6-½

UNIT - 3 :: Classification

What is classification? What are the classification models? Explain.

3.1 Basic Concept of Classification: In a Data warehousing, large database has huge amount of raw data, which is analyzed and predicted to retrieve useful information to make decisions.

For example, if we want to know the performance of the JNT University, we classify the student's database based on their performance as *above average, average,* and *below average* students. From this category, if the classification shows *below average* students are more than above average, then the decision making says "to improve JNT university performance".

Definition of Classification: Classification is the task of learning a target function f that maps each attribute set x to one of the predefined class labels y. The target function is also known as a classification model. A classification model can use in two purposes. Example in Note book.

1. Descriptive Modeling: A classification model can serve as an explanatory tool to distinguish b/w objects of different classes. Consider the following example, that a Training set consists descriptive data with based on condition. i.e.
   "If(Age=65 AND Heard rate >70) OR (Age > 60 AND blood pressure >140/70) THEN Heart Problem = yes".

Training set

Age	Heart rate	Blood pressure	Heart problem
65	78	150/70	Yes
37	83	112/76	No
71	67	108/65	No

2. Predictive Modeling: A classification model can also be used to predict the class label of unknown records. Based on the above table, a classification model can be treated as a black box that automatically assigns a class label when presented with the attribute set of an unknown record.

Prediction set

Age	Heart rate	Blood pressure	Heart problem
43	98	147/89	?
65	58	106/63	?
84	77	150/65	?

Classification techniques are most suited for predicting or describing data sets with binary or nominal categories. They are less effective for ordinal categories. For example, to classify a person as a member of high, medium, low income.

Q. Give an explanation for solving a classification problem:

3.2 General Approach to solving a Classification Problem:

- A *Classification technique* is a systematic approach to building classification models from an input data set.
- The following figure shows a general approach for solving classification problem. In this, first a training set consisting of records whose class labels are known.

- Using the training set building a classification model and find unknown class labels. Second, using a training set can desire the class label with unknown data in the test set.

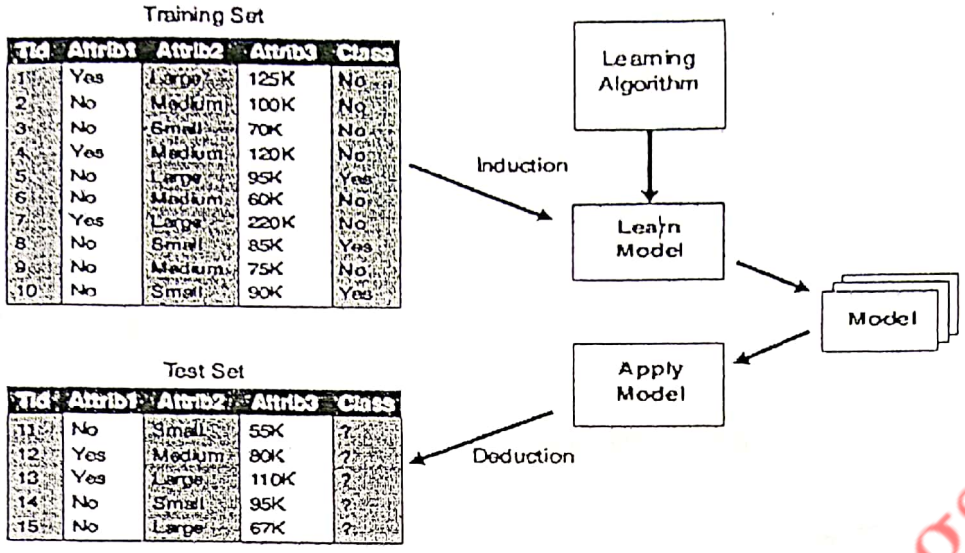


Figure 4.3. General approach for building a classification model.

3.2.1. Types of Classification techniques:

- Classification techniques are Decision tree classifiers, rule-based classifiers, neural networks classifiers, support vector machines classifiers, and Navie Bayes classifiers.
- The classification techniques using a learning algorithm to identify a model that best fits the relationship b/w the attribute set and class label of the input data.
- Therefore, the model which is generated by *a learning algorithm* should fit the input data well and correctly predict the class labels of previously unknown records. That is, a learning algorithm is to build models with good generalization capability.

3.3 Decision Tree Induction:

3.3.1 How a Decision Tree Induction Works: A Decision Tree Induction has 3 types of nodes. They are.

- A *root node* that has no incoming edges and zero or more outgoing edges.
- *Internal nodes* each of which has exactly one incoming edge and two or more outgoing edges.
- *Leaf* or *terminal* nodes, each of which has exactly one incoming edge and no outgoing edges.

Consider a following table that has "training set" with known class label.

Name	Body Temperature	Gives Birth	Aquatic Creature	Aerial Creature	Has legs	Class label
Human	warm-blooded	yes	no	no	yes	mammal
Python	cold- blooded	no	no	no	no	reptile
Salmon	cold- blooded	no	yes	no	no	fish
Whale	warm- blooded	yes	yes	no	no	mammal
Frog	cold- blooded	no	semi	no	yes	amphibian
Bat	warm- blooded	yes	no	no	yes	mammal
Pigeon	warm- blooded	no	no	yes	yes	bird
Cat	warm- blooded	yes	yes	no	yes	mammal
penguin	warm- blooded	no	semi	yes	no	bird
eel	cold- blooded	no	yes	no	no	fish

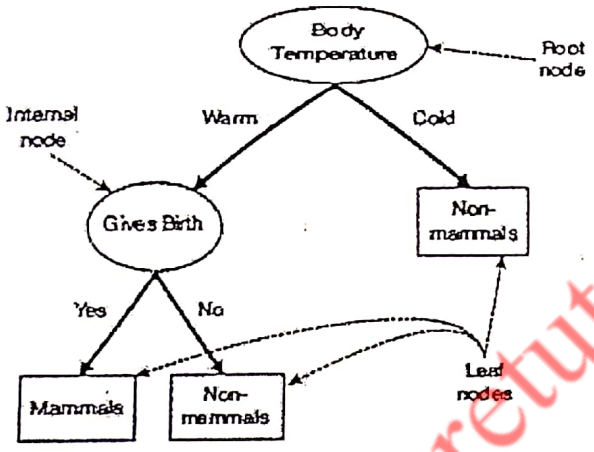
- In the above table, the *class label* is classified into 5 distinct groups of species (related item). But, they assigned into two categories such as *mammal* and *non-mammal*.
  - In this the first question is whether the species in cold or warm blooded. If it is cold-blooded, then it is definitely not a mammal otherwise it is either a bird or a mammal.
  - The second question is if they give birth or not. If they gives birth then they definitely mammals otherwise they are not mammals.

