UNIT - 2

Distributions, Continuous Distributions, Probability, Decisions Trees, Machine Learning, Probability and Distributions, Descriptive Statistics and Graphs.

R. K. Sharma, CSED, TIET Patiala

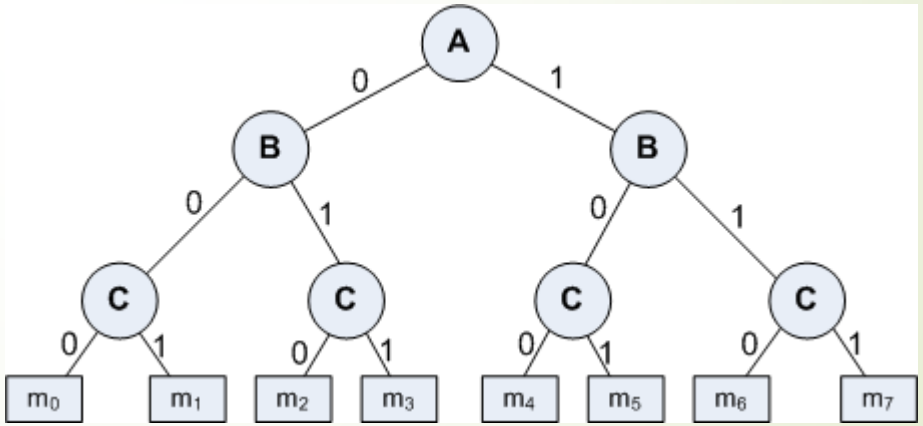
Decision Trees

Basic Concepts

- A Decision Tree is an important data structure known to solve many computational problems

Example 1: Binary Decision Tree

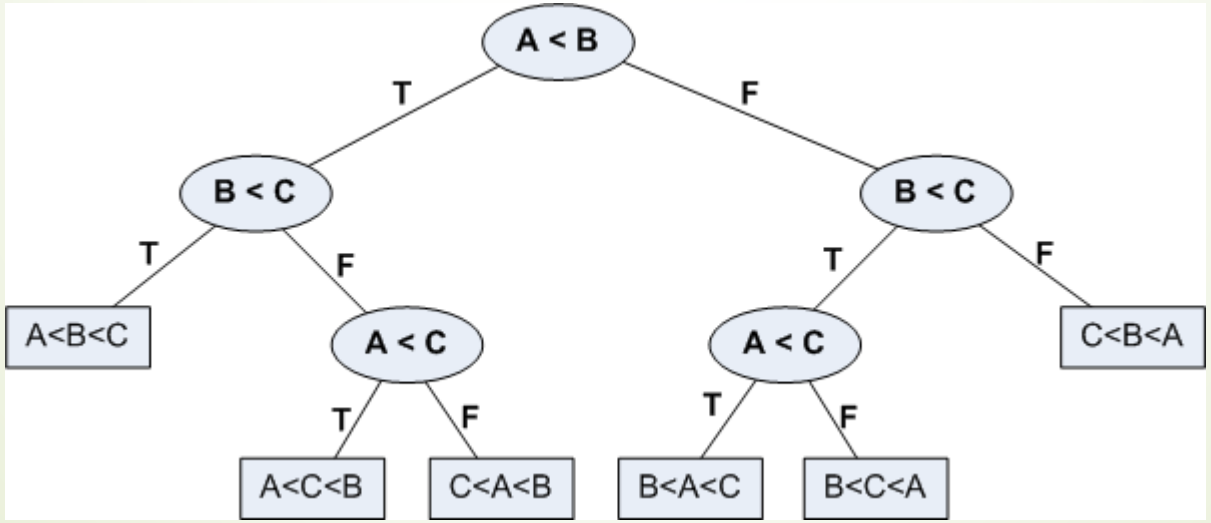
A	*B*	*C*	*f*
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7



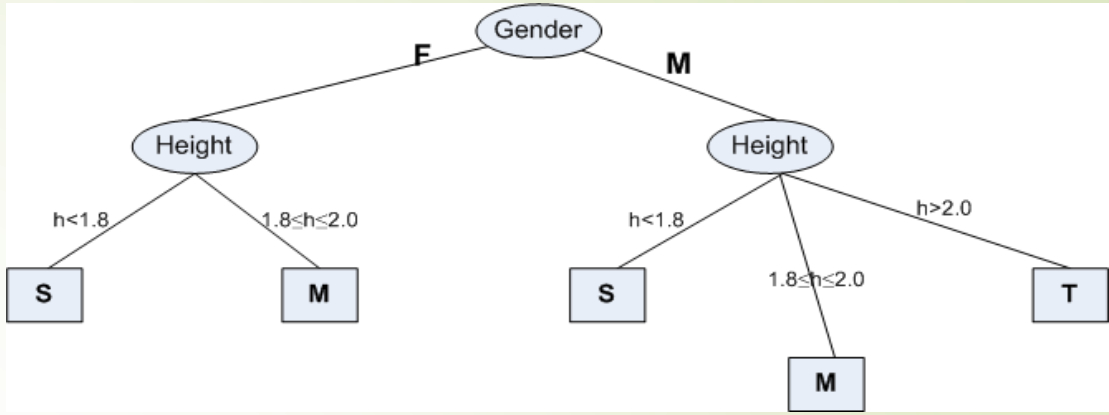
Basic Concepts

- In Example 1, we have considered a decision tree where values of any attribute is binary only. Decision tree is also possible where attributes are of continuous data type

