Data Science – Semester 5 – Fall 2020/2021

INTRODUCTION TO DATA MINING & WAREHOUSING

Lecture 5
Classification: Basic Concepts and Techniques (1)

Supervised vs. Unsupervised 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
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Classification: Definition

- **Given a collection of records (training set)**
  - Each record is characterized by a tuple ($\boldsymbol{x}$,y), where $\boldsymbol{x}$ is the attribute set and y is the class label
    - $\boldsymbol{x}$: attribute, predictor, independent variable, input, feature
    - y: class, response, dependent variable, output
- **Task:**
  - Learn a model that maps each attribute set $\boldsymbol{x}$ into one of the predefined class labels y

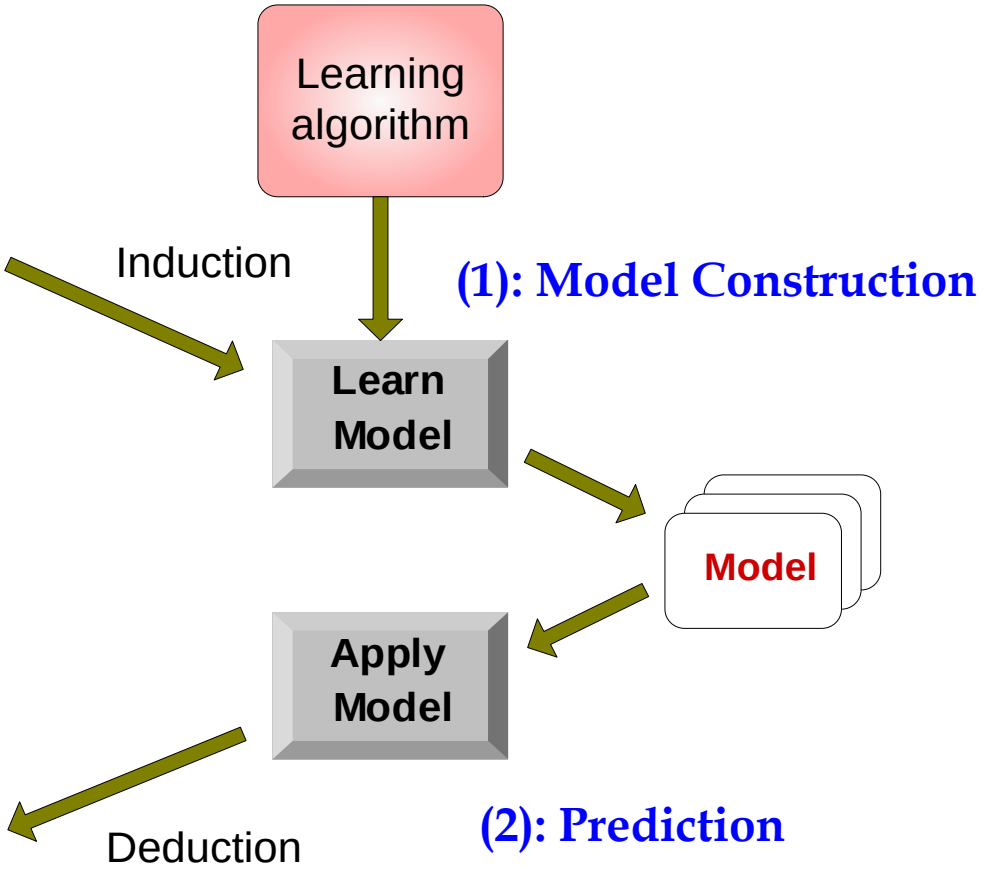
General Approach for Building Classification Model

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

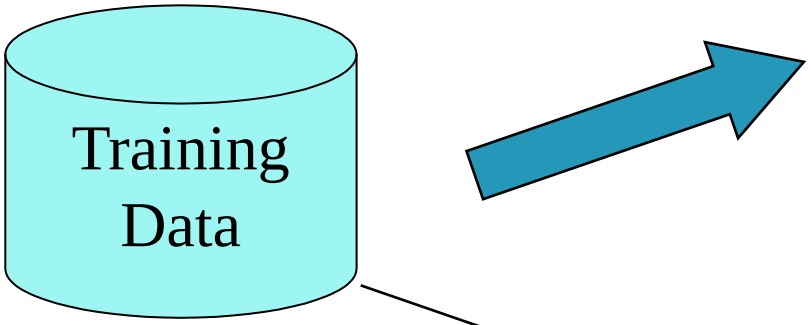


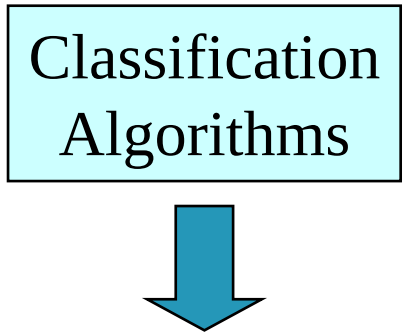
Classification—A Two-Step Process

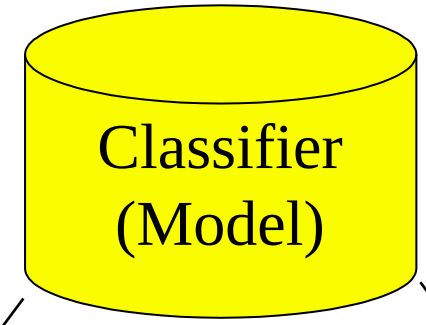
- **Model construction:** describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is **training set**
  - The model is represented as classification rules,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

- **Model usage:** for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample 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
    - **Test set** is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data

Process (1): Model Construction



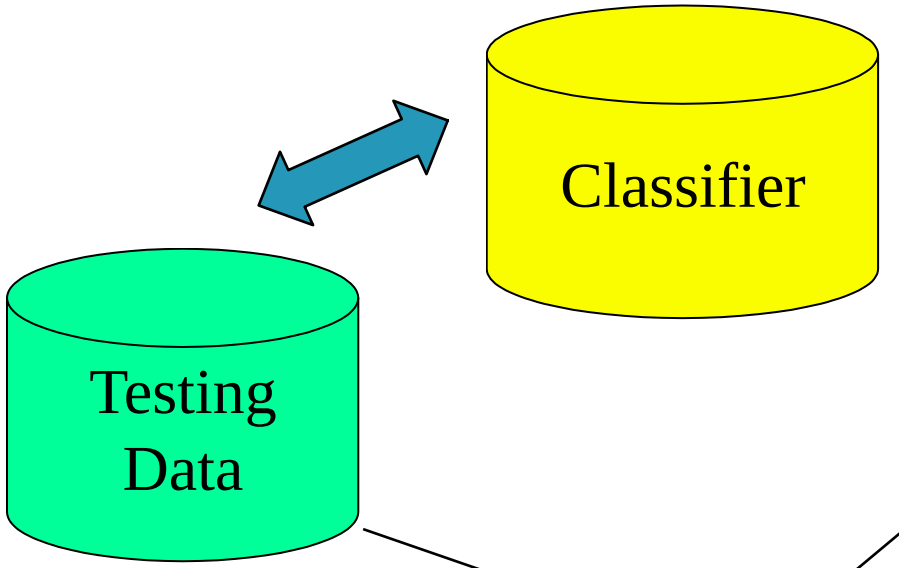




NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

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

Process (2): Using the Model in Prediction



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

(Jeff, Professor, 4)

Tenured?

Yes

Classification vs. Regression

- Classification
  - predicts categorical class labels (discrete or nominal)
  - 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/loan approval:
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is

Classification Techniques

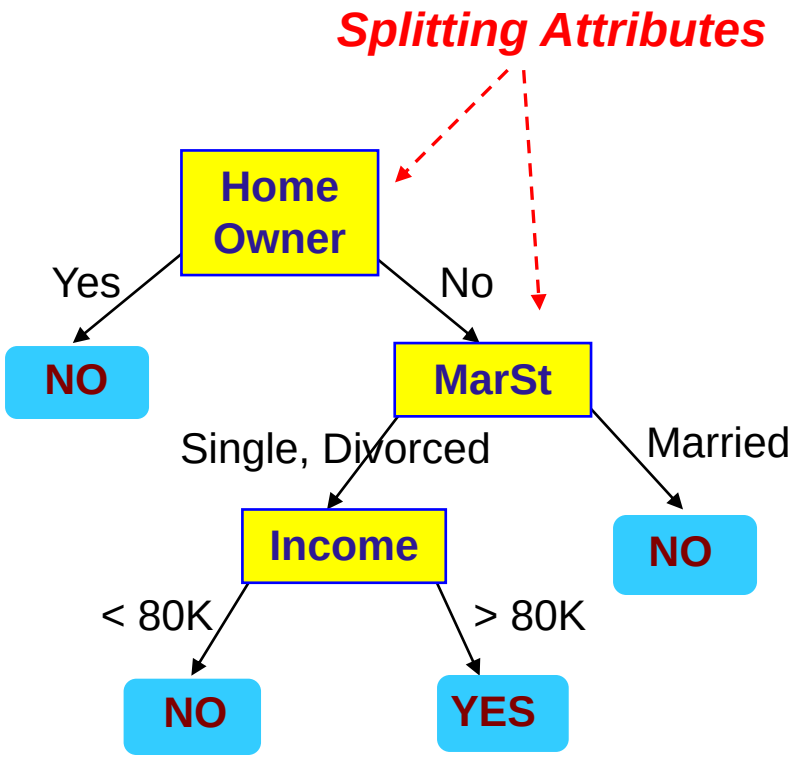
- **Base Classifiers**
  - **Decision Tree based Methods**
  - Rule-based Methods
  - Nearest-neighbor
  - Neural Networks, Deep Neural Nets
  - Naïve Bayes and Bayesian Belief Networks
  - Support Vector Machines

- **Ensemble Classifiers**
  - Boosting, Bagging, Random Forests

Example of a Decision Tree

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
	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

