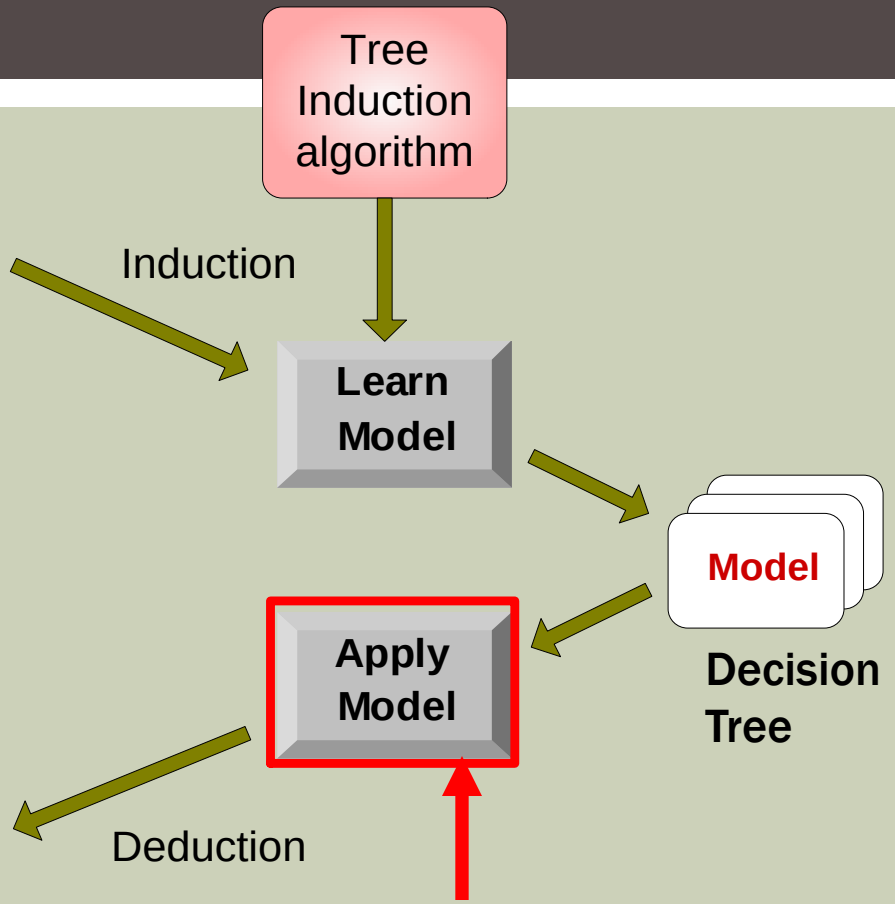
DECISION TREE CLASSIFICATION TASK

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

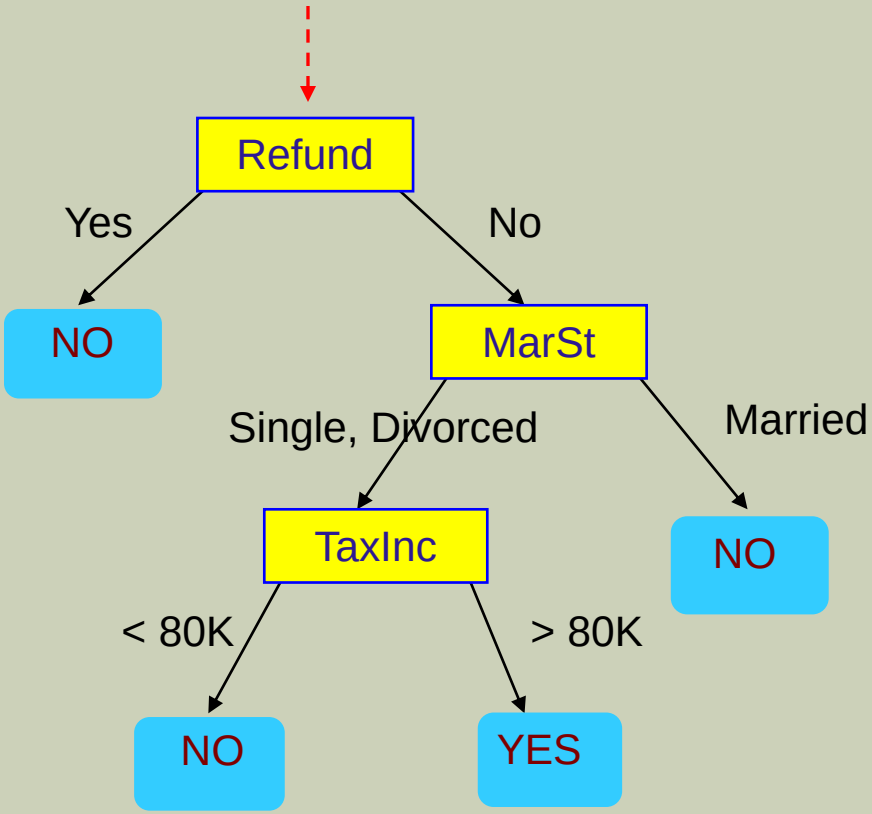
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



