CHAPTER 5

Unit V

Classification

Syllabus Topics

Introduction to : Classification and Regression for Predictive Analysis, Decision Tree Induction, Rule-Based Classification : using IF-THEN Rules for Classification, Rule Induction Using a Sequential Covering Algorithm. Bayesian Belief Networks, Training Bayesian Belief Networks, Classification Using Frequent Patterns, Associative Classification, Lazy Learners-k-Nearest-Neighbor Classifiers, Case-Based Reasoning.

Syllabus Topic : Introduction to : Classification and Regression for Predictive Analysis

5.1 Introduction to : Classification and Regression for Predictive Analysis

- Classification constructs the classification model based on training data set and using that model classifies the new data.
- It predicts the value of classifying attribute or class label.

Typical applications

- Classify credit approval based on customer data.
- Target marketing of product.
- Medical diagnosis based on symptoms of patient.
- Treatment effectiveness analysis of patient based on their treatment given.

Various classification techniques

- Regression
- Decision trees
- Rules
- Neural networks

5.1.1 Classification is a Two Step Process

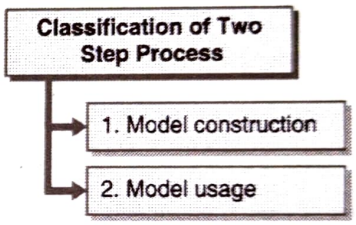


Fig. 5.1.1 : Classification of two step process

1. Model construction

- Every sample tuple or object has assigned a predefined class label.
- Those set of sample tuples or subset data set is known as training data set.
- The constructed model based on training data set is represented as classification rules, decision trees or mathematical formulae.

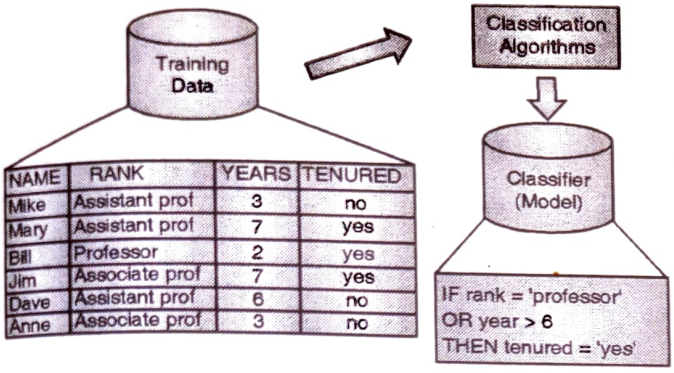
2. Model usage

- For classifying unknown objects or new tuple use the constructed model.
- Compare the class label of test sample with the resultant class label.
- Estimate accuracy of the model by calculating the percentage of test set samples that are correctly classified by the model constructed.

- Test sample data and training data samples are always different, otherwise over-fitting will occur.

☞ **Example**

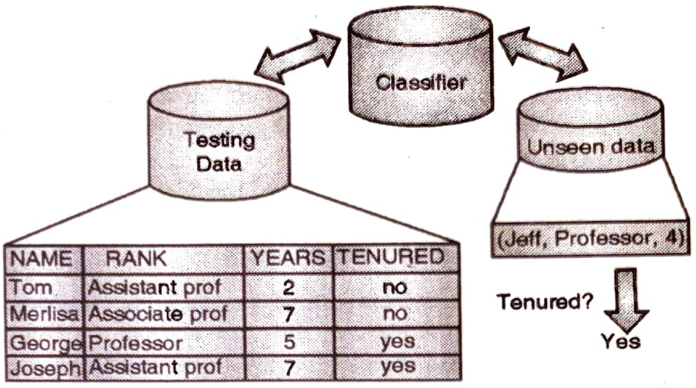
Classification process : (1) Model construction



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

Fig. 5.1.2 : Learning : Training data are analyzed by a classification algorithm

Classification process : (2) Model usage (Use the model in prediction)



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

Fig. 5.1.3 : Classification : Test data are used to estimate the accuracy of the classification rule

For example

How to perform classification task for classification of medical patients by their disease ?