Example 2: Decision Tree with numeric data

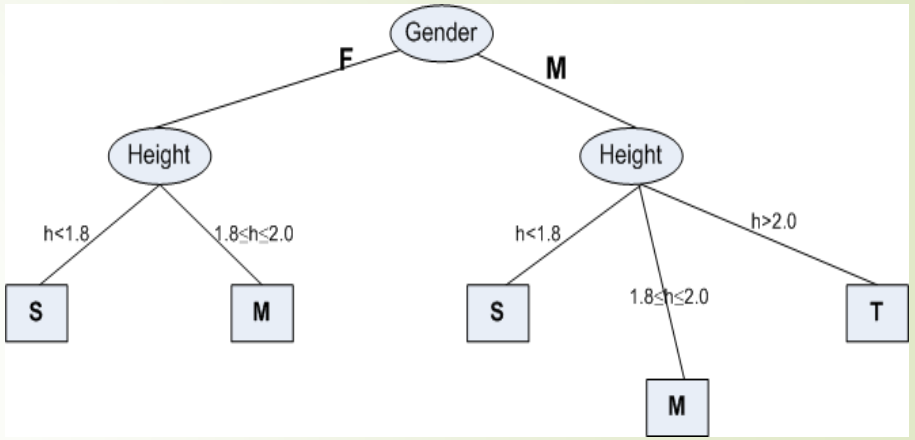


Some Characteristics



- Decision tree may be *n*-ary, $n \geq 2$.
- There is a special node called root node.
- All nodes drawn with circle (ellipse) are called internal nodes.
- All nodes drawn with rectangle boxes are called terminal nodes or leaf nodes.
- Edges of a node represent the outcome for a value of the node.
- In a path, a node with same label is never repeated.
- Decision tree is not unique, as different ordering of internal nodes can give different decision tree.

Decision Tree and Classification Task



- Decision tree helps us to classify data.
  - Internal nodes are some attribute
  - Edges are the values of attributes
  - External nodes are the outcome of classification
- Such a classification is, in fact, made by posing questions starting from the root node to each terminal node.

Decision Tree and Classification Task

- We can solve a classification problem by asking a series of questions about the attributes.
  - Each time we receive an answer, a follow-up question is asked until we reach a conclusion about the class-label of the test.
- The series of questions and their answers can be organized in the form of a decision tree
  - As a hierarchical structure consisting of nodes and edges
- Once a decision tree is built, it is applied to any test to classify it.

Definition of Decision Tree

Definition: **Decision Tree**

Given a dataset D = $\{t_1, t_2, \ldots\ldots, t_n\}$, where t_i denotes a tuple, which is defined by a set of attribute $A = \{A_1, A_2, \ldots\ldots, A_m\}$. Also, given a set of classes C = $\{c_1, c_2, \ldots\ldots, c_k\}$.

A decision tree ***T*** is a tree associated with *D* that has the following properties:

- Each internal node is labeled with an attribute A_i
- Each edge is labeled with predicate that can be applied to the attribute associated with the parent node of it
- Each leaf node is labeled with class c_j

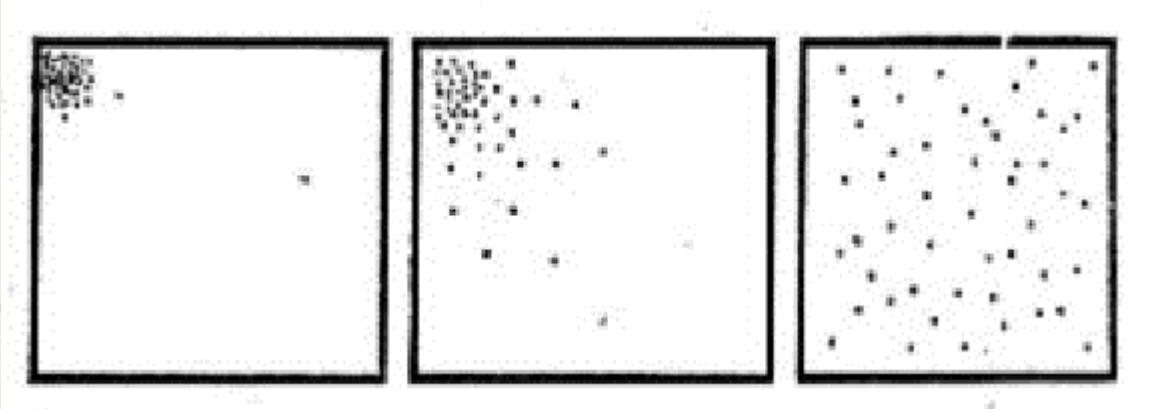
Building Decision Tree

- In principle, there are exponentially many decision trees that can be constructed from a given dataset (also called training data).
  - Some of the trees may not be optimum
  - Some of them may give inaccurate result
- Two approaches are known
  - **Greedy strategy**
    - A top-down recursive divide-and-conquer
  - **Modification of greedy strategy**
    - ID3
    - CART
    - C4.5, etc.

Measurements of Impurity in a Dataset

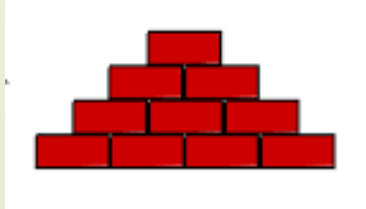
1. Entropy
2. Gini Index

Concept of Entropy

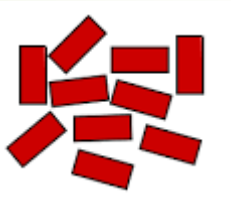


If a point represents a gas molecule, then which system has the more entropy?