APPLY MODEL TO TEST DATA

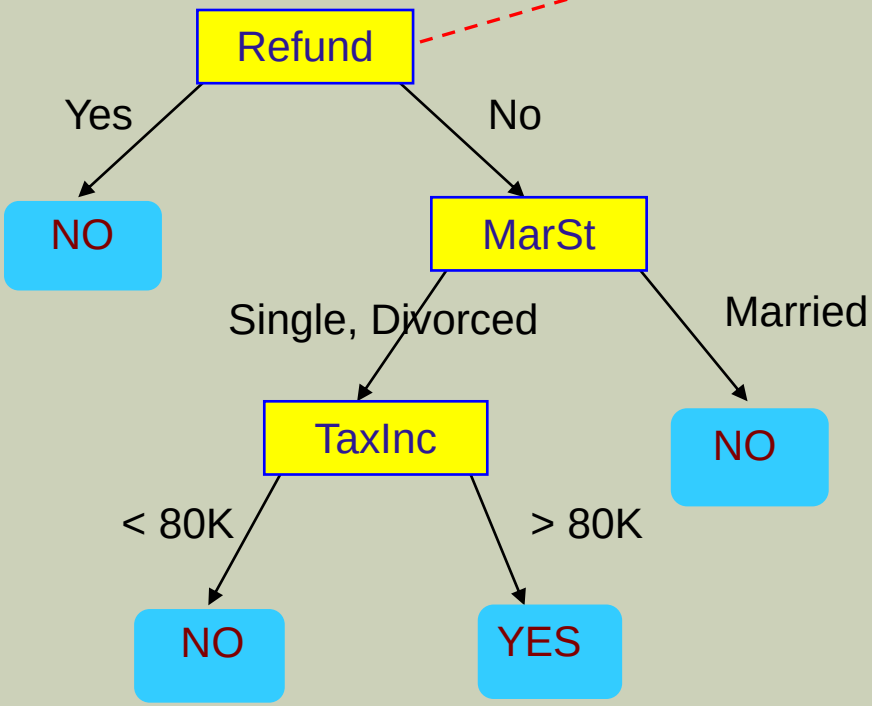
Start from the root of tree.



Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

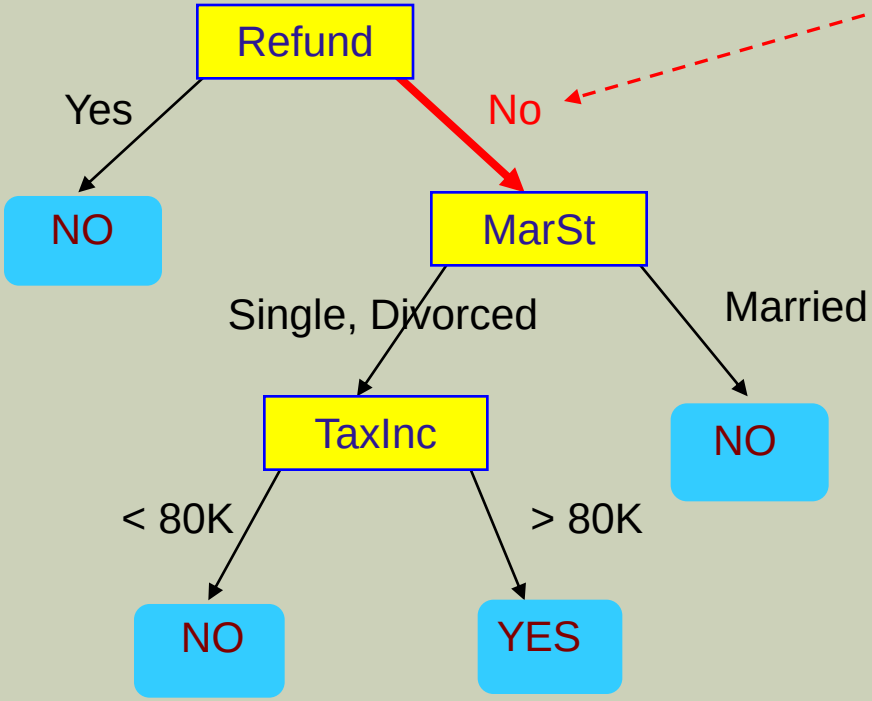
APPLY MODEL TO TEST DATA

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



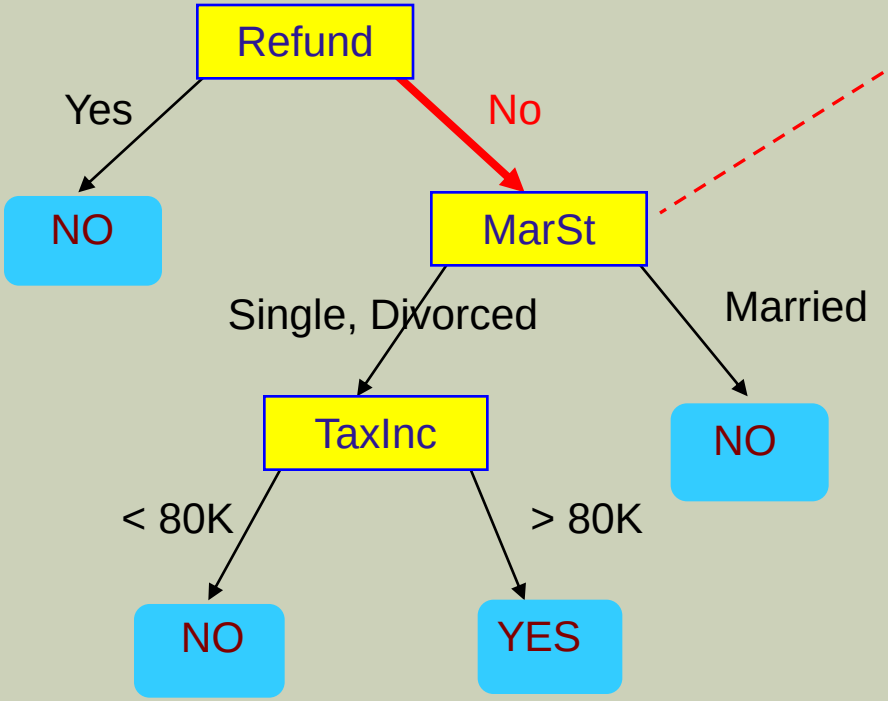
APPLY MODEL TO TEST DATA

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



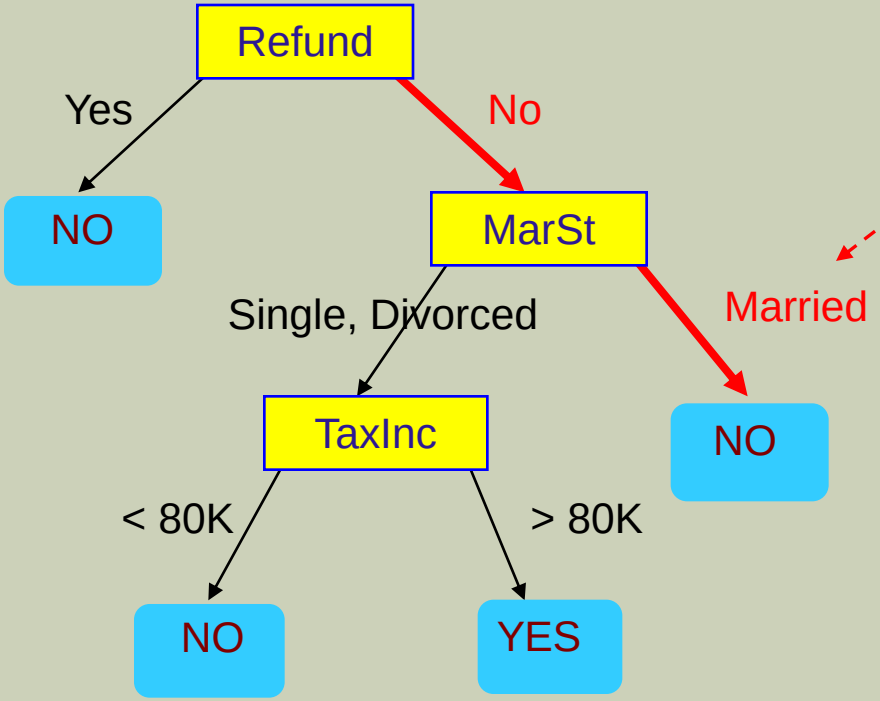
APPLY MODEL TO TEST DATA

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



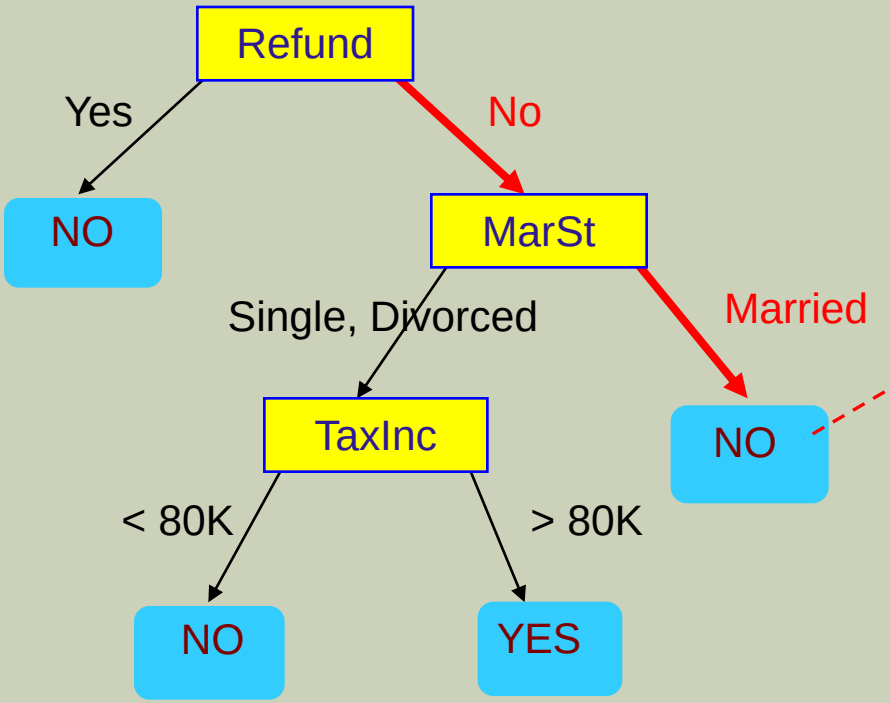
APPLY MODEL TO TEST DATA

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



APPLY MODEL TO TEST DATA

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

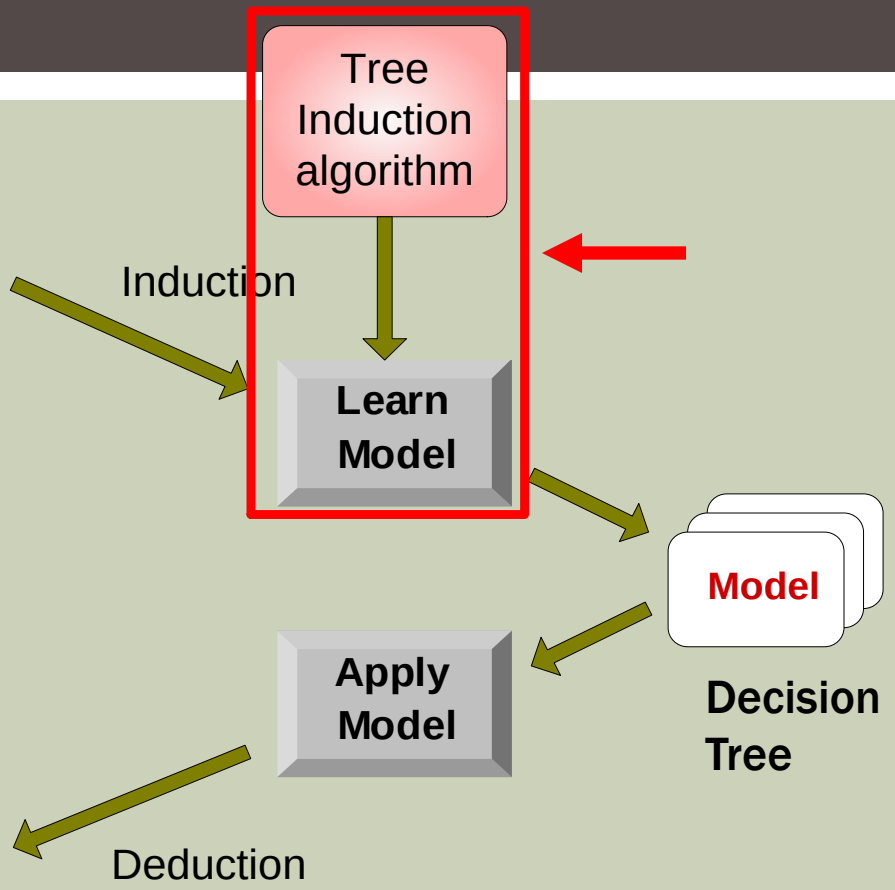


Assign Cheat to “No”

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

DECISION TREE INDUCTION

- Many Algorithms:
  - Hunt's Algorithm (one of the earliest)
  - CART (Classification And Regression trees)