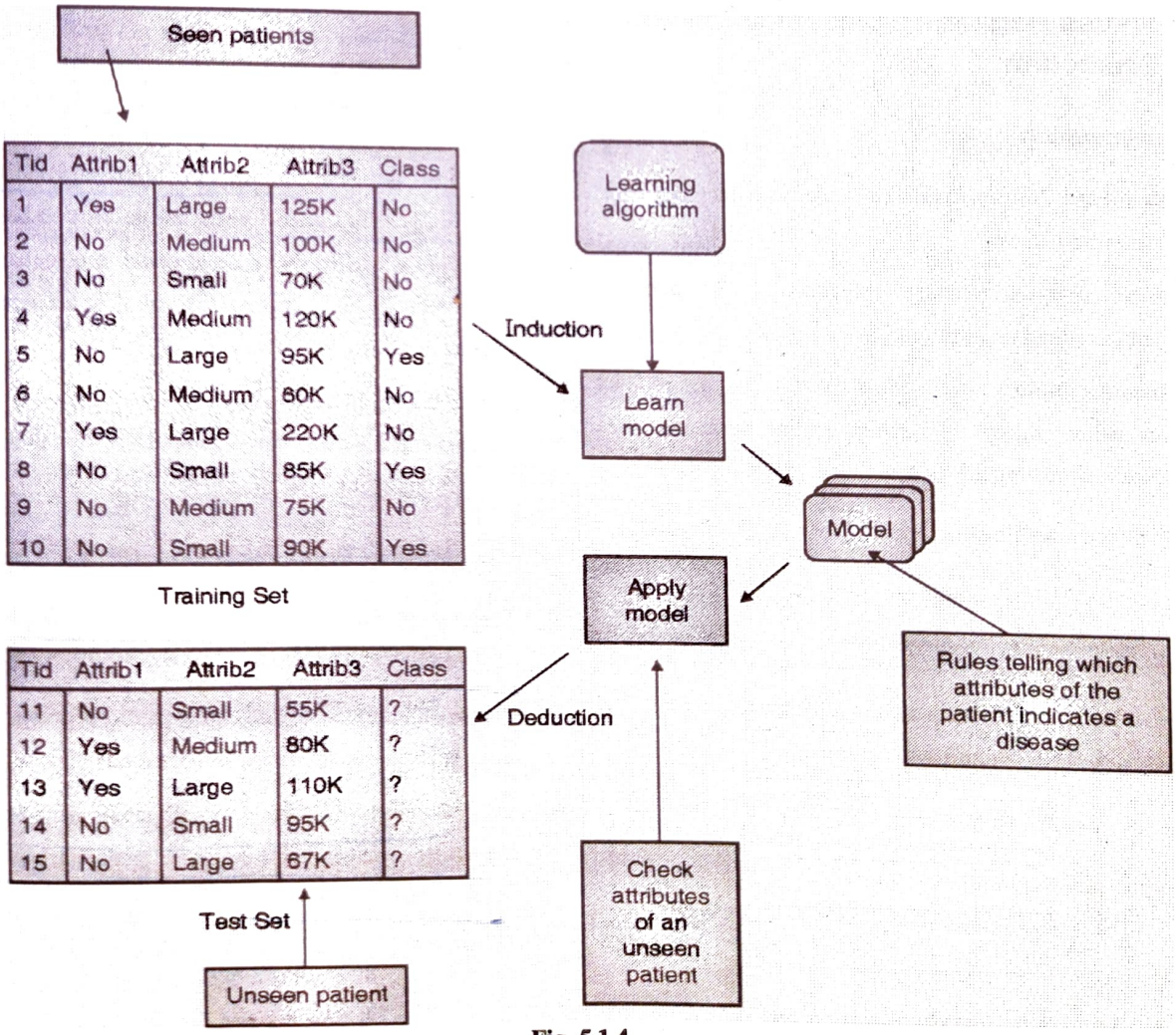


Fig. 5.1.4

5.1.2 Difference between Classification and Prediction

Sr. No.	Classification	Prediction
1.	Classification is a major type of prediction problem where classification is used to predict discrete or nominal values.	Prediction can be viewed as the construction and use of a model to assess the class of an unlabeled sample.
2.	Classification is the use of prediction to predict class labels.	It is used to assess the values or value ranges of an attribute that a given sample is likely to have.
3.	E.g. Group patients based on their known medical data and treatment outcome then it's a classification.	E.g. if a classification model is used to predict the treatment outcome for a new patient, then it would be a prediction.

5.1.3 Issues Regarding Classification and Prediction

☞ **Data preparation**

- **Data cleaning :** Pre-process data in order to reduce noise and handle missing values.
- **Relevance analysis (feature selection) :** Remove the irrelevant or redundant attributes.
- **Data transformation :** Generalize the data to higher level concepts using concept hierarchies and/or normalize data which involves scaling the values.

☞ **Evaluating classification methods**

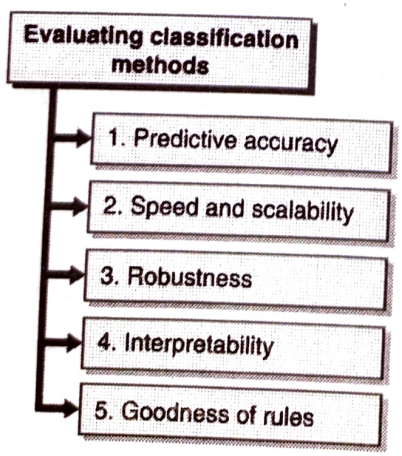


Fig. 5.1.5 : Evaluating classification methods

➜ **1. Predictive accuracy**

- This refers the ability of the model to correctly predict the class label of new or previously unseen data.