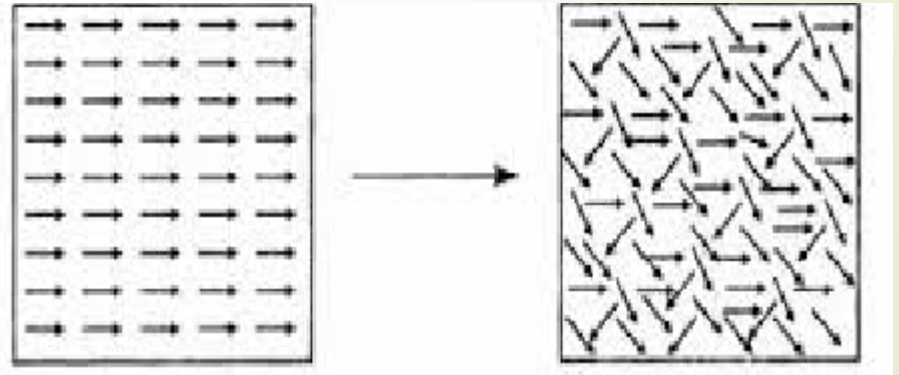
How to measure? $\Delta S = \frac{\Delta Q}{T}$?



More **ordered**
less **entropy**



Less ordered
higher entropy



More organized or
ordered (less **probable**)

Less organized or
disordered (**more** probable)

Entropy and its Meaning

- Entropy is an important concept used in Physics in the context of heat and thereby uncertainty of the states of a matter.
- At a later stage, with the growth of Information Technology, entropy becomes an important concept in Information Theory.
- To deal with the classification job, entropy is an important concept, which is considered as:
  - An information-theoretic measure of the “uncertainty” contained in a training data
    - due to the presence of more than one classes.

Entropy in Information Theory

- The entropy concept in information theory first time coined by Claude Shannon (1850).
- The first time it was used to measure the “information content” in messages.
- These days entropy is widely being used as a way of representing messages for efficient transmission by Telecommunication Systems.

Measure of Information Content

Example:

a) Guessing a birthday of your classmate

It is with uncertainty ~ $\frac{1}{365}$

Whereas guessing the day of his/her birthday is $\frac{1}{7}$.

This uncertainty, we may say varies between 0 to 1, both inclusive.

b) As another example, a question related to event with eventuality (or impossibility) will be answered with 0 or 1 uncertainty.

- Does sun rises in the East? (answer is with 0 uncertainty)
- Will mother give birth to male baby? (answer is with ½ uncertainty)
- Is there a planet like earth in the galaxy? (answer is with an extreme uncertainty)

Entropy of a Training Set

- If there are k classes $c_1, c_2 \ldots\ldots, c_k$ and p_i for $i = 1\ to\ k$ denotes the number of occurrences of classes c_i divided by the total number of instances (i.e., the relative frequency of occurrence of c_i, can also be called as the probability of class c_i) in the training set, then entropy of the training set is denoted by:

$$E = -\sum_{i=1}^{k} p_i \log_2 p_i$$

Notes:

- The above formula should be summed over the non-empty classes only, that is, classes for which $p_i \neq 0$
- E is always a positive quantity
- E takes it's minimum value (zero) if and only if all the instances have the same class (i.e., the training set with only one non-empty class, for which the probability is 1).
- Entropy takes its maximum value when the instances are equally distributed among k possible classes. In this case, the maximum value of E is $\log_2 k$.

//R Illustration//

Entropy Calculations

- For the set X = {a, a, a, b, b, b, b, b}
  - Total instances : 8
  - Instances of b : 5
  - Instances of a : 3

- Entropy $E = (-3/8)\log_2(3/8) + (-5/8)\log_2(5/8)$

  $= -[0.375 * (-1.415) + 0.625 * (-0.678)]$

  $=-(-0.53-0.424)$

  $= 0.954$

Entropy of a Training Set: One more example

Example: OPTH dataset

Consider the OTPH data shown in the following table with total 24 instances in it.

Age	Eye sight	Astigmatic	Use Type	Class
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	2
2	2	1	1	3
2	2	1	2	1
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3
3	1	1	1	3
3	1	1	2	2
3	1	2	1	3
3	1	2	2	3
3	2	1	1	3
3	2	1	2	3
3	2	2	1	3
3	2	2	2	3

A coded forms for all values of attributes are used to avoid the cluttering in the table.

Entropy of a training set

Specification of the attributes are as follows.

Age	Eye Sight	Astigmatic	Use Type
1: Young	1: Myopia	1: No	1: Frequent
2: Middle-aged	2: Hypermetropia	2: Yes	2: Less
3: Old			

Class: 1: Contact Lens 2:Normal glass 3: Nothing

In the OPTH database, there are 3 classes and 4 instances with class 1, 5 instances with class 2 and 15 instances with class 3. Hence, entropy E of the database is:

$$E = -\frac{4}{24}\log_2\frac{4}{24} - \frac{5}{24}\log_2\frac{5}{24} - \frac{15}{24}\log_2\frac{15}{24} = 1.3261$$

Note:

- The entropy of a training set implies the number of yes/no questions, on the average, needed to determine an unknown test to be classified.
- It is very crucial to decide the series of questions about the value of a set of attribute, which collectively determine the classification. Sometimes it may take one question, sometimes many more.
- Decision tree induction helps us to ask such a series of questions. In other words, we can utilize entropy concept to build a better decision tree.

How entropy can be used to build a decision tree is our next topic of discussion.

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 split, and
  2. Splitting criteria

- Several algorithms have been proposed for the above tasks.
  - **ID3**
  - **CART**
  - **C 4.5 (To be completed as *Home Work*)**

ID3 Algorithm: Illustration

ID3: Decision Tree Induction Algorithms

- 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.

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.

Defining Information Gain

- We consider the following symbols and terminologies to define information gain.
- $D \equiv$ denotes the training set at any instant
- $|D| \equiv$ denotes the size of the training set D
- $E(D) \equiv$ denotes the entropy of the training set D
- The entropy of the training set D

$$E(D) = -\sum_{i=1}^{k} p_i \, log_2(p_i)$$

  - where the training set D has $c_1, c_2, \ldots, c_k$, the k number of distinct classes and
  - p_i, $0 < p_i \leq 1$ is the probability that an arbitrary tuple in D belongs to class c_i ($i = 1, 2, \ldots, k$).