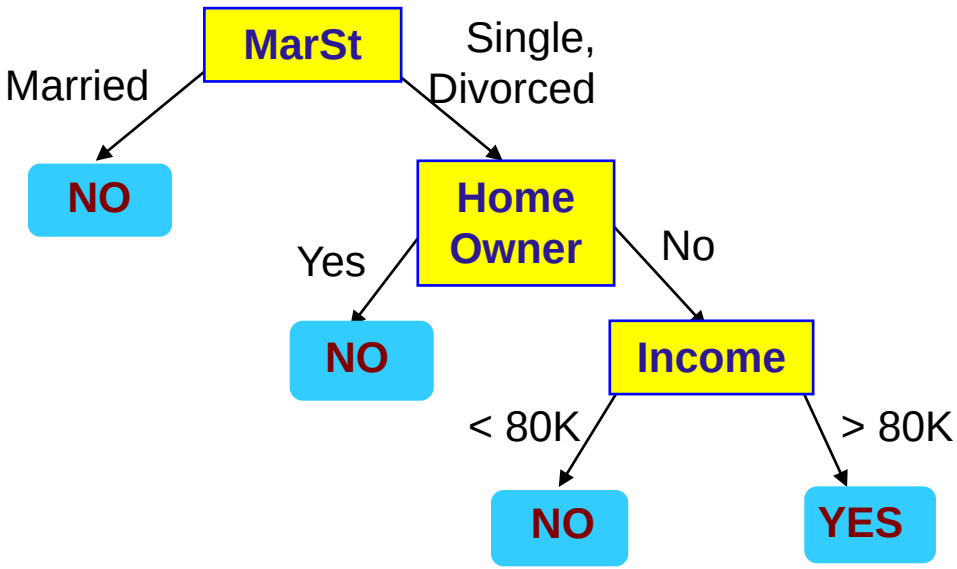
Training Data



Model: Decision Tree

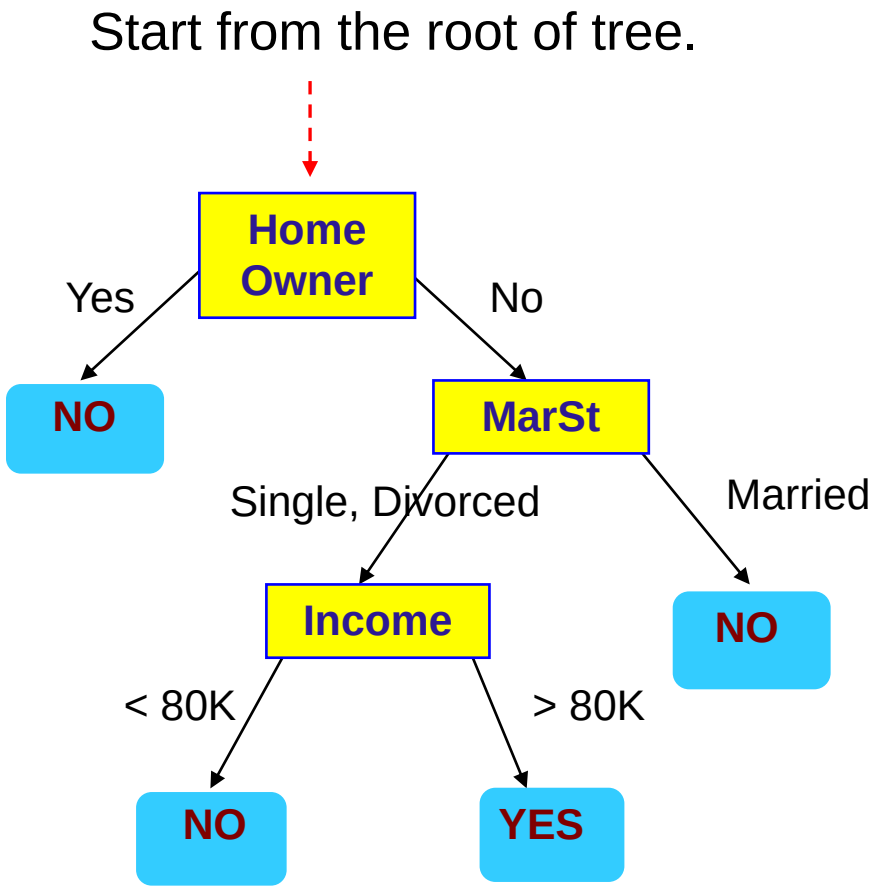
Another Example of Decision Tree

	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



There could be more than one tree that fits the same data! Which is the best?

Apply Model to Test Data



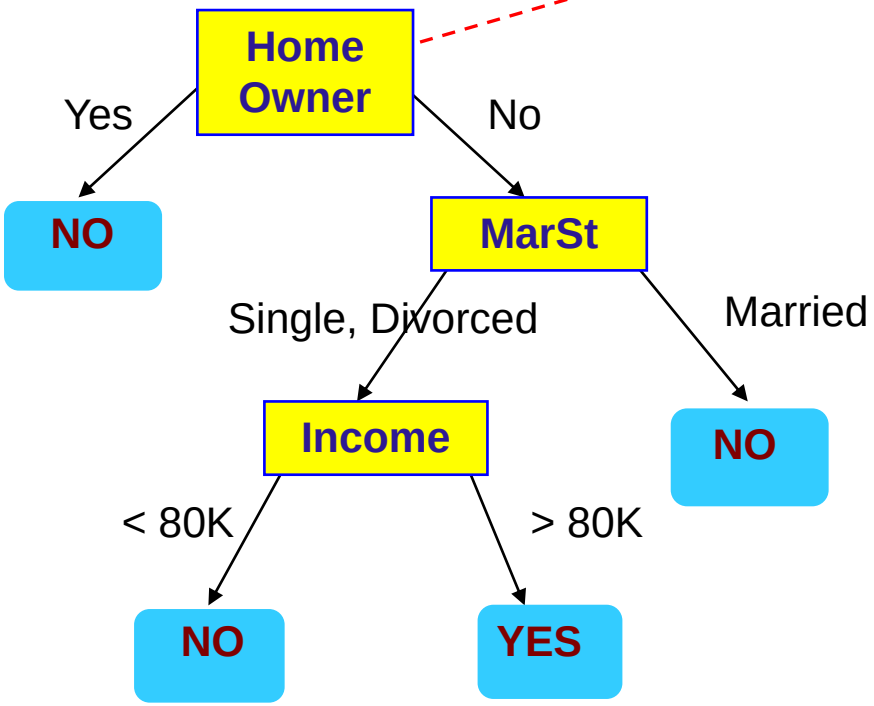
Test Data

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

Apply Model to Test Data

Test Data

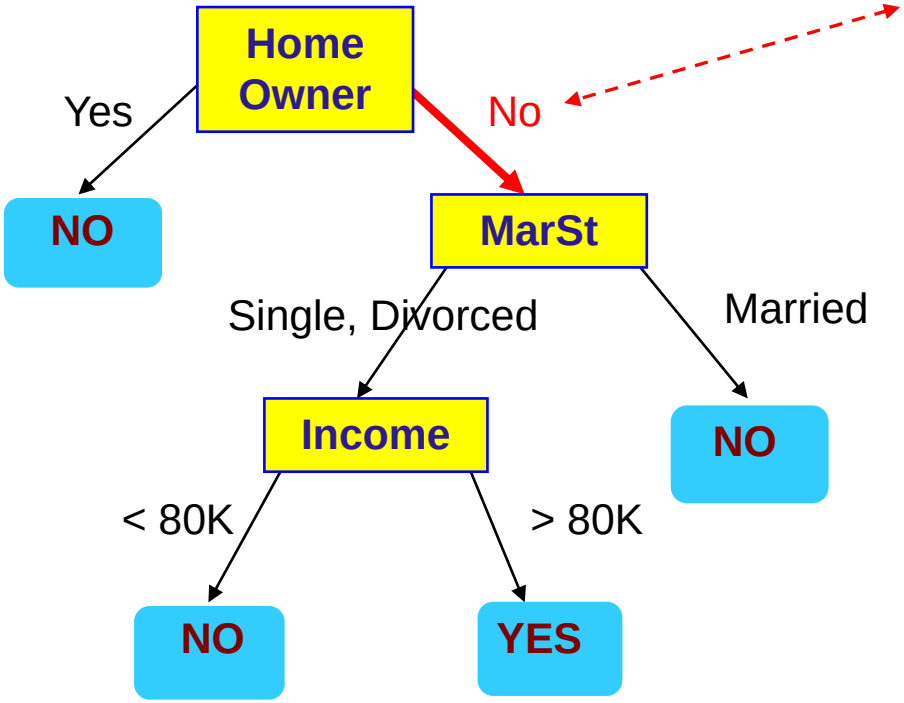
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