Thus, using above association rules we can solve the classification problem for the given input data.

For example,

Name	Body Temperature	Gives Birth	Aquatic Creature	Aerial Creature	Has legs	Class label
Sparrow	warm-blooded	no	no	yes	yes	?
pulasa	cold- blooded	no	no	no	no	?

- In a decision tree, each leaf node is assigned a class label. The non-mammal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics. For example, the root node shown in figure that uses the attribute *Body Temperature* to separate "warm-blooded" from "cold-blooded".



R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds

R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals

R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

Figure A decision tree for the mammal classification problem.

- In this, all cold-blooded are non-mammals and created *non-mammals* as a leaf node at the right child of the root node.
- If they are *warm-blooded* then a subsequent attribute "gives birth" is used to distinguish mammals from other warm-blooded creatures (i.e. they are mostly birds).

3.3.2. How to Build a Decision Tree: The decision tree is build by using efficient decision tree algorithm. Such as Hunt's algorithm. Using this algorithm, many decision tree induction algorithms were developed. They are ID3, 4.5, and CART.

Hunt's Algorithm: In Hunt's algorithm, a decision tree is constructed in a recursive fashion by partitioning the training records into successively purer subsets.

→ Let D_t be the set of training records that are associated with node 't' any $y = \{y_1, y_2, y_3, \ldots y_c\}$ be the class labels.

Definition:

Step 1: If all the records in D_t belong to the same class y_t then t is a leaf node labeled as y_t.

Step 2: If D_t contains records that belongs to more than one class, an "attribute test condition" is selected to partition the records into smaller subsets.

- A child node is created for each outcome of the test condition and the records in D_t are distributed to the children based on the outcomes.
- Thus, the algorithm is then recursively applied to each child node.

Consider the following training set for predicting borrowers who will default on loan payments.

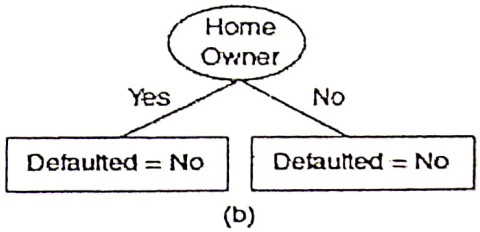
	binary	categorical	continuous	class
Tid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

From the table predicting outcome data whether a loan applicant will repay the loan or not. Using the training set table, the problem can be constructed by examining the records of previous borrowers.

- In this each record contains the personal information of a borrower along with a class label indicating whether the borrower has defaulted on loan payments.
- The initial tree for the classification problem contains a single node with class label "defaulted = No". It means that most of the borrowers successfully repaid their loans.

Fig (a) Defaulted = No

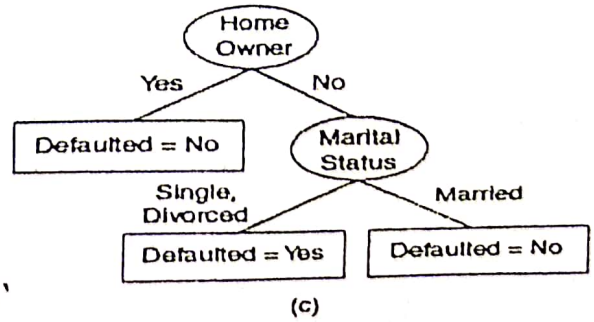
- The records are subsequently divided into smaller subsets based on the outcomes of the "Home Owner" test condition. In this if Home Owner = Yes then that has all records in same class.



(b)

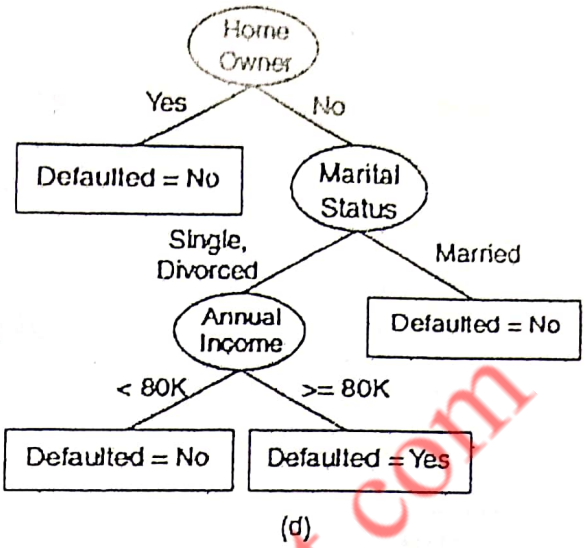
- If test condition is No [i.e. "Home Owner = No"] then the Hunt's algorithm apply the step recursively. Because Home Owner is No has all records are in different class.

→In the tree, the left child of the root node is labeled "Defaulted = Yes" then it is not extended recursively.

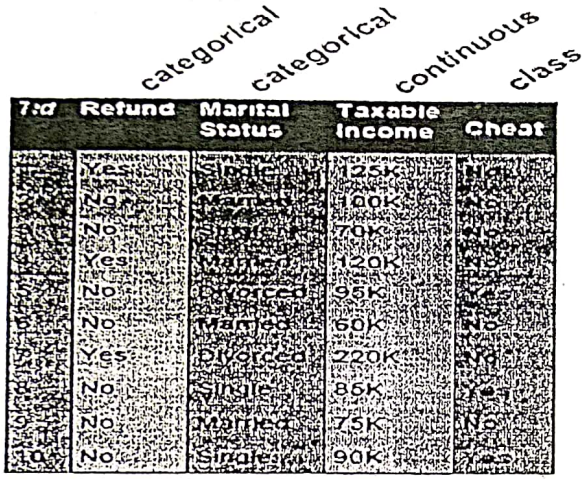


(c)

→The right child of the root node is continued by applying the recursive step of Hunt's algorithm until all the records belongs to the same class. The tree result the recursive steps as shown in fig (d).

