# Classification: Basic Concepts

- Basics
- Decision Tree Classifier
- Rule Based Classifier
- Nearest Neighbor Classifier
- Bayesian Classifier
- Artificial Neural Network Classifier

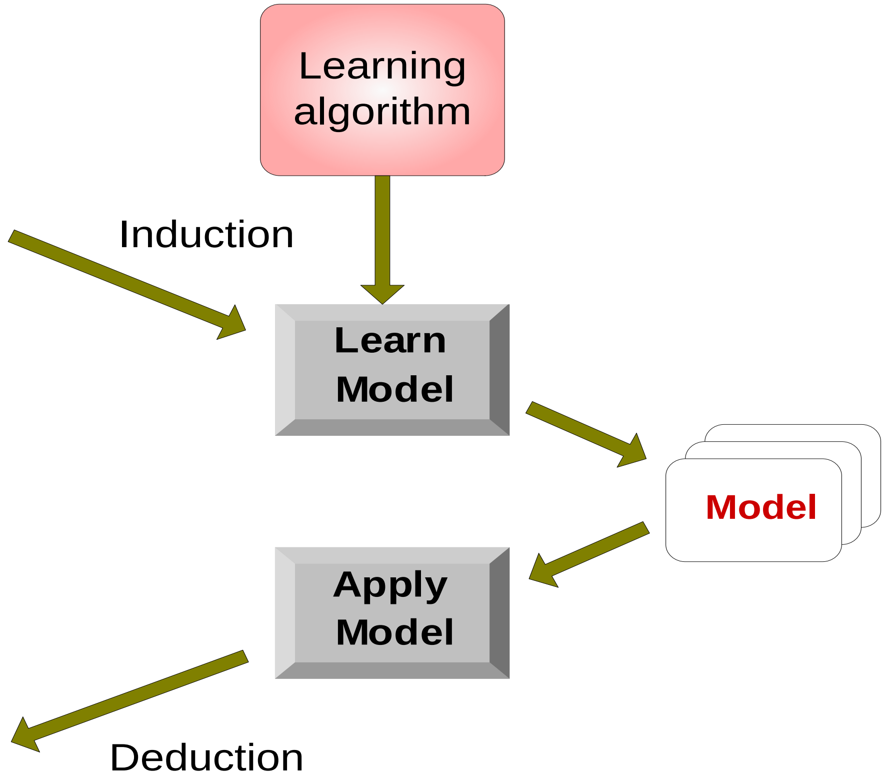
Issues : Over-fitting, Validation, Model Comparison

# Supervised Learning

- Supervised learning (classification)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set

| *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 vs. Prediction

- Classification:
  - predicts categorical class labels
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Regression:
  - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical Applications
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis

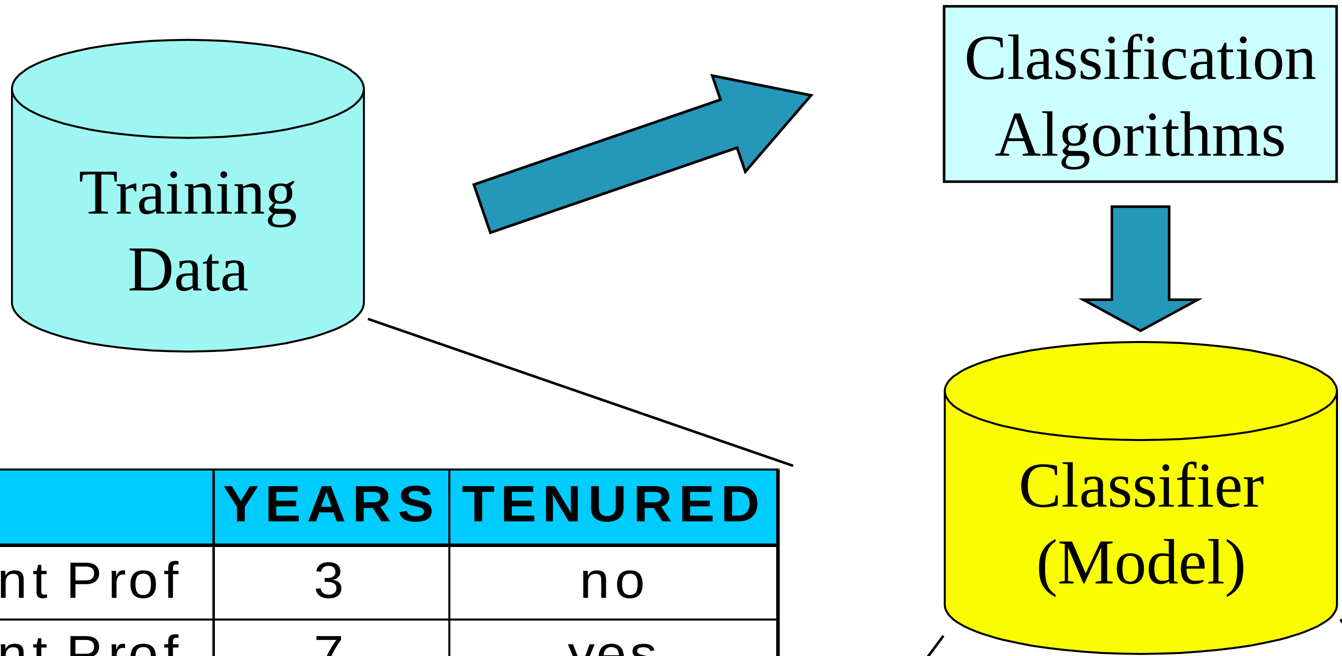
# Why Classification? A motivating application