  - ID3 (Iterative dichotomiser), C4.5
  - SLIQ,SPRINT

TREE INDUCTION

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

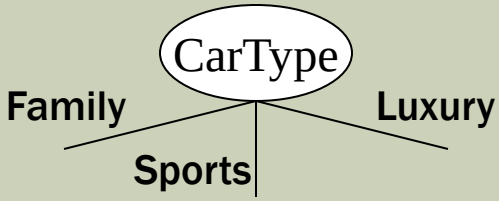
- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

HOW TO SPECIFY TEST CONDITION?

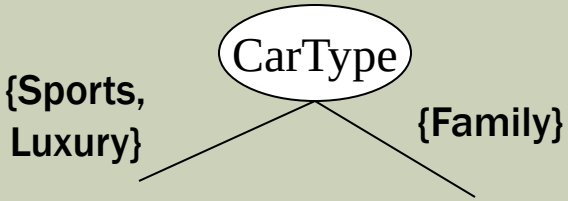
- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

SPLITTING BASED ON NOMINAL ATTRIBUTES

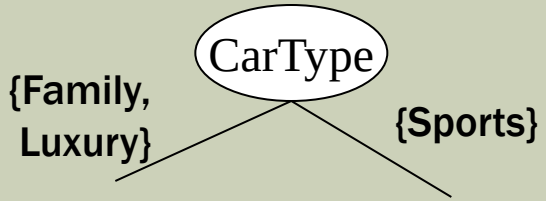
- **Multi-way split:** Use as many partitions as distinct values.



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

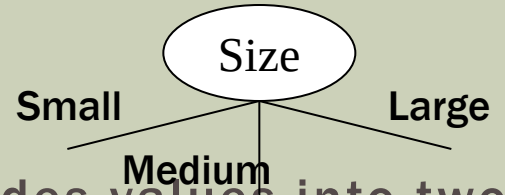


OR



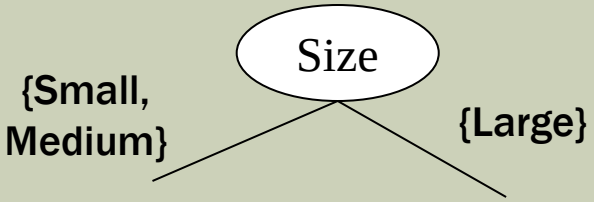
SPLITTING BASED ON ORDINAL ATTRIBUTES

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

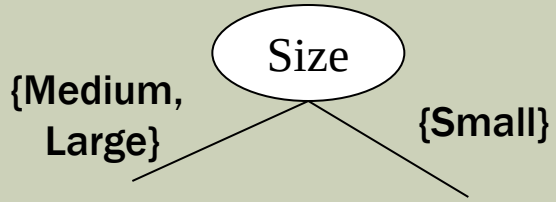


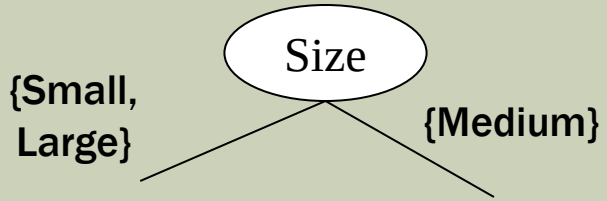
- **Binary split:** Divides values into two subsets.
  Need to find optimal partitioning.