Defining Information Gain

- p_i can be calculated as

$$p_i = \frac{|C_{i,D}|}{|D|}$$

- where $C_{i,D}$ is the set of tuples of class c_i in D.

- Suppose, we want to partition D on some attribute A having m distinct values $\{a_1, a_2, \ldots, a_m\}$.

- Attribute A can be considered to split D into m partitions $\{D_1, D_2, \ldots, D_m\}$, where D_j $(j = 1,2, \ldots ,m)$ contains those tuples in D that have outcome a_j of A.

Defining Information Gain

Definition: **Weighted Entropy**

The weighted entropy denoted as $E_A(D)$ for all partitions of D with respect to A is given by:

$$E_A(D) = \sum_{j=1}^{m} \frac{|D_j|}{|D|} E(D_j)$$

Here, the term $\frac{|D_j|}{|D|}$ denotes the weight of the j-th training set.

More meaningfully, $E_A(D)$ is the expected information required to classify a tuple from D based on the splitting of A.

Defining Information Gain

- **Our objective is to take A on splitting to produce an exact classification (also called pure), that is, all tuples belong to one class.**
- **However, it is quite likely that the partitions is impure, that is, they contain tuples from two or more classes.**
- **In that sense, $E_A(\mathrm{D})$ is a measure of impurities (or purity). A lesser value of $E_A(\mathrm{D})$ implying more power the partitions are.**

Definition 9.5: **Information Gain**

Information gain $g(D, A)$ of the training set D splitting on the attribute A is given by

$$g(D, A) = E(D) - E_A(D)$$

In other words, $g(D, A)$ gives us an estimation how much would be gained by splitting on A. The attribute A with the highest value of information gain should be chosen as the splitting attribute for D.

Calculating Information gain

- Consider the set D:

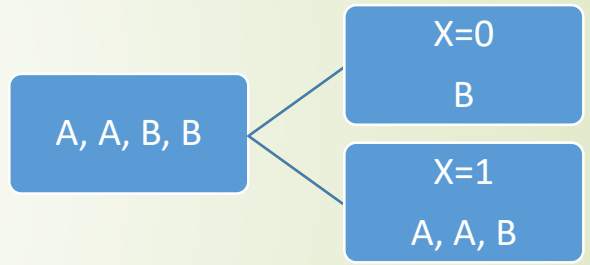
X	Y	Z	Class
1	1	1	A
1	1	0	A
0	0	1	B
1	0	0	B

- This example has 3 features, 2 classes, and 4 patterns
- Entropy(D) = -(1/2)*log(1/2, 2) – ½*log(1/2, 2) = -log(1/2, 2) = 1

Calculating Information gain

Split on X:

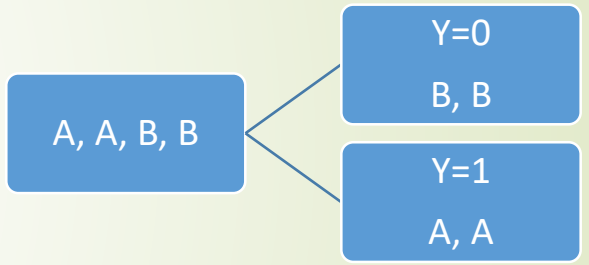
- **Entropy(Child1) = 0**
- **Entropy(Child2) = -(2/3) log(2/3, 2) – (1/3) log(1/3, 2) = 0.918**
- **g(D, X) = 1 – (1/4)*0 – (3/4)*0.918 = 0.3112**



Calculating Information gain

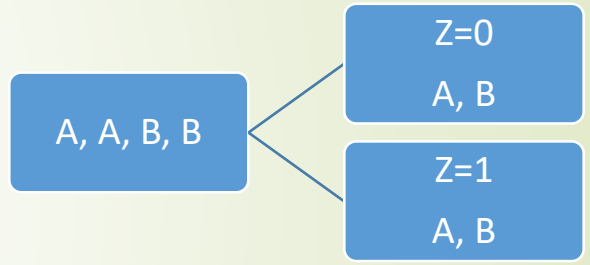
Split on Y:

- **Entropy(Child1) = 0**
- **Entropy(Child2) = 0**
- **g(D, Y) = 1 – (2/4)*0 – (2/4)*0 = 1**



Calculating Information gain

Split on Z:



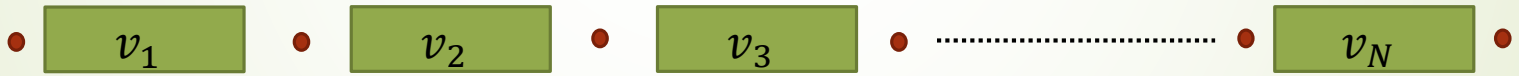
- Entropy(Child1) = 1
- Entropy(Child2) = 1
- g(D, Z) = 1 – (2/4)*1 – (2/4)*1 = 0

- Maximum Information gain is 1 when we split at Y. So this is the best feature to split.

Splitting of Continuous Attribute Values

- In the foregoing discussion, we assumed that an attribute to be splitted is with a finite number of discrete values. Now, there is a great deal if the attribute is not so, rather it is a continuous-valued attribute.
- There are two approaches mainly to deal with such a case.