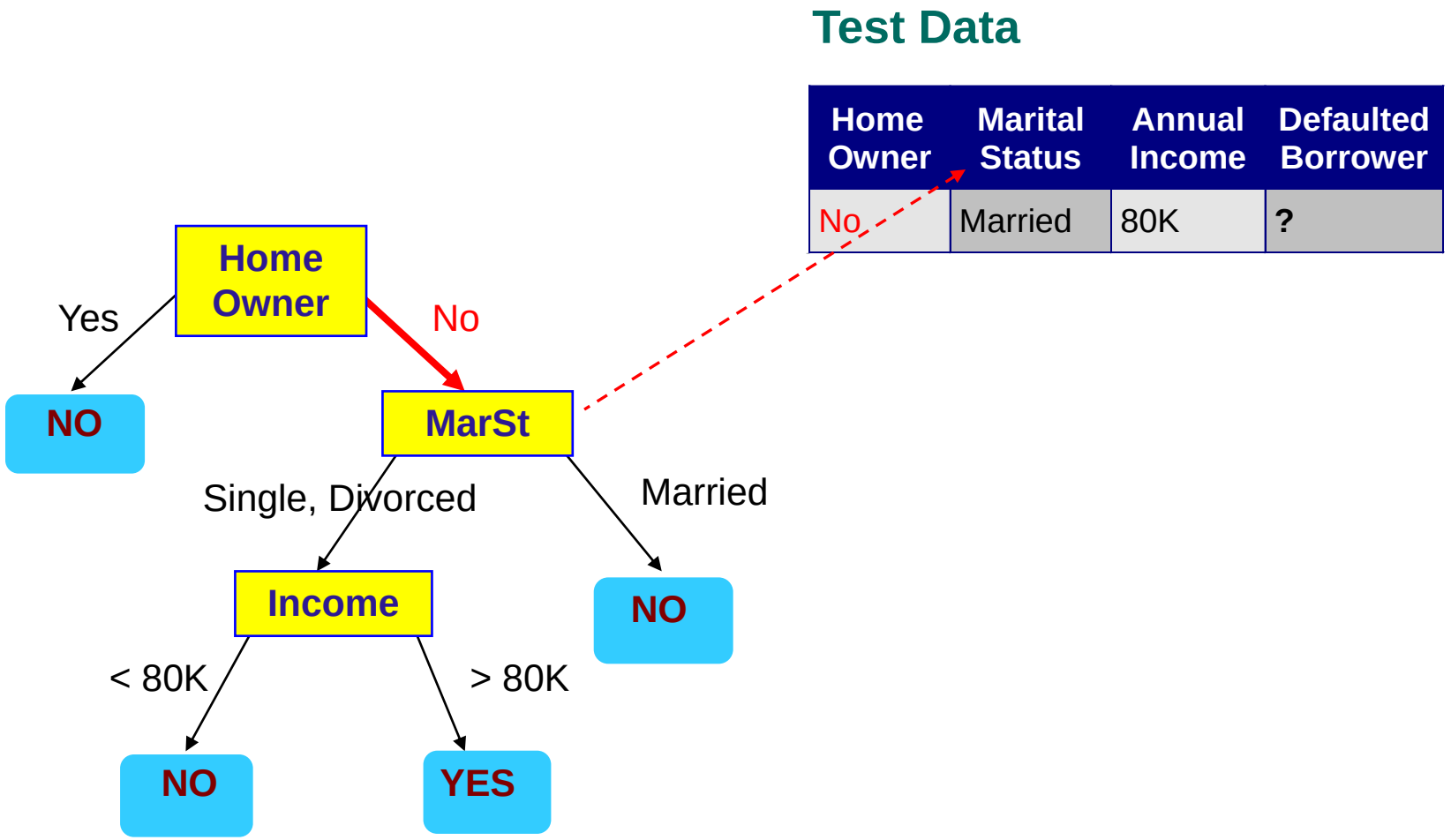
Apply Model to Test Data

Test Data

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



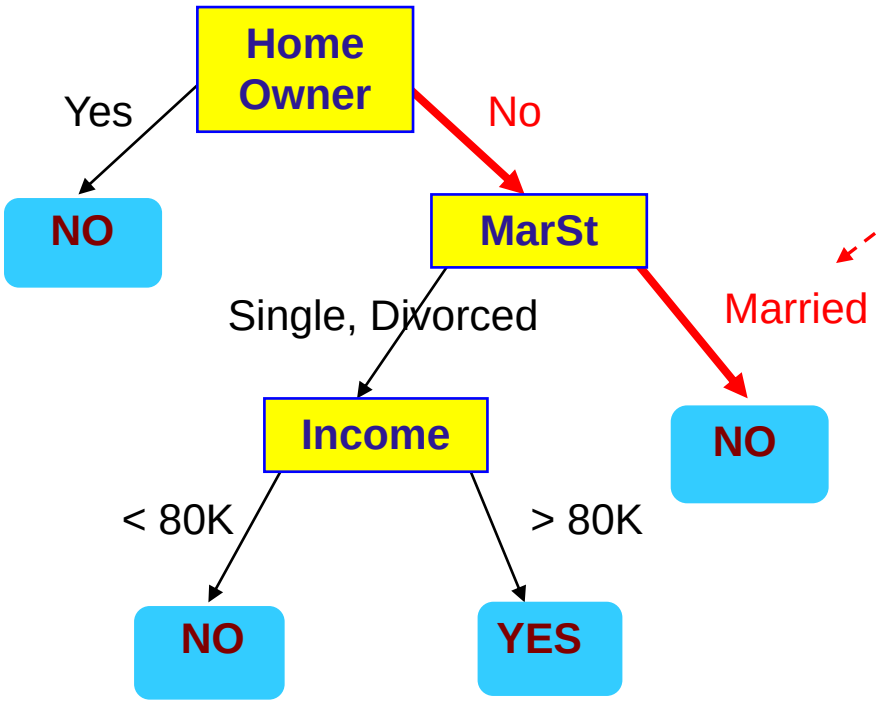
Apply Model to Test Data



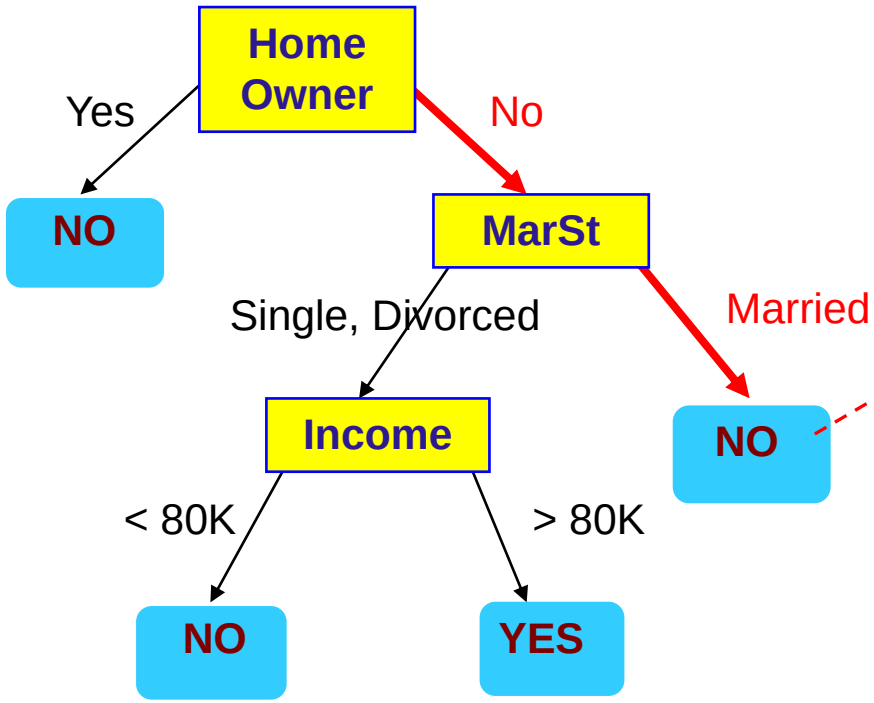
Apply Model to Test Data

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"

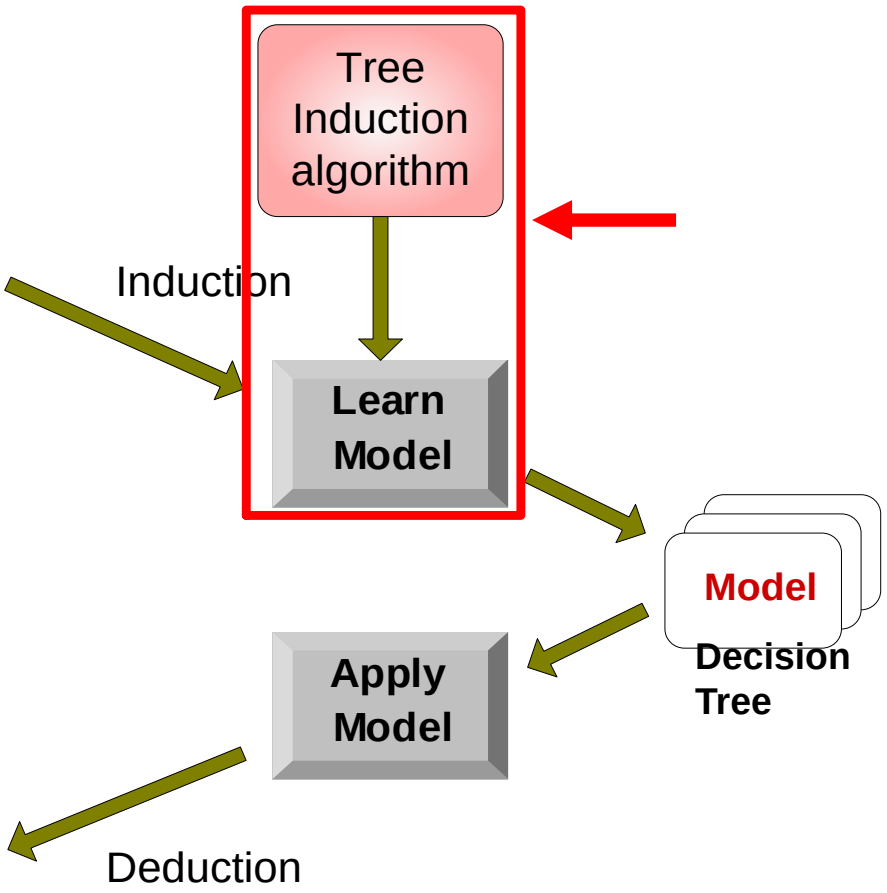
Decision Tree Classification Task

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



Decision Tree Induction

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

Algorithm for Decision Tree Induction

- **Basic algorithm (a greedy algorithm)**
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Base case: If all data belong to the same class, create a leaf node with that label
  - Otherwise:
    - calculate the "**score**" for each feature if we used it to split the data
    - pick the feature with the highest score, partition the data based on that data value and call recursively

- **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

Example

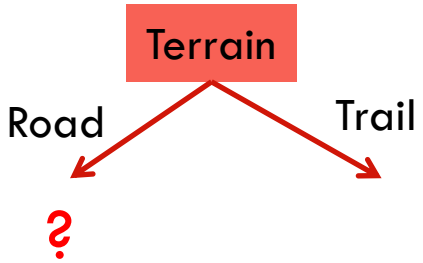
- Training data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Build a decision tree

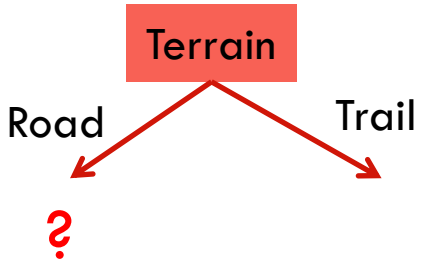
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



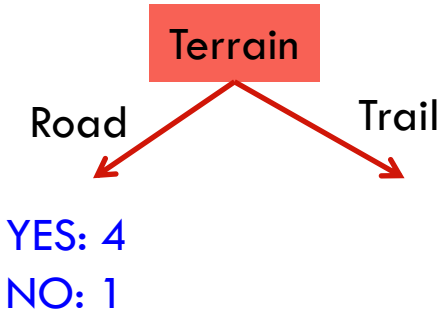
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



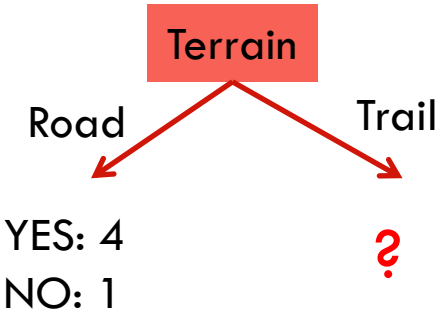
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



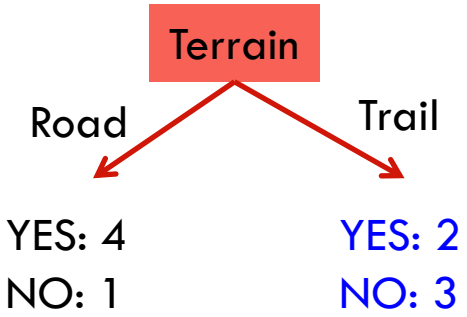
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



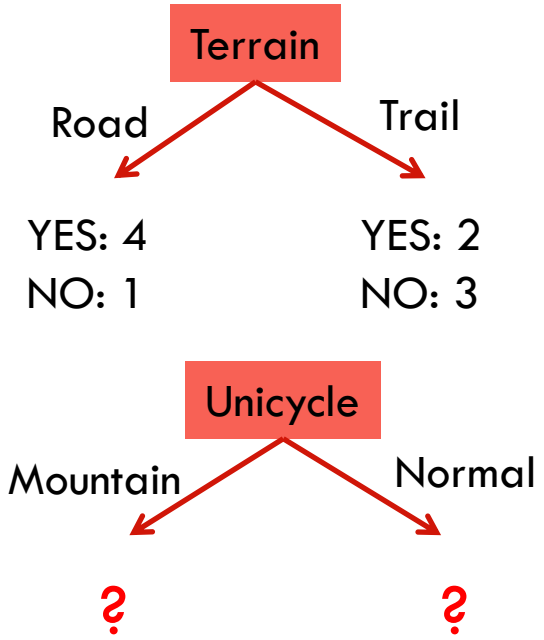
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



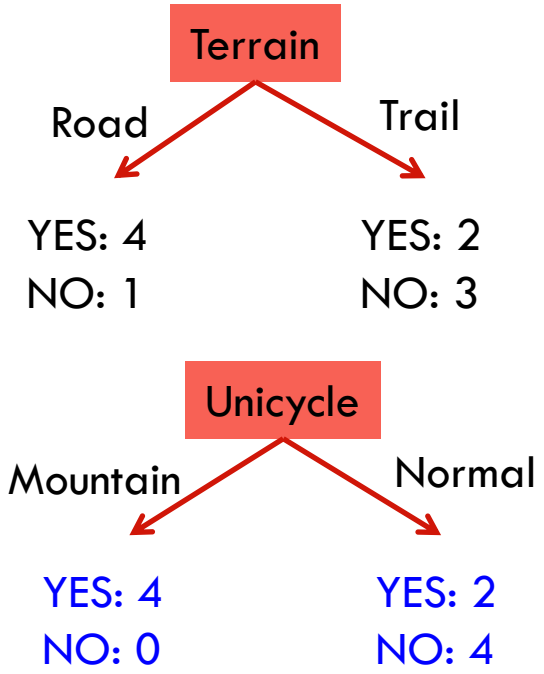
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