➜ **2. Speed and scalability**

- Time to construct the model.
- Time to use the model.
- Efficiency in disk-resident databases.

➜ **3. Robustness**

Handling noise and missing values.

➜ **4. Interpretability**

Understanding and insight provided by the model.

➜ **5. Goodness of rules**

- Decision tree size.
- Compactness of classification rules.

5.1.4 Regression

- Suppose an employee needs to predict how much rise he will get in his salary after 5 years, means he bother to predict the numeric value. In this case a model is constructed based on his previous salary values that predicts a continuous-valued function or ordered value.
- Prediction is generally about the future values or the unknown events and it models continuous-valued functions.
- Most commonly used methods for prediction is regression.

☞ **Structure of Regression Model**

- Regression Model represents reality by using the system of equations.
- Regression model explains relationship between variables and also enables quantification of these relationships.
- It determines the strength of relationship between one dependent variable with the other independent variable using some statistical measure.
- Dependent variable is usually denoted by Y.

- The two basic types of regression :

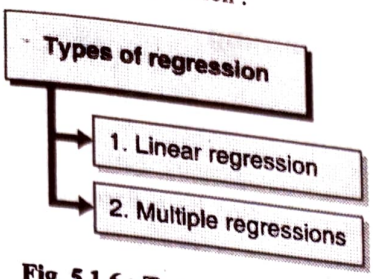


Fig. 5.1.6 : Types of Regression

The general form of regression is :

Linear regression : $Y = m + nX + u$

Multiple regression : $Y = m + n_1X_1 + n_2X_2 + n_3X_3 + ... + n_tX_t + u$

Where :

Y = The dependent variable which we are trying to predict

X = The independent variable that we are using to predict variable Y

m = The intercept

n = The slope

u = The regression residual.

- In multiple regressions each variable is differentiated with subscripted numbers.
- Regression uses a group of random variables for prediction and finds a mathematical relationship between them. This relationship is depicted in the form of a straight line (linear regression) that approximates all the points in the best way.
- Regression may be used to determine for e.g. price of a commodity, interest rates, the price movement of an asset influenced by industries or sectors.

☞ Linear Regression

Regression tries to find the mathematical relationship between variables, if it is a straight line then it is a linear model and if it gives a curved line then it is a non linear model.

(A) Simple linear regression

- The relationship between dependent and independent variable is described by straight line and it has only one independent variable.

$$Y = \alpha + \beta X$$

- Two parameters, α and β specify the (Y-intercept and slope of the) line and are to be estimated by using the data at hand.
- The value of Y increases or decreases in a linear manner as the value of X changes accordingly.
- Draw a line relating to Y and X which is well fitted to given data set.
- The idea situation is that if the line which is well fitted for all the data points and no error for prediction.
- If there is random variation of data points, which are not fitted in a line then construct a probabilistic model related to X and Y.
- Simple linear regression model assumes that data points deviates about the line, as shown in the Fig. 5.1.7.

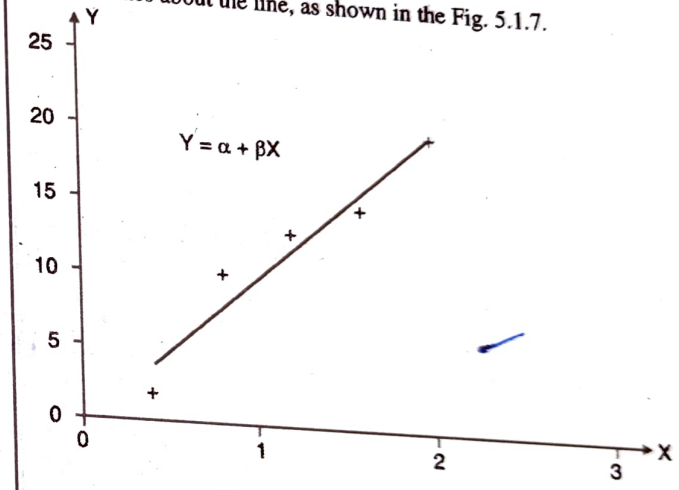


Fig. 5.1.7 : Linear regression

(B) Multiple Linear Regression

- Multiple linear regression is an extension of simple linear regression analysis .
- It uses two or more independent variables to predict the outcome and a single continuous dependent variable.

$$Y = a_0 + a_1X_1 + a_2X_2 + ... + a_kX_k + e$$

Where, Y is the dependent variable or response variable

$X_1, X_2 .....X_k$ are the independent variables or predictors e is random error.

