UNIT IV

CHAPTER 4

Classification

Syllabus Topics

Decision trees - Overview, general algorithm, decision tree algorithms, evaluating a decision tree. Naïve Bayes -Bayes' Algorithm, Naïve Bayes' Classifier, smoothing, diagnostics. Diagnostics of classifiers, additional classification methods.

4.1 Decision Trees : Introduction

Q. 4.1.1 Write note on decision Tree. (Refer section 4.1) **(8 Marks)**

- A **decision tree** which is also known as **prediction tree** refers a tree structure to mention the sequences of decisions as well as consequences.
- Considering the input $X = \{x_1, x_2, ... x_n\}$, the aim is to predict a response or output variable Y.
- Each element in the set $\{x_1, x_2, ...x_n\}$ is known as **input variable**.
- It is possible to achieve the prediction by the process of building a decision tree which has test points as well as branches.
- At each test point, it is decided to select a particular branch and traverse down the tree.
- Ultimately, a final point is reached, and it will be easy to make prediction.
- In a decision tree, all the test points exhibit testing specific input variables (or attributes), and the developed decision tree is represented by the branches.
- Because of flexibility as well as simple visualization, decision trees are most probably deployed in data mining applications for the purpose of classification.
- In the decision tree, the input values are considered as categorical or continuous.
- A structure of test points (known as nodes) and branches is established by the decision tree by which the decision being made will be represented.
- Leaf node is the one which do not have further branches. The returning value of leaf nodes is class labels while in some cases they return the probability scores.
- It is possible to convert decision tree into a set of decision rules.
- See the following example in which income and mortgage_amount are input variables, and the response is nothing but the output variable default which posses the a probability score.

IF income < $50,000 **AND** mortgage_amount > $100K
THEN default = True **WITH PROBABILITY** 75%

- There are two types of Decision trees: **classification trees** and **regression trees**
- Classification trees are generally applied to output variables which are categorical and mostly binary in nature, for example yes or no, sale or not, and so on.
- Whereas regression trees are applied to output variables which are numeric or continuous, for example predicted price of a consumer good.
- In variety of situations, it is possible to apply decision tree. It is easy to represent them in a visual way, and the analogous decision rules are simply straightforward.

- Also as the result is a sequence of logical if-then statements, there is no any presence of underlying assumption regarding a linear or nonlinear relationship between the input variables and the response variable.

Syllabus Topic : Decision Trees- Overview

4.1.1 Decision Trees : Overview

- In the Fig. 4.1.1 we can observe an example in which a decision tree is used to predict the customer's possibility of buying a product.
- The outcome of a decision is referred by the term branch and is visualized as a line which joins two nodes.
- In the situation when a decision is numerical, the "greater than" branch is generally positioned on the right while the "less than" branch is positioned on the left.
- Based on variable's nature, there may be need to include an "equal to" component.
- Decision or test points are represented by the internal nodes. The internal nodes are considered as references to input variables or attributes.
- The top internal node is known as the root.
- In the Internal nodes, the decision tree is a binary tree in which each internal node has maximum two branches. The branching of a node is called as a split.

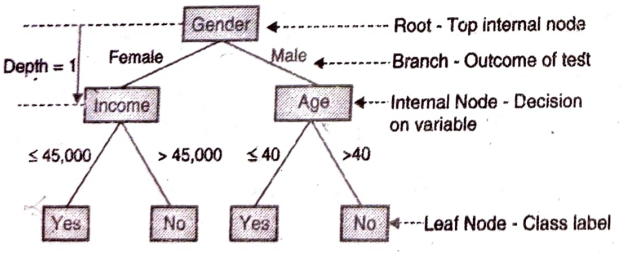

Fig. 4.1.1 : Example of Decision Tree

- Sometimes there may be more than two branches to decision trees which are stemming from a node.
- For example, if an input variable *Weather* is categorical and posses 3 options; *Sunny, Rainy*, and *Snowy*, then the subsequent node *Weather* in the decision tree may contain 3 branches captioned as as Sunny, Rainy, and Snowy, respectively.
- The **depth** of a node is nothing but the minimum number of steps which are necessary to reach the node from the root.
- In Fig. 4.1.1 the depth of nodes Income and Age is one, and the leaf nodes without child have a depth of two.
- **Leaf nodes** are located at the end of the last branches on the tree. Class labels are represented by them that is the outcome of all the previous decisions.
- In the path from the root to a leaf node, there is a series of decisions which are made at several internal nodes.
- In Fig. 4.1.1 the root node is basically divided into two branches having a Gender test.
- All the records whose gender is male are in the right branch while the records having gender as female present in left branch to create the depth 1 of internal nodes.
- Each and every internal node efficiently works as the root of a subtree, and a most excellent test considered for all the nodes is the independent determination of the other internal nodes.
- The LHS (left hand side) internal node is divided on a question depending upon the Income variable to produce leaf nodes at depth 2, while the RHS (right-hand side) is divided on a question depending upon the Age variable.
- The decision tree in Fig. 4.1.1 illustrates that females who have income less than or equal to $45,000 and males who are forty years old or younger are termed as individuals who would purchase the product.
- In the process of traversing the tree, age is not important for females, and income is not important for males.
- Now to understand working of decision tree, we will consider a case of a bank which is eager to market its term deposit products like Certificates of Deposit to the suitable customers.
- Provided the details about the clients and their reactions to prior campaign which was conducted by phone calls, the aim of bank is to predict which clients may be ready for term deposit.

- Here we are considering a dataset of a bank on directed marketing campaigns.
- Fig. 4.1.2 displays a subset of the updated bank marketing dataset. In this dataset, there are two thousand instances which are randomly drawn from the original dataset, and all the instances are corresponding to customers.
- For simplification purpose, the subset only keeps the categorical variables as follows:
  - job,
  - marital status,
  - education level,
  - if the credit is in default,
  - if there is a housing loan,
  - if the customer currently has a personal loan, contact type result of the previous marketing campaign contact (poutcome),
  - if the client actually subscribed to the term deposit.
- Attributes (1) through (8) are considered here as input variables and (9) is outcome.
- The outcome subscribed is either yes which indicates that the customer is ready to subscribe to the term deposit or no which indicates that the customer will not subscribe to the term deposit.
- All the variables listed here are categorical.

	job	marital	education	default	housing	loan	contact	poutcome	subscribed
1.	management	single	tertiary	no	yes	no	cellular	unknown	no
2.	entrepreneur	married	tertiary	no	yes	yes	cellular	unknown	no
3.	services	divorced	secondary	no	no	no	cellular	unknown	yes
4.	management	married	tertiary	no	yes	no	ceilular	unknown	no
5.	management	married	secondary	no	yes	no	unknown	unknown	no
6.	management	single	tertiary	no	yes	no	unknown	unknown	no
7.	entrepreneur	married	tertiary	no	yes	no	cellular	failure	yes
8.	admin.	married	secondary	no	no	no	cellular	unknown	no
9.	blue-collar	married	secondary	no	yes	no	cellular	unknown	no
10.	management	married	tertiary	yes	no	no	cellular	unknown	no
11.	blue-collar	married	secondary	no	yes	no	cellular	unknown	no
12.	management	divorced	secondary	no	no	no	unknown	unknown	no
13.	blue-caller	married	secondary	no	yes	no	cellular	unknown	no
14	retired	married	secondary	no	no	no	cellular	unknown	no
15.	management	single	tertiary	no	yes	no	cellular	unknown	no
16.	retired	married	secondary	yes	yes	no	cellular	unknown	no
17.	unemployed	married	secondary	no	yes	no	telephone	unknown	no
18.	management	divorced	tertiary	no	yes	no	cellular	unknown	no
19.	management	married	tertiary	no	yes	no	cellular	unknown	no
20.	blue-collar	married	secondary	no	yes	no	unknown	unknown	no
21.	management	divorced	tertiary	no	yes	yes	cellular	failure	yes
22.	blue-collar	divorced	secondary	no	yes	no	cellular	failure	no
23.	blue-collar	single	secondary	no	yes	no	cellular	failure	no
24.	admin	Single	secondary	no	no	no	unknown	unknown	no
25.	blue-collar	married	secondary	no	yes	no	cellular	Failure	no
26.	blue-collar	single	secondary	no	yes	no	unsown	Unknown	no
27.	housemaid	married	secondary	no	no	no	cellular	unknown	no
28.	technician	married	tertiary	no	no	no	cellular	unknown	no