- Credit approval
  - A bank wants to classify its customers based on whether they are expected to pay back their approved loans
  - The history of past customers is used to train the classifier
  - The classifier provides rules, which identify potentially reliable future customers
  - Classification rule:
    - If age = "31...40" and income = high then credit_rating = excellent
  - Future customers
    - Paul: age = 35, income = high $\Rightarrow$ excellent credit rating
    - John: age = 20, income = medium $\Rightarrow$ fair credit rating

# Classification—A Two-Step Process

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction: training set
  - The model is represented as classification rules, decision trees, or mathematical formulae

# Classification Process (1): Model Construction: E.g.




| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Mike | Assistant Prof | 3 | no |
| Mary | Assistant Prof | 7 | yes |
| Bill | Professor | 2 | yes |
| Jim | Associate Prof | 7 | yes |
| Dave | Assistant Prof | 6 | no |
| Anne | Associate Prof | 3 | no |

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

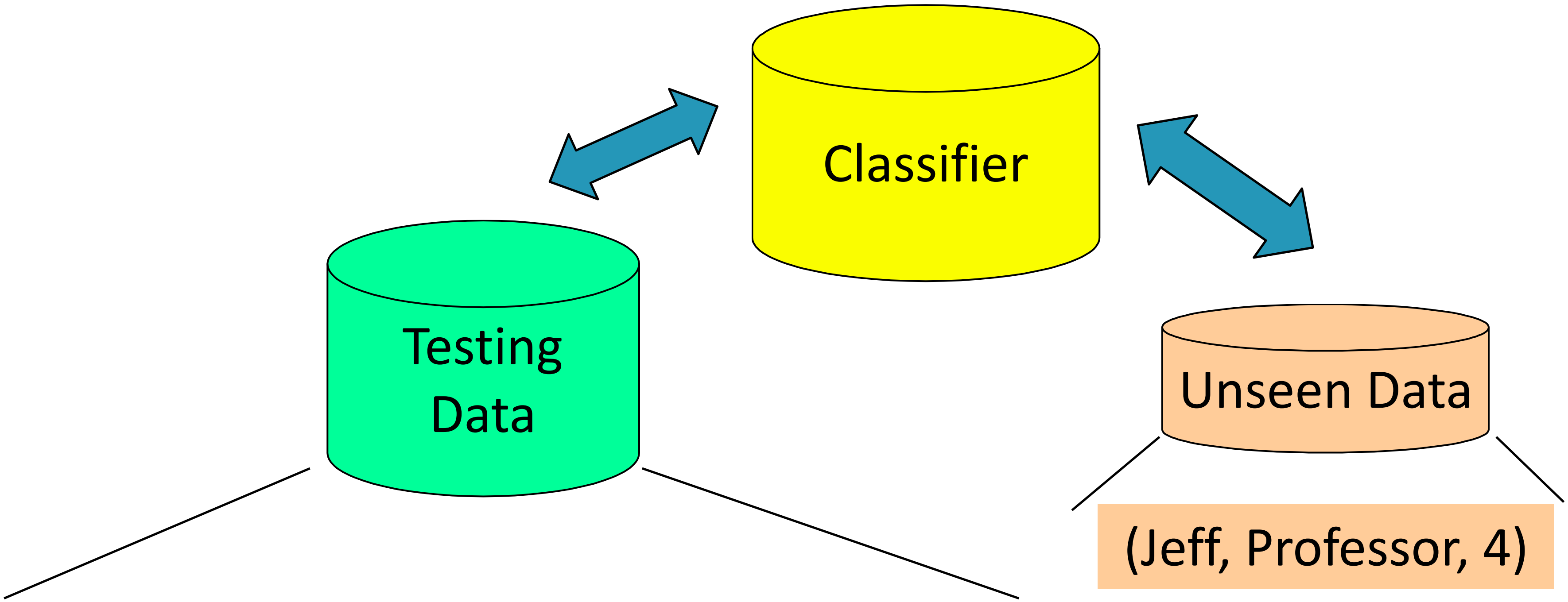
# Classification—A Two-Step Process

- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test samples is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model