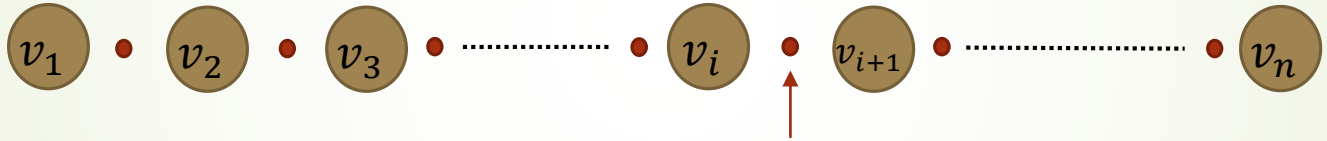
1. **Data Discretization**: All values of the attribute can be discretized into a finite number of group values and then split point can be decided at each boundary point of the groups.



So, if there are $n - groups$ of discrete values, then we have $(n + 1)$ split points.

Splitting of Continuous attribute values

2. **Mid-point splitting:** Another approach to avoid the data discretization.
   - It sorts the values of the attribute and take the distinct values only in it.
   - Then, the mid-point between each pair of adjacent values is considered as a split-point.



   - Here, if n-distinct values are there for the attribute A, then we choose $n-1$ split points as shown above.
   - For example, there is a split point $s = \frac{v_i + v_{(i+1)}}{2}$ in between v_i and $v_{(i+1)}$
   - For each split-point, we have two partitions: $A \leq s$ *and* $A > s$, and finally the point with maximum information gain is the desired split point for that attribute.

CART Algorithm: Illustration

CART Algorithm

- It is observed that information gain measure used in ID3 is biased towards test with many outcomes, that is, it prefers to select attributes having a large number of values.
- L. Breiman, J. Friedman, R. Olshen and C. Stone in 1984 proposed an algorithm to build a binary decision tree also called CART decision tree.
  - CART stands for Classification and Regression Tree
  - In fact, invented independently at the same time as ID3 (1984).
  - ID3 and CART are two cornerstone algorithms spawned a flurry of work on decision tree induction.
- CART is a technique that generates a binary decision tree; That is, unlike ID3, in CART, for each node only two children is created.
- ID3 uses Information gain as a measure to select the best attribute to be splitted, whereas CART do the same but using another measurement called Gini index . It is also known as Gini Index of Diversity and is denote as γ.

Gini Index of Diversity

Definition: **Gini Index**

Suppose, D is a training set with size $|D|$ and $C = \{c_1, c_2, \dots, c_k\}$ be the set of k classifications and $A = \{a_1, a_2, \dots, a_m\}$ be any attribute with m different values of it. Like entropy measure in ID3, CART proposes Gini Index (denoted by G) as the measure of impurity of D. It can be defined as follows.

$$G(D) = 1 - \sum_{i=1}^{k} p_i^2$$

where p_i is the probability that a tuple in D belongs to class c_i and p_i can be estimated as

$$p_i = \frac{|C_{i,D}|}{|D|}$$

where $|C_{i,D}|$ denotes the number of tuples in D with class c_i.

Gini Index Calculations

- For the set X = {a, a, a, b, b, b, b, b}
  - Total instances : 8
  - Instances of b : 5
  - Instances of a : 3
  - $G(D) = 1 - \sum_{i=1}^{k} p_i^2$

    = 1 – (5/8)^2 – (3/8)^2

    = 0.46875

Gini Index of Diversity

Note

- $G(D)$ measures the "impurity" of data set D.
- The smallest value of $G(D)$ is zero
  - which it takes when all the classifications are same.
- It takes its largest value $= 1 - \frac{1}{k}$
  - when the classes are evenly distributed between the tuples, that is the frequency of each class is $\frac{1}{k}$.

Gini Index of Diversity

Definition 9.7: **Gini Index of Diversity**

Suppose, a binary partition on A splits D into D_1 and D_2, then the weighted average Gini Index of splitting denoted by $G_A(D)$ is given by

$$G_A(D) = \frac{|D_1|}{|D|}.G(D_1) + \frac{|D_2|}{|D|}.G(D_2)$$

This binary partition of D reduces the impurity and the reduction in impurity is measured by

$$\gamma(A, D) = G(D) - G_A(D)$$

Gini Index of Diversity and CART

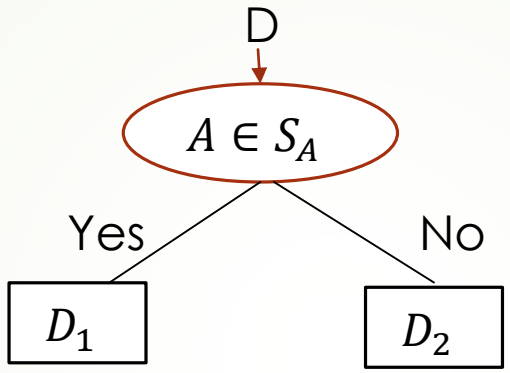
- This $\gamma(A, D)$ is called the Gini Index of diversity.
- It is also called as “impurity reduction”.
- The attribute that maximizes the reduction in impurity (or equivalently, has the minimum value of $G_A(D)$) is selected for the attribute to be splitted.

n-ary Attribute Values to Binary Splitting

- The CART algorithm considers a binary split for each attribute.
- We shall discuss how the same is possible for attribute with more than two values.
- **Case 1: Discrete valued attributes**
- Let us consider the case where A is a discrete-valued attribute having m discrete values $a_1, a_2, \ldots, a_m$.
- To determine the best binary split on A, we examine all of the possible subsets say 2^A of A that can be formed using the values of A.
- Each subset $S_A \in 2^A$ can be considered as a binary test for attribute A of the form "$A \in S_A$?“.

n-ary Attribute Values to Binary Splitting

- Thus, given a data set *D*, we have to perform a test for an attribute value *A* like



- This test is satisfied if the value of *A* for the tuples is among the values listed in S_A.