Fig. 4.1.2 : A subset of the bank marketing dataset

- Following statistics are displayed by the summary of the dataset
- For simplification, the summary just contains the top six most frequently appearing values for each attribute. The remaining are shown as (Other).

job	marital	education	default
blue-collar : 435	divorced : 228	primary : 335	no : 1961
management : 423	married:1201	secondary : 1010	yes : 39
Technician : 339	single : 571	Tertiary : 564	
admin. : 235		unknown : 91	
services : 168			
retired : 92			
(other) : 308			

housing	loan	contact	month	poutcome	subscribed
no : 916	no : 1717	cellular : 1287	may : 581	failure : 210	no : 1789
yes : 1084	yes : 283	telephone : 136	jul : 340	other : 79	yes : 211
		unknown : 577	aug : 278	success : 58	
			jun : 232	unkown :1653	
			nov : 183		
			apr : 118		
			(other) : 268		

- Attribute job includes the following values.

admin.	blue-collar	entrepreneur	housemaid
235	435	70	63
management	retired	self-employed	services
423	92	69	168
student	technician	unemployed	Unknown
36	339	60	10

- Fig. 4.1.3 shows a decision tree built over the bank marketing dataset.
- The root of the tree illustrates that the general fraction of the clients who are not interesting in subscribing the term deposit is 1,789 out of the total number of clients (2,000).
- At each split, out of all the attributes, the most informative attribute is picked by the decision tree algorithm. The extent to informativeness of an attribute is decided by measures like entropy and information gain.

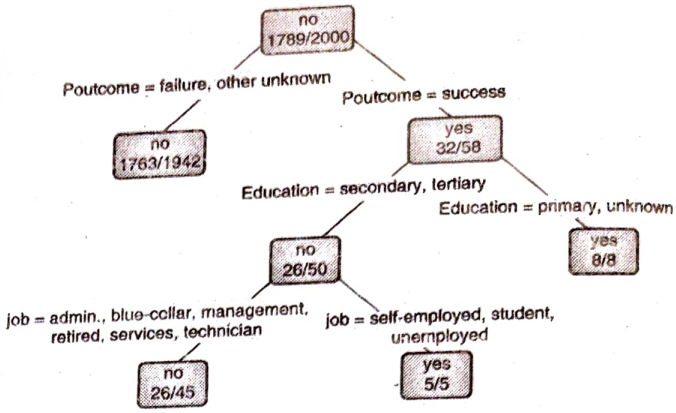

Fig. 4.1.3 : Using a decision tree to predict if a client will subscribe to a term deposit

- When the first split is done, the attribute poutcome is selected by the decision tree algorithm.
- At depth one, there are 2 nodes. The left node is a leaf node which represents a group for which the outcome of the previous marketing campaign contact is a failure, other, or unknown.
- For this group, near about 1,763 clients out of 1,942 have no subscription to the term deposit.
- The rest of the population is represented by the right node.
- The right node represents the remaining population, for which the result of the prior marketing campaign contact is a success.
- For the population of this node, thirty two clients out of fifty eight have subscription to the term deposit.
- Further depending upon education level, this node is divided into two nodes.
- In case of secondary or tertiary education level, twenty six clients out of fifty have no subscription to the term deposit.
- In case of primary education level, eight clients out of eight have subscription to the term deposit.
- At depth two, the LHS node splits depending upon attribute job.
- In case of occupation admin, technician, management, retired, services, twenty six clients out of forty five have no subscription.

- In case of occupation self-employed, student, or unemployed, five clients out of five have subscription to the term deposit.

Syllabus Topic : General Algorithm

4.1.2 General Algorithm

Q. 4.1.2 Explain the term general algorithm. (Refer section 4.1.2) **(8 Marks)**

- The main objective of a decision tree algorithm is to build a tree T from a training set S.
- When all of the records in training set S are related to any class C (e.g. subscribed = yes), or if training set S is satisfactorily pure (more than a preset threshold), then that node is assumed as a leaf node and label c is assigned to it.
- The **purity** of a node can be described as its probability of the subsequent class.
- For example, in Fig. 4.1.3, the root P(subscribed = yes) $= 1 - \frac{1789}{2000} = 10.55\%$; hence, the purity of root is just 10.55% on the subscribed = yes class.
- While its purity is 89.45% on the subscribed = no class.
- On the other hand, if all the records in S does not belong to class C or if purity of S is not satisfactorily, the algorithm choose subsequently most informative attribute A (duration, marital, etc.) and do the partitions S as per A's values.
- A subtrees T_1, T_2 ... will be constructed by algorithm for the subsets of S recursively unless one of the following criteria get satisfied:
- The minimum purity threshold is fulfilled by all the present leaf nodes.
- The preset minimum purity threshold is not sufficient to split the tree further.
- Another stopping criterion is satisfied (e.g. the maximum depth of the tree).
- In the process of constructing a decision tree, the first step is to select the most informative attribute.
- Entropy-based methods are the common option to identify the most informative attribute
- The most informative attribute is selected by the entropy methods based on two basic measures:
  - **Entropy**, it is used to measure the impurity of an attribute
  - **Information gain**, it is used to measure the purity of an attribute
- Provided a class X with label $x \in X$, consider P(x) be the probability of x. the entropy of X is H_x. The definition is as follows:

$$H_x = - \sum_{\forall x \in X} P(X) \log_2 P(x) \quad ...(4.1.1)$$

- The given Equation 4.1.1 illustrates that entropy H_X will be 0 when all P(x) is 0 or 1.
- In case of classification in binary form (true or false), H_X is zero when P(x) that is the probability of each label x is either 0 or 1.
- Whereas, H_X will get the maximum entropy in the case when all the class labels have same probability.
- In case of classification in binary form, $H_x = 1$ if all class labels have probability as 50/50.
- The increase in maximum entropy depends on increase in number of possible outcomes.
- As a binary random variable example, think of tossing a coin with known, not essentially fair, probabilities of coming up heads or tails.
- Fig. 4.1.4 shows corresponding entropy graph.
- Consider $x = 1$ stand for heads and $x = 0$ stand for tails. The entropy of the output which is not known of the next toss is maximized when the coin is fair.
- Means when probabilities of heads and tails are equal,

$$P(x = 1) = P(x = 0) = 0.5,$$

entropy $H_X = -(0.5 \times \log_2 0.5 + 0.5 \times \log_2 0.5) = 1$.

- Whereas, if the coin is not fair, the heads and tails will not have equal probabilities and uncertainty will be less.

- In an extreme situation, when the probability regarding the tossing a head is equal to zero or one, the entropy is minimized to zero.
- Hence, a completely pure variable will have entropy as zero and is one for a set having same occurrences for both of the classes (head and tail, or yes and no).

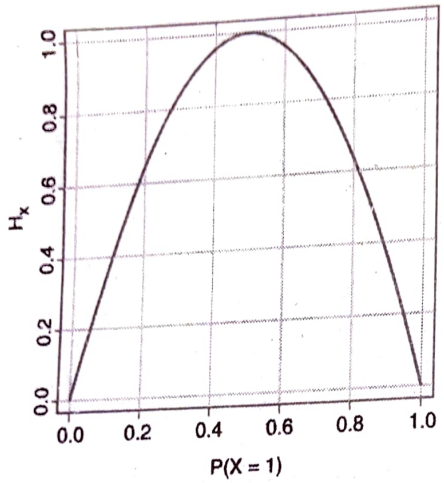

Fig. 4.1.4 : Entropy of coin flips, where X = 1 represents heads

- In the bank marketing scenario which has seen in previous section, the output variable is subscribed.
- The base entropy is defined as entropy of the output variable, that is $H_{subscribed}$.
- As we know, P(subscribed = yes) = 0.1055 and P(subscribed = no) = 0.8945.
- As per the Equation (4.1.1), the base entropy

$$H_{subscribed} = -0.1055 \cdot \log_2 0.1055 - 0.8945 \cdot \log_2 0.8945 \approx 0.4862$$

- In the further step, the conditional entropy should be identified for each attribute.
- Provided an attribute X, its value x, its outcome Y, and its value y, conditional entropy $H_{Y|X}$ is the remaining entropy of Y given X, the definition is given in Equation (4.1.2).

$$H_{\eta x} = \sum_{x} P(x)\, H(Y|X = x)$$

$$= \sum_{\forall x \in x} P(X) \sum_{\forall y \in y} P(y|x) \log_2 P(y|x) \quad \ldots(4.1.2)$$

- In the banking marketing scenario, if the selection of attribute contact is done, X = {cellular, telephone, unknown}. All the three values are considered by the conditional entropy of contact.
- Table 4.1.1 lists the probabilities related to the contact attribute. The top row of the table displays the probabilities of each value of the attribute. The next two rows contain the probabilities of the class labels conditioned on the contact.