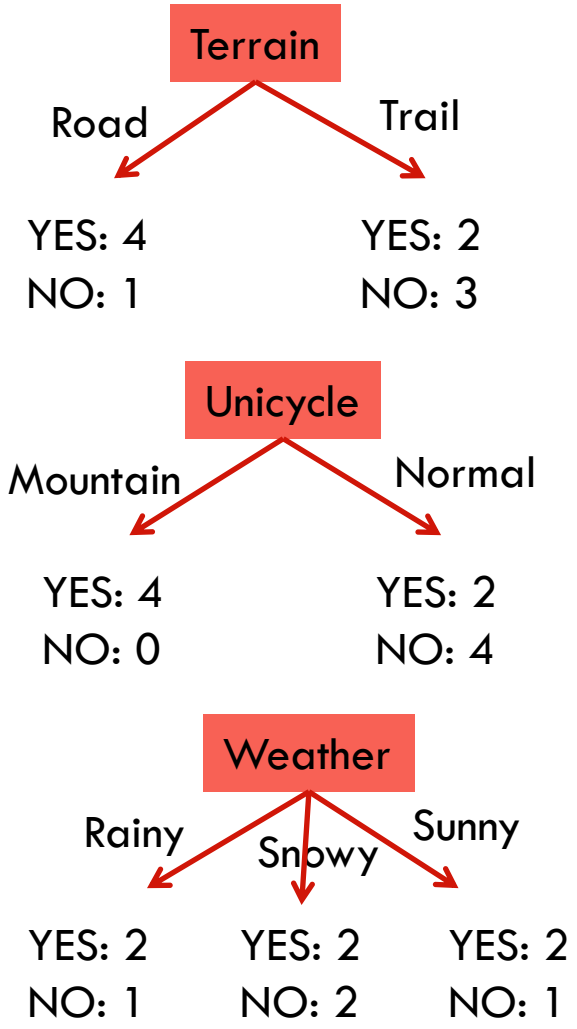
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



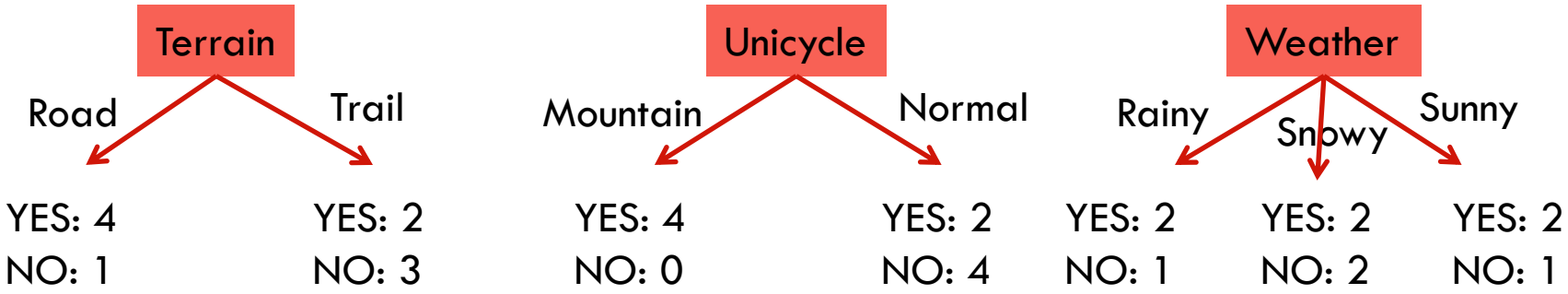
Splitting the data

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES



Splitting the data

- On which attribute to split first?



calculate the "score" for each feature
if we used it to split the data

What score should we use?

Splitting the data – What score should we use?

- **Many variants:**
  - from machine learning: ID3 (Iterative Dichotomizer), C4.5 (Quinlan 86, 93)
  - from statistics: CART (Classification and Regression Trees) (Breiman et al 84)
  - from pattern recognition: CHAID (Chi-squared Automated Interaction Detection) (Magidson 94)

- **Main difference: divide (split) criterion**
  - Which attribute to test at each node in the tree ? The attribute that is most useful for classifying examples

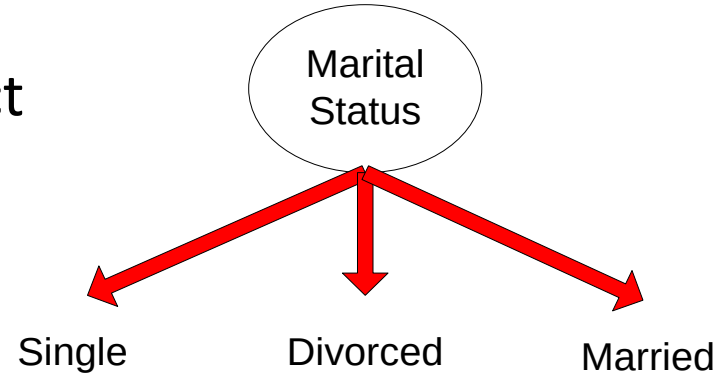
Split criterion

- Depends on attribute types
  - Binary
  - Nominal
  - Ordinal
  - Continuous

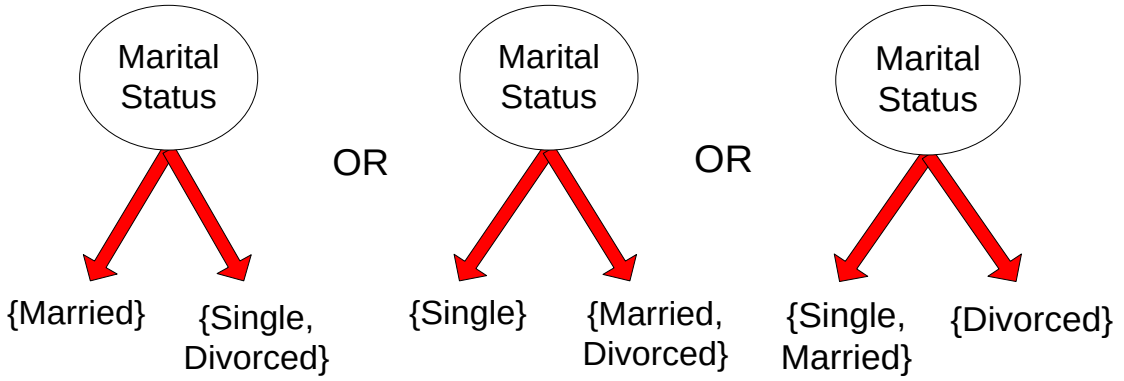
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Split Criterion for Nominal Attributes

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

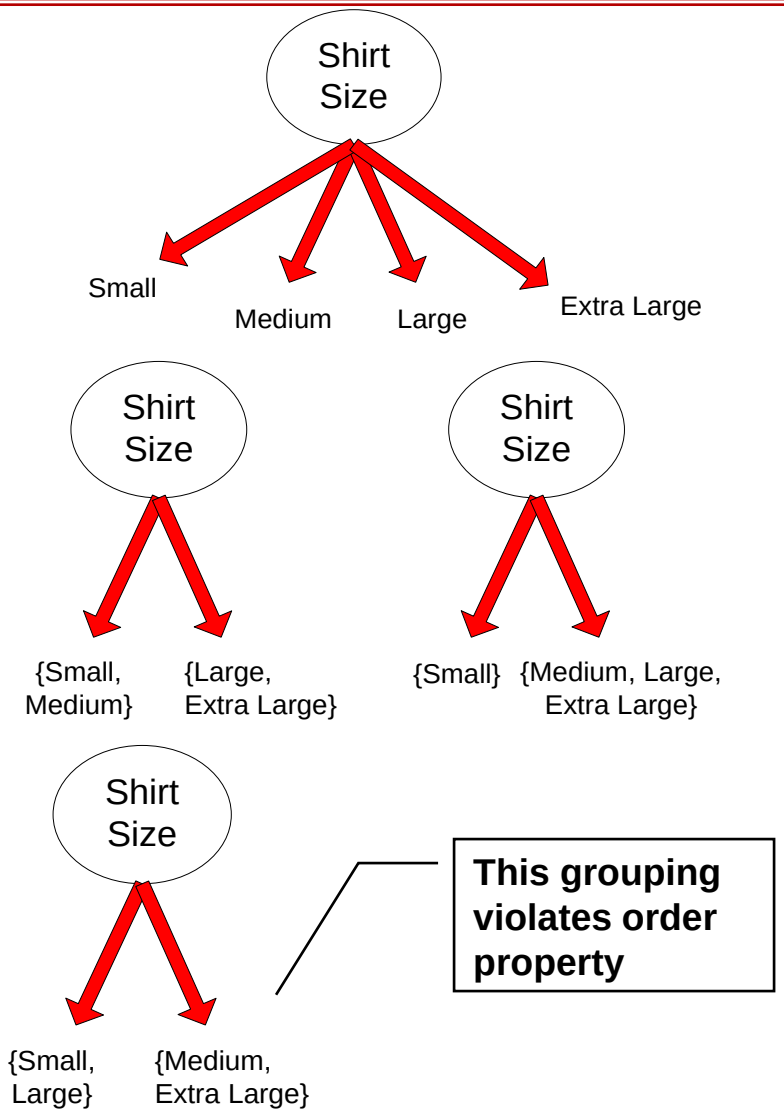


- Binary split:
  - Divides values into two subsets

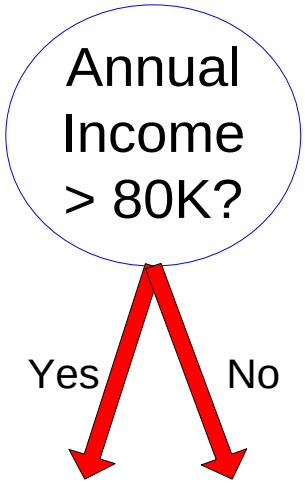


Split Criterion for Ordinal Attributes

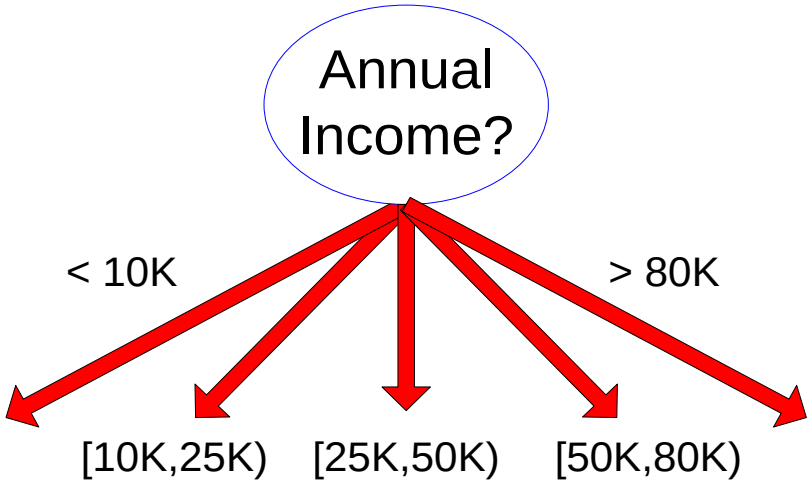
- Multi-way split:
  - Use as many partitions as distinct values
- Binary split:
  - Divides values into two subsets
  - Preserve order property among attribute values



Split Criterion for Continuous Attributes



(i) Binary split



(ii) Multi-way split

Binary Decision: (A < v) or (A > v)

- consider all possible splits and finds the best cut
- can be more compute intensive

Discretization to form an ordinal categorical attribute

Split criterion

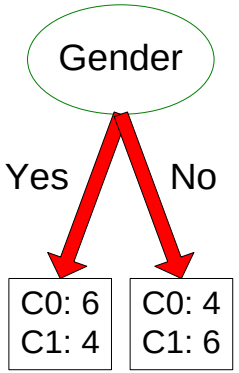
- **Information gain (based on Entropy)**
  - All attributes are assumed to be categorical (ID3)
  - Can be modified for continuous-valued attributes (C4.5)

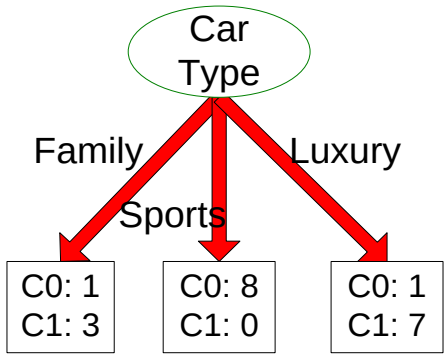
- **Gini index (CART, IBM IntelligentMiner)**
  - All attributes are assumed continuous-valued
  - Assume there exist several possible split values for each attribute
  - Can be modified for categorical attributes

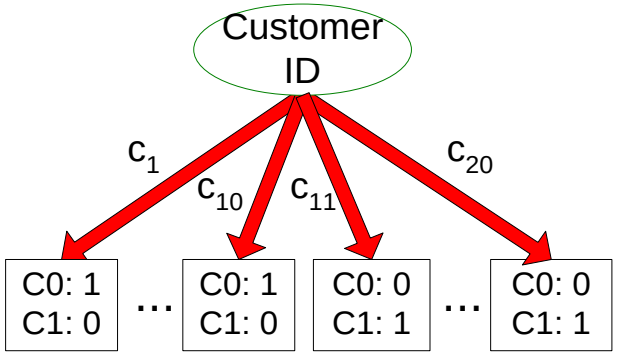
How to determine the Best Split

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

Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1







Which test condition is the best?