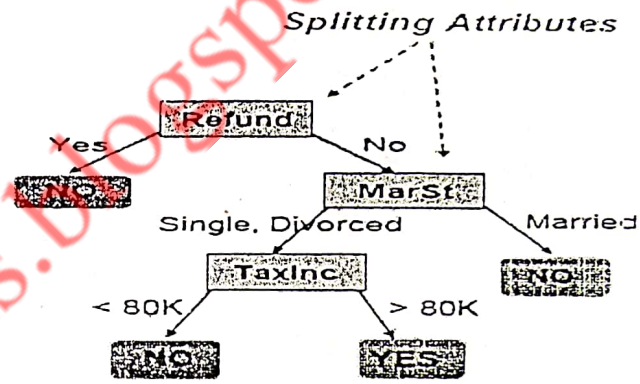


(d)



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

Training Data



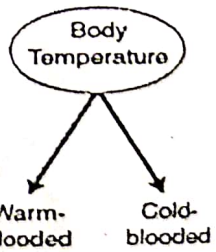
Model: Decision Tree

3.4. Methods for Expressing Test Conditions Attribute test conditions:

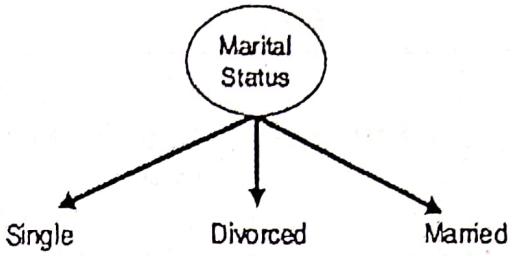
Decision tree induction algorithm expressing an attribute test condition and it can be done different attribute type.

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

1. **Binary attribute:** The test condition for a binary attribute generate two potential outcomes. Shown in fig.

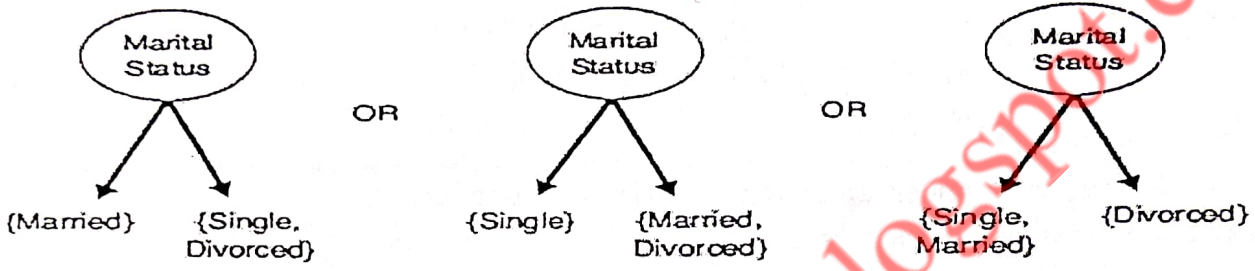


2. Nominal Attribute: A nominal attribute can have many related values.
   - Its test condition can be expressed in two ways. 1. Multi-way, 2. Two-way.
   - For a *multi-way split* the number of outcomes depends on the number of distinct values for the corresponding attribute.
     - For example, if marital status has three distinct values such single, married and divorced. Its test condition will produce a three-way split.



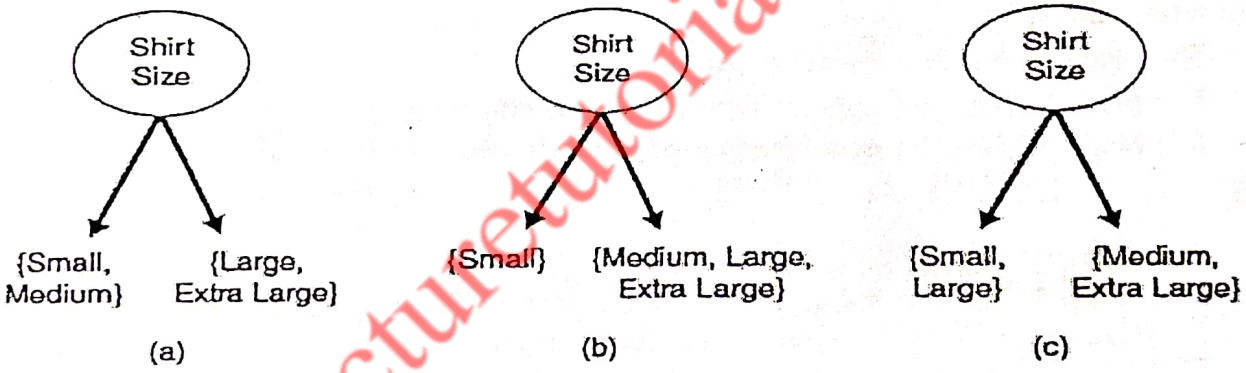
(a) Multiway split

   - The test condition can be split into two-ways i.e. binary attribute. This is shown in fig.



(b) Binary split {by grouping attribute values}

3. Ordinal Attribute (Group): Ordinal attributes can also produce binary (or) multi-way splits. Ordinal attribute values can be grouped as long as the grouping does not violate the order

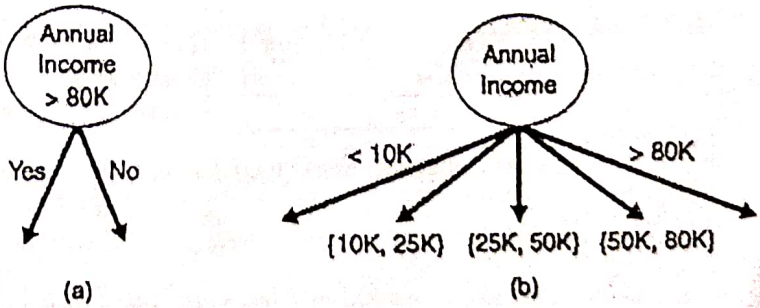


Different ways of grouping ordinal attribute values.

property of the attribute values. The figure shows two-way split.