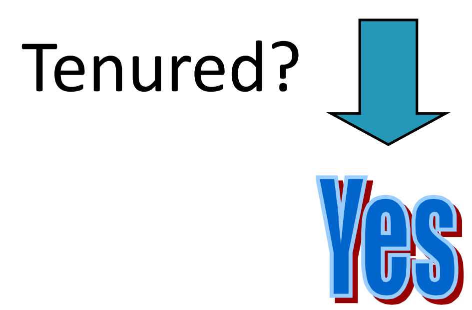
$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$

    - Test set is independent of training set, otherwise over-fitting will occur

# Classification Process (2): Use the Model in Prediction

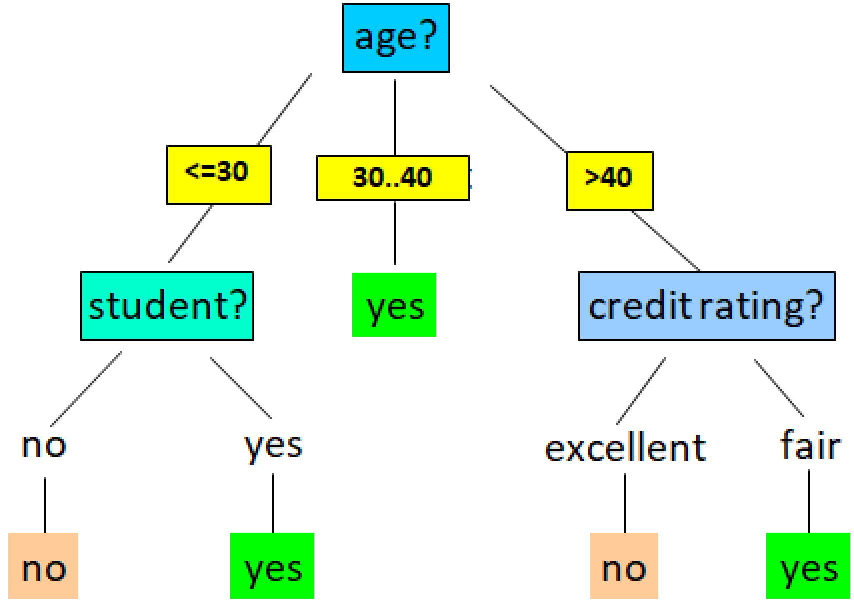



| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Tom | Assistant Prof | 2 | no |
| Mellisa | Associate Prof | 7 | no |
| George | Professor | 5 | yes |
| Joseph | Assistant Prof | 7 | yes |



# Classification by Decision Tree Induction

- 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



# Classification by Decision Tree Induction

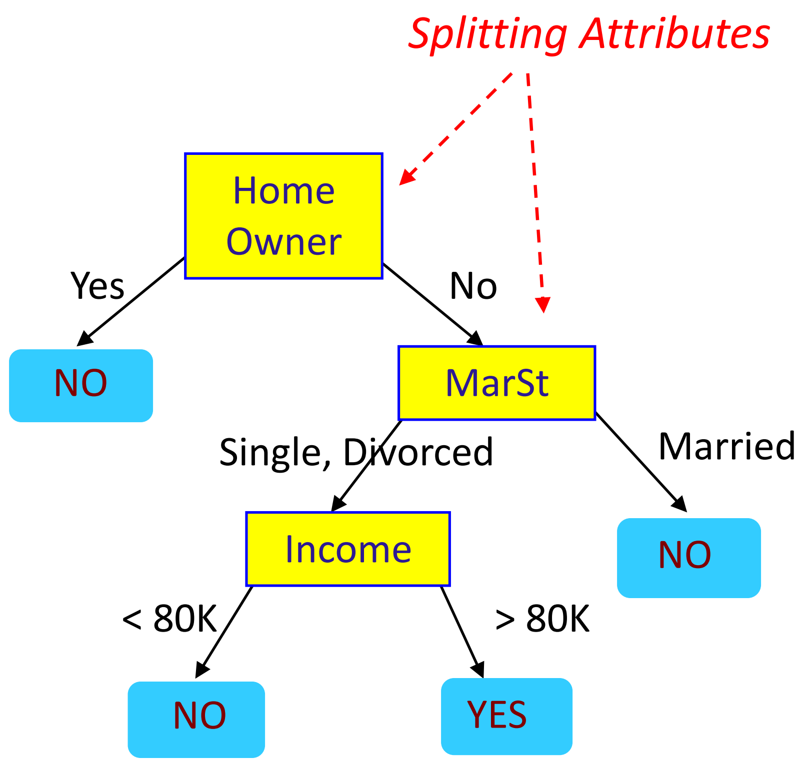
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