How to determine the Best Split

- Greedy approach:
  - Nodes with purer class distribution are preferred
- Entropy and Gini index are measures of node impurity

C0: 5 C1: 5	C0: 9 C1: 1
High degree of impurity	Low degree of impurity

Measures of Node Impurity

- Gini Index

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

Where $\boldsymbol{p_i(t)}$ is the frequency of class $\boldsymbol{i}$ at node t, and $\boldsymbol{c}$ is the total number of classes

- Entropy

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

- Misclassification error

$$Classification\ error = 1 - \max[p_i(t)]$$

Finding the Best Split

1. Compute impurity measure (B) before splitting
2. Compute impurity measure (A) after splitting
   - » Compute impurity measure of each child node
   - » A is the weighted impurity of child nodes

3. Choose the attribute test condition that produces the highest gain

   Gain = B - A

   or equivalently, lowest impurity measure after splitting (A)

Example using Entropy

- Dataset S

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Before split:

- B= {yes^6, no^4}

- E(B)=-p(yes)$\log_2$p(yes) – p(no)$\log_2$p(no)

  = -(6/10)$\log_2$(6/10) – (4/10)$\log_2$(4/10)

  = 0.44 + 0.53 = 0.97

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

Example using Entropy

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

After split

- **Gain of attribute Terrain**

- $Terrain_{trail} = \{ yes^2 , no^3 \}$

$E(Terrain=trail) = -(2/5)\log_2(2/5)-(3/5)\log_2(3/5)$

$= 0.53+0.44=0.97$

- $Terrain_{Road} = \{ yes^4 , no^1 \}$

$E(Terrain=Road) = -(4/5)\log_2(4/5) - (1/5)\log_2(1/5)$

$= 0.26+0.46 = 0.72$

- Average entropy of attribute Terrain:

$E(Terrain) = (5/10)\, E(Terrain=Trail) + (5/10)E(Terrain=Road)$

$= 0.485 + 0.36 = 0.845$

- **Gain(S,Terrain) = E(S) - E(Terrain) = 0.97 – 0.845 = 0.125**

Example using Entropy

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

- **Gain of attribute Unicycle-type**

- Unicycle-Type$_{Normal}$ = { yes^2 , no^4 }

E(Unicycle-Type=Normal)
= -(2/6)log$_2$(2/6) – (4/6)log$_2$(4/6)
= 0.53+0.39=0.92

- Unicycle-Type$_{Moutain}$ = { yes^4 , no^0 }

E(Unicycle-Type=Mountain) = 0 (maximal homogeneity → minimal impurity → minimal entropy)

- Average entropy

E(Unicycle-Type) = (6/10)E(Unicycle-type=Normal) + (4/10)E(Unicycle-type=Mountain) = 0.552

- **Gain(S, Unicycle-type) = E(S) – E(Unicycle-type) = 0.97 – 0.552 = 0.418**

Example using Entropy

- **Gain of attribute Weather**

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

- $Weather_{Rainy} = \{ yes^2 , no^1 \}$

$E(Weather=Rainy) = -(2/3)\log_2(2/3) - (1/3)\log_2(1/3)$
$= 0.39+0.53 = 0.92$

- $Weather_{Sunny} = \{ yes^2 , no^1 \}$

$E(Weather=Sunny) = 0.92$

- $Weather_{Snowy} = \{ yes^2 , no^2 \}$

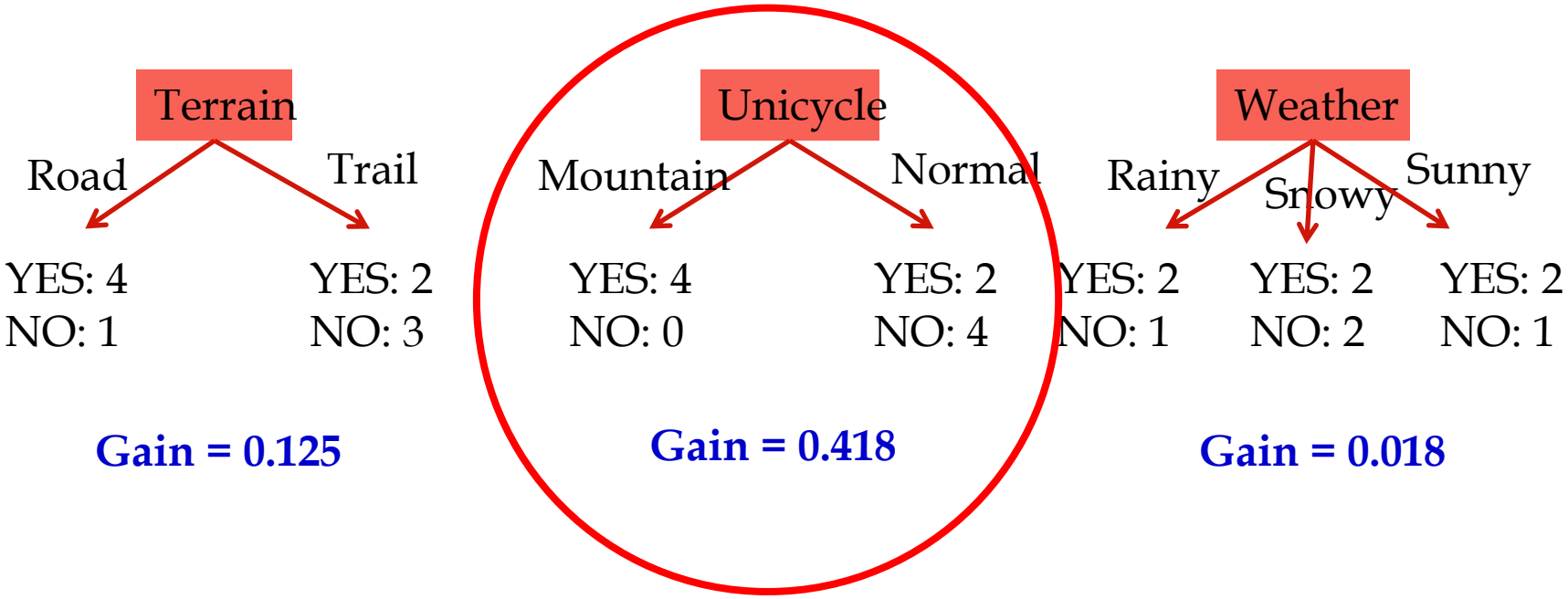
$E(Weather=Snowy) = 1$ (minimal homogeneity → maximal impurity → maximal entropy)

- Average entropy

$E(Weather) = (3/10)*0.92 + (3/10)*0.92 + (4/10)*1$
$= 0.952$

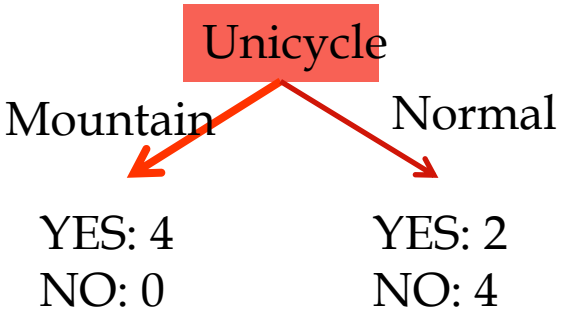
- **Gain(S, Weather) = E(S) – E(Weather) = 0.97 – 0.952 = 0.018**

Example using Entropy

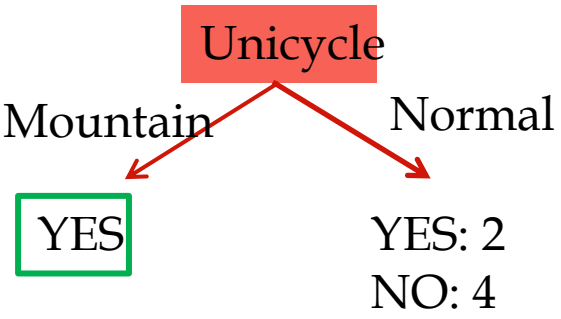


Highest gain Split on this attribute and recurse

Example using Entropy

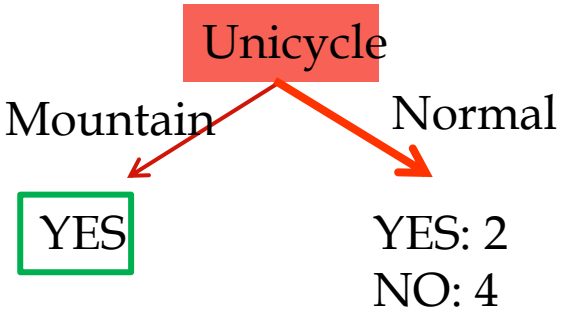


Base case: all data belong to the same class, stop and create a leaf node with that label



Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Example using Entropy



Our new dataset set S filtered on "Normal"

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Before splitting

- B = { yes^2, no^4 }

$E(B) = -(2/6)\log_2(2/6) - (4/6)\log_2(4/6)$

$= 0.53 + 0.39 = 0.92$

Example using Entropy

Our new dataset set S filtered on "Normal"

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

- **Gain of Terrain**

- $Terrain_{trail} = \{ yes^0 , no^3 \}$

E(Terrain=trail) = 0

- $Terrain_{Road} = \{ yes^2 , no^1 \}$
- $E(Terrain=Road) = -(2/3)\log_2(2/3) - (1/3)\log_2(1/3)$
  $= 0.39+0.53= 0.92$

- Average entropy of attribute Terrain:

E(Terrain) = (3/6) E(Terrain=Trail) + (3/6)E(Terrain=Road)
= 0.46

- **Gain(S, Terrain) = E(S) – E(Terrain) = 0.92 – 0.46= 0.46**

Example using Entropy

- **Gain of attribute Weather**

- $Weather_{Rainy} = \{ yes^1 , no^1 \}$

E(Weather=Rainy) = 1

- $Weather_{Sunny} = \{ yes^1 , no^1 \}$

E(Weather=Sunny) = 1

- $Weather_{Snowy} = \{ yes^0 , no^2 \}$

E(Weather=Snowy) = 0

- Average Entropy

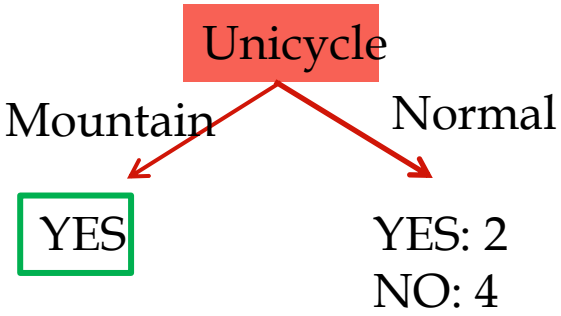
E(Weather) = 4/6 = 0.66

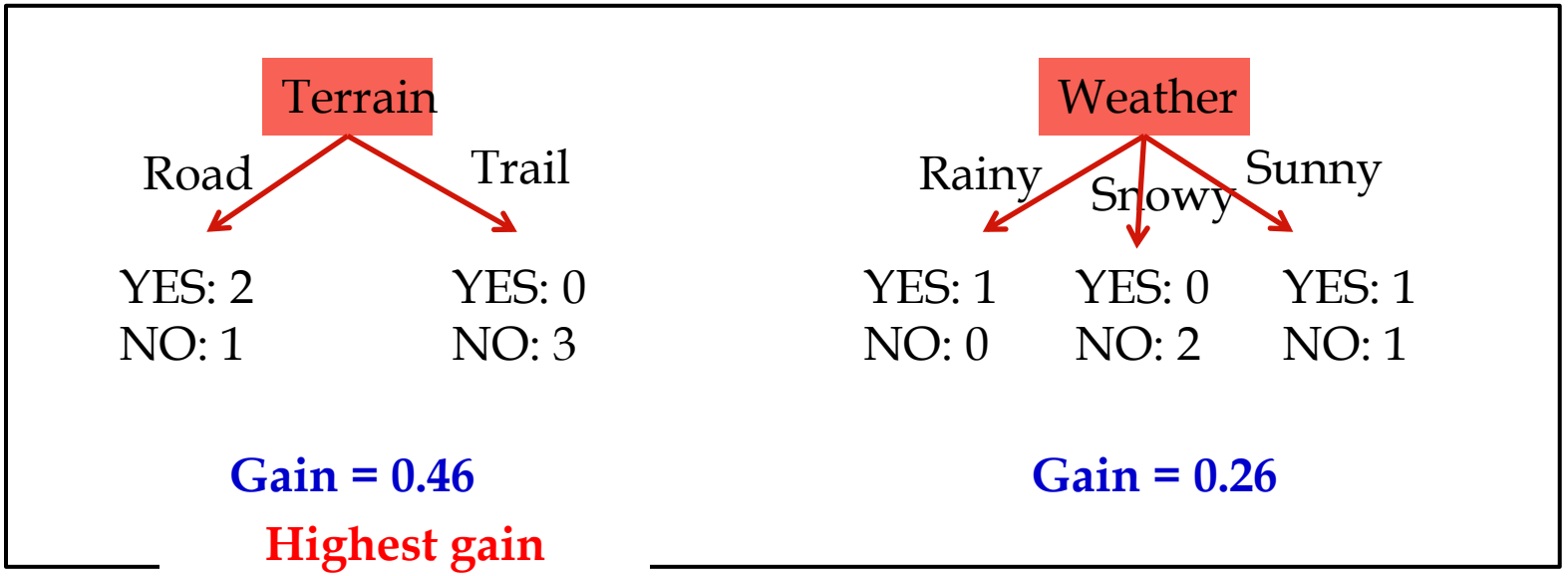
- **Gain(S, Weather) = 0.92 – 0.66 = 0.26**

Our new dataset set S filtered on "Normal"

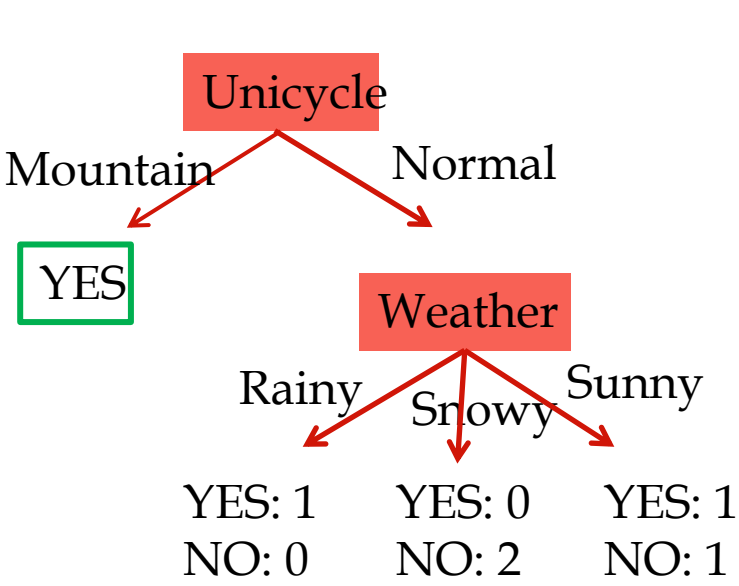
Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

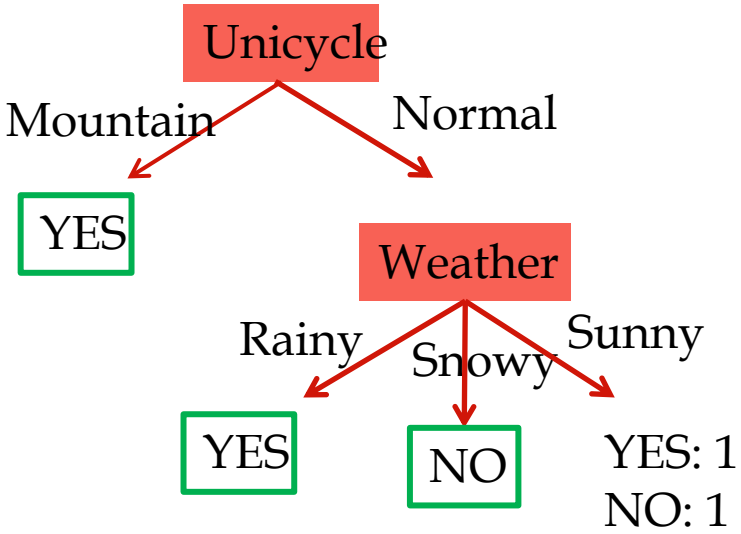
Example using Entropy



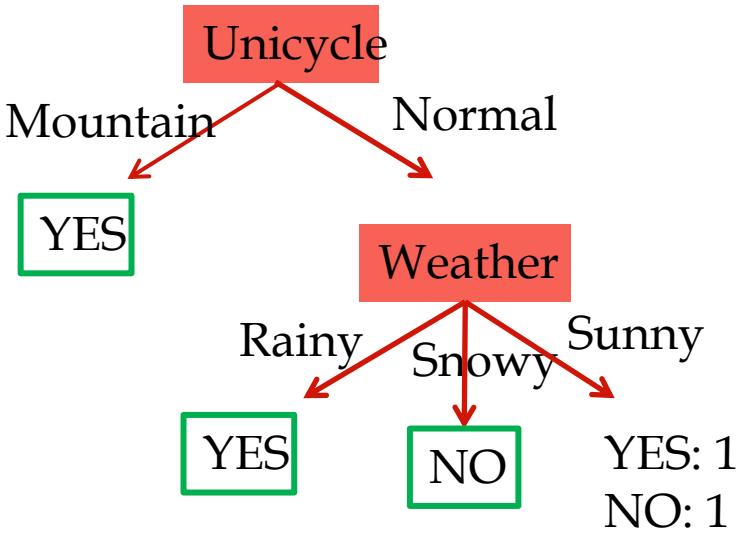


Example using Entropy





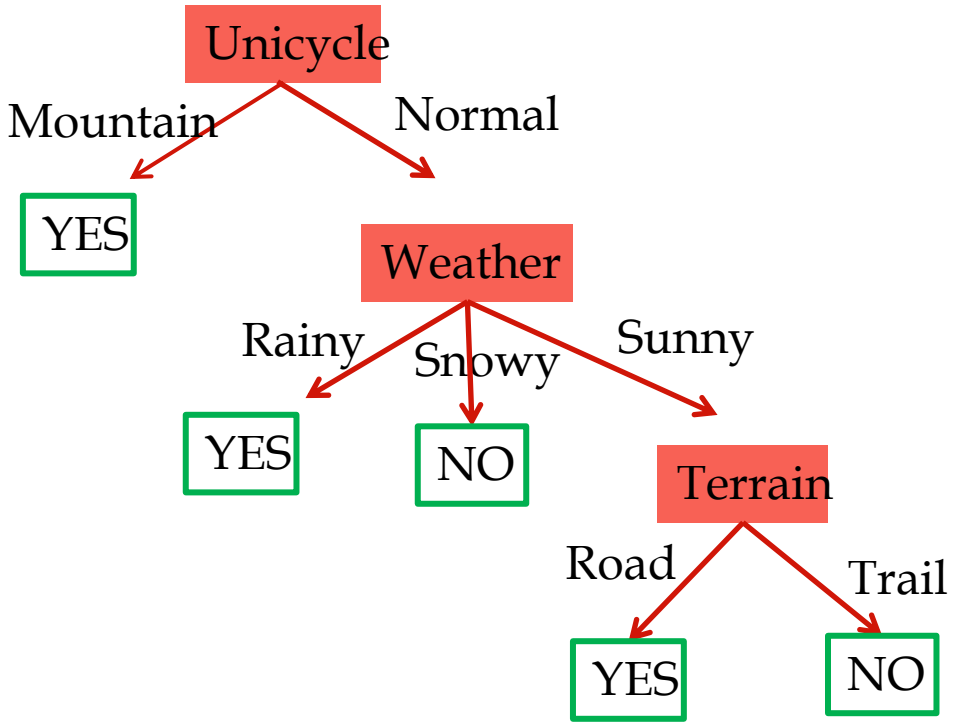
Example using Entropy



Our new dataset set S filtered on "Normal" and "Sunny"

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Example using Entropy



Our new dataset set S filtered on "Normal" and "Sunny"

Terrain	Unicycle-type	Weather	Go-For-Ride?
Trail	Normal	Rainy	NO
Road	Normal	Sunny	YES
Trail	Mountain	Sunny	YES
Road	Mountain	Rainy	YES
Trail	Normal	Snowy	NO
Road	Normal	Rainy	YES
Road	Mountain	Snowy	YES
Trail	Normal	Sunny	NO
Road	Normal	Snowy	NO
Trail	Mountain	Snowy	YES

Exercise

Build a decision tree that splits the data set based on just one predictor variable. Take as response variable the attribute *Car*, i.e., we are interested to learn what influences the type of car most.

Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

Exercise

Family = F, Sport = S, None = N

Before split:

- B= {F^8, S^4, N^4}

- E(S) = -(8/16 log(8/16) + 4/16 log(4/16) + 4/16 log(4/16))= 0.5+0.5+0.5 = 1.5

Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

Exercise

After split: Gain of Income

- $Income_{Low}$: $\{F^4, S^0, N^2\}$

E(Income=Low) = -(4/6 log(4/6) + 2/6 lo(2/6)) = 0.39+0.53 = 0.92

- $Income_{Medium}$: $\{F^2, S^0, N^2\}$

E(Income = medium) = 1

- $Income_{High}$: $\{F^2, S^4, N^0\}$

E(Income=High) = -(2/6 log(2/6) + 4/6 lo(4/6)) = 0.53+0.39 = 0.92

Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

- **Average entropy:**

E(Income) = 6/16*E(Income=low) + 4/16*E(Income=Medium) + 6/16*E(Income=High)= 0.345 + 0.25 + 0.345 = 0.94

IG(B, Income) = 1.5 – 0.94 = 0.56

Exercise

After split: Gain of NbChildren

- $NBC_{2 \text{ or more}}$: $\{F^8, S^0, N^2\}$

E(NBC = 2 or more) = -(8/10 log(8/10) + 2/10 lo(2/10)) = 0.26+0.46 = 0.72

- NBC_0: $\{F^0, S^2, N^2\}$

- NBC_1: $\{F^0, S^2, N^0\}$

E(Income=High) = -(2/6 log(2/6) + 4/6 lo(4/6)) = 0.53+0.39 = 0.92

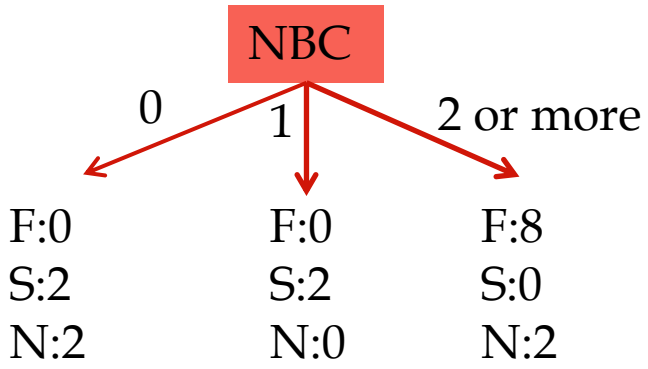
Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

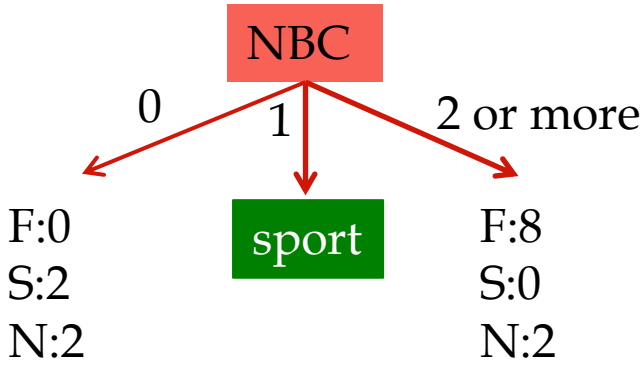
- **Average entropy:**

E(NBC) = 10/16*0.72 + 4/16*1 +2/16*0 = 0.45+0.25=0.7

IG(B, NBC) = 1.5 – 0.7 = 0.8

Exercise

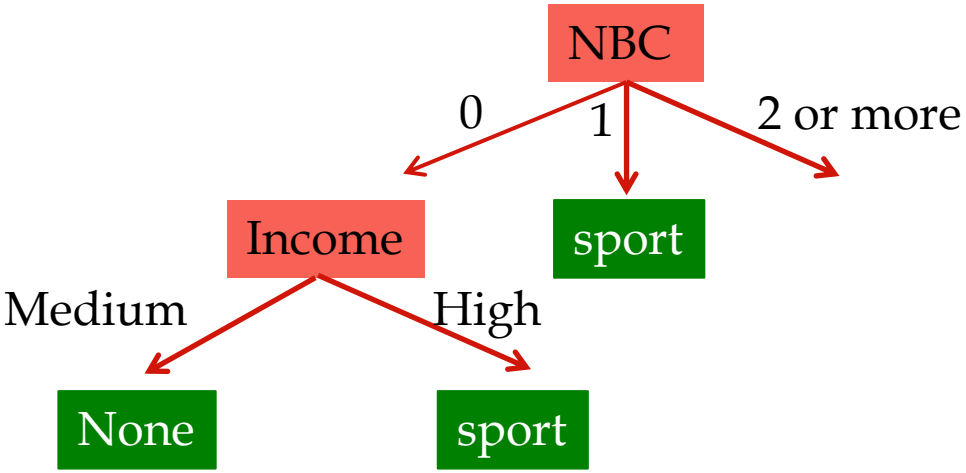




Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

Exercise

- Since there is only 1 attribute left you can deduce the tree without making the calculations.

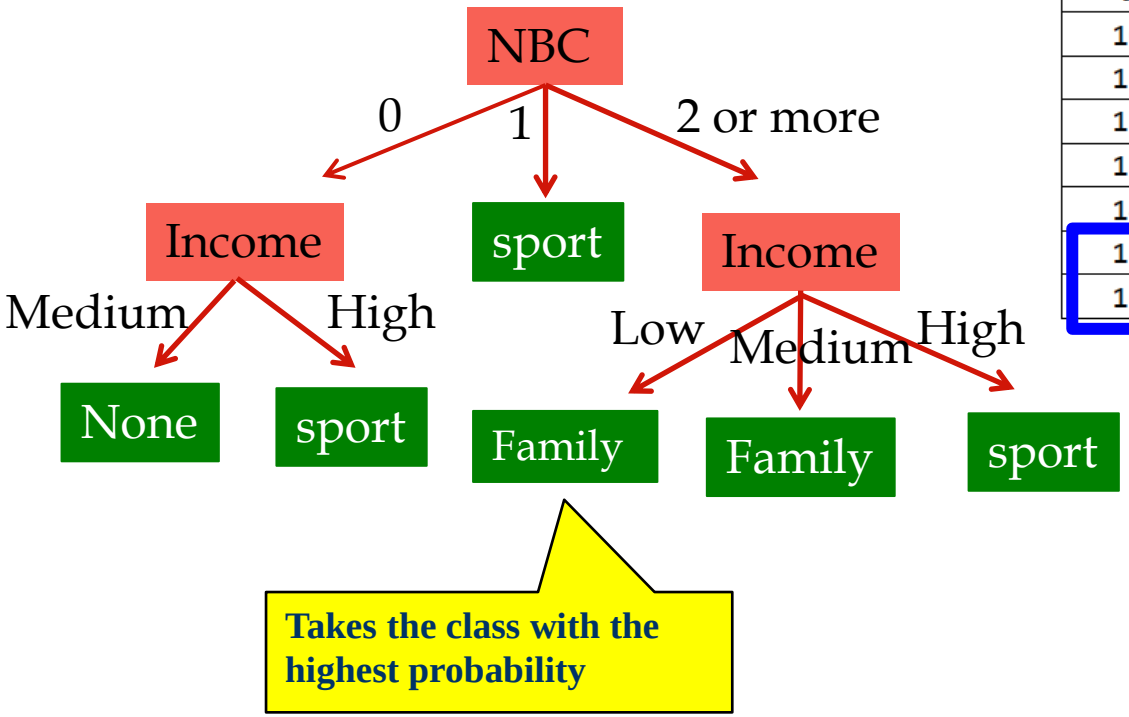


Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

Exercise

- Since there is only 1 attribute left you can deduce the tree without making the calculations.

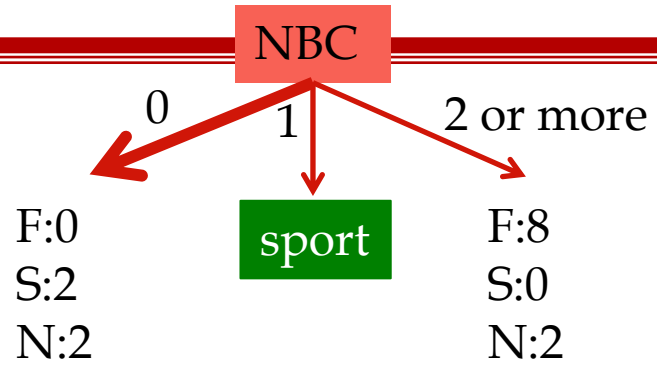
Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None



Exercise

- **Before split:**
- B= {F0, S^2, N^2}

E(B) = 1



Row no.	Income	NumberOfChildren	Car
1	Low	2 or more	Family
2	Low	2 or more	Family
3	Low	2 or more	Family
4	Low	2 or more	Family
5	Medium	2 or more	Family
6	Medium	2 or more	Family
7	High	2 or more	Family
8	High	2 or more	Family
9	High	0	Sport
10	High	0	Sport
11	High	1	Sport
12	High	1	Sport
13	Medium	0	None
14	Medium	0	None
15	Low	2 or more	None
16	Low	2 or more	None

Continuous Attributes: Computing Entropy

- Must determine **the *best split point*** for a continuous attribute A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - » $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the minimum expected information requirement for A is selected as the split-point for A

ID	Home Owner	Marital Status	Annual Income	Defaulted
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 Entropy

- Sort the attribute on values
- Compute midpoints: $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
- Choose the split position that has the least entropy

	Annual Income																			
Sorted Values →	60		70		75		85		90		95		100		120		125		220	
	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
Entropy	0.420		0.400		0.375		0.343		0.417		0.400		***0.300***		0.343		0.375		0.400		0.420	

Gain Ratio

- C4.5 (a successor of ID3) uses **gain ratio** to overcome the problem (normalization to information gain)
- The attribute with the maximum gain ratio is selected as the splitting attribute

$$GainRatio(B, A) = \frac{IG(B, A)}{Entropy(A)}$$

Decision Tree Advantages

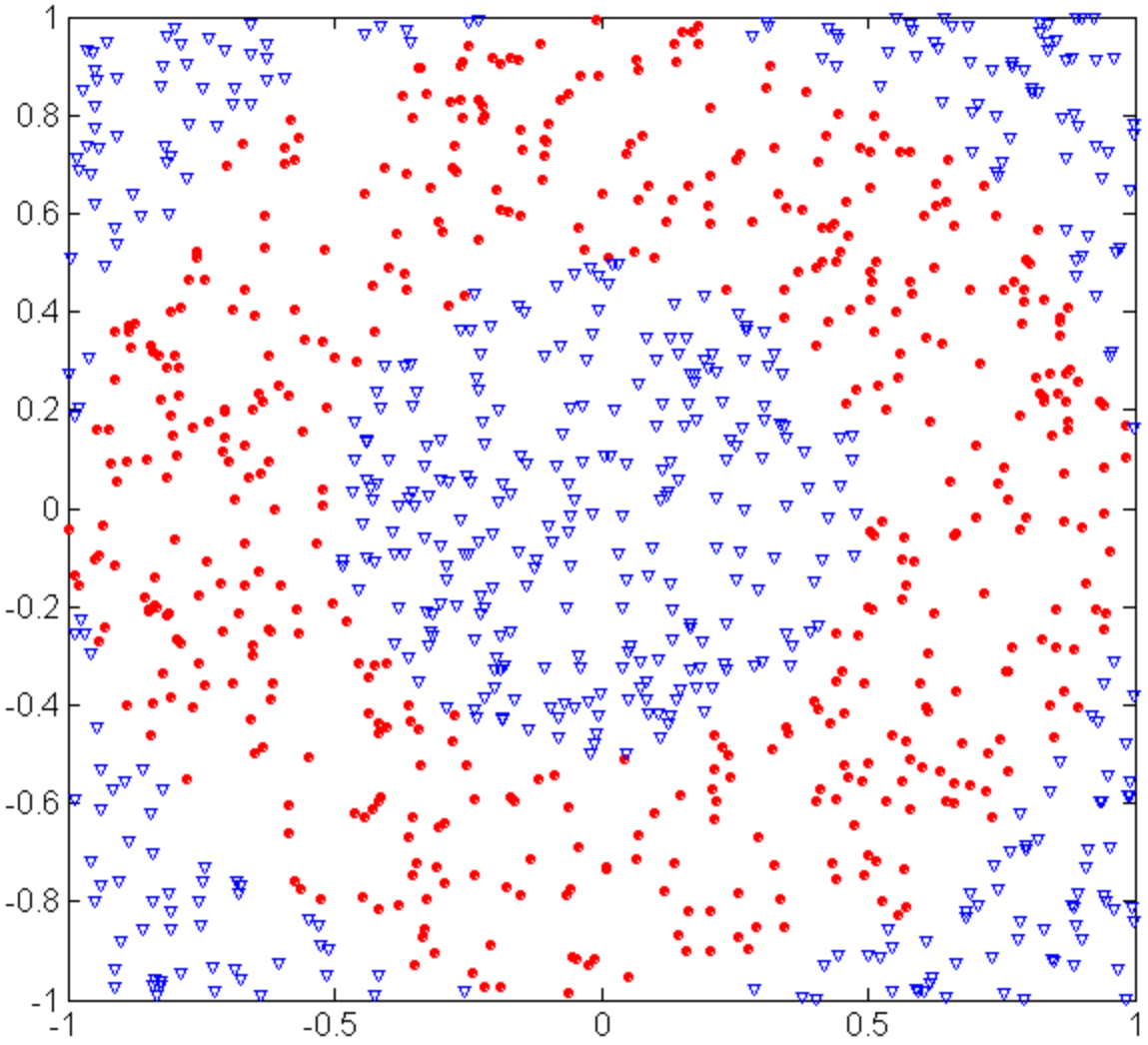
- Inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Accuracy is comparable to other classification techniques for many simple data sets