4. Continuous Attributes: For continuous attributes, the test condition can be expressed as a comparison test with binary outcomes (yes or no)

In Multi-way, the test condition is done based on Annual Income in the given ranges.



Test condition for continuous attributes.

3.5. Measures for selecting the Best Split:

There are many measures that can be used to determine the best way to split the records. These measures are defined in terms of the class distribution of the records before and after splitting.

→Let p(i/t) denote the fraction of records belonging to class *i* at a given node *t*.

In a two-class problem, the class distribution at any node can be written as (p_0, p_1), where $p_1 = 1 - p_0$.

Measures :

$$\text{1. Entropy}(t) = -\sum_{i=0}^{c-1} p(i/t) \log_2 p(i/t)$$

$$\text{2. Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i/t)]^2$$

3. Classification error(t) = 1 – max [p(i/t)]

1. Compute impurity measure (P) before splitting.
2. Compute impurity measure (M) after splitting.
3. Compute impurity measure of each child node.
   - M is the weighted impurity of children
4. Choose the attribute test condition that produces the highest gain Gain = P – M or equivalently, lowest impurity measure after splitting (M).

- Greedy approach:
  - Nodes with purer class distribution are preferred
- Need a measure of node impurity:

C0: 5
C1: 5

High degree of impurity

C0: 9
C1: 1

Low degree of impurity

In the below examples,

→ The Node N1 has the lowest impurity value.
→ The Node N2 has the approximately Low impurity value.
→ The Node N3 has the equal number of records. So is high impurity.

Eg:

Node N_1	Count
Class=0	0
Class=1	6

Gini = $1 - (0/6)^2 - (6/6)^2 = 0$
Entropy = $-(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$
Error = $1 - \max[0/6, 6/6] = 0$

Node N_2	Count
Class=0	1
Class=1	5

Gini = $1 - (1/6)^2 - (5/6)^2 = 0.278$
Entropy = $-(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$
Error = $1 - \max[1/6, 5/6] = 0.167$

Node N_3	Count
Class=0	3
Class=1	3

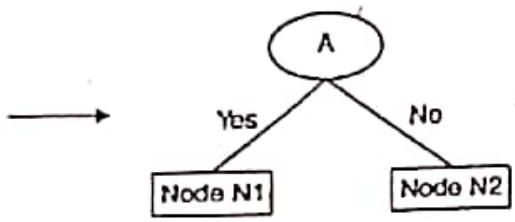
Gini = $1 - (3/6)^2 - (3/6)^2 = 0.5$
Entropy = $-(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$
Error = $1 - \max[3/6, 3/6] = 0.5$

Example to illustrate the best split using following fig.

→ The class distribution before splitting is (0.500). Class distribution after splitting is (0.333)

Before Split

	Parent
C1	6
C2	6
Gini = 0.500	



After Split

	N1	N2
C0	4	2
C1	3	3
Gini = 0.486		

Before Split:

→ $1 - (6/12)^2 - (6/12)^2$

→ 1 - 0.25 - 0.25

→ 0.5

After Split:

N1 → $1 - (4/7)^2 - (3/7)^2$

N1 → $1 - (0.571429)^2 - (0.428571)^2$

N1 → 1 - (0.3265) - (0.1837)

N1 → 0.489

N2 → $1 - (2/5)^2 - (3/5)^2$

N2 → $1 - (0.4)^2 - (0.6)^2$

N2 → 1 - (0.16) - (3/5)

N2 → 0.480

The Gini Index of entire tree → 7/12 * 0.489 + 5/12 * 0.480 → 0.486

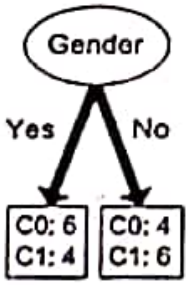
Spliting of Binary Attribute.

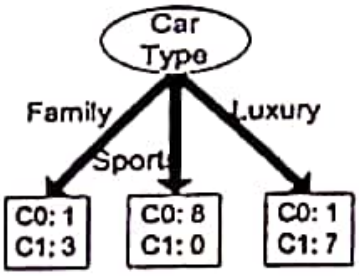
3.6. How to determine the Best Split:

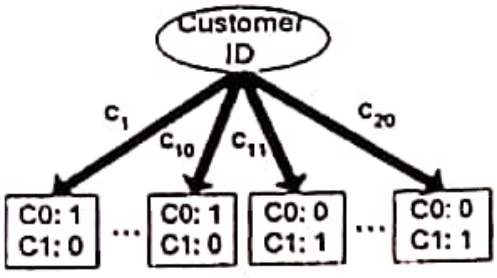
Consider the following example that explains the best split.