# Example of a Decision Tree

categorical categorical continuous class

| ID | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |

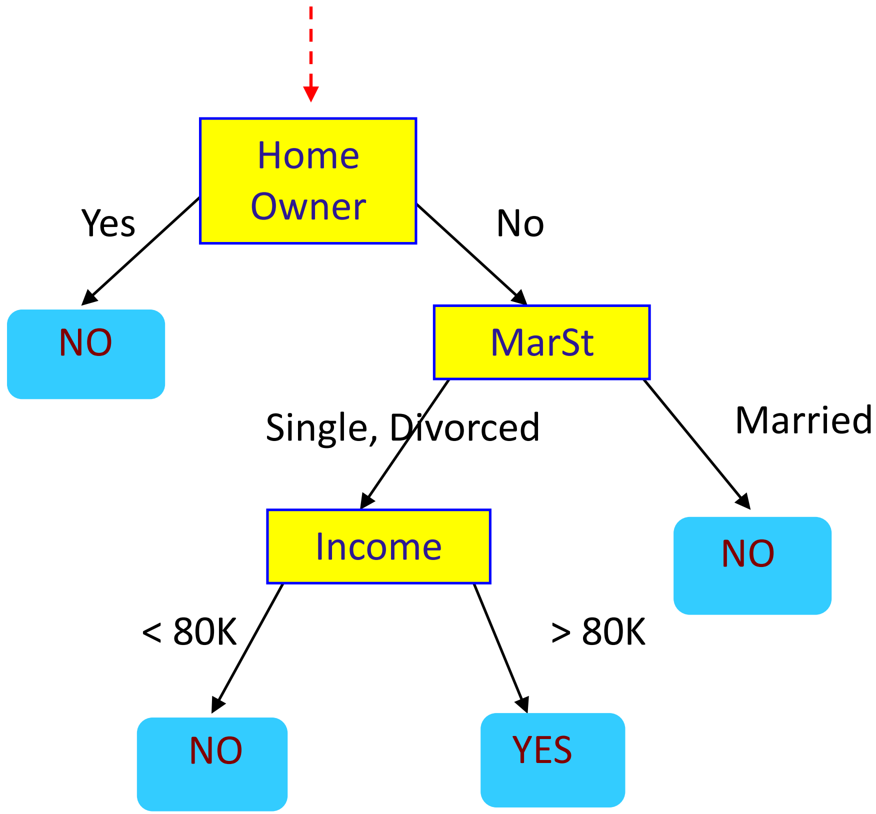
Training Data




Model: Decision Tree

# Apply Model to Test Data

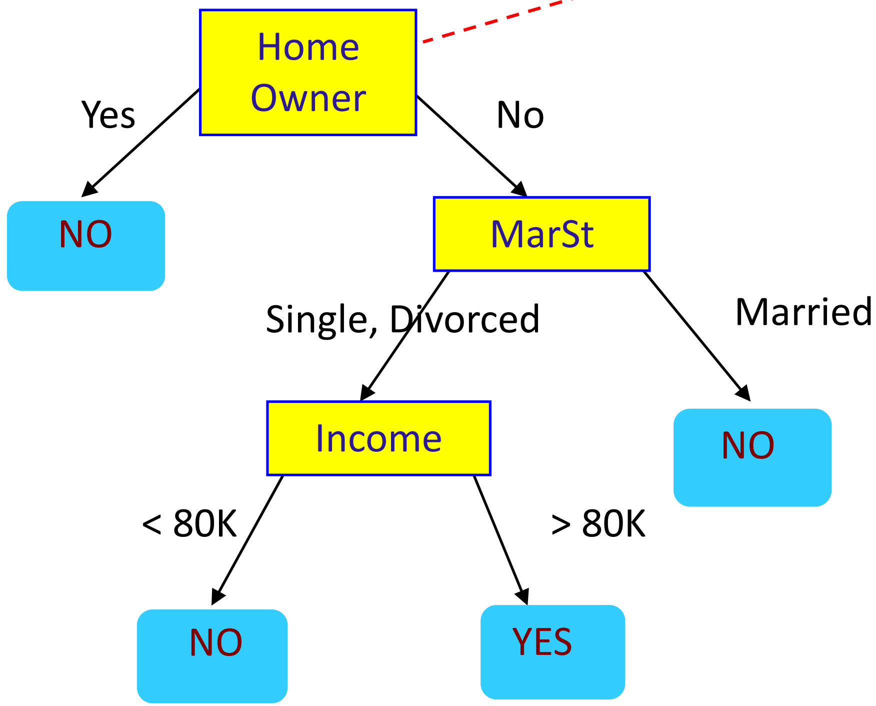
Start from the root of tree.




Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | **?** |

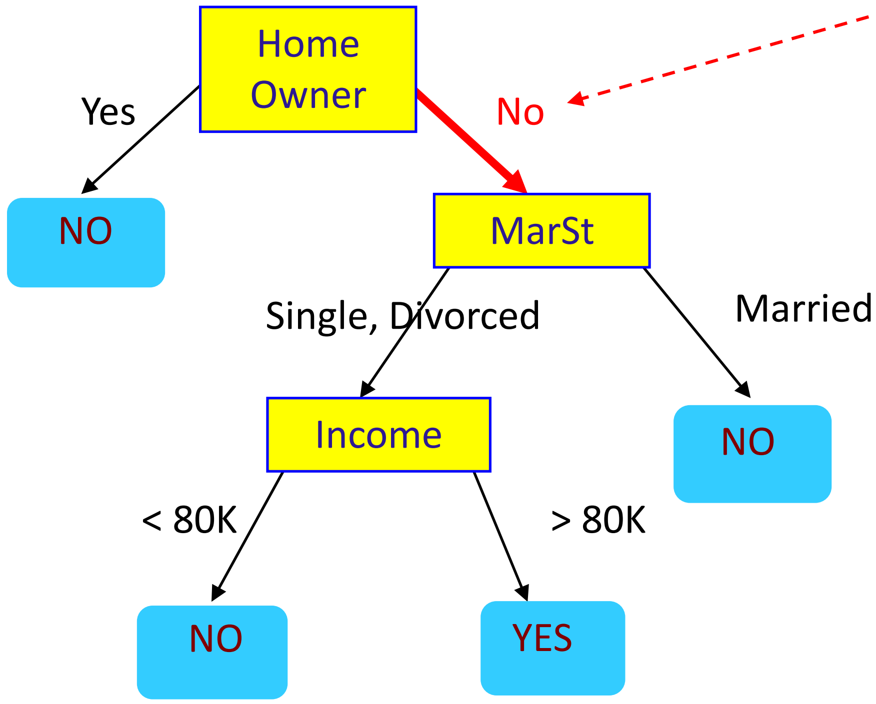
# Apply Model to Test Data




Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | **?** |

# Apply Model to Test Data



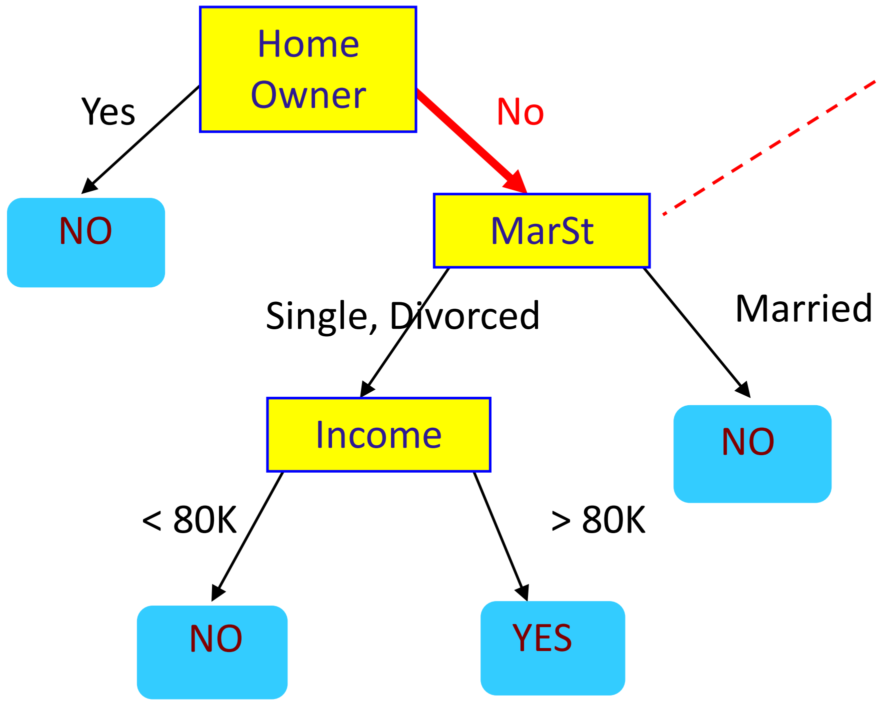

Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Test Data

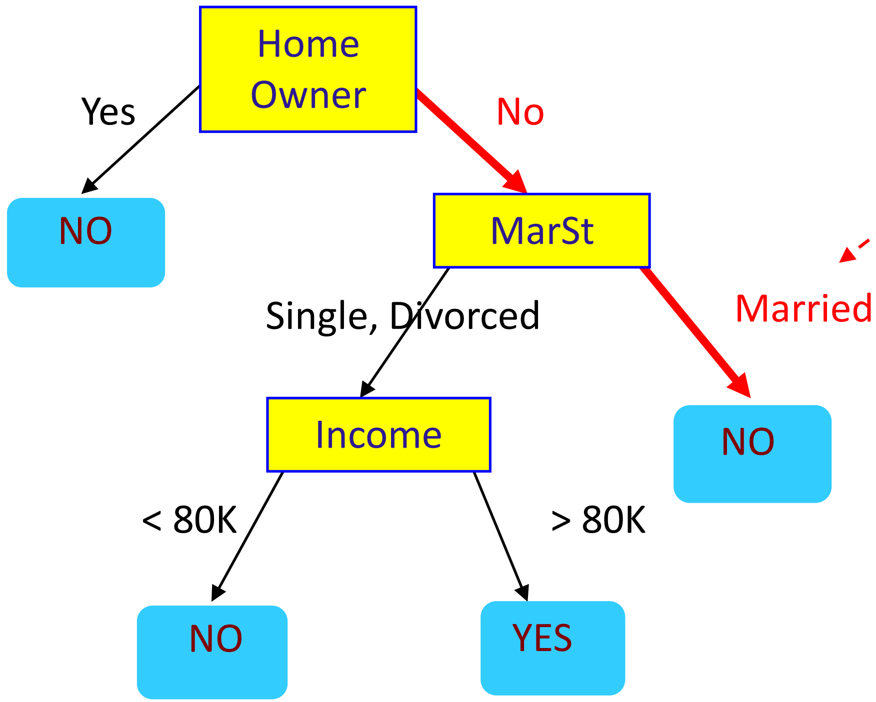
| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | ? |