$a_0, a_1, a_2.....a_k$ are the regression coefficients

☞ Other Regression Model

- In log linear regression a best fit between the data and a loglinear model is found.

- Major assumption : A linear relationship exists between the log of the dependent and independent variables.
- Loglinear models are models that postulate a linear relationship between the independent variables and the logarithm of the dependent variable, for example :

$$\log(y) = a_0 + a_1 x_1 + a_2 x_2 \ldots + a_N x_N$$

where y is the dependent variable; x_i, i=1,...,N are independent variables and $\{a_i$, i=0,...,N$\}$ are parameters (coefficients) of the model.

- For example, log linear models are widely used to analyze categorical data represented as a contingency table. In this case, the main reason to transform frequencies (counts) or probabilities to their log-values is that, provided the independent variables are not correlated with each other, the relationship between the new transformed dependent variable and the independent variables is a linear (additive) one.

Syllabus Topic : Decision Tree Induction

5.2 Decision Tree Induction Classification Methods

Classifications methods are given below :

1. Decision Tree Induction : Attribute selection measures, tree pruning.
2. Bayesian Classification : Naïve Bayes' classifier

- Training dataset should be class-labeled for learning of decision trees in decision tree induction.
- A decision tree represents rules and it is very a popular tool for classification and prediction.
- Rules are easy to understand and can be directly used in SQL to retrieve the records from database.
- To recognize and approve the discovered knowledge got from decision model is very crucial task.
- There are a many algorithms to builddecision trees :
  - ID3 (Iterative Dichotomiser 3)
  - C4.5 (Successor of ID3)
  - CART (Classification And Regression Tree)
  - CHAID (CHi-squared Automatic Interaction Detector

5.2.1 Appropriate Problems for Decision Tree Learning

Decision tree learning is appropriate for the problems having the characteristics given below :

- Instances are represented by a fixed set of attributes (e.g. gender) and their values (e.g. male, female) described as **attribute-value pairs**.
- If the attribute has small number of disjoint possible values (e.g. high, medium, low) or there are only two possible classes (e.g. true, false) then decision tree learning is easy.
- Extension to decision tree algorithm also handles real value attributes (e.g. salary).
- Decision tree gives a class label to each instance of dataset.
- Decision tree methods can be used even when some training examples have unknown values (e.g. humidity is known for only a fraction of the examples).
- Learned functions are either represented by a decision tree or re-represented as sets of if-then rules to improve readability.

5.2.2 Decision Tree Representation

Decision tree classifier has tree type structure which has leaf nodes and decision nodes.

- **A leaf node** is the last node of each branch and indicates class label or value of target attribute.
- **A decision node** is the node of tree which has leaf node or sub-tree. Some test to be carried on the each value of decision node to get the decision of class label or to get next sub-tree.

Example : Decision tree representation for play tennis.

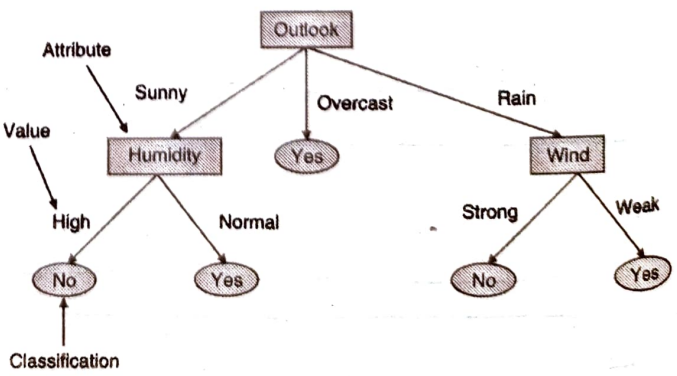


Fig. 5.2.1 : Representation of decision tree

Other representation for play tennis

- Logical expression for Play tennis=Yes is given below,
  - (Outlook= sunny ∧humidity=normal) ∨ (outlook=overcast) ∨outlook = rain ∧wind=weak)
- **If-then rules :**
  - IF outlook = sunny ∧ humidity = normal THEN play tennis = Yes
  - IF outlook = sunny ∧ humidity = high THEN play tennis = No
  - IF outlook = overcast THEN play tennis = Yes
  - IF outlook = rain ∧ wind = weak THEN play tennis = Yes
  - IF outlook = rain ∧ wind = strong THEN play tennis = No

5.2.3 Algorithm for Inducing a Decision Tree

The Basic ideas behind ID3 :

- C4.5 is an extension of ID3.
- C4.5 accounts for unavailable values, continuous attribute value ranges, pruning of decision trees, rule derivation and so on.
- C4.5 is a designed by Quinlan to address the following issues not given by ID3 :
  - It avoids over fitting the data.
  - It determines the depth of decision tree and reduces the error pruning.
  - It also handles continuous value attributes e.g. Salary or temperature.
  - It works for missing value attribute and handles suitable attribute selection measure.
  - It gives better the efficiency of computation. The algorithm to generate decision tree is given by Jiawei Han et al. as below :

Q. Write a pseudo code for the construction of Decision Tree State and justify its time complexity also. **Dec. 15, 4 Marks**

Algorithm : Generate_decision_tree : Generate a decision tree from the training tuples of data partition, D.