- If *A* has *m* distinct values in *D*, then there are 2^m possible subsets, out of which the empty subset { } and the power set $\{a_1, a_2, \ldots, a_n\}$ should be excluded (as they really do not represent a split).

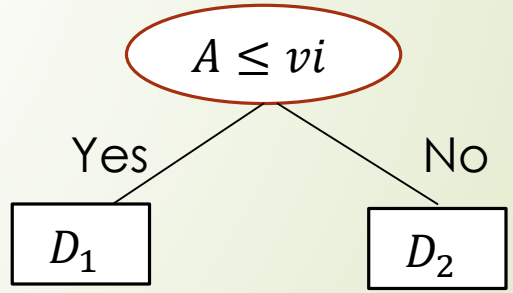
- Thus, there are $2^m - 2$ possible ways to form two partitions of the dataset *D*, based on the binary split of *A*.

n-ary Attribute Values to Binary Splitting

Case2: Continuous valued attributes

- For a continuous-valued attribute, each possible split point must be taken into account.
- The strategy is similar to that followed in ID3 to calculate information gain for the continuous –valued attributes.
- According to that strategy, the mid-point between a_i and a_{i+1} , let it be v_i, then

$$v_i = \frac{a_i + a_{i+1}}{2}$$



n-ary Attribute Values to Binary Splitting

- Each pair of (sorted) adjacent values is taken as a possible split-point say v_i.
- D_1 is the set of tuples in D satisfying $A \leq v_i$ and D_2 in the set of tuples in D satisfying $A > v_i$.
- The point giving the minimum Gini Index $G_A(D)$ is taken as the split-point of the attribute A.

Note

- The attribute A and either its splitting subset S_A (for discrete-valued splitting attribute) or split-point v_i (for continuous valued splitting attribute) together form the splitting criteria.

CART Algorithm : Illustration

Example: CART Algorithm

Suppose we want to build decision tree for the data set as given in this table.

Age
Y : young
M : middle-aged
O : old

Salary
L : low
M : medium
H : high

Job
G : government
P : private

Performance
A : Average
E : Excellent

Class : Select
Y : yes
N : no

Tuple#	Age	Salary	Job	Performance	Select
1	Y	H	P	A	N
2	Y	H	P	E	N
3	M	H	P	A	Y
4	O	M	P	A	Y
5	O	L	G	A	Y
6	O	L	G	E	N
7	M	L	G	E	Y
8	Y	M	P	A	N
9	Y	L	G	A	Y
10	O	M	G	A	Y
11	Y	M	G	E	Y
12	M	M	P	E	Y
13	M	H	G	A	Y
14	O	M	P	E	N

CART Algorithm : Illustration

For this dataset,

$$G(EMP) = 1 - \sum_{i=1}^{2} p_i^2$$

$$= 1 - \left[\left(\frac{9}{14}\right)^2 + \left(\frac{5}{14}\right)^2\right]$$

$$= \mathbf{0.4592}$$

Now let us consider the calculation of $G_A(EMP)$ for **Age**, **Salary**, **Job** and **Performance**.

CART Algorithm : Illustration

Attribute of splitting: Age

The attribute age has three values, namely Y, M and O. So there are 6 subsets, that should be considered for splitting as:

$$\underset{age_1}{\{Y\}}, \underset{age_2}{\{M\}}, \underset{age_3}{\{O\}}, \underset{age_4}{\{Y,M\}}, \underset{age_5}{\{Y,O\}}, \underset{age_6}{\{M,O\}}$$

$$G_{age_1}(D) = \frac{5}{14} * \left(1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2\right) + \frac{9}{14} * \left(1 - \left(\frac{2}{9}\right)^2 - \left(\frac{7}{9}\right)^2\right) = \mathbf{0.3637}$$

$\gamma(G_{age_1}(D))$ **= 0.4592 – 0.3637 = 0.0955**

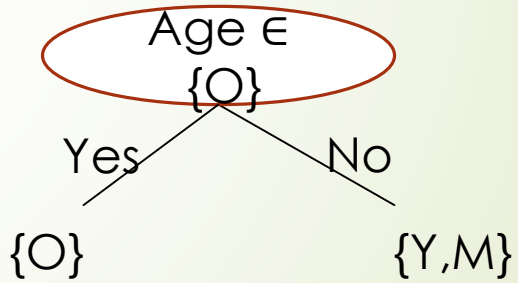
$G_{age_2}(D) =$ **??**

$G_{age_3}(D) =$ **??**

$G_{age_4}(D) = G_{age_3}(D)$

$G_{age_5}(D) = G_{age_2}(D)$

$G_{age_6}(D) = G_{age_1}(D)$



The best value of Gini Index while splitting attribute Age is $G_{age_2}(D) = \mathbf{0.3571}$

CART Algorithm : Illustration

Attribute of splitting: Age

The attribute age has three values, namely Y, M and O. So there are 6 subsets, that should be considered for splitting as:

$$\underset{age_1}{\{Y\}}, \underset{age_2}{\{M\}}, \underset{age_3}{\{O\}}, \underset{age_4}{\{Y,M\}}, \underset{age_5}{\{Y,O\}}, \underset{age_6}{\{M,O\}}$$

$$G_{age_1}(D) = \frac{5}{14} * \left(1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2\right) + \frac{9}{14}\left(1 - \left(\frac{2}{9}\right)^2 - \left(\frac{7}{9}\right)^2\right) = \mathbf{0.3937}$$

$\gamma(G_{age_1}(D))$ **= 0.4592 – 0.3937 = 0.0655**

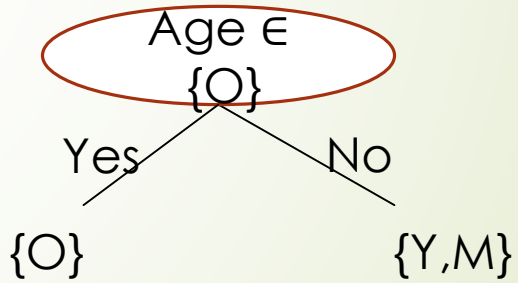
$G_{age_2}(D) =$ **0.3571, so** $\gamma(G_{age_2}(D))$ **= 0.1021**