- **What about this split?**



OR

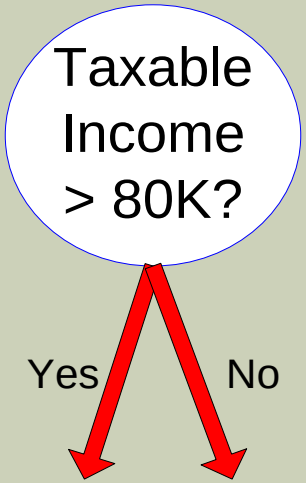




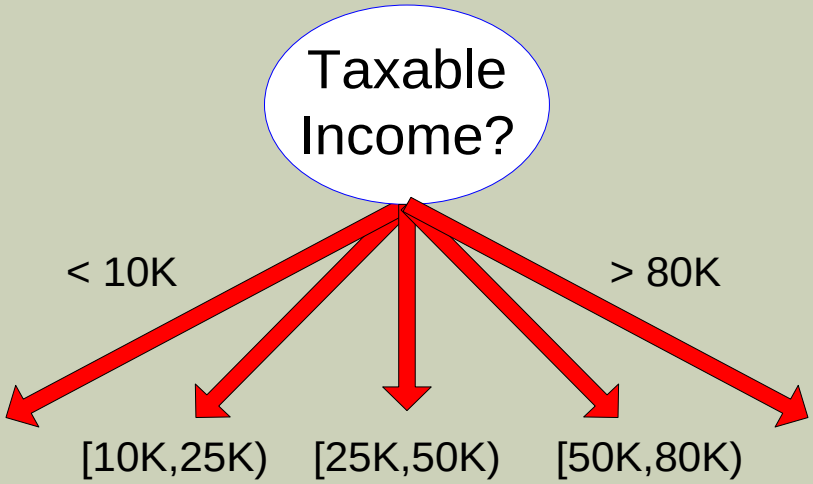
SPLITTING BASED ON CONTINUOUS ATTRIBUTES

- 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



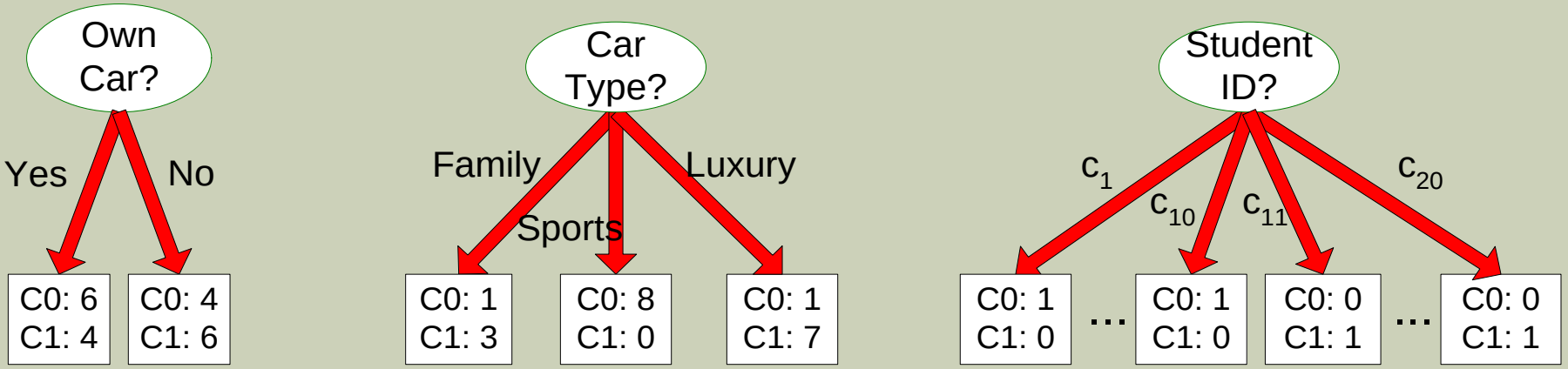
(ii) Multi-way split

TREE INDUCTION

- 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 0,
10 records of class 1



Which test condition is the best? The measures developed for selecting the best split are often based on the degree of impurity of child nodes. Impurity measures include Entropy(t), Gini(t), Classification error(t) etc.

HOW TO DETERMINE THE BEST SPLIT

- Greedy approach:
  - Nodes with **homogeneous** class distribution are 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

CHOOSING ATTRIBUTES

- The order in which attributes are chosen determines how complicated the tree is.
- ID3 uses information theory to determine the most informative attribute.
- A measure of the information content of a message is the inverse of the probability of receiving the message:

$$information(M) = 1/probability(M)$$

- Taking logs (base 2) makes information correspond to the number of bits required to encode a message:

$$information(M) = -log_2(probability(M))$$

ENTROPY

- Different messages have different probabilities of arrival.
- Overall level of uncertainty (termed entropy) is:

$$-\Sigma_i P_i \log_2 P_i$$

- Frequency can be used as a probability estimate.
  - E.g. if there are 5 positive examples and 3 negative examples in a node the estimated probability of positive is 5/8 = 0.625.

SPLITTING CRITERION

- Work out entropy based on distribution of classes.
- Trying splitting on each attribute.
- Work out expected information gain for each attribute.
- Choose best attribute.

ID3 ALGORITHM

- Constructs a decision tree by using top-down recursive approach.
- Main aim is to choose that splitting attribute which is having the highest information gain.

- The tree starts as a single node, representing all the training samples
- If all samples are of the same class, then the node becomes a leaf node and is labeled with that class.
- Else, an entropy-based algorithm, known as information gain, is used for selecting the attribute that will best separate the samples into individual classes. This attribute becomes the test attribute at the node.

- Branch is now constructed for each value of the test attribute and samples are partitioned accordingly.
- The algorithm then recursively applies the same process to form a decision tree for samples at each node.
- This recursive partitioning stops when either:
  - It is a leaf node
  - Splitting attributes is over

CALCULATE INFORMATION GAIN

- Entropy:
  - Given a collection S of c outcomes
  - Entropy(S) = S -p(I) log2 p(I)
  - Where p(I) is the proportion of S belonging to class I. S is over c. Log2 is log base 2

- Gain(S, A) is information gain of example set S on attribute A is defined as
  - Gain(S, A) = Entropy(S) - S (($|S_v|$ / $|S|$) * Entropy(S_v))
- Where:
  - S is each value v of all possible values of attribute A
  - S_v = subset of S for which attribute A has value v
  - $|S_v|$ = number of elements in S_v $|S|$ = number of elements in S

TRAINING SET

RID	Age	Income	Student	Credit	Buys
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	31-40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	31-40	Low	Yes	Excellent	Yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	31-40	Medium	No	Excellent	Yes
13	31-40	High	Yes	Fair	Yes
14	>40	Medium	No	Excellent	No