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



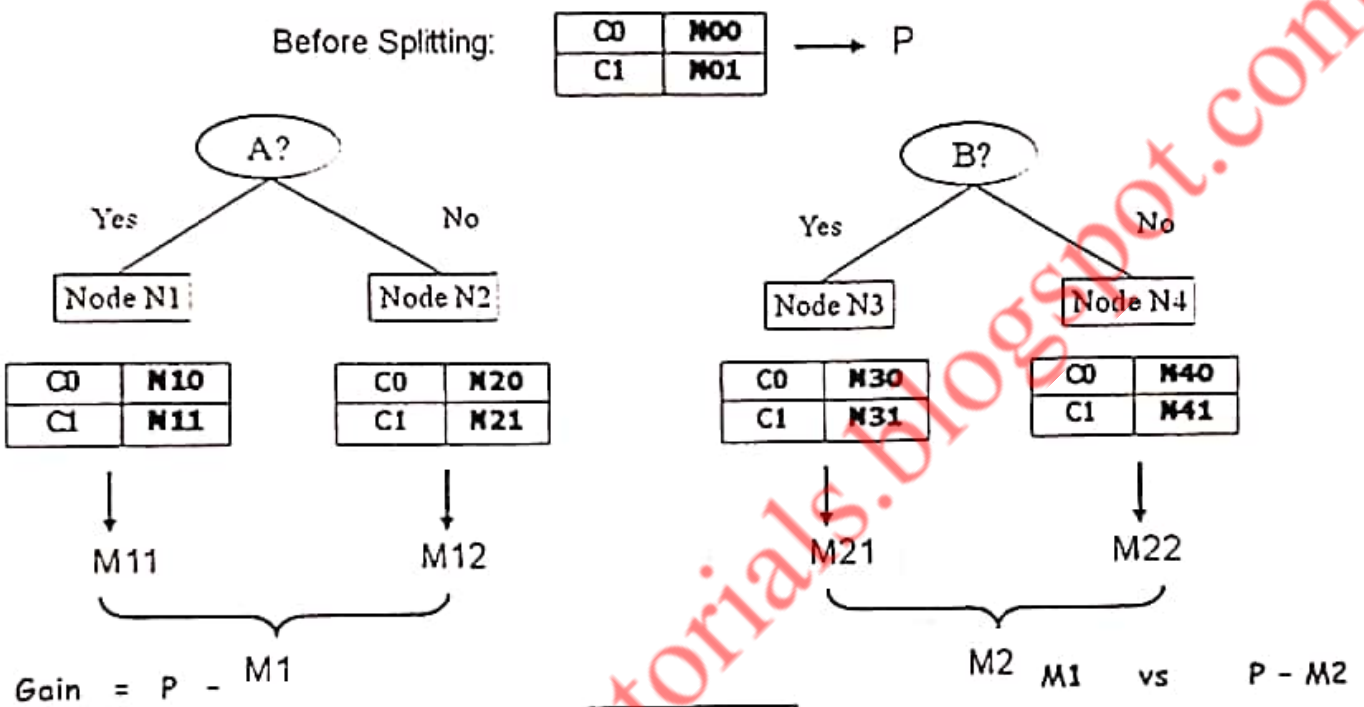


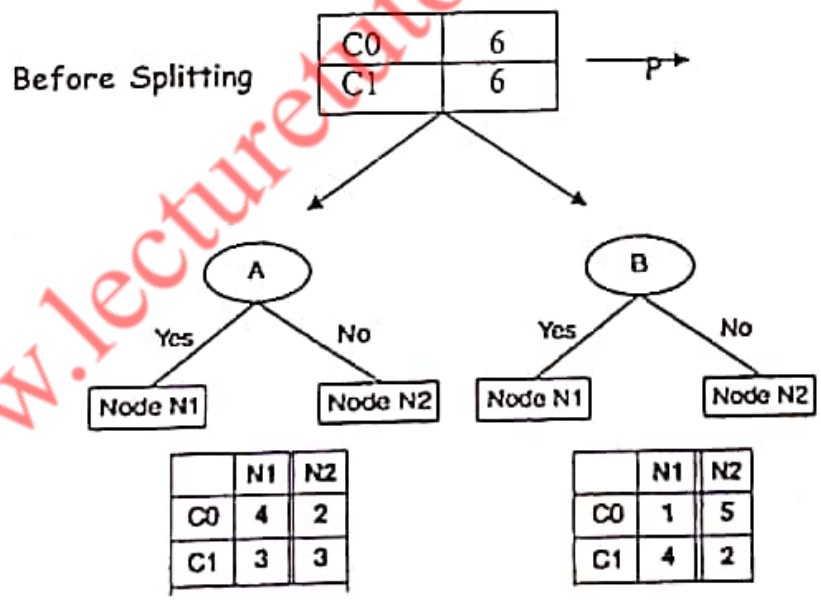


Which test condition is the best?

In the above example,

- The Gender = yes then M has 6 in class label C0 and 4 in class label C1. Similarly, the gender = no then F has 4 in class label C0 and 6 in class label C1.
- The Car Type = Family then C0 has 1 and C1 has 3. Therefore, the Car Type = sports then C0 has 8 and C1 has 0. The Car Type = Luxury then C0 has 1 and C1 has 7.
- The Customer id = 1 then C0 has 1 and C1 has 0. Similarly remaining.





3.7. How to determine the performance of test condition: The test condition can be performed by using to compare the degree of impurity of the parent node (before splitting) with the degree of impurity of the child nodes (after splitting). So, the gain can be used to determine the goodness of a split.

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

→ Where $I(.)$ is the impurity measure of a given node.
→ N is the total number of records at the parent node.
→ K is the number of attribute values and
→ $N(v_j)$ is the number of records associated with the child node v_j.

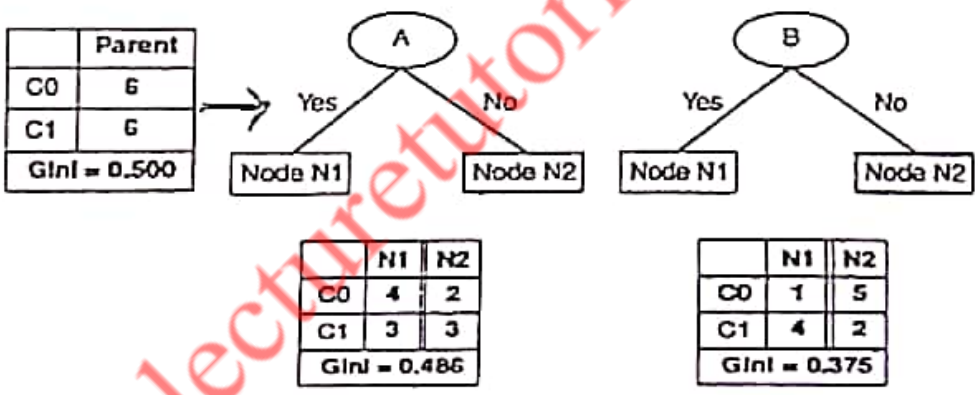
Splitting the attribute values at various types:

→ Splitting of Binary Attributes: The attribute which is having two category values then they will split into two smaller subsets. Before Splitting, the Gini index is 0.5. It is having equal number of records from both classes.

If an attribute A is choosen to split the data as: the Gini index for node N1 is 0.4898 and for node N2 is 0.480.

The weighted average of the Gini index is (7/12) x 0.4898 + (5/12) x 0.480 = 0.486.