Input

- Data partition, D, which is a set of training tuples and their associated class labels;
- Attribute_list, the set of candiadate attributes;
- Attribute_selection_method, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a splitting_attribute and, possibly, either a split-point or splitting subset.

Output

A decision tree.

Method

1. create a node N;
2. if tuples in D are all of the same class, C, then
3. return N as a leaf node labeled with the class C;
4. if attribute_list is empty then
5. return N as a leaf node labeled with the majority class in D; // majority voting
6. applyAttribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
7. label node N with splitting_criterion;
8. if splitting_attribute is discrete-valued and Multiway splits allowed then // not restricted to binary trees
9. Attribute_list←attribute_list – splitting_attribute; // remove splitting_attribute
10. for each outcome j of splitting_criterion // partition the tuples and grow subtrees for each partition
11. let D_j be the set of data tuples in D satisfying outcome j; // a partition
12. if D_j is empty then
13. attach a leaf labeled with the majority class in D to node N;
14. else attach the node returned by Generate_decision_tree (D_j, attribute_list) to node N; endfor
15. return N;

- **Time complexity :** For a normal style decision tree such as C4.5 the time complexity is O(N D^2), where D is the number of features. A single level decision tree would be O(N D)
- It gives better the efficiency of computation.

5.2.4 Tree Pruning

- Because of noise or outliers, the generated tree may overfit due to many branches.
- To avoid overfitting, prune the tree so that it is not too specific.

☞ **Prepruning**

- Start pruning in the beginning while building the tree itself.
- Stop the tree construction in early stage.
- Avoid splitting a node by checking the threshold with the goodness measure falling below a threshold.
- Selection of correct threshold is difficult in prepruning.

☞ **Postpruning**

- Build the full tree then start pruning, remove the branches.
- Use different set of data than training data set to get the best pruned tree.

5.2.5 Examples of ID3

Ex. 5.2.1 : Apply ID3 on the following training dataset from all electronics customer database and extract the classification rule from the tree.

Table P. 5.2.1 : Training data of customer

Age	Income	Student	Credit_rating	Class: 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

Soln. :

Class P : buys_computer = "yes"

Class N : buys_computer = "no"

Total number of records 14.

Count the number of records with "yes" class and "no" class.

So number of records with "yes" class = 9 and "no" class = 5

So Information gain = I (p, n)

$$= -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$I(p, n) = I(9, 5) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940$

Step 1 : Compute the entropy for age :

For age <=30,

p_i = with "yes" class = 2 and n_i = with "no" class = 3

Therefore, $I(p_i, n_i) = I(2,3) = 0.971$.

Similarly for different age ranges $I(p_i, n_i)$ is calculated as given below :

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

So, the expected information needed to classify a given sample if the samples are partitioned according to age is,

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

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

Hence

$$Gain(age) = I(p, n) - E(age)$$

$$= 0.940 - 0.694 = 0.246$$

Similarly, Gain (income) = 0.029

Gain (student) = 0.151

Gain (credit_rating) = 0.048

Now the age has the highest information gain among all the attributes, so select age as test attribute and create the node as age and show all possible values of age for further splitting.

Since Age has three possible values, the root node has three branches (<=30, 31...40, >40).

Step 2 :

The next question is "what attribute should be tested at the Age branch node?" Since we have used Age at the root, now we have to decide on the remaining three attributes:income, student, or credit_rating.

Consider Age : <= 30 and count the number of tuples from the original given training set

$S_{<=30}$ = 5 (Age: <=30)

Age	Income	Student	Credit_rating	Buys_computer
<=30	High	No	Fair	No
<=30	High	No	Excellent	No
<=30	Medium	No	Fair	No
<=30	Low	Yes	Fair	Yes
<=30	Medium	Yes	Excellent	Yes

Note : Refer above table :
Total number of Yes tuple = 2 and total number of No tuple = 3
I (p, n) = I (2, 3) = – (2/5) $\log_2(2/5)$ – (3/5) $\log_2(3/5)$ = 0.971

(i) Compute the entropy for income : (High, medium, low)

For Income = High,

p_i = with "yes" class = 0 and n_i = with "no" class = 2

Therefore , $I(p_i, n_i) = I(0,2) = -(0/2)\log_2(0/2) - (2/2)\log_2(2/2) = 0$.