- S is a collection of 14 examples with 9 YES and 5 NO examples then

  $\text{Entropy}(S) = -(9/14)\,\text{Log}_2(9/14) - (5/14)\,\text{Log}_2(5/14)$

  $= 0.940$

- Let us consider Age as the splitting attribute

  $\text{Gain}(S, \text{Age}) = \text{Entropy}(S) - (5/14)*\text{Entropy}(S_{<30}) - (4/14)*\text{Entropy}(S_{31\text{-}40}) - (5/14)*\text{Entropy}(S_{>40})$

  $= 0.940 - (5/14)*0.971 - (4/14)*0 - (5/14)*0.971$

  $= 0.246$

- Similarly consider Student as the splitting attribute

  Gain(S, Student) = Entropy(S) - (7/14)*Entropy(S_{YES})
  - (7/14)*Entropy(S_{NO})

  = 0.151

- Similarly consider Credit as the splitting attribute

  Gain(S, Credit) = Entropy(S) - (8/14)*Entropy(S_{FAIR})
  - (6/14)*Entropy($S_{EXCELLENT}$)

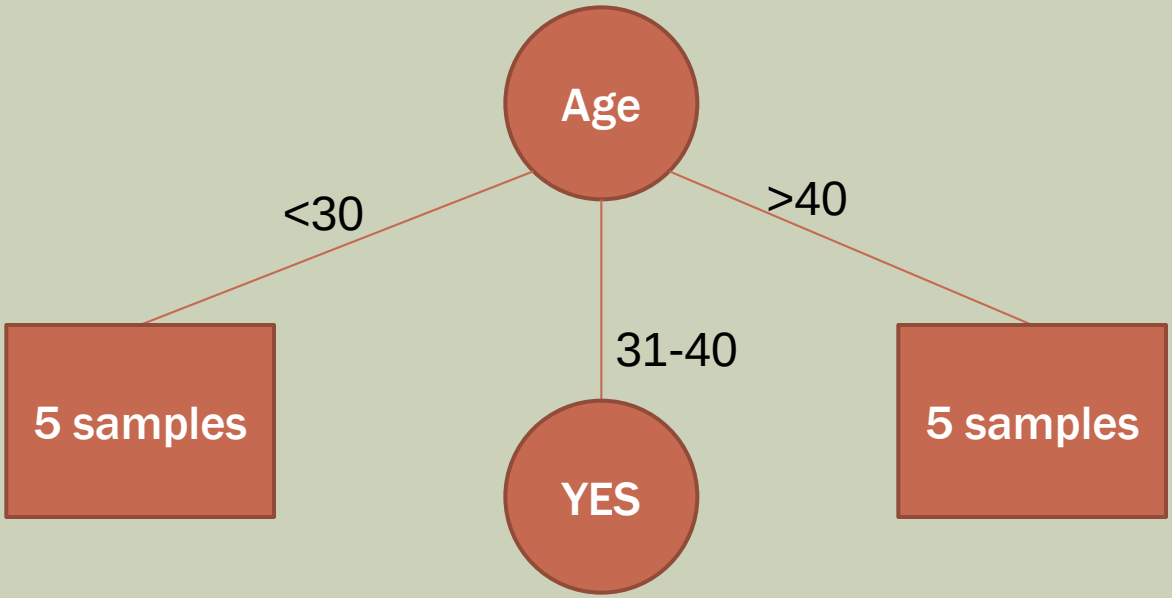
  = 0.046

- Similarly consider Income as the splitting attribute

  Gain(S, Income) = Entropy(S) - (4/14)*Entropy(S_{HIGH})
  - (5/14)*Entropy(S_{MED})
  - (5/14)*Entropy(S_{LOW})

  = 0.029

- We see that Gain (S, Age) is max with 0.246.
- Hence Age is chosen as the splitting attribute.



- Recursively we find the splitting attributes at the next level.
- Let us consider Student as the next splitting attribute

$$\textbf{Gain}(\textbf{Age}_{<30}, \textbf{Student}) = \text{Entropy}(\text{Age}_{<30}) - (2/5)*\text{Entropy}(S_{YES}) - (3/5)*\text{Entropy}(S_{NO})$$

$$= 0.971 - (2/5)*0 - (3/5)*0$$

$$= \mathbf{0.971}$$

- Similarly Credit as the next splitting attribute

$$\textbf{Gain}(\textbf{Age}_{<30}, \textbf{Credit}) = \text{Entropy}(\text{Age}_{<30}) - (3/5)*\text{Entropy}(S_{FAIR}) - (2/5)*\text{Entropy}(S_{EXCELLENT})$$