# Apply Model to Test Data

Test Data

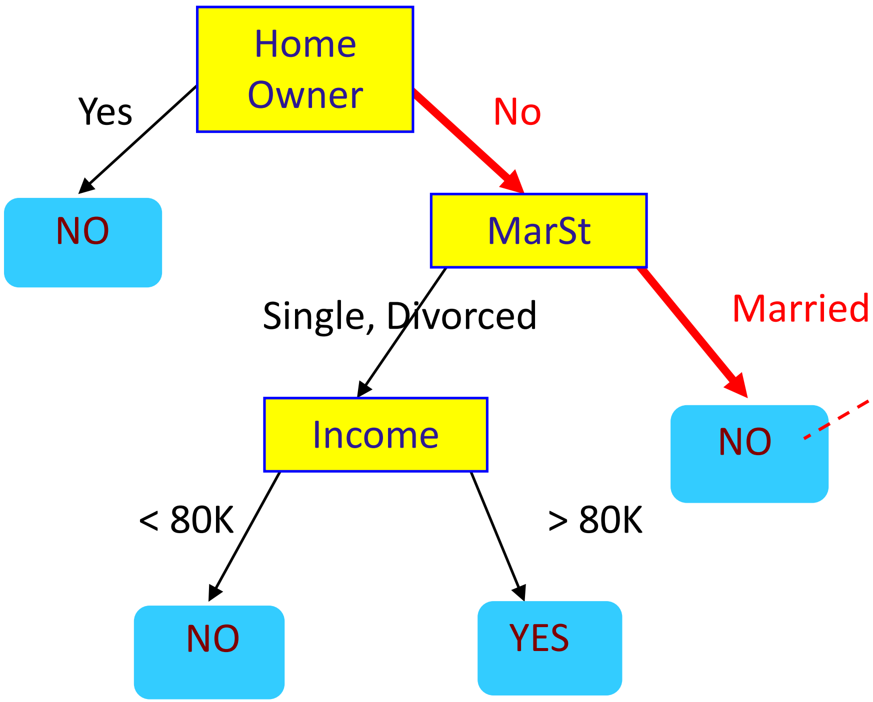
| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | ? |



# Apply Model to Test Data

Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|
| No | Married | 80K | ? |

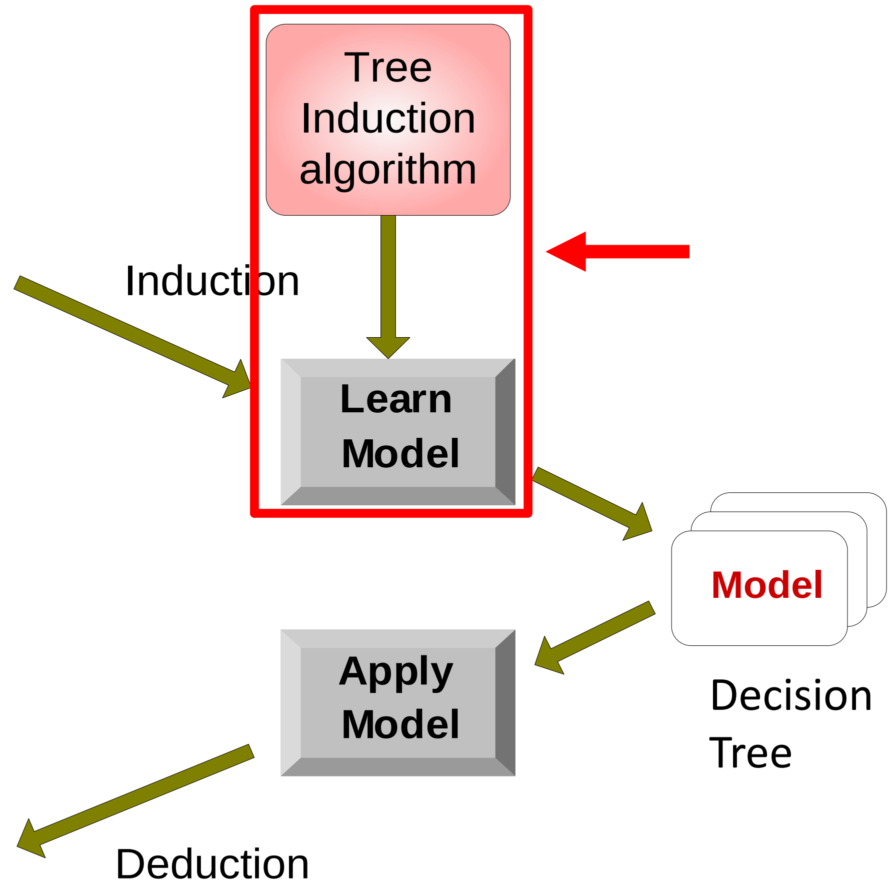



Assign Defaulted to “No”