If an attribute B is choosen to split the data as: the Gini index B is 0.375

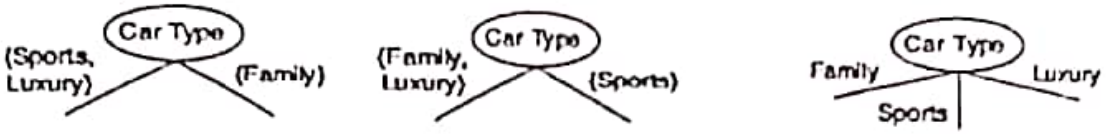


Splitting binary attributes.

→ Splitting of Nominal Attributes: The attribute which having two or more distinct category values then they will split into two or multi way split. This can be split into binary grouping of *car type attribute* with 3 categories such as (sports, luxury, and family).

The Gini index of Car type (sports, luxury) is 0.4922 and
the Gini index of {family} is 0.3750.

The weighted average Gini index for the grouping is equal to 16/20 x 0.4922 + 4/20 x 0.3750 = 0.468.



	Car Type	
	{Sports, Luxury}	{Family}
C0	9	1
C1	7	3
Gini	0.468	

	Car Type	
	{Sports}	{Family, Luxury}
C0	8	2
C1	0	10
Gini	0.167	

	Car Type		
	Family	Sports	Luxury
C0	1	8	1
C1	3	0	7
Gini	0.163		

(a) Binary split

(b) Multiway split

Splitting nominal attributes.

Similarly, Car type (sports) and (family, luxury) is 0.167 and
Car type (family) , (sports) , (luxury) is 0.163.

→ Splitting of Continuous Attributes: The attribute which having range of values. For example, Consider the following example that shown the test condition Annual Income <= v is used to split the training records for the loan default classification problem.

Class	No		No		No		Yes		Yes		Yes		No		No		No		No		No	
Annual Income																						
Sorted Values →	60		70		75		85		90		95		100		120		125		220			
Split Positions →	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
Gini	0.420		0.400		0.375		0.343		0.417		0.400		0.300		0.343		0.375		0.400		0.420	

Splitting continuous attributes.

3.8. What is Gain Ratio? Explain with an example.

Information gain is generally used for attributes which have large set of outcomes. But certain attributes like "Fruit", have only one outcome for each partition and thus have maximum information gain for all attributes. In such cases we use gain ratio. This gain ratio is calculated as

- Gain ratio (Att) = $\frac{\text{Gain (Att)}}{\text{Split information (Att)}}$

Where Split information (Att) is = $-\sum_{i=1}^{n} p(vi) \log_2 p(vi)$ Where 'n' is the total number of splits.

For example, if each attribute value has the same number of records, then p(vi) = 1/k and the split information would be equal to $\log_2 k$.

Consider the following example that explain Gain ratio.

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

→ Class P: buys_computer = "yes"

→ Class N: buys_computer = "no"

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

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

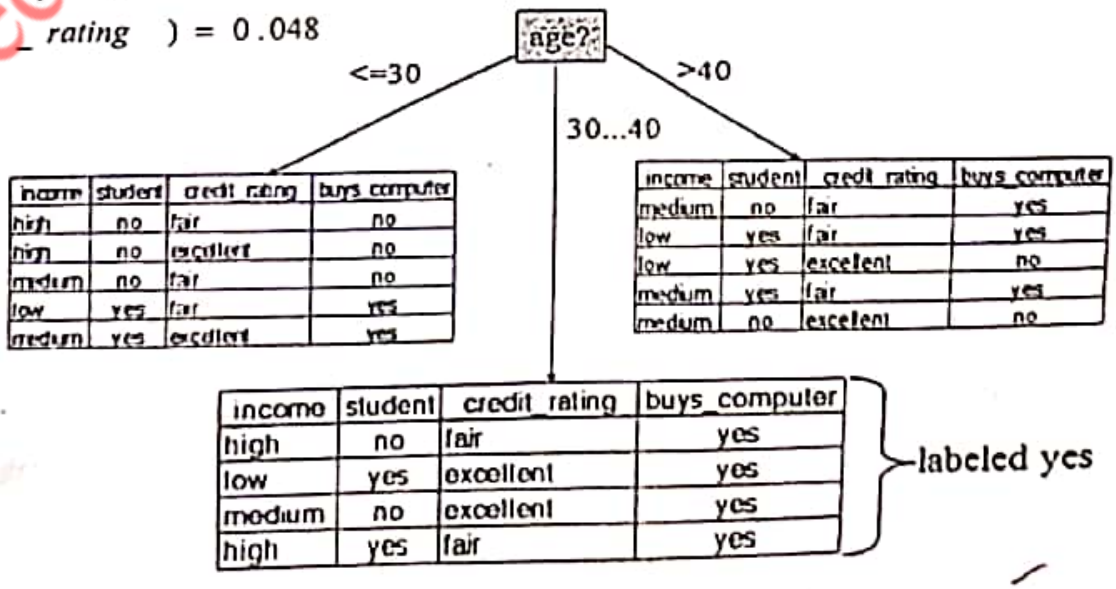
$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

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

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

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



BVSR ENGIN

.9. Algorithm for Decision Tree Induction:

→ A skeletion decision tree induction algorithm called TreeGrowth, shown in Algorithm.
→ The input to this algorithm consists of the training records E and the attribute set F.
→ The algorithm works by recursively selecting the best attribute to split the data and expanding the
leaf nodes of the tree (steps 11 & 12) until the stopping _con is true.

```
GenDecTree(Sample E, Features F)
Step  1. If stopping_condition(S,F) = true then
      2.    a. leaf = createNode()
      3.    b. leaf.label= Classify(S)
      4.    c. return leaf
      5. else
      6. root = createNode()
      7. root.test_condition = findBestSplit(S,F)
      8. V = {v| v a possible outcome of root.test_condition}
      9. for each value veV:
      10.    a. Sv: = {s | root.test_condition(s) = v and s ∈ S};
      11.    b. child = TreeGrowth(Sv ,F) :
      12.    c. Add child as a descent of root and label the edge (root→child) as v
      13.    end for
      14.    end if
      15.    return root
```

The detail of each function is explained as

1. The *createNode()* function extends the decision tree by creating a new node. A node in the decision tree has either a test condition (denoted as *node.test_cond)* or a class label (denoted as *node.label*).
2. The *find_best_split()* function determines which attribute should be selected as the test condition for splitting the training records. The " test condition " depends on measures such entropy, Gini index, and the x^2 statistic.
3. The *classify()* function determines the class label to be assigned to a leaf node. For each leaf node t, let *p(i/t)* denote the fraction of training records from class I associated with the node t.

   i.e. leaf.label = argmax p(i/t)

4. The *stopping_cond()* function is used to terminate the tree-growing process by testing whether all the records have either the same class label or the same attribute values.