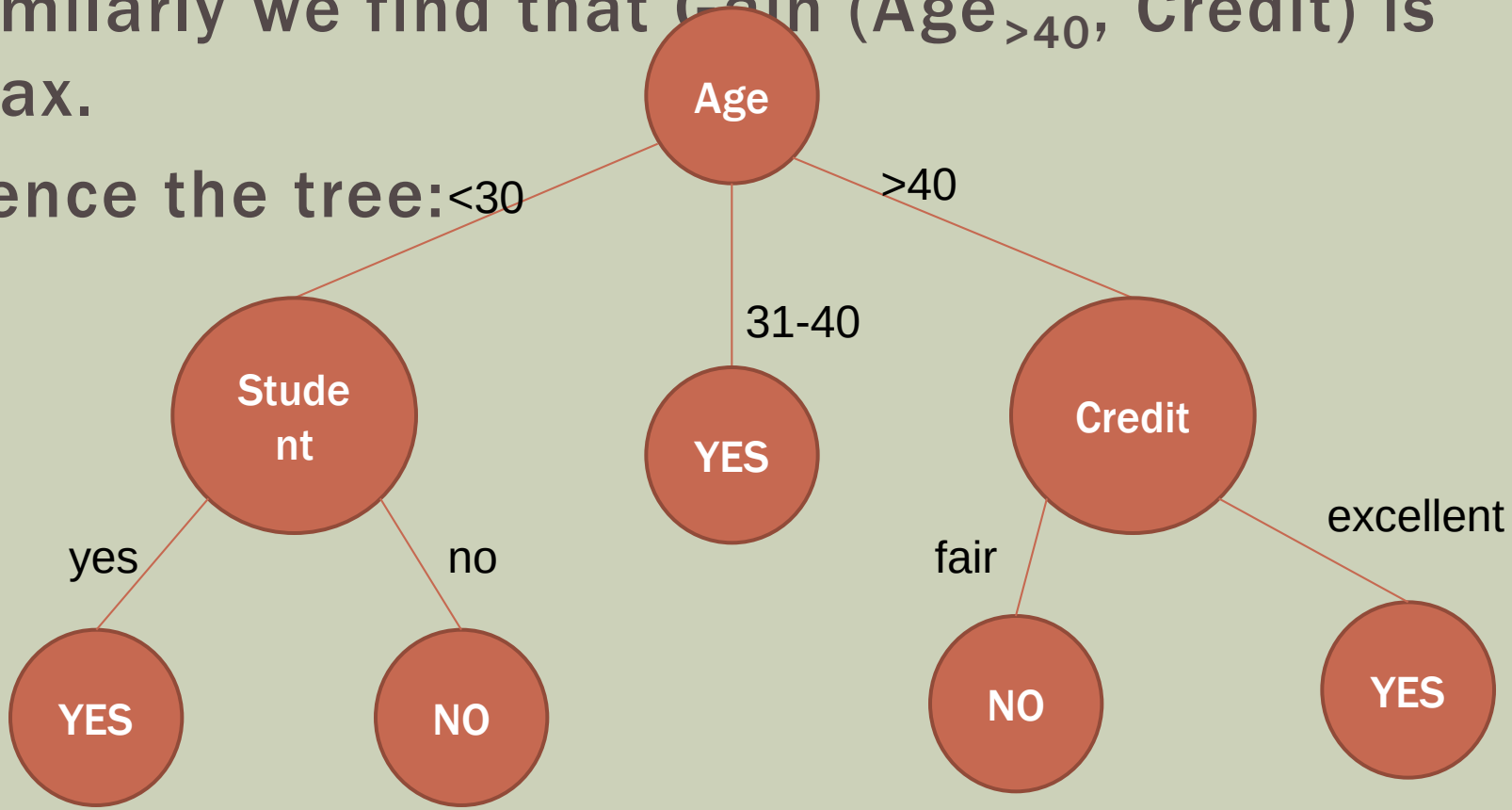
$$= \mathbf{0.57}$$

- Similarly Income as the next splitting attribute

$$\textbf{Gain}(\textbf{Age}_{<30}, \textbf{Income}) = \text{Entropy}(\text{Age}_{<30}) - (2/5)*\text{Entropy}(S_{HIGH}) - (2/5)*\text{Entropy}(S_{MED}) - (1/5)*\text{Entropy}(S_{LOW})$$

$$= \mathbf{0.19}$$

- We see that Gain ($Age_{<30}$, Student) is max with 0.971.
- Hence Student is chosen as the next splitting attribute.
- Similarly we find that Gain ($Age_{>40}$, Credit) is max.
- Hence the tree:



EXTRACTING CLASSIFICATION RULES FROM TREES

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example

IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"

IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

IF *age* = "31…40" THEN *buys_computer* = "*yes*"

IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"

IF *age* = ">40" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

CONSTRUCT A DECISION TREE FOR THE BELOW EXAMPLE:

- Suppose we want ID3 to decide whether the weather is amenable to playing baseball. Over the course of 2 weeks, data is collected to help ID3 build a decision tree. The target classification is "should we play baseball?" which can be yes or no.
- The weather attributes can have the following values:
  - outlook = { sunny, overcast, rain }
  - temperature = {hot, mild, cool }
  - humidity = { high, normal }
  - wind = {weak, strong }

Table 1

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

- We need to find which attribute will be the root node in our decision tree. The gain is calculated for all four attributes:
- Gain(S, Outlook) = 0.246
- Gain(S, Temperature) = 0.029
- Gain(S, Humidity) = 0.151
- Gain(S, Wind) = 0.048