# 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

# Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Samples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

# Algorithm for Decision Tree Induction (pseudocode)

Algorithm GenDecTree(Sample S, Attlist A)

1. create a node N
2. If all samples are of the same class C then label N with C; terminate;
3. If A is empty then label N with the most common class C in S (majority voting); terminate;
4. Select $a \in A$, with the highest information gain; Label N with a;
5. For each value v of a:
    a. Grow a branch from N with condition a=v;
    b. Let $S_v$ be the subset of samples in S with a=v;
    c. If $S_v$ is empty then attach a leaf labeled with the most common class in S;
    d. Else attach the node generated by GenDecTree($S_v$, A-a)

# Attribute Selection Measure: Information Gain (ID3)

- Select the attribute with the highest information gain
- Let $p_i$ be the probability that an arbitrary tuple in D (data set) belongs to class $C_i$, estimated by $|C_{i, D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

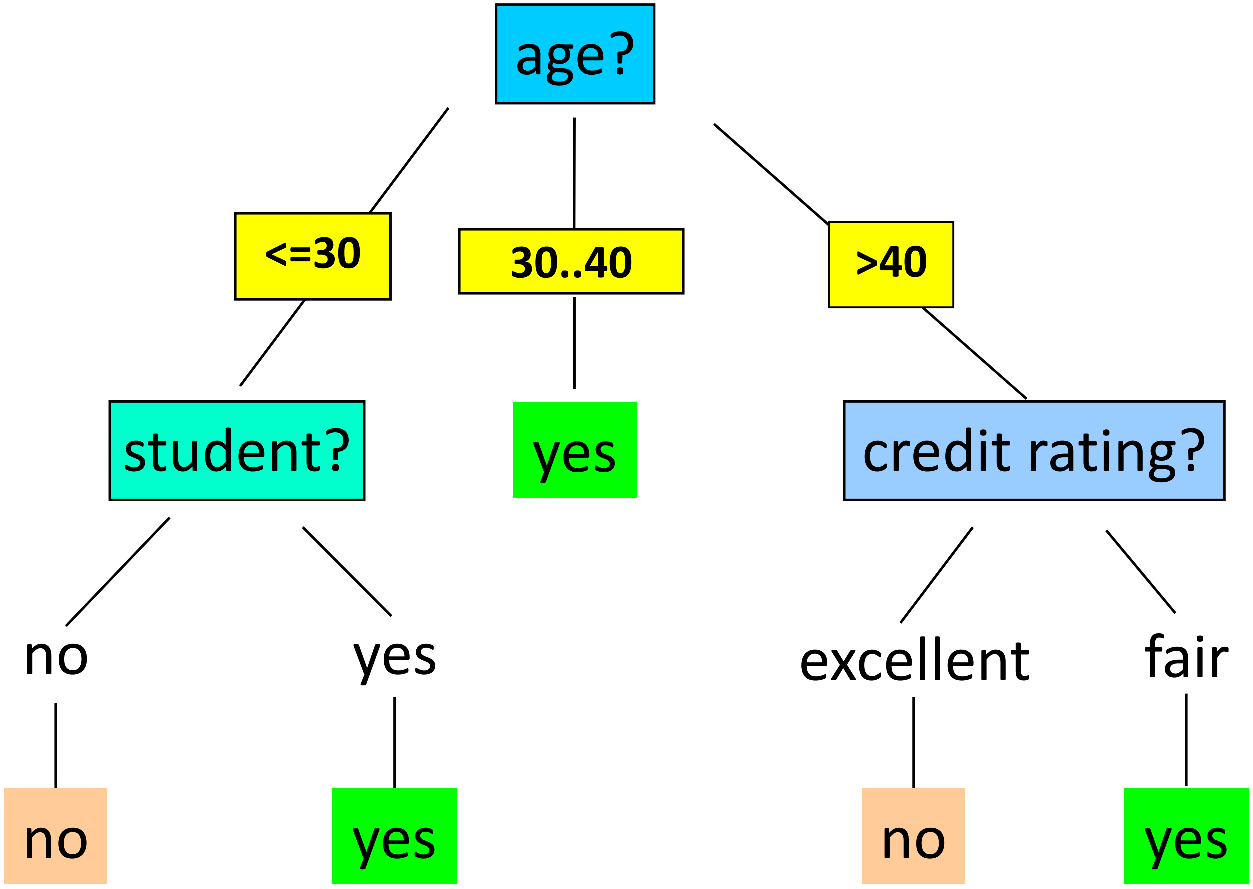
- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

# Input: Training Dataset

| 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 |

# Output: A Decision Tree for "*buys_computer*"



# Attribute Selection: Information Gain

$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

| age | $p_i$ | $n_i$ | $I(p_i, n_i)$ |
|---|---|---|---|
| <=30 | 2 | 3 | 0.971 |
| 31…40 | 4 | 0 | 0 |
| >40 | 3 | 2 | 0.971 |

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

| 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 |

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

# Splitting the samples using age

- Because age has the highest information gain among the attributes, it is selected as the splitting attribute

age?