$G_{age_3}(D) =$ **0.4571** $\gamma(G_{age_3}(D)) =$ **0.0021**

$G_{age_4}(D) = G_{age_3}(D)$ //

$G_{age_5}(D) = G_{age_2}(D)$ //

$G_{age_6}(D) = G_{age_1}(D)$ //



The best value of Gini Index while splitting attribute Age is $G_{age_2}(D) = \mathbf{0.3571}$

CART Algorithm : Illustration

Attribute of Splitting: Salary

The attribute salary has three values namely L, M and H. So, there are 6 subsets, that should be considered for splitting as:

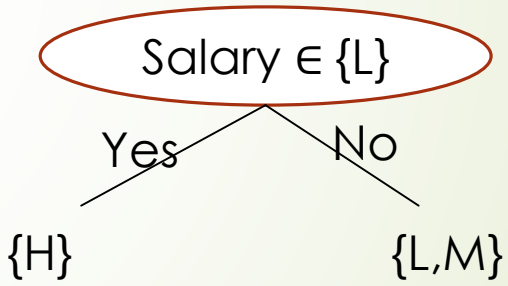
$$\underset{sal_1}{\{L\}}, \quad \underset{sal_2}{\{M,H\}}, \quad \underset{sal_3}{\{M\}}, \quad \underset{sal_4}{\{L,H\}}, \quad \underset{sal_5}{\{H\}}, \quad \underset{sal_6}{\{L,M\}}$$

$G_{sal_1}(D) = G_{sal_2}(D) =$

(4/14)*(1-(3/4)^2-(1/4)^2) + (10/14)*(1-(6/10)^2-(4/10)^2) = **0.45**

$G_{sal_3}(D) = G_{sal_4}(D) = 0.4583$

$G_{sal_5}(D) = G_{sal_6}(D) = 0.4429$



$$\gamma(salary_{(5,6)}, D) = 0.4592 - 0.4429 = 0.0163$$

CART Algorithm : Illustration

Attribute of Splitting: job

Job being the binary attribute , we have

$$G_{job}(D) = \frac{7}{14} G(D_1) + \frac{7}{14} G(D_2)$$

$$= \frac{7}{14}\left[1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2\right] + \frac{7}{14}\left[1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2\right] = ?$$

$$\gamma(job, D) = ?$$

CART Algorithm : Illustration

Attribute of Splitting: job

Job being the binary attribute , we have

$$G_{job}(D) = \frac{7}{14} G(D_1) + \frac{7}{14} G(D_2)$$

$$= \frac{7}{14}\left[1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2\right] + \frac{7}{14}\left[1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2\right] = \mathbf{0.3673}$$

$$\gamma(job, D) = 0.4592 - 0.3673 = 0.0919$$

CART Algorithm : Illustration

Attribute of Splitting: Performance

Job being the binary attribute , we have

$G_{Performance}(D) = 0.4286$

$\gamma(performance, D) = 0.0306$

Out of these $\gamma(Age, D)$ gives the maximum value and hence, the attribute **Age** would be chosen for splitting subset $\{M\}$ or $\{Y, O\}$.

Summary of Algorithms

Algorithm	Splitting Criteria	Remark
ID3	**Information Gain** $\alpha(A, D) = E(D) - E_A(\mathrm{D})$ Where $E(D)$ = Entropy of D (a measure of uncertainty) = $-\sum_{i=1}^{k} p_i \log_2 p_i$ where D is with set of k classes $c_1, c_2, \ldots, c_k$ and $p_i = \frac{\lvert C_{i,D} \rvert}{\lvert D \rvert}$; Here, $C_{i,D}$ is the set of tuples with class c_i in D. $E_A(\mathrm{D})$ = Weighted average entropy when D is partitioned on the values of attribute A = $\sum_{j=1}^{m} \frac{\lvert D_j \rvert}{\lvert D \rvert} E(D_j)$ Here, m denotes the distinct values of attribute A.	• The algorithm calculates $\alpha(A_i,D)$ for all A_i *in* D and choose that attribute which has **maximum** $\alpha(A_i,D)$. • The algorithm can handle both categorical and numerical attributes. • It favors splitting those attributes, which has a large number of distinct values.

Algorithm	Splitting Criteria	Remark
CART	**Gini Index** $\gamma(A, D) = G(D) - G_A(\mathrm{D})$ where $G(D)$ = Gini index (a measure of impurity) $= 1 - \sum_{i=1}^{k} {p_i}^2$ Here, $p_i = \frac{\lvert C_{i,D} \rvert}{\lvert D \rvert}$ and D is with k number of classes and $G_A(D) = \frac{\lvert D_1 \rvert}{\lvert D \rvert} G(D_1) + \frac{\lvert D_2 \rvert}{\lvert D \rvert} G(D_2)$, when D is partitioned into two data sets D_1 and D_2 based on some values of attribute A.	• The algorithm calculates all binary partitions for all possible values of attribute A and choose that binary partition which has the **maximum** $\gamma(A, D)$. • The algorithm is computationally very expensive when the attribute A has a large number of values.