Similarly for different age ranges $I(p_i, n_i)$ is calculated as given below :

Income	p_i	n_i	$I(p_i, n_i)$
High	0	2	0
Medium	1	1	1
Low	1	0	0

Calculate entropy using the values from the above table and the formula given as :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

E (Income) = 2/5 * I (0, 2) + 2/5* I(1, 1) + 1/5 *I(1, 0) = 0.4

Note : $S_{<=30}$ is the total training set.

Hence

Gain($S_{<=30}$, Income) = I (p, n) – E (Income)

= 0.971 – 0.4 = 0.571

(ii) Compute the entropy for Student : (No , yes)

For Student = No,

p_i = with "yes" class = 0 and n_i = with "no" class = 3

Therefore, $I(p_i, n_i) = I(0,3) = -(0/3)\log_2(0/3) - (3/3)\log_2(3/3) = 0$.

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Student	p_i	n_i	$I(p_i, n_i)$
No	0	3	0
Yes	2	0	0

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

E (Student) = 3/5 * I (0, 3) + 2/5* I(2, 0) = 0

Note : $S_{<=30}$ is the total training set.

Hence

Gain($S_{<=30}$, Student) = I (p, n) – E (Student)

= 0.971 – 0 = 0.971

(iii) Compute the entropy for credit_rating: (Fair, excellent)

For credit_rating = Fair,

p_i = with "yes" class = 1 and n_i = with "no" class = 2

Therefore

I(pi , ni) = I(1, 2)

= – (1/3) log2 (1/3) – (2/3) log2 (2/3) = 0.918

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Credit_rating	p_i	n_i	$I(p_i, n_i)$
Fair	1	2	0.918
Excellent	1	1	1

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

E (Credit_rating) = 3/5 * I (1, 2) + 2/5* I(1, 1) = 0.951

Note : $S_{<=30}$ is the total training set.

Hence

Gain($S_{<=30}$, credit_rating) = I (p, n) – E (credit_rating)
= 0.971 – 0.951 = 0.02

Therefore,

Gain($S_{<=30}$, student) = 0.970

Gain($S_{<=30}$, income) = 0.570

Gain($S_{<=30}$, credit_rating) = 0.02

Student has the highest gain; therefore, it is below Age : "<=30".

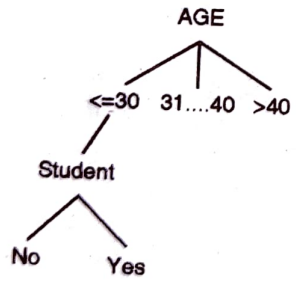


Fig. P. 5.2.1(a)

Step 3 :

Consider now only income and credit rating for age : 31...40 and count the number of tuples from the original given training set

$S_{31...40}$ = 4 (age : 31...40)

Age	Income	Student	Credit_rating	Buys_computer
31...40	High	No	Fair	Yes
31...40	Low	Yes	Excellent	Yes
31...40	Medium	No	Excellent	Yes
31...40	High	Yes	Fair	Yes

Since for the attributes income and credit_rating, buys_computer = yes, so assign class 'yes' to 31...40

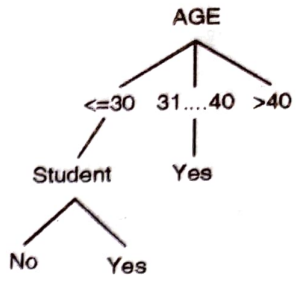


Fig. P. 5.2.1(b)

Step 4 :

Consider income and credit_rating for age: >40 and count the number of tuples from the original given training set

$S_{>40}$ = (age : > 40)

Age	Income	Student	Credit_rating	Buys_computer
>40	Medium	No	Fair	Yes
>40	Low	Yes	Fair	Yes
>40	Low	Yes	Excellent	No
>40	Medium	Yes	Fair	Yes
>40	Medium	No	Excellent	No

Consider the above table as the new training set and calculate the Gain for income and credit_rating

Class P :buys_computer = "yes"

Class N: buys_computer = "no"

Total number of records 5.

Count the number of records with "yes" class and "no" class.

So number of records with "yes" class = 3 and "no" class = 2

So Information gain = I (p, n)

$$= -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

I (p, n) = I (3, 2) = – (3/5) $\log_2$ (3/5) – (2/5) $\log_2$ (2/5)

= 0.970

(iv) Compute the entropy for credit_rating

For credit_rating = Fair

p_i = with "yes" class = 3 and n_i = with "no" class = 0

Therefore , $I(p_i, n_i) = I(3,0) = 0$.