Table 4.1.1 : Conditional Entropy Example

	Cellular	Telephone	Unknown
P(contact)	0.6435	0.0680	0.2885
P(subscribed=yes|contact)	0.1399	0.0809	0.0347
P(subscribe=no|contact)	0.8601	09192	0.9653

- The computation of contact attribute's conditional entropy is as follows :

$$H_{subscribed|contact} = -[0.6435 \cdot (0.1399 \log_2 0.1399 + 0.8601 \cdot \log_2 0.8601) + (0.0680) \cdot (0.0809 \cdot \log_2 0.0809 + 0.9192 \log_2 0.9192) + 0.2885\ (0.0347 \log_2 0.0347 + 0.9653 \log_2 0.9653)] = 0.4661$$

- Computation which is given inside the parentheses is regarding the entropy of the class labels within a single contact value.
- We have to remember that the value of conditional entropy is at all times less than or equal to the base entropy. i.e., $H_{subscribed|marital} \leq H_{subscribed}$.
- When there is relation between attribute and the outcome, the conditional entropy is less as compared to the base entropy.
- In the worst case, when there is no relation between attribute and the outcome, the conditional entropy is same as of the base entropy.
- The information gain by an attribute A is defined as the difference between the attribute's base entropy and its conditional entropy. This is illustrated in as shown in Equation (4.1.3).

$$InfoGain_A = H_S - H_{S|A} \quad \ldots(4.1.3)$$

- In the previous example of bank marketing, the information gain of the contact attribute is as follows :

$$\text{InfoGain}_{\text{contact}} = H_{\text{subscribed}} - H_{\text{contact|subscribed}}$$
$$= 0.4862 - 0.4661 = 0.0201 \quad \ldots(4.1.4)$$

- Information gain is basically used to compare the parent node's degree of purity before a split with the child node's degree of purity after a split.
- At each and every split, an attribute which is having the maximum information gain is assumed the most informative attribute.
- Purity of an attribute is indicated by the Information gain.
- For all the input variables, the output of information gain is as shown in Table 4.1.2.
- The most informative variable is Attribute poutcome and it has the most information gain.
- Hence, poutcome is selected for the first split of the decision tree. It can be seen in Fig. 4.1.3.
- In Table 4.1.2, the information gain values are small but the important factor is relative difference.
- The algorithm splits on the attribute which is having the major information gain in every round.

Table 4.1.2 : Calculating Information Gain of Input Variables for the First Split

Attribute	Information Gain
poutcome	0.0289
contact	0.0201
housing	0.0133
job	0.0101
education	0.0034
marital	0.0018
loan	0.0010
default	0.0005

Detecting Significant Splits

- Most of the times it is essential to measure the significance of a split in a decision tree, particularly in the case of small information gain.
- Consider N_A as the number of class A and N_B be the number of class B in the parent node.
- Let N_{AL} stand for the number of class A will be the left child node, N_{BL} stand for the number of class B will be the left child node, N_{AR} stand for the number of class B will be the right child node, and N_{BR} stand for the number of class B will be the right child node.
- Let p_L represents the proportion of data going to the left node and p_R represents the proportion of data going to the right node.

$$P_L = \frac{N_{AL} + N_{BL}}{N_A + N_B}$$

$$P_R = \frac{N_{AR} + N_{BR}}{N_A + N_B}$$

- The significance of a split is computed by the following measure.
- That means, it measures how much the split diverges from the expectation in the random data.

$$K = \frac{(N'_{AL} - N_{AL})^2}{N'_{AL}} + \frac{(N'_{BL} - N_{BL})^2}{N'_{BL}} + \frac{(N'_{AR} - N_{AR})^2}{N'_{AR}} + \frac{(N'_{BR} - N_{BR})^2}{N'_{BR}}$$

Where

$$N'_{AL} = N_A \times p_L$$

$$N'_{BL} = N_B \times p_L$$

$$N'_{AR} = N_A \times p_R$$

$$N'_{BR} = N_B \times p_R$$

- The information gain from the split is not important if the value of K is small. It is significant only when the value of K is big.
- For example consider the first split of the decision tree in Fig. 4.1.3 on variable poutcome :

$$N_A = 1789, N_B = 211, N_{AL} = 1763,$$

$$N_{BL} = 179, N_{AR} = 26, N_{BR} = 32.$$

- Following are the proportions regarding the data which is going to the left and right node :

$$p_L = \frac{1942}{2000} = 0.971 \text{ and } p_R = \frac{58}{2000} = 0.029$$

- The N'_{AL}, N'_{BL}, N'_{AR} and N'_{BR} are used to represent the number of each class going to the LHS or RHS node in the case of random data. Their values :

$N'_{AL} = 1737.119$, $N'_{BL} = 204.881$, $N'_{AR} = 51.881$ and $N'_{BR} = 6.119$

- Hence value of K is 126.0324, which indicates that the split on poutcome is significant.
- After each and every split, the algorithm observes every record at a leaf node, and the calculation of information gain of each candidate attribute is done once more over these records.
- The next split is on the attribute which is having the highest information gain.
- After all the splits, it may be possible that a record can only belong to single leaf node, based on the implementation, an attribute may appear in multiple splits of the tree.
- This partitioning of the records and searching the attribute which is most informative is repeatedly done until the nodes are sufficiently pure, or the information gain by splitting on more attributes is not enough.
- Alternatively, the growth of the tree can be stopped if each node at a leaf node belong to a specific class (e.g. subscribed = yes) or values of all the records are identical.
- For simplification in previous bank marketing example, only categorical variables are included in the data set.
- Consider a continuous variable called duration is included in the dataset which represents the number of seconds regarding the last phone call with the bank lasted as part of the prior marketing campaign.
- For a continuous variable, its division is important into a disjoint set of regions which is having the highest information gain.
- A brute-force method assumes each value of the continuous variable in the training data as a candidate split position.
- Computationally, the brute-force method is not efficient.
- For simplification purpose, it is possible to sort the training records based on the duration, and the process of candidate splitting can be identified by taking the midpoints among two neighbouring sorted values.
- An examples is if the sorted values in the duration are {140, 160, 180, 200} and the candidate splits are 150, 170, and 190.
- Fig. 4.1.5 shows the view of decision tree when the duration attribute is considered.
- There will two part of roots :: such clients which have duration < 456 seconds, and another clients which have duration ≥ 456 seconds.

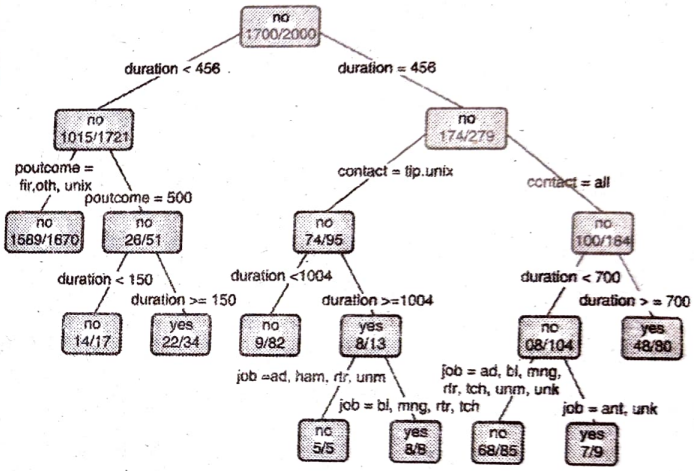

Fig. 4.1.5 : Decision tree with attribute duration

- With the decision tree which is illustrated in Fig. 4.1.5, it will be very easy to predict if a new client is going to subscribe to the term deposit.
- E.g. provided the record of a newly introduced client shown in Table 4.1.3, the prediction is that this client will subscribe to the term deposit.