After building the decision tree, a tree-pruning step can be performed to reduce the size of the decision tree.

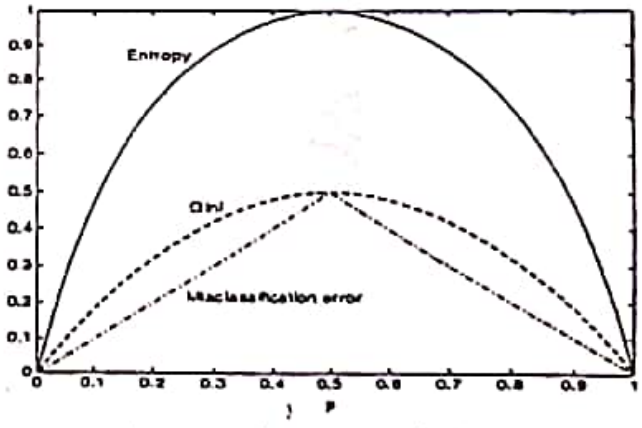
3.10. Characteristics of Decision Tree Induction:

1. Decision tree induction is a nonparametric approach for building classification models.
2. Techniques developed for constructing decision trees. But they are computationally inexpensive.
3. Small sized trees are relatively easy to interpret.
4. Decision trees provide an expressive representation for learning discrete-valued functions.
5. Decision tree algorithms are quite robust to the presence of noise.
6. Decision tree algorithm employ a top-down, recursive portioning approach.
7. Sub-trees can be replicated multiple times in 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

Comparison of Best Split: The figure compares the values of the impurity measures for binary classification problems. Observe that all three measures attain their maximum value when the class distribution is uniform (i.e. when p = 0.5).



Comparison among the impurity measures for binary classification problems.

Exercises: Consider the following Confusion matrix for classification problem with 2 class categories. Such as Class 0 & 1.

		Predicted Class	
		$Class = 1$	$Class = 0$
Actual Class	$Class = 1$	f_{11}	f_{10}
	$Class = 0$	f_{01}	f_{00}

- The performance of above classification model is calculated based on the test records correctly and incorrectly predicted by the model. From the above "confusion matrix", each entry of f_{ij} is denoted the number of records from class i predicted to be of class j. Therefore, the total number of records are correctly predicted by the model is $(f_{11} + f_{00})$ and the total number of records are incorrectly predicted by the model is $(f_{10} + f_{01})$.
- The Performance of model for above matrix with accuracy is expressed as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{00} + f_{10} + f_{01}}$$

- The Performance of mode above matrix with error rate is expressed as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{00} + f_{10} + f_{01}}$$

1. Draw the full decision tree for the parity function of four Boolean attributes, A, B, C, and D. Is it possible to simplify the tree?

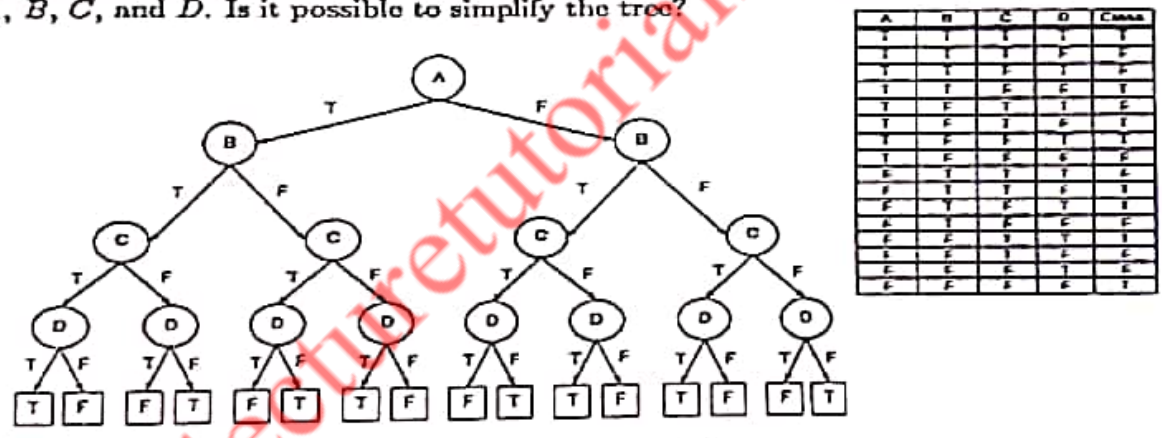


Figure 4.1. Decision tree for parity function of four Boolean attributes.

2. Consider the training examples shown in Table 4.1 for a binary classification problem.

Table 4.1. Data set for Exercise 2.

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

(a) Compute the Gini index for the overall collection of training examples.

Answer:

Gini $= 1 - 2 \times 0.5^2 = 0.5$.

(b) Compute the Gini index for the Customer ID attribute.

Answer:

The gini for each Customer ID value is 0. Therefore, the overall gini for Customer ID is 0.

(c) Compute the Gini index for the Gender attribute.

Answer:

The gini for Male is $1 - 2 \times 0.5^2 = 0.5$. The gini for Female is also 0.5. Therefore, the overall gini for Gender is $0.5 \times 0.5 + 0.5 \times 0.5 = 0.5$.

Table 4.2. Data set for Exercise 3.

Instance	a_1	a_2	a_3	Target Class
1	T	T	1.0	+
2	T	T	6.0	+
3	T	F	5.0	−
4	F	F	4.0	+
5	F	T	7.0	−
6	F	T	3.0	−
7	F	F	8.0	−
8	T	F	7.0	+
9	F	T	5.0	−

(d) Compute the Gini index for the Car Type attribute using multiway split.