For credit_rating = Excellent

p_i = with "yes" class = 0 and n_i = with "no" class = 2

Therefore, $I(p_i, n_i) = I(0,2) = 0$

Similarly for different age ranges $I(p_i, n_i)$ is calculated as given below :

Credit_rating	p_i	n_i	$I(p_i, n_i)$
Fair	3	0	0
Excellent	0	2	0

Calculate entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$$E(\text{Credit\_rating}) = \frac{3}{5} I(3,0) + \frac{2}{5} I(0,2) = 0$$

Hence

$$\text{Gain}(S_{>40}, \text{credit\_rating}) = I(p, n) - E(\text{credit\_rating})$$

$$= 0.970 - 0 = 0.970$$

(v) **Compute the entropy for income : (High, medium, low)**

For Income = High,

p_i = with "yes" class = 0 and n_i = with "no" class = 0

Therefore, $I(p_i, n_i) = I(0,0) = 0$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Income	p_i	n_i	$I(p_i, n_i)$
High	0	0	0
Medium	2	1	0.918
Low	1	1	1

Calculate Entropy using the values from the above table and the formula given below

$$E(\text{Income}) = 0/5 * I(0, 0) + 3/5 * I(2, 1) + 2/5 * I(1, 1) = 0.951$$

Note : $S_{>40}$ is the total training set.

Hence

$$\text{Gain}(S_{>40}, \text{income}) = I(p, n) - E(\text{income})$$

$$= 0.970 - 0.951 = 0.019$$

Therefore,

$$\text{Gain}(S_{>40}, \text{income}) = 0.019$$

$$\text{Gain}(S_{>40}, \text{Credit\_rating}) = 0.970$$

Credit_rating has the highest gain; therefore, it is below Age: ">40".

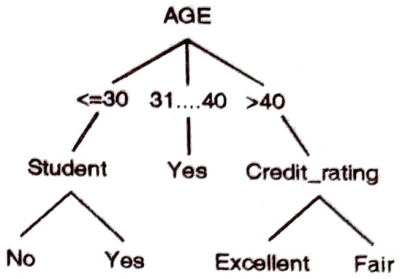


Fig. P. 5.2.1(c)

Output : A Decision Tree for "buys_computer"

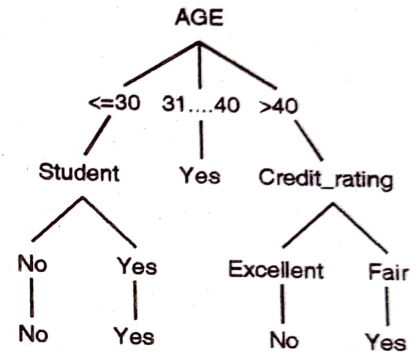


Fig. P. 5.2.1(d) : Decision tree for "buys computer"

Extracting classification rules from trees

Example

IF age = "<=30" AND student = "no"

THEN buys_computer = "no"

IF age = "<=30" AND student = "yes"

THEN buys_computer = "yes"

IF age = "31…40" THEN buys_computer = "yes"

IF age = ">40" AND credit_rating = "excellent"

THEN buys_computer= "no"

IF age = ">40" AND credit_rating = "fair"

THEN buys_computer = "yes"

Ex. 5.2.2 The weather attributes are outlook, temperature, humidity, and wind speed. They can have the following values :

Outlook = {sunny, overcast, rain}

temperature = {hot, mild, cool}

humidity = {high, normal}

wind = {weak, strong}

Sample data set S are :

Table P. 5.2.2 : Training data set for Play Tennis

Day	Outlook	Temperature	Humidity	Wind	Play ball
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

We need to find which attribute will be the root node in our decision tree. The gain is calculated for all four attributes using formula of gain (A).

Soln. :

Class P : Playball = "yes"

Class N : Playball = "no"

Total number of records 14.

Count the number of records with "yes" class and "no" class.

So number of records with "yes" class = 9 and "no" class = 5

So Information gain = I (p, n)

$$= -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$I(p, n) = I(9, 5)$

$= -(9/14)\log_2(9/14) - (5/14)\log_2(5/14)$

$= (-0.643) * (-0.637) + (-0.357) * (-1.485)$

$= 0.409 + 0.530 = 0.940$

Step 1 : Compute the entropy for outlook : (Sunny, overcast , rain)

For outlook = sunny,

p_i = with "yes" class = 2 and n_i = with "no" class = 3

Therefore ,

$I(p_i , n_i) = I(2,3)$

$= -(2/5)\log_2(2/5) - (3/5)\log_2(3/5) = 0.971.$

Similarly for different outlook ranges $I(p_i , n_i)$ is calculated as given below :

Outlook	p_i	n_i	$I(p_i, n_i)$
Sunny	2	3	0.971
Overcast	4	0	0
Rain	3	2	0.971

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

$$E(\text{outlook}) = \frac{5}{14} I(2, 3) + \frac{4}{14} I(4, 0) + \frac{5}{14} I(3, 2) = 0.694$$

Note : T is the total training set.