The paths which are traversed in the decision tree are as follows :

- duration ≥ 456
- contact = cll (cellular)
- duration < 700
- job = ent (entrepreneur), rtr (retired)

Table 4.1.3 : Record of a New Client

Job	Marital	Education	Default	Housing	Loan	Contact	Duration	Poutcome
retired	married	secondary	no	yes	no	cellular	598	unknown

Syllabus Topic : Decision Tree Algorithms

4.1.3 Decision Tree Algorithms

Q. 4.1.3 Explain decision Tree algorithm in detail. (Refer section 4.1.3) (8 Marks)

- For the implementation of decision tree, there are number of algorithms there is difference in the methods of tree construction with different algorithms.
- Some important algorithms are ID3, C4.5 and CART.

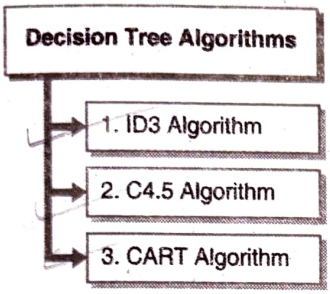

Fig. 4.1.6 : Decision tree algorithm

1. ID3 Algorithm

- In decision tree learning, ID3 (Iterative Dichotomiser 3) is an algorithm invented by Ross Quinlan used to generate a decision tree from a dataset.
- ID3 is the precursor to the C4.5 algorithm, and is typically used in the machine learning and natural language processing domains.
- The ID3 algorithm begins with the original set S as the root node. On each iteration of the algorithm, it iterates through every unused attribute of the set S and calculates the entropy H(S) (or information gain IG(S) of that attribute.
- It then selects the attribute which has the smallest entropy (or largest information gain) value. The set S is then split by the selected attribute (e.g. age is less than 50, age is between 50 and 100, age is greater than 100) to produce subsets of the data.
- The algorithm continues to recurse on each subset, considering only attributes never selected before. Recursion on a subset may stop in one of these cases:
- Every element in the subset belongs to the same class (+ or -), then the node is turned into a leaf and labelled with the class of the examples
- There are no more attributes to be selected, but the examples still do not belong to the same class (some are + and some are -), then the node is turned into a leaf and labelled with the most common class of the examples in the subset
- There are no examples in the subset, this happens when no example in the parent set was found to be matching a specific value of the selected attribute, for example if there was no example with age = 100. Then a leaf is created, and labelled with the most common class of the examples in the parent set.
- Throughout the algorithm, the decision tree is constructed with each non-terminal node representing the selected attribute on which the data was split, and terminal nodes representing the class label of the final subset of this branch.

Summary

- Calculate the entropy of every attribute using the data set {\displaystyle S} S
- Split the set {\displaystyle S} S into subsets using the attribute for which the resulting entropy (after splitting) is minimum (or, equivalently, information gain is maximum)
- Make a decision tree node containing that attribute
- Recurse on subsets using remaining attributes.

Algorithm

```
ID3 (A, P, T)
   if T ∈ϕ
     return ϕ
   if all records in T have the same value for P
     return a single node with that value
   if A ∈ϕ
   return a single node with the most frequent value of P in T
   Compute information gain for each attribute in A relative
to T
   Pick attribute D with the largest gain
   Let {
d1, d2... dm
```

} be the values of attribute D

Partition T into {

T1, T2... Tm

} according to the values of D

12 return a tree with root D and branches labeled

d1, d2... dm

going respectively to trees ID3(A-{D}, P, T1),

ID3(A-{D}, P, T2),

... ID3(A-{D}, P, Tm)

Usage

- The ID3 algorithm is used by training on a dataset S to produce a decision tree which is stored in memory.
- At runtime, this decision tree is used to classify new unseen test cases by working down the decision tree using the values of this test case to arrive at a terminal node that tells you what class this test case belongs to.

2. C4.5 Algorithm

- C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan.
- C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier.
- Authors of the Weka machine learning software described the C4.5 algorithm as "a landmark decision tree program that is probably the machine learning workhorse most widely used in practice to date".
- It became quite popular after ranking #1 in the Top 10 Algorithms in Data Mining pre-eminent paper published by SpringerLNCS in 2008.

Algorithm

- C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy.
- The training data is a set S = s1, s2... of already classified samples. Each sample s_i consists of a p-dimensional vector $(x_{1,i}, x_{2,i}, \ldots. x_{p,i})$ where the x_j represent attribute values or features of the sample, as well as the class in which s_i falls.
- At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other.
- The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision.
- The C4.5 algorithm then recurs on the smaller sublists. This algorithm has a few base cases.
- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value

Pseudocode

- In pseudocode, the general algorithm for building decision trees is :
  1. Check for the above base cases.
  2. For each attribute a, find the normalized information gain ratio from splitting on a.
  3. Let a_best be the attribute with the highest normalized information gain.
  4. Create a decision node that splits on a_best.
  5. Recur on the sublists obtained by splitting on a_best, and add those nodes as children of node.

3. CART Algorithm

- CART (or Classification And Regression Trees) is usually used as a generic acronym for the decision tree, even if it is a precise implementation.

- Just like C4.5, CART is also able to handle continuous attributes.
- For rank tests C4.5 uses entropy-based criteria while CART uses the Gini diversity index.

$$Gini_x = 1 - \sum_{\forall x \in X} P(X)^2 \quad \ldots(4.1.5)$$

- While C4.5 implements stopping rules, CART build a series of subtrees, uses cross-validation for the purpose of estimating the misclassification cost of each subtree, and select the lowest cost option.

Syllabus Topic : Evaluating a Decision Tree

4.1.4 Evaluating a Decision Tree

- ***Greedy algorithms*** are used in Decision trees in which they all the time select the best option.
- In every step the algorithm choose attribute to use for the purpose of splitting the remaining records.
- It may be possible that overall the selection may not be the best, but it assures to be the best at that step.
- This property strengthens the effectiveness of decision trees. But, if a bad split is taken, it impacts the rest of the tree.
- So as to solve such problem, an ensemble technique like random forest may help to randomize the splitting or even randomize data and provide a multiple tree structure.
- Then for all the classes voting is done by these trees and obviously the class getting maximum votes is selected as the predicted class.
- For evaluation of a decision tree some ways are available.
- The initial important task is to check whether to split the tree will be useful.

Sanity checks are conducted by the process of validating the decision rules with domain experts, and decide the viability of decision rules.

In the next step, one has to observe the depth as well as nodes of the tree.

- If there is presence of large number of layers and nodes with a small number of members, then it can be considered as overfitting.
- In overfitting, the training set is well fitted in the model, but its performance is poor on the new samples in the testing set.
- We can see the performance of an overfit model in Fig. 4.1.7.
- The amount of data is represented by the *x*-axis while the errors are represented by the *y*-axis.
- The upper curve represents testing set and lower curve represents the training set.
- The left side of the vertical line which is gray color indicates that the prediction of model is good on the testing set. But on the RHS of the gray line, the performance of model is getting worse as new data is introduced.

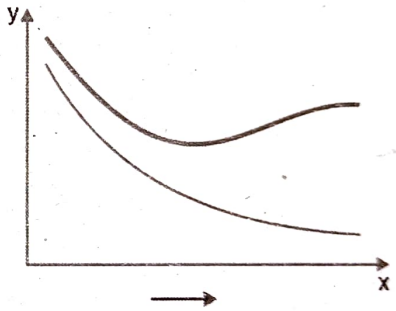

Fig. 4.1.7 : An overfit model describes the training data well but predicts poorly on unseen data

- In the process of learning decision tree, overfitting may occur because of either the lack of training data or the biased data present in the training set.
- There are 2 approaches which are able to avoid overfitting in decision tree learning.
- Prevent growth of the tree prior to its reaching the point where perfect classification of all the training data is done.
- Complete the growth of the full tree, and then post-prune the tree with the help of different methods like reduced-error pruning and rule-based post pruning.
- Computationally, the Decision trees are considered as low cost and the data classification is simple.