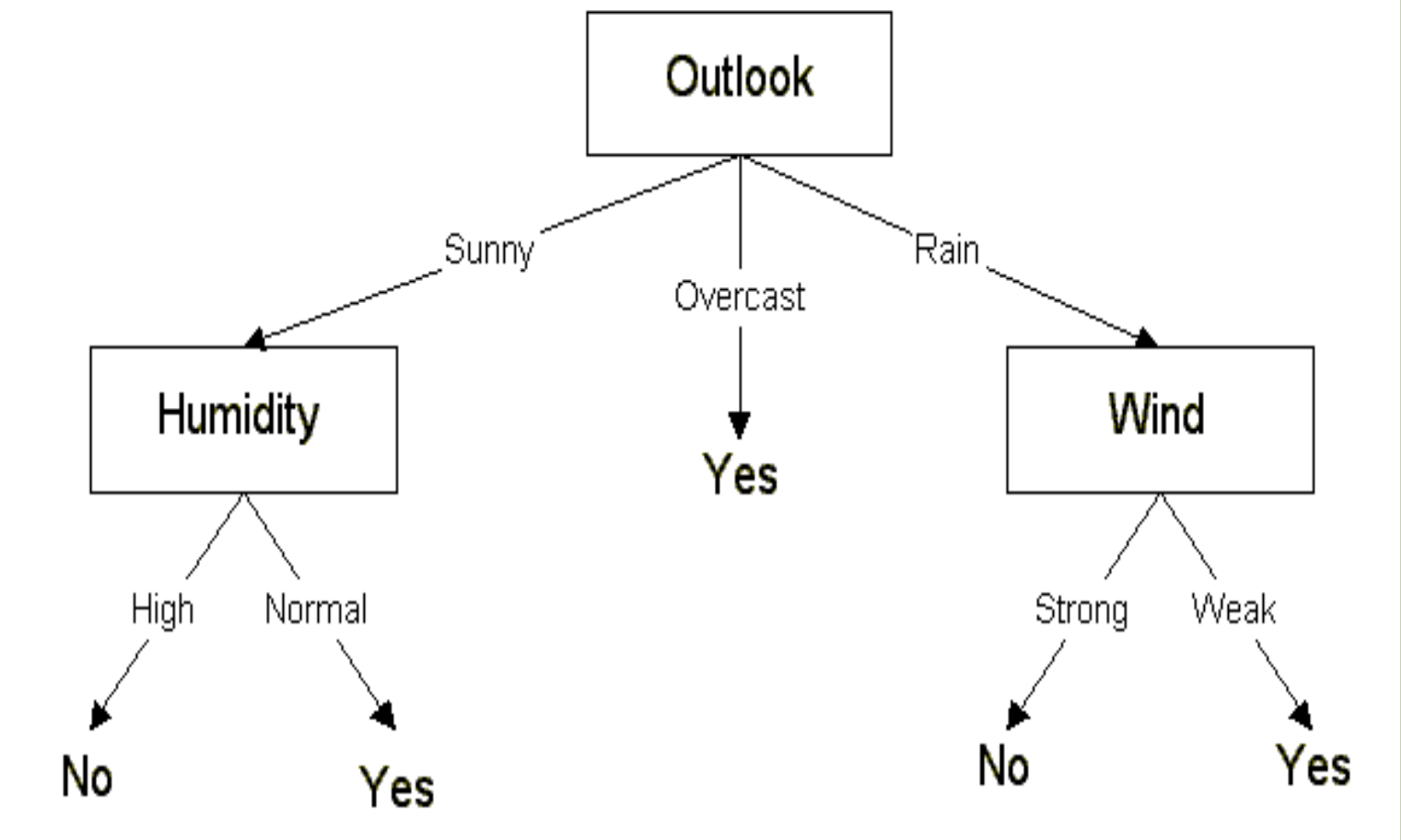
- Gain(S,Wind)=Entropy(S)-(8/14)*Entropy(Sweak)-(6/14)*Entropy(Sstrong)

  = 0.940 - (8/14)*0.811 - (6/14)*1.00

  = 0.048

  - Entropy(Sweak) = - (6/8)*log2(6/8) - (2/8)*log2(2/8) = 0.811
  - Entropy(Sstrong) = - (3/6)*log2(3/6) - (3/6)*log2(3/6) = 1.00

- Outlook attribute has the highest gain, therefore it is used as the decision attribute in the root node.
- Since Outlook has three possible values, the root node has three branches (sunny, overcast, rain). The next question is "what attribute should be tested at the Sunny branch node?" Since we have used Outlook at the root, we only decide on the remaining three attributes: Humidity, Temperature, or Wind.
- Ssunny = {D1, D2, D8, D9, D11} = 5 examples from table 1 with outlook = sunny
- Gain(Ssunny, Humidity) = 0.970
- Gain(Ssunny, Temperature) = 0.570
- Gain(Ssunny, Wind) = 0.019
- Humidity has the highest gain; therefore, it is used as the decision node. This process goes on until all data is classified perfectly or we run out of attributes.


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

The decision tree can also be expressed in rule format:

- IF outlook = sunny AND humidity = high THEN playball = no
- IF outlook = rain AND humidity = high THEN playball = no
- IF outlook = rain AND wind = strong THEN playball = yes
- IF outlook = overcast THEN playball = yes
- IF outlook = rain AND wind = weak THEN playball = yes

BAYESIAN CLASSIFICATION: WHY?

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

BAYESIAN THEOREM: BASICS

- Let **X** be a data sample ("*evidence*"): class label is unknown
- Let H be a *hypothesis* that X belongs to class C
- Classification is to determine P(H|**X**), the probability that the hypothesis holds given the observed data sample **X**
- P(H) (*prior probability*), the initial probability
  - E.g., **X** will buy computer, regardless of age, income, …
- P(**X**): probability that sample data is observed
- P(**X**|H) (*posteriori probability*), the probability of observing the sample **X**, given that the hypothesis holds
  - E.g., Given that **X** will buy computer, the prob. that X is aged 31..40, medium income

BAYESIAN THEOREM

- Given training data **X**, *posteriori probability of a hypothesis* H, P(H|**X**), follows the Bayes theorem

$$P(H \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid H)P(H)}{P(\mathbf{X})}$$

- Informally, this can be written as

  posteriori = likelihood x prior/evidence

- Predicts **X** belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|\mathbf{X})$ for all the *k* classes

- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

TOWARDS NAÏVE BAYESIAN CLASSIFIER

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector $\mathbf{X} = (x_1, x_2, \ldots, x_n)$
- Suppose there are *m* classes $C_1, C_2, \ldots, C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(X) is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

NAÏVE BAYESIAN CLASSIFIER: TRAINING DATASET

Class:
C1:buys_computer = ‘yes’
C2:buys_computer = ‘no’

Data sample
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

age	income	student	credit_rating	comp
<=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

NAÏVE BAYESIAN CLASSIFIER: AN EXAMPLE

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357

- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4

- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")