Decision Tree Disadvantages

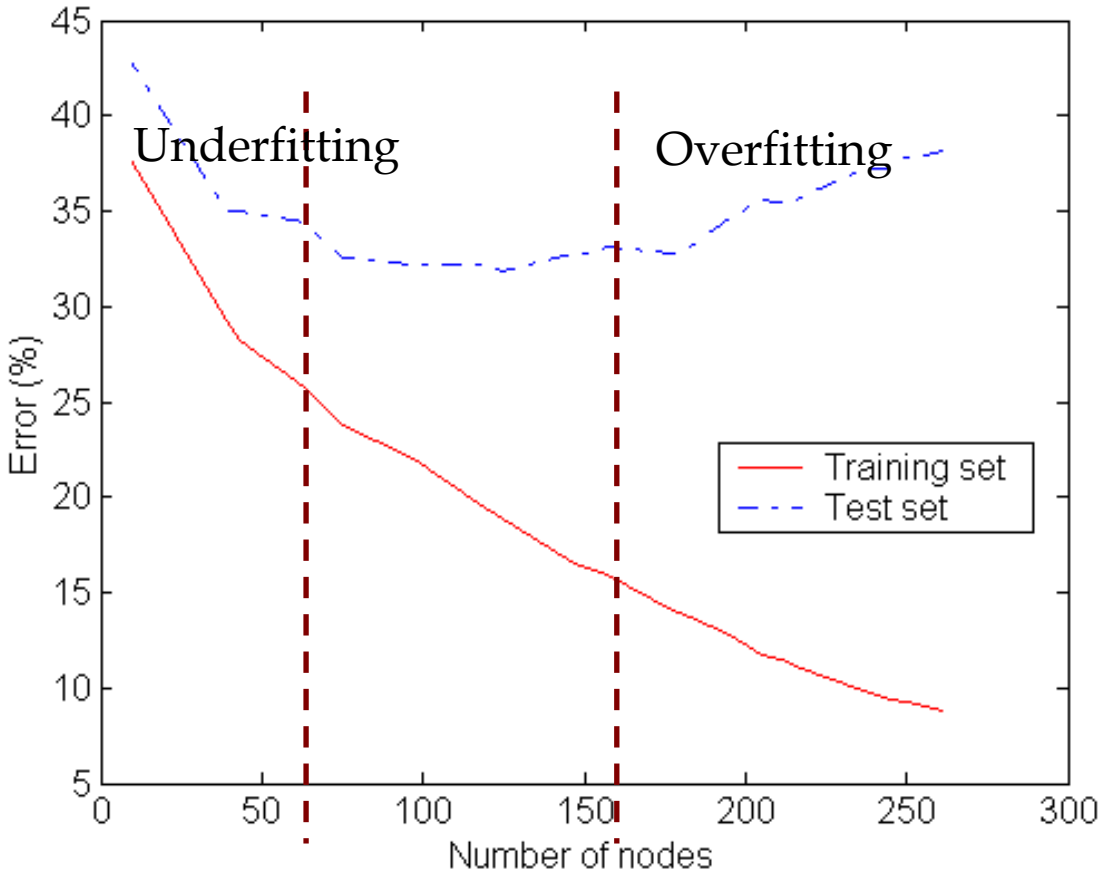
- **Overfitting**: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples

- Two approaches to avoid overfitting
  - Prepruning: *Halt tree construction early*—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: *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"

Few notes on Underfitting/Overfitting



Few notes on Underfitting/Overfitting



Underfitting: when model is too simple, both training and test errors are large

Overfitting: when model is too complex it models the details of the training set and fails on the test set

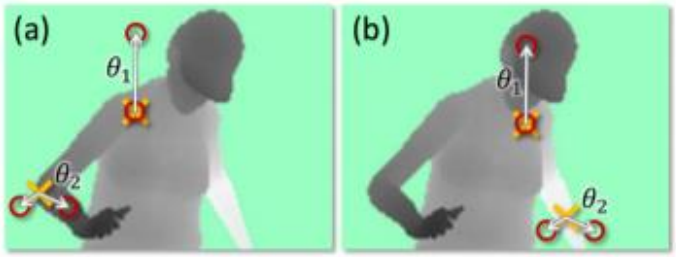
Applications of Decision Trees

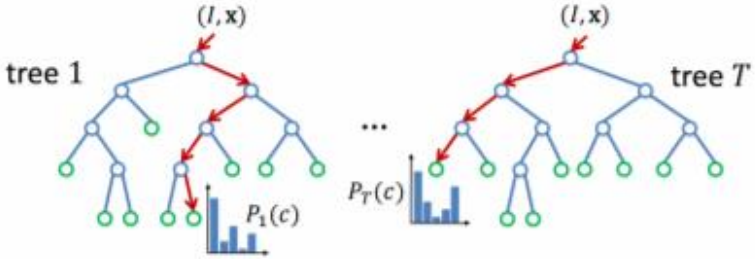
- Decision Trees are used in X-box

[J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake. Real-Time Human Pose

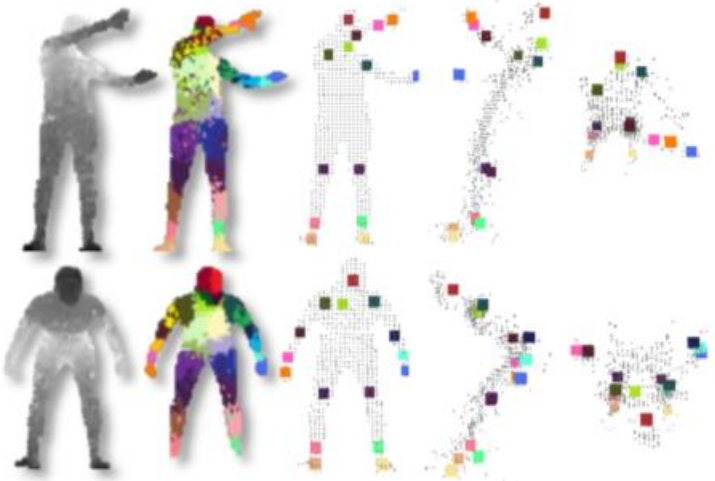
Applications of Decision Trees

- Trained on million(s) of examples



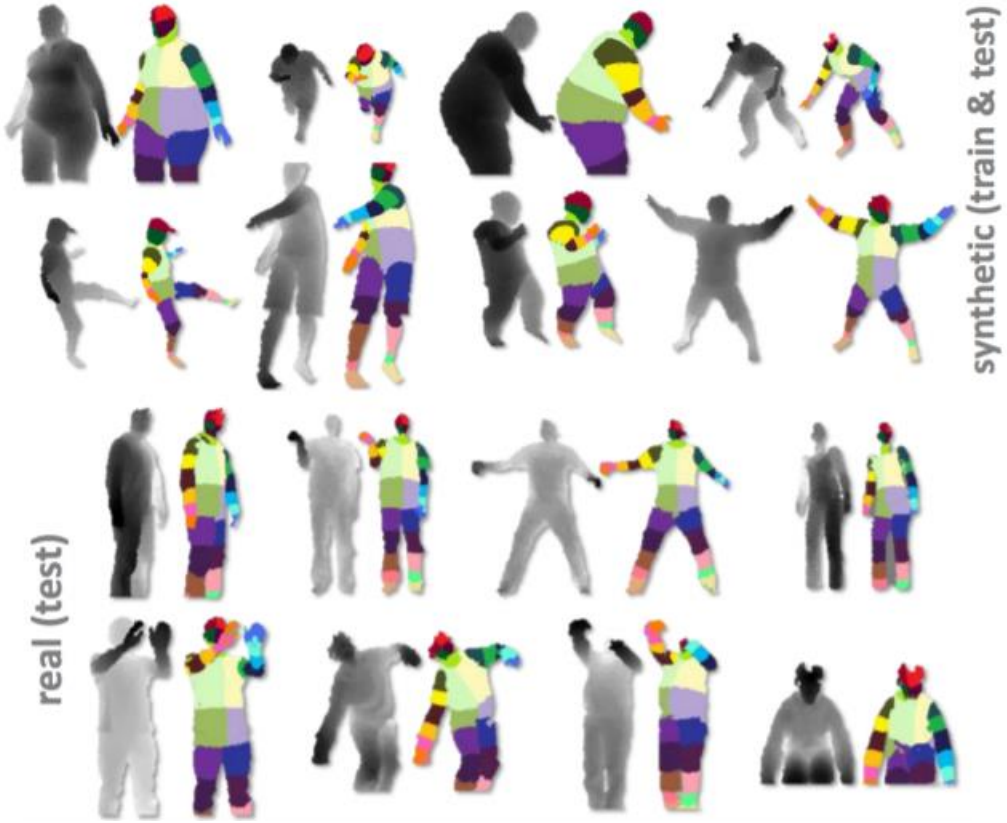


- Results



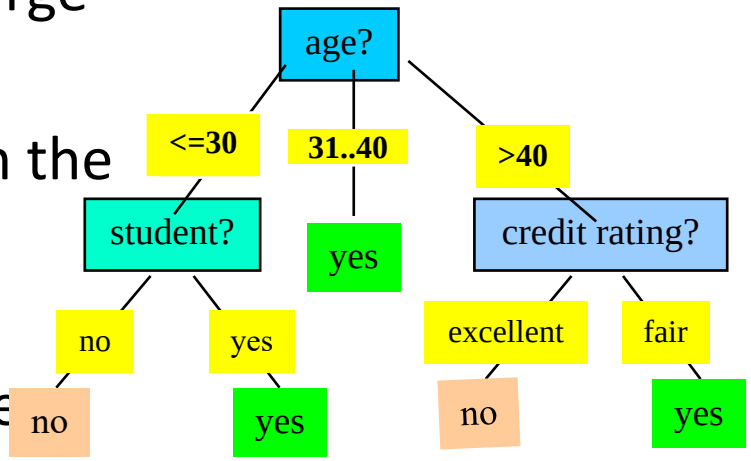
Applications of Decision Trees

- Trained on million(s) of examples



Rule-Based Classification

- Rules are easier to understand than large trees
- 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 holds the class prediction



- Example: Rule extraction from our *buys_computer* decision-tree

IF *age* = young AND *student* = *no* THEN *buys_computer* = *no*

IF *age* = young AND *student* = *yes* THEN *buys_computer* = *yes*

IF *age* = mid-age THEN *buys_computer* = *yes*

IF *age* = old AND *credit_rating* = *excellent* THEN *buys_computer* = *no*

IF *age* = old AND *credit_rating* = *fair* THEN *buys_computer* = *yes*

Rule Coverage and Accuracy

- Coverage of a rule:
  - Fraction of records that satisfy the antecedent of a rule
- Accuracy of a rule:
  - Fraction of records that satisfy the antecedent that also satisfy the consequent of a rule

Tid	Refund	Marital Status	Taxable Income	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**

(Status=Single) → No

Coverage = 40%, Accuracy = 50%