Answer:

The gini for Family car is 0.375, Sports car is 0, and Luxury car is 0.2188. The overall gini is 0.1625.

(e) Compute the Gini index for the Shirt Size attribute using multiway split.

Answer:

The gini for Small shirt size is 0.48, Medium shirt size is 0.4898, Large shirt size is 0.5, and Extra Large shirt size is 0.5. The overall gini for Shirt Size attribute is 0.4914.

(f) Which attribute is better, Gender, Car Type, or Shirt Size?

Answer:

Car Type because it has the lowest gini among the three attributes.

(g) Explain why Customer ID should not be used as the attribute test condition even though it has the lowest Gini.

Answer:

The attribute has no predictive power since new customers are assigned to new Customer IDs.

3. Consider the training examples shown in Table 4.2 for a binary classification problem.

(a) What is the entropy of this collection of training examples with respect to the positive class?

Answer:

There are four positive examples and five negative examples. Thus, $P(+) = 4/9$ and $P(-) = 5/9$. The entropy of the training examples is $-4/9 \log_2(4/9) - 5/9 \log_2(5/9) = 0.9911$.

(b) What are the information gains of a_1 and a_2 relative to these training examples?

Answer:

For attribute a_1, the corresponding counts and probabilities are:

a_1	+	-
T	3	1
F	1	4

The entropy for a_1 is

$$\frac{4}{9}\Big[-(3/4)\log_2(3/4)-(1/4)\log_2(1/4)\Big] + \frac{5}{9}\Big[-(1/5)\log_2(1/5)-(4/5)\log_2(4/5)\Big] = 0.7616.$$

Therefore, the information gain for a_1 is $0.9911 - 0.7616 = 0.2294$.

For attribute a_2, the corresponding counts and probabilities are:

a_2	+	-
T	2	3
F	2	2

The entropy for a_2 is

$$\frac{5}{9}\Big[-(2/5)\log_2(2/5)-(3/5)\log_2(3/5)\Big] + \frac{4}{9}\Big[-(2/4)\log_2(2/4)-(2/4)\log_2(2/4)\Big] = 0.9839.$$

Therefore, the information gain for a_2 is $0.9911 - 0.9839 = 0.0072$.

(c) For a_3, which is a continuous attribute, compute the information gain for every possible split.

Answer:

a_3	Class label	Split point	Entropy	Info Gain
1.0	+	2.0	0.8484	0.1427
3.0	-	3.5	0.9885	0.0026
4.0	+	4.5	0.9183	0.0728
5.0	-			
5.0	-	5.5	0.9839	0.0072
6.0	+	6.5	0.9728	0.0183
7.0	+			
7.0	-	7.5	0.8889	0.1022

The best split for a_3 occurs at split point equals to 2.

(d) What is the best split (among a_1, a_2, and a_3) according to the information gain?

Answer:

According to information gain, a_1 produces the best split.

(e) What is the best split (between a_1 and a_2) according to the classification error rate?

Answer:

For attribute a_1: error rate = 2/9.

For attribute a_2: error rate = 4/9.

Therefore, according to error rate, a_1 produces the best split.

(f) What is the best split (between a_1 and a_2) according to the Gini index?

Answer:

For attribute a_1, the gini index is

$$\frac{4}{9}\left[1-(3/4)^2-(1/4)^2\right]+\frac{5}{9}\left[1-(1/5)^2-(4/5)^2\right]=0.3444.$$

For attribute a_2, the gini index is

$$\frac{5}{9}\left[1-(2/5)^2-(3/5)^2\right]+\frac{4}{9}\left[1-(2/4)^2-(2/4)^2\right]=0.4889.$$

Since the gini index for a_1 is smaller, it produces the better split.

5. Consider the following data set for a binary class problem.

A	B	Class Label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−

(a) Calculate the information gain when splitting on A and B. Which attribute would the decision tree induction algorithm choose?

Answer:

The contingency tables after splitting on attributes A and B are:

	$A=T$	$A=F$
+	4	0
−	3	3

	$B=T$	$B=F$
+	3	1
−	1	5

The overall entropy before splitting is:

$$E_{orig} = -0.4 \log 0.4 - 0.6 \log 0.6 = 0.9710$$

The information gain after splitting on A is:

$$\begin{aligned} E_{A=T} &= -\frac{4}{7}\log\frac{4}{7} - \frac{3}{7}\log\frac{3}{7} = 0.9852 \\ E_{A=F} &= -\frac{3}{3}\log\frac{3}{3} - \frac{0}{3}\log\frac{0}{3} = 0 \\ \Delta &= E_{orig} - 7/10 E_{A=T} - 3/10 E_{A=F} = 0.2813 \end{aligned}$$

The information gain after splitting on B is:

$$\begin{aligned} E_{B=T} &= -\frac{3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4} = 0.8113 \\ E_{B=F} &= -\frac{1}{6}\log\frac{1}{6} - \frac{5}{6}\log\frac{5}{6} = 0.6500 \\ \Delta &= E_{orig} - 4/10 E_{B=T} - 6/10 E_{B=F} = 0.2565 \end{aligned}$$

Therefore, attribute A will be chosen to split the node.

(b) Calculate the gain in the Gini index when splitting on A and B. Which attribute would the decision tree induction algorithm choose?

Answer:

The overall gini before splitting is:

$$G_{orig} = 1 - 0.4^2 - 0.6^2 = 0.48$$

The gain in gini after splitting on A is:

$$\begin{aligned} G_{A=T} &= 1 - \left(\frac{4}{7}\right)^2 - \left(\frac{3}{7}\right)^2 = 0.4898 \\ G_{A=F} &= 1 = \left(\frac{3}{3}\right)^2 - \left(\frac{0}{3}\right)^2 = 0 \\ \Delta &= G_{orig} - 7/10 G_{A=T} - 3/10 G_{A=F} = 0.1371 \end{aligned}$$

The gain in gini after splitting on B is:

$$\begin{aligned} G_{B=T} &= 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2 = 0.3750 \\ G_{B=F} &= 1 = \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 = 0.2778 \\ \Delta &= G_{orig} - 4/10 G_{B=T} - 6/10 G_{B=F} = 0.1633 \end{aligned}$$

Therefore, attribute B will be chosen to split the node.