- It is easy to interpret outputs as a fixed series of simple tests. For several decision tree algorithms it is possible to show the importance of each and every input variable.
- Most of the statistical software packages provides the basic measures like information gain.
- Both numerical as well as categorical attributes can be handled by the Decision trees and are robust with redundant or correlated variables.
- Decision trees are able to handle categorical attributes with several dissimilar values like country codes for telephone numbers.
- Also variables which have nonlinear effect on the outcome can also be handled by the Decision trees. Hence in case of highly nonlinear problems their working is better than linear models (for example, linear regression and logistic regression).
- Variable interactions are naturally handled by the Decision trees. The dependence of each and every node is on its preceding nodes in the tree.
- The decision regions are denoted as rectangular surfaces in a decision tree.
- Fig. 4.1.7 illustrates an example in which 5 rectangular decision surfaces (A, B, C, D, and E) are defined with the help of four values - $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\}$ - of 2 attributes (x_1 and x_2).
- The subsequent decision tree is present on RHS of the Fig. 4.1.8.

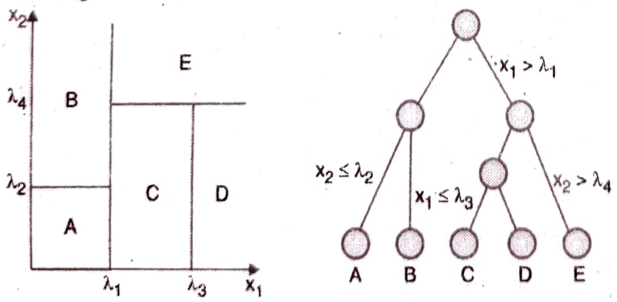

Fig. 4.1.8 : Decision surfaces can only be axis-aligned

- A decision surface corresponds to a leaf node (lowest node without child) of the tree, and it can be reached by performing the traversing from the root of the tree following by a series of decisions which depends upon the value of an attribute.
- For the decision tree, decision surface can only be axis-aligned.
- In the training data, the structure of a decision tree is considered as responsive to small variations.
- Even for the same dataset, construction of 2 decision trees based on 2 dissimilar subsets may generate completely different trees.
- If depth of tree is more, then overfitting may occur, since every split decreases the training data for subsequent splits.
- If there are many irrelevant variables in the dataset, then it will not be good option to select the choice of Decision trees.
- It is apart from the notion that the Decision trees are robust with redundant variables as well as correlated variables.
- If there are redundant variables in the dataset, the resultant decision tree avoids all but one of these variables since the algorithm is unable to detect information gain by adding more redundant variables.
- Whereas, if there are irrelevant variables in the dataset, and if the selection of these variables is accidental as splits in the tree, the growth of tree may too large and may end up with small amount of data at every split, where overfitting is likely to occur.
- To solve such problem, we can introduce feature selection in the phase of data preprocessing to remove the irrelevant variables.
- Even if decision trees are considered as good to handle correlated variables, but if maximum variables in the training set are correlated then decision trees are not well suited, as overfitting is likely to occur.
- To get rid of the problem regarding instability and potential overfitting of deep trees, it is good to merge the decisions of various randomized shallow decision trees which is considered as the basic idea of another classifier known as random forest or take help of ensemble methods to merge various weak learners for better classification.

- If in a Decision tree there are records with an even probability of each result, a decision tree works better for binary decisions.
- That means we can say that the root of the tree has a fifty percent chance of either classification.
- This happens by randomly choosing training records from all the possible classifications in similar numbers.
- If methods like logistic regression on a dataset with a lot of variables are used, decision trees are able to identify the variables which are the most helpful to select based on information gain.
- Then for the logistic regression, such variables can be selected.
- One more use of Decision trees is to prune redundant variables.

Syllabus Topic : Naïve Bayes

4.2 Naïve Bayes

Q. 4.2.1 What is Naïve Bayes ? (Refer section 4.2) (4 Marks)

- Naïve Bayes is a probabilistic classification method based on Bayes' theorem (or Bayes' law) with a few tweaks.
- Bayes' theorem provides the relationship among the probabilities of 2 events with their conditional probabilities.
- Bayes' law is named after the English mathematician Thomas Bayes.
- A naïve Bayes classifier considers that the existence or nonexistence of a specific feature of a class is not related to the existence or nonexistence of other features.
- E.g. classification of object can be done on the basis of its attributes like shape, color, and weight.
- A logical classification of an object which is spherical, yellow and having weigh less than 60 grams may be a tennis ball.
- Even though these features depend on each other or upon the presence or absence of the other features, the naïve Bayes classifier assumes all the properties should have contribution without any dependence to the probability that the object is a tennis ball.
- Usually the input variables are categorical, but continuous variables are accepted by the variations of the algorithm.
- Also there are some methods to convert continuous variables into categorical variables. This mechanism is known as **discretization of continuous variables**.
- In the tennis ball example, to convert into a categorical variable, a continuous variable like weight can be grouped into intervals.
- The conversion of attribute is as follows considering attribute income.

  Low Income : income < $10,000

  Working Class : $10,000 ≤ income < $50,000

  Middle Class : $50,000 ≤ income < $1,000,000

  Upper Class : income ≥ $1,000,000
- In the output, most often a class label and its corresponding probability score is present.
- The probability score is not considered as the true probability of the class label, but it is assumed that the probability score is proportional to the true probability.
- As the implementation of naïve Bayes classifiers is easy and can execute effectively even without previous knowledge of the data, they are considered as the frequently used algorithms for classifying text documents.
- Spam filtering can be assumed as an example of classic use case of naïve Bayes text classification.
- Nowadays Bayesian spam filtering is considered as the best mechanism for the purpose of distinguishing spam e-mail from legitimate e-mail.
- Number of modern mail clients likes to implement variants of Bayesian spam filtering.
- Fraud detection is an example where Naïve Bayes classifiers can also be used.

Syllabus Topic : Bayes' Algorithm

4.2.1 Bayes' Algorithm

Q. 4.2.2 Explain Algorithm of Bayes' Algorithm. (Refer section 4.2.1) **(8 Marks)**

- The ***conditional probability*** regarding the event C happening, provided that event A has previously occurred, is represented as $P(C|A)$, which can be found with the help of formula in Equation (4.2.1).

$$P(C|A) = \frac{P(A \cap C)}{P(A)} \quad ...(4.2.1)$$

- Now we can get Equation (4.2.2) with some minor algebra and substitution of the conditional probability:

$$P(C|A) = \frac{P(A|C) \cdot P(C)}{P(A)} \quad ...(4.2.2)$$

- Here C is the class label $C \in \{c_1, c_2, ...c_n\}$ and A is the observed attributes $A = \{a_1, a_2, ...a_m\}$.
- Equation (4.2.2) can assumes as the most common form of the ***Bayes' theorem***.
- Mathematical point of view Bayes' theorem provides the relationship among the probabilities of C and A, P(C) and P(A), as well as the conditional probabilities of C provided A and A provided C, namely P(C|A) and P(A|C).
- Bayes' theorem is significant because quite often P(C|A) is much more difficult to compute than P(A|C) and P(C) from the training data. By using Bayes' theorem, this problem can be circumvented.
- We will see an example to illustrate the use of Bayes' theorem.
- Kunal flies regularly and always wants to upgrade his seat to first class.
- He has observed that if he entered the airport for his flight minimum two hours before, the probability of getting an upgrade is 0.75; or else, the probability of getting an upgrade is 0.35.
- Because of busy schedule, entering the airport minimum two hours early is possible for him only 40% of the time.
- Consider that in a most recent attempt Kunal unable to receive an upgrade. What is the probability that he did not arrive two hours early?
- Let C = {Kunal arrived at least two hours early}, and A = {Kunal received an upgrade}, then $\neg C$ = {Kunal did not arrive two hours early}, and $\neg A$ = {Kunal did not receive an upgrade}.
- Kunal checked in minimum two hours before only 40% of the time, or P(C) = 0.4.

Hence, $P(\neg C) = 1 - P(C) = 0.6$.