<=30

| income | student | credit_rating | buys_computer |
|---|---|---|---|
| high | no | fair | no |
| high | no | excellent | no |
| medium | no | fair | no |
| low | yes | fair | yes |
| medium | yes | excellent | yes |

30...40

| income | student | credit_rating | buys_computer |
|---|---|---|---|
| high | no | fair | yes |
| low | yes | excellent | yes |
| medium | no | excellent | yes |
| high | yes | fair | yes |

labeled yes

>40

| income | student | credit_rating | buys_computer |
|---|---|---|---|
| medium | no | fair | yes |
| low | yes | fair | yes |
| low | yes | excellent | no |
| medium | yes | fair | yes |
| medium | no | excellent | no |

# Over-fitting and Tree Pruning

- Over-fitting: An induced tree may over-fit the training data
  - Good accuracy on training data but poor on test data
  - Symptoms: Too many branches, some may reflect anomalies due to noise or outliers
  - Results in Poor accuracy for unseen samples
- Two approaches to avoid over-fitting
  - Pre-pruning: Halt tree construction early—do not split a node if this would result in the goodness measure(Information gain) falling below a threshold
    - Difficult to choose an appropriate threshold
  - Post-pruning: Remove branches from a “fully grown” tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the “best pruned tree”

# Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Robust to noise (especially when methods to avoid over-fitting are employed)
  - Can easily handle redundant or irrelevant attributes (unless the attributes are interacting)

# Decision Tree Based Classification

- Disadvantages:
  - Space of possible decision trees is exponentially large. Greedy approaches are often unable to find the best tree.
  - Does not take into account interactions between attributes
  - Each decision boundary involves only a single attribute

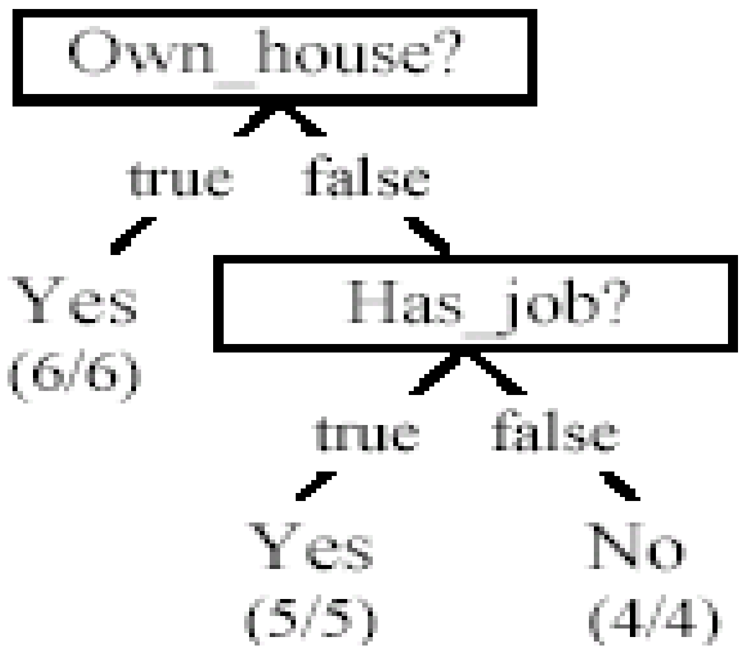
# Class work

- Study the table given below and construct a decision tree based on the greedy algorithm using information gain.

| ID | Age | Has_Job | Own_House | Credit_Rating | Class |
|---|---|---|---|---|---|
| 1 | young | false | false | fair | **No** |
| 2 | young | false | false | excellent | **No** |
| 3 | young | true | false | good | **Yes** |
| 4 | young | true | true | good | **Yes** |
| 5 | young | false | false | fair | **No** |
| 6 | middle | false | false | fair | **No** |
| 7 | middle | false | false | good | **No** |
| 8 | middle | true | true | good | **Yes** |
| 9 | middle | false | true | excellent | **Yes** |
| 10 | middle | false | true | excellent | **Yes** |
| 11 | old | false | true | excellent | **Yes** |
| 12 | old | false | true | good | **Yes** |
| 13 | old | true | false | good | **Yes** |
| 14 | old | true | false | excellent | **Yes** |
| 15 | old | false | false | fair | **No** |

# Contd..

We build the final tree



# Home work

- Study the table given below and construct a decision tree based on the greedy algorithm using information gain.

| Day | Outlook | Temperature | Humidity | Wind | Play Tennis |
|---|---|---|---|---|---|
| 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 |