Hence Gain(T, outlook) = I (p, n) – E (outlook)

= 0.940 – 0.694 = 0.246

Similarly,

Gain (T, Temperature) = 0.029

Gain (T, Humidity) = 0.151

Gain (T, Wind) = 0.048

Outlook shows the highest gain, so it is used as the decision attribute in the root node.

As Outlook has only values "sunny, overcast, rain", the root node has three branches

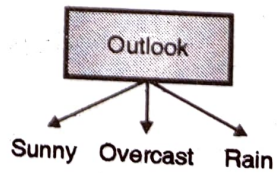


Fig. P. 5.2.2(a)

Step 2 :

As attribute outlook at root, we have to decide on the remaining three attribute for sunny branch node.

Consider outlook = Sunny and count the number of tuples from the original given training set

S_{sunny} = {1, 2, 8, 9, 11}

= 5 (From Table. P.5.2.2, outlook = sunny)

Day	Outlook	Temperature	Humidity	Wind	Play ball
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Note : Refer Table P. 5.2.2 :

Total number of Yes tuple = 2 and total number of No tuple= 3

I (p, n) = I (2, 3)

$= -(2/5)\log_2(2/5) - (3/5)\log_2(3/5) = 0.971$

(i) Compute the entropy for temperature: (Hot, mild, cool)

For Temperature = Hot,

p_i = with "yes" class = 0 and n_i = with "no" class = 2

Therefore ,

$I(p_i, n_i)$ = $I(0,2) = -(0/2)\log_2(0/2) - (2/2)\log_2(2/2) = 0$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Temperature	p_i	n_i	$I(p_i, n_i)$
Hot	0	2	0
Mild	1	1	1
Cool	1	0	0

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

E (Temperature) = 2/5 * I (0, 2) + 2/5* I(1, 1) + 1/5 *I(1, 0)

= 0.4

Note : T_{sunny} is the total training set.

Hence

Gain(T_{sunny}, temperature) = I (p, n) – E (temperature)

= 0.971 – 0.4 = 0.571

(ii) Compute the entropy for humidity : (High, normal)

For Humidity = High,

p_i = with "yes" class = 0 and n_i = with "no" class = 3

Therefore ,

I(pi , ni) = I(0,3)

$= -(0/3)\log2(0/3) - (3/3)\log2(3/3) = 0$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Humidity	p_i	n_i	$I(p_i, n_i)$
High	0	3	0
Normal	2	0	0

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

E (Humidity) = 3/5 * I (0, 3) + 2/5* I(2, 0) = 0

Note : T_{sunny} is the total training set.

Hence

Gain(T_{sunny}, Humidity) = I (p, n) – E (Humidity)

= 0.971 – 0 = 0.971

(iii) Compute the entropy for wind : (Weak, strong)

For wind = weak,

p_i = with "yes" class = 1 and n_i = with "no" class = 2

Therefore,

$I(p_i, n_i)$ = I(1,2)

$= -(1/3)\log_2(1/3) - (2/3)\log_2(2/3) = 0.918$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Wind	p_i	n_i	$I(p_i, n_i)$
Weak	1	2	0.918
Strong	1	1	1

Calculate Entropy using the values from the above table and the formula given as:

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

E (Wind) = 3/5 * I (1, 2) + 2/5* I (1, 1) = 0.951

Note : T_{sunny} is the total training set.

Hence Gain(T_{sunny}, Wind) = I (p, n) – E (Wind)

= 0.971 – 0.951 = 0.02

Therefore,

Gain(T_{sunny}, Humidity) = 0.970

Gain(T_{sunny}, Temperature) = 0.570

Gain(T_{sunny}, Wind) = 0.02

Humidity has the highest gain; therefore, it is below Outlook = "sunny".

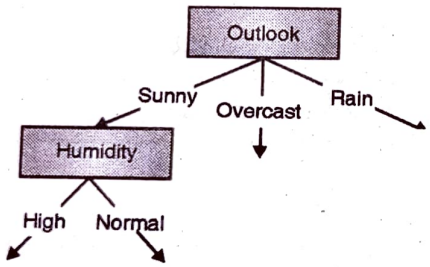


Fig. P. 5.2.2(b)

Step 3 :

Consider now only temperature and wind for outlook = Overcast and count the number of tuples from the original given training set

$T_{overcast}$ = {3,7,12,13}

= 4 (From Table. P.5.2.2, outlook = overcast)

Day	Outlook	Temperature	Humidity	Wind	Play ball
3	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes

Since for the attributes temperature and wind, playball = yes, so assign class 'yes' to overcast.