- The probability that Kunal received an upgrade given that he checked in early is 0.75, or P(A|C) = 0.75.
- The probability that Kunal received an upgrade given that he did not arrive two hours early is 0.35, or $P(A|\neg C) = 0.35$. Therefore, $P(\neg A|\neg C) = 0.65$.
- The probability that Kunal received an upgrade P(A) can be calculated as illustrated in Equation (4.2.3).

$$\begin{aligned} P(A) &= P(A \cap C) + P(A \cap \neg C) \\ &= P(C) \cdot P(A|C) + P(\neg C) \cdot P(A|\neg C) \\ &= 0.4 \times 0.75 + 0.6 \times 0.35 \\ &= 0.51 \end{aligned} \quad ...(4.2.3)$$

- In this way, the probability which Kunal did not receive an upgrade $P(\neg A) = 0.49$.
- With the help of Bayes' theorem, the probability that Kunal not able to arrive two hours before provided that he did not get his upgrade illustrated in Equation (4.2.4).

$$\begin{aligned} P(\neg C|\neg A) &= \frac{P(\neg A|\neg C) \cdot P(\neg C)}{P(\neg A)} \\ &= \frac{0.65 \times 0.6}{0.49} \approx 0.796 \end{aligned} \quad ...(4.2.4)$$

Syllabus Topic : Naïve Bayes Classifier

4.2.2 Naïve Bayes Classifier

Q. 4.2.3 Explain Naïve Bayes Classifier. (Refer section 4.2.2) **(8 Marks)**

- Now with two simplifications, it will be possible to extend Bayes' theorem to become a naïve Bayes classifier.

- The first simplification includes the use of conditional independence assumption. That means, every attribute is conditionally not dependent on each other attribute provided a class label c_i. Observe the Equation (4.2.5).

$$P(a_1, a_2, ..., a_m|c_1) = P(a_1|c_1)... P(a_m | c_1) = \prod_{j=1}^{m} P(a_j|c_i) \quad ...(4.2.5)$$

- Hence, the naïve assumption simplifies the computation of $P(a_1, a_2,..., a_m|c_i)$.
- The next (second) simplification is skipping the denominator $P(a_1, a_2,..., a_m)$. Since $P(a_1, a_2,..., a_m)$ present in the denominator of $P(c_i|A)$ for each and every value of i, skipping the denominator will not impact the relative probability scores and calculations will be simplified.
- The classification of Naïve Bayes applies the 2 simplifications stated here and, as a result, $P(c_i|a_1, a_2,..., a_m)$ is proportional to the product of $P(a_j|c_i)$ times $P(c_i)$. It is illustrated in Equation (4.2.6).

$$P(c_1|A) \propto P(c_1) \cdot \prod_{j=1}^{m} P(a_j|c_1) \quad i = 1, 2, ... n \quad ...(4.2.6)$$

- The mathematical symbol $\propto$ states that the LHS $P(c_i|A)$ is directly proportional to the RHS.
- In the previous section we have seen a bank marketing dataset (Fig. 4.1.2).
- Here we will see how to use the naïve Bayes classifier on that given dataset to predict whether the clients will like to subscribe to a term deposit.
- To build naïve Bayes classifier, we require knowledge of certain statistics, all computes from the training set.
- In the first requirement we have to gather the probabilities of all class labels, $P(c_i)$.
- In the given example, these are the probability which a client will like or not to subscribe to the term deposit. From the data available in the training set,

$P(\text{subscribed} = \text{yes}) \approx 0.11$ and $P(\text{subscribed} = \text{no}) \approx 0.89$.

- One more element the naïve Bayes classifier requires to know is the conditional probabilities of every attribute a_j provided every class label c_i, namely $P(a_j|c_i)$.
- The training set includes number of attributes : job, marital, education, default, housing, loan, contact, and poutcome.
- For all the .attributes and their possible values, calculating the conditional probabilities provided subscribed = yes or subscribed =no is necessary.
- E.g., relative to the marital attribute, calculation of conditional probabilities is done as follows :

$P(\text{single}|\text{subscribed} = \text{yes}) \approx 0.35$

$P(\text{married}|\text{subscribed} = \text{yes}) \approx 0.53$

$P(\text{divorced}|\text{subscribed} = \text{yes}) \approx 0.12$

$P(\text{single} | \text{subscribed} = \text{no}) \approx 0.28$

$P(\text{married}|\text{subscribed} = \text{no}) \approx 0.61$

$P(\text{divorced}|\text{subscribed} = \text{no}) \approx 0.11$

- After the process of training the classifier and computation of all the necessary statistics, it is possible to test the naïve Bayes classifier over the testing set.
- In the testing set, for all the records, the classifier label c_i is assigned by the naïve Bayes classifier which maximizes $P(c_i) \cdot \prod_{j=1}^{m} P(a_j|c_i)$
- In Table 4.2.1, we can observe that a single record is present of a client having career in management, is marital status is married, education is secondary degree, has credit not in default, has only housing loan no personal loan, contact preferences are cellular, and whose outcome of the preceding marketing campaign contact carried out was a success.
- Is this client would subscribe to the term deposit?

Table 4.2.1 : Record of an Additional Client

Job	Marital	Education	Default	Housing	Loan	Contact	Poutcome
Management	married	secondary	no	yes	no	cellular	success

- The calculation of conditional probabilities shown Table 4.2.2 can be done after establishing the classifier with the training set.

Table 4.2.2 : Compute Conditional Probabilities for the New Record

j	a_j	P(a_j|subscribed = yes)	P(a_j|subscribed = no)
1	job = management	0.22	0.21
2.	marital = married	0.53	0.61
3.	education = secondery	0.46	0.51
4.	default = no	0.99	0.98
5.	housing = yes	0.35	0.57
6.	loan = no	0.90	0.85
7.	contact = cellular	0.85	0.62
8.	poutcome = success	0.15	0.01

since $P(c_i|a_1, a_2,..., a_m)$ is proportional to the product of $P(a_j|c_i)(j \in [1, m])$ times (c_i), the naïve Bayes classifier assigns the class label c_i, which gives output in the maximum value over all i. In this way, $P(c_i|a_1, a_2,..., a_m)$ is calculated for every c_i with

$$P(c_i|A) \propto P(c_i) \cdot \prod_{i-1}^{m} P(a_j|c_i) \quad ...(4.2.7)$$

- For A = {management, married, secondary, no, yes, no, cellular, success},

$$P(yes|A) \propto 0.11\ (0.22 \cdot 0.53 \cdot 0.46 \cdot 0.99 \cdot 0.35 \cdot 0.90 \cdot 0.85 \cdot 0.15) \approx 0.00023$$

$$P(no|A) \propto 0.89\ (0.21 \cdot 0.61 \cdot 0.51 \cdot 0.98 \cdot 0.57 \cdot 0.85 \cdot 0.62 \cdot 0.01) \approx 0.00017$$

- Since P(subscribed = yes|A) > P(subscribed = no|A), the client displayed in Table 4.2.1 is assigned with the label subscribed = yes. It indicates that the client is classified as he/she would subscribe to the term deposit.
- Even if the the magnitude of scores is small, it is the ratio of *P(yes|A)* and *P(no|A)* which is important.
- Practically, the scores of P(yes|A) and P(no|A) are not considered as the true probabilities but are only assumed to be proportional to the true probabilities, as illustrated in Equation (4.2.7).
- Ultimately, if the scores are certainly the true probabilities, then the addition of P(yes|A) and P(no|A) would be equal to one.
- If problems having a huge number of attributes, or attributes which have high number of levels are considered, the magnitude of these values will be very small (near about zero) which results in even more small differences of the scores.
- This problem is considered of **numerical underflow**, which occurs because of multiplying several probability values having values close to zero.
- To solve the problem, one easy way is to compute the logarithm of the products, which is equal to the addition of the logarithm of the probabilities.
- In this way, the naïve Bayes formula can be rewritten as revealed in Equation (4.2.8).

$$P(c_1|A) \propto \log P(c_1) + \sum_{i=1}^{m} \log P(a_j|c_1) \quad i = 1, 2, ... n \quad ...(4.2.8)$$

- Even though there may be increase in risk of underflow as there is increase in number of attributes, the use of logarithms is generally applied irrespective of the number of attribute dimensions.

Syllabus Topic : Smoothing

4.2.3 Smoothing

Q. 4.2.4 Explain Smoothing process. (Refer section 4.2.3) **(8 Marks)**

- If one of the attribute values is not present in one of the class labels in the training set, the subsequent $P(a_j|c_i)$ will equal to zero.
- In this situation, the resulting $P(c_i|A)$ from multiplying every $P(a_j|c_i)(j \in [1, m])$ instantly becomes zero irrespective of how large few of the conditional probabilities are.
- Hence overfitting occurs. Smoothing techniques are used to adjust the probabilities of $P(a_j|c_i)$ and to assure a nonzero value of $P(c_i|A)$.
- A small nonzero probability is assigned by the smoothing technique to unusual events not contained in the training dataset.

- Additionally, the smoothing takes care of possibility of taking the logarithm of zero which may happen in Equation (4.2.8).
- Number smoothing techniques are available. One of them is the **Laplace smoothing** (or add-one) technique which pretend to look for all the outcomes again than it actually appears.
- This technique can be seen in Equation (4.2.9) :

$$P^*(\dot{x}) = \frac{\text{count}(x) + 1}{\Sigma x\,[\text{count}(x) + 1]} \quad \ldots(4.2.9)$$

- E.g., consider that there is subscription of hundred clients to the term deposit, in which twenty are single, seventy married, and ten divorced.
- The "raw" probability is :

P(singlelsubscribed=yes) = 20/100 = 0.2

- By the use of Laplace smoothing which adds one to the counts, the adjusted probability will be :

P' (*singlelsubscribed* = *yes*)

= (20 + 1)/[(20 + 1) + (70 + 1) + (10 + 1)] ≈ 0.2039.

- One issue regarding the Laplace smoothing is that it may assign more than expected probability to unseen events.
- To solve this issue, Laplace smoothing can be generalized to use ε instead of 1, where usually ε∈[0,1].
- Observe Equation (4.2.10) :

$$P^{**}(x) = \frac{\text{count}(x) + \varepsilon}{\Sigma x\,[\text{count}(x) + \varepsilon]} \quad \ldots(4.2.10)$$

- For naïve Bayes classifiers, the smoothing techniques are offered in various types of standard software packages.
- On the other hand, if because of some reason (such as performance concerns) the naïve Bayes classifier requires to be coded directly into an application, then incorporation of smoothing as well as logarithm calculations should be done into the implementation.

Syllabus Topic : Diagnostics

4.2.4 Diagnostics

- Naïve Bayes classifiers are able to handle missing values which is not possible to logistic regression.
- Naïve Bayes is also considered as robust regarding the irrelevant variables variables which are scattered between all the classes whose inmpacts are not pronounced.
- The model of Naïve Bayes classifiers is easy to implement even if libraries are not available.
- The base of prediction is counting the occurrences of events, making the classifier competent to execute.
- Naïve Bayes is considered as efficient computationally and can also handle high-dimensional data in an efficient manner.
- Research study indicates that the naive Bayes classifier in number of cases is competitive with several learning algorithms such as decision trees and neural networks.
- In some cases, the output of naïve Bayes is superior to other methods.
- Naïve Bayes classifiers are able to handle categorical variables, which is not possible to logistic regression.
- Handling categorical variables is also possible in decision trees but increase in number of levels may result in a deep tree.
- The overall performance of naïve Bayes classifier is better than decision trees regarding the categorical values with many levels.
- Naïve Bayes is more resistant to overfitting as compared to decision trees, particularly when there is involvement of a smoothing technique.
- Despite the advantages of naïve Bayes, this method also has some drawbacks.
- The variables in the data are considered by the Naïve Bayes as conditionally independent. Hence, correlating variables becomes sensitive since the algorithm may double count the effects.
- E.g., consider that individuals having low income and low credit tend to default. If there is requirement of scoring "default" based on both income as well as credit as two different attributes, there may be double-counting effect in naïve Bayes on the default outcome, which will affect the accuracy of the prediction.

- Even if probabilities are offered as part of the output for the purpose of prediction, then usually the naïve Bayes classifiers are not considered as very trustworthy for probability estimation and their use should be limited to assigning class labels.
- In its simple form, we can use the Naïve Bayes only with categorical variables. If there are any continuous variables then they must be converted into categorical variables b using the process called as discretization.

Syllabus Topic : Diagnostics of Classifiers

4.2.5 Diagnostics of Classifiers

Q. 4.2.5 Write note on diagnostics of classifiers. (Refer section 4.2.5) **(8 Marks)**

- Up till now we have seen three classifiers: logistic regression, decision trees, and naïve Bayes. These three techniques are used to classify instances into distinct sets based on the same characteristics they share.
- All these classifiers have the same issue: how to evaluate if they perform well.
- Some tools are available to evaluate the performance of a classifier.
- A specific table layout is available known as **confusion matrix** which allows visualization of the performance of a classifier for a two-class classifier, confusion matrix is shown in Table 4.2.3.
- **True Positives** (TP) are the count of positive instances to which the classifier is correctly able to identify as positive.
- **False Positives** (FP) are the count of instances in which the identification of classifier is positive but practically they are negative.
- **True Negatives** (TN) are the count of negative instances to which the classifier is correctly able to identify as negative.
- **False Negatives** (FN) are the count of instances in which the identification of classifier is negative but practically they are positive.
- In a two-class classification, we can use a preset threshold to separate positives from negatives.
- True positives and True negatives are considered as the correct guesses. Sign of a good classifier is that it should have large True positives and True negatives and small (ideally zero) numbers for False positives and False negatives.

Table 4.2.3 : Confusion Matrix

		Predicted Class	
		Positive	Negative
Actual Class	Positive	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- In the bank marketing example, there are two thousand instances in the training set.
- Also hundred instances are added as the testing set. Table 4.2.4 displays the confusion matrix related to a naïve Bayes classifier on hundred clients for predicting their possibility to subscribe to the term deposit.
- Out of the eleven clients whose subscription is for the term deposit, the prediction of model is that three subscribed while eight not subscribed.
- In same way, of the eighty nine clients whose subscription is not for the term, the prediction of model is that two subscribed and eighty seven not subscribed.
- In the table all the accurate speculations are placed from top left to bottom right.
- Inspecting the table visually is simple for errors, since they will be denoted by any nonzero values outside the diagonal.

Table 4.2.4 : Confusion Matrix of Naïve Bayes from the Bank Marketing Example

		Predicted Class		
		Subscribed	Not Subscribed	Total
Actual Class	Subscribed	3	8	11
	Not Subscribed	2	87	89
Total		5	95	100

- The **accuracy** (also termed as **overall success rate**) is a metric which defines the rate at which classification of records is done by the model accurately.
- It is defined as the addition of TP (True Positive) and TN (True negative) divided by the total number of instances. It is described in Equation (4.2.11)

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad ...(4.2.11)$$

- The accuracy score of a good model should be height, but just this is not sufficient to assure that the model is well established.
- The performance of a classifier is evaluated better by the following measures.
- The TPR (**True positive rate**) elaborates the exact percent of positive instances identified by the classifier.

  It is illustrated in Equation (4.2.12)

$$TPR = \frac{TP}{TP + FN} \quad ...(4.2.12)$$

- FPR (The **false positive rate**) elaborates the exact percent of negatives identified by the classifier.

  The FPR is also termed as the **false alarm rate** or the **type I error rate.** It is illustrated in Equation (4.2.13)

$$FPR = \frac{FP}{FP + TN} \quad ...(4.2.13)$$

- The **false negative rate** (FNR) elaborates the exact percent of positives to which the classifier marked as negatives.
- It is also termed as the **miss rate** or **type II error rate.** It is illustrated in Equation (4.2.14). Note that the addition of TPR and FNR is 1.

$$FNR = \frac{FN}{TP + FN} \quad ...(4.2.14)$$

- The TPR of a well-performed model should be high which is ideally considered as one and a low FPR and FNR which are ideally considered as zero.
- Practically, this is rare to have TPR = 1, FPR = 0, and FNR = 0, but the use of these measures are important to compare the performance of several models which are designed for the purpose of getting solution for the same problem.
- Remember that usually, the model which is whose preference is more may depends on the business situation.
- In the lifecycle of data analytics, while performing the discovery phase, it is important for the team to learn from the business which type of errors can be tolerated.
- In some business situations type I errors are more tolerated, while in some other business situations type II errors are more tolerated.
- In some situations, a model having a TPR of 0.95 and an FPR of 0.3 is considered as more suitable than a model having a TPR of 0.9 and an FPR of 0.1 even though the accuracy of second model is overall more.
- Assume example of e-mail spam filtering. Some individual like busy executives just desire important e-mail in their inbox and are tolerant of getting a few less important e-mail in their spam folder.
- Some individuals may not desire any important or less important e-mail to be specified as spam.
- There are two accuracy metrics known as precision and recall which are used by the information retrieval community, but their use is limited to characterize classifiers in general.
- **Precision** is the percentage of instances which are identified positive that actually are positive.
- It is illustrated in Equation (4.2.15)

$$\text{Precision} = \frac{TP}{TP + FP} \quad ...(4.2.15)$$

- **Recall** is the percentage of positive instances which were identified accurately. Recall is equivalent to the TPR.
- Provided the confusion matrix from Table 4.2.4, the computation of metrics is as follows :

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

$$= \frac{3 + 87}{3 + 87 + 2 + 8} \times 100\% = 90\%$$

$$\text{TPR (or Recall)} = \frac{TP}{TP + FN} = \frac{3}{3 + 8} \approx 0.273$$

$$FPR = \frac{FP}{FP + TN} = \frac{2}{2 + 87} \approx 0.022$$

$$FNR = \frac{FN}{TP + FN} = \frac{8}{3 + 8} \approx 0.727$$

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{3}{3+2} = 0.6$$

- These metrics help to illustrate that for the bank marketing example, the performance of the naïve Bayes classifier is good with accuracy and FPR measures and precision is also good.
- On the other hand, its performance is not good on TPR and FNR.
- For the purpose of improving the performance, one can include extra attributes in the datasets so that distinguishing the characteristics of the records will be better.
- Other ways are available for evaluating the performance of a classifier like N-fold cross validation or bootstrap.
- One option is **ROC (receiver operating characteristic) curve**, which is a common tool use for the evaluation of classifiers.
- It is possible for any classifier to reach the bottom left of the graph where TPR = FPR = 0 by the process of classifying everything as negative.
- In the same way, it is possible for any classifier to reach the top right of the graph where TPR = FPR = 1 by the process of classifying everything as positive.
- If a performance of any classifier is "at chance" by random speculating the results, it can reach to any point on the diagonal line TPR = FPR by selecting a suitable threshold of positive/negative.
- An ideal classifier must be able to accurately separate positives from negatives and in this way achieve the top-left corner (TPR = 1, FPR = 0).
- In such classifiers, the ROC curve moves straight up from TPR = FPR = 0 to the top-left corner and goes straight right to the top-right corner.
- Practically it is hard to achieve the top-left corner. But for a better classifier, it is responsibility to be closer to the top left, which distinguish it from other types of classifiers which are nearer to the diagonal line.
- AUC (**area under the curve**) is related to the ROC curve which computed by measuring the area under the ROC curve.
- Higher AUC indicates that the performance of classifier is better. The value of score is from 0.5 (for the diagonal line TPR=FPR) to 1.0 (with ROC going through the top-left corner).
- In the bank marketing example, there are two thousand instances in the training set.
- Also hundred instances are added as the testing set. Fig. 4.2.1 displays a ROC curve of the naïve Bayes classifier which is basically built on the training set of two thousand instances and whose testing is implemented on the testing set of hundred instances.
- Following R script is used to generate the figure.
- There is need of ROCR package for the purpose of plotting the ROC curve.
- The two thousand instances are present in a data frame known as banktrain, and the extra hundred instances are present in a data frame known as banktest.

```
library(ROCR)

# training set
banktrain <- read.table("bank-
sample.csv",header=TRUE,sep=",")
# drop a few columns
drops <- c("balance", "day", "campaign", "pdays", "previous",
"month")
banktrain <- banktrain [,!(names(banktrain) %in% drops)]

# testing set
banktest <- read.table("bank-sample-
test.csv",header=TRUE,sep=",")
banktest <- banktest [,!(names(banktest) %in% drops)]
# build the naïve Bayes classifier
nb_model <- naiveBayes(subscribed~.,
                       data=banktrain)
# perform on the testing set
nb_prediction <- predict(nb_model,
                         # remove column "subscribed"
                         banktest[,-ncol(banktest)],
```

```
                type='raw')
score <- nb_prediction[, c("yes")]
actual_class <- banktest$subscribed == 'yes'

pred <- prediction(score, actual_class)

perf <- performance(pred, "tpr", "fpr")
plot(perf, lwd=2, xlab="False Positive Rate (FPR)",
   ylab="True Positive Rate (TPR)")
abline(a=0, b=1, col="gray50", lty=3)
```

- The following R code illustrates that the subsequent AUC score of the ROC curve is about 0.915.

```
auc <- performance(pred, "auc")
auc <- unlist(slot(auc, "y.values"))
auc
[1] 0.9152196
```

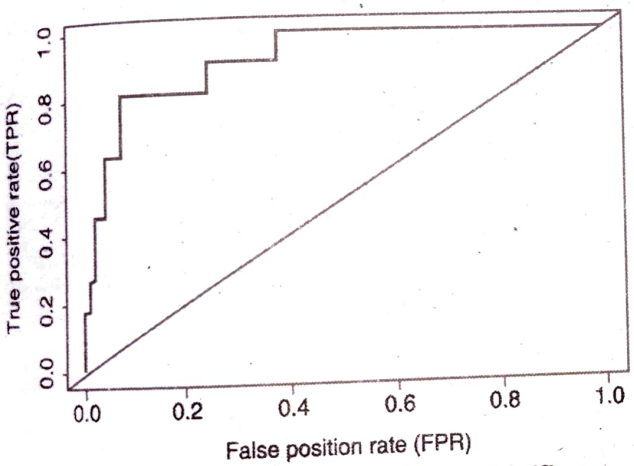

Fig. 4.2.1 : ROC curve of the naïve Bayes classifier on the bank marketing dataset

Syllabus Topic : Additional Classification Methods

4.2.6 Additional Classification Methods

Q. 4.2.6 Explain methods of additional classification. **(8 Marks)**
(Refer section 4.2.6)

- In addition to classifiers which we have seen here, there are number of methods available which are commonly used for classification purpose such as bagging, boosting, random forest, and support vector machines (SVM).
- There are some ensemble methods such as bagging, boosting, and random forest which use multiple models to get better predictive performance which can be easily gained from any of the constituent models.
- Bootstrap technique is used by the Bagging or bootstrap aggregating which repetitively samples with replacement from a dataset based on the uniform probability distribution.
- "With replacement" indicates that when a sample is choose for a training or testing set, then it is still stored in the dataset and there is a possibility that it may be selected again.
- Since the sampling is with replacement, few samples may emerge number of times in a training or testing set, while others may be absent.
- The training of a model or base classifier is implemented independently on every bootstrap sample, and a assignment of test sample done to the class which got the maximum number of votes.
- Just like bagging, boosting technique also uses votes for the purpose of classification to combine the output of individual models.
- Additionally, it integrates models of the similar type. However, boosting is considered as an iterative procedure in which there is influence of previously built models on new model.
- In addition, a weight is assigned by the boosting to every training sample which is going to reflect its importance, and it may be possible that the weight may change adaptively at the end of every boosting round.
- Random forest is considered as a class regarding the ensemble methods by the use of decision tree classifiers.
- It is a combination of tree predictors in such a way that each and every tree depends on the values of a random vector sampled separately and with the similar distribution for each and every tree in the forest.
- Sometimes bagging on decision trees is used by a unusual case of random forest, in which samples are randomly selected with replacement from the original training set.

RBRS

- SVM (Support vector machines) is one more common classification method which integrates linear models and instance-based learning techniques.
- SVM choose a small number of critical boundary instances which are known as support vectors from all the classes and make a linear decision method which do their separation as widely as possible.
- By default SVM can effectively carry out linear classifications and can be configured to carry out nonlinear classifications also.

❑❑❑