Algorithm	Splitting Criteria	Remark
C4.5	**Gain Ratio** $$\beta(A,D) = \frac{\alpha(A,D)}{E_A^*(\mathrm{D})}$$ where $\alpha(A,D)$ = Information gain of D (same as in ID3, and $E_A^*(\mathrm{D})$ = splitting information $= -\sum_{j=1}^{m} \frac{\lvert D_j \rvert}{\lvert D \rvert} log_2 \frac{\lvert D_j \rvert}{\lvert D \rvert}$ when D is partitioned into D_1, D_2, … , D_m partitions corresponding to m distinct attribute values of A.	• The attribute A with **maximum** value of $\beta(A,D)$ is selected for splitting. • Splitting information is a kind of normalization, so that it can check the biasness of information gain towards the choosing attributes with a large number of distinct values.

Important Notes

1. **Optimal Decision Tree:** Finding an optimal decision tree is an NP-complete problem. Hence, decision tree induction algorithms employ a heuristic based approach to search for the best in a large search space. Majority of the algorithms follow a greedy, top-down recursive divide-and-conquer strategy to build decision trees.

2. **Missing data and noise:** Decision tree induction algorithms are quite robust to the data set with missing values and presence of noise. However, proper data pre-processing can be followed to nullify these discrepancies.

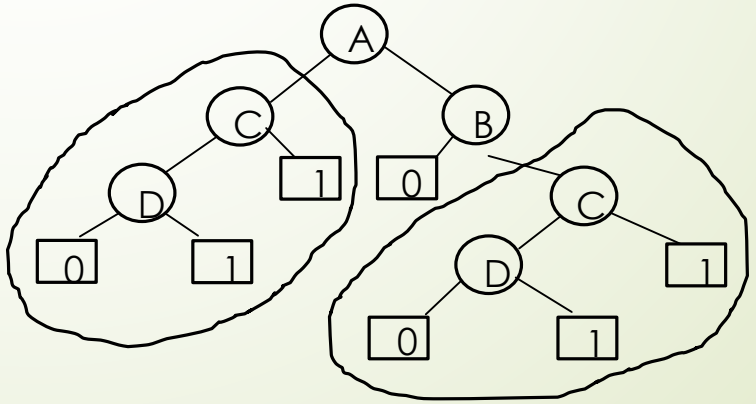
3. **Redundant Attributes:** The presence of redundant attributes does not adversely affect the accuracy of decision trees. It is observed that if an attribute is chosen for splitting, then another attribute which is redundant is unlikely to chosen for splitting.

4. **Computational complexity:** Decision tree induction algorithms are computationally inexpensive, in particular, when the sizes of training sets are large, Moreover, once a decision tree is known, classifying a test record is extremely fast, with a worst-case time complexity of O(d), where d is the maximum depth of the tree.

Important Notes

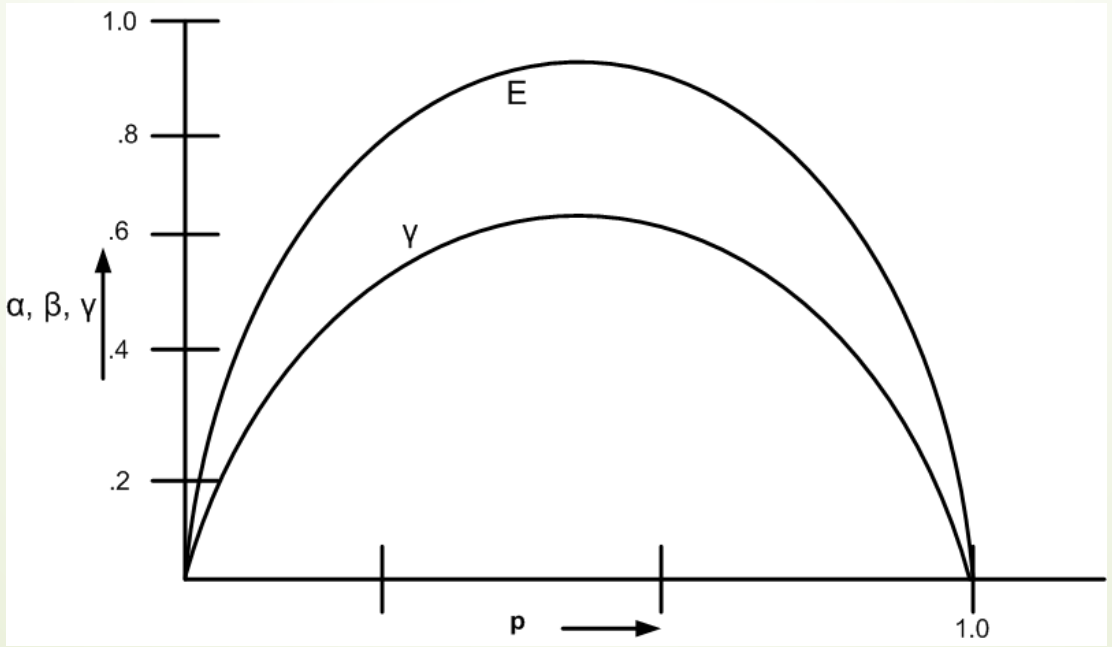
5. **Data Fragmentation Problem:** Since the decision tree induction algorithms employ a top-down, recursive partitioning approach, the number of tuples becomes smaller as we traverse down the tree. At a time, the number of tuples may be too small to make a decision about the class representation, such a problem is known as the data fragmentation. To deal with this problem, further splitting can be stopped when the number of records falls below a certain threshold.

6. **Tree Pruning:** A sub-tree can replicate two or more times in a decision tree (see figure below). This makes a decision tree unambiguous to classify a test record. To avoid such a sub-tree replication problem, all sub-trees except one can be pruned from the tree.



Important Notes

7. **Decision tree equivalence:** The different splitting criteria followed in different decision tree induction algorithms have little effect on the performance of the algorithms. This is because the different heuristic measures (such as information gain (α), Gini index (γ) and Gain ratio (β) are quite consistent with each other); also see the figure below.



Decision Tree Illustration Ends

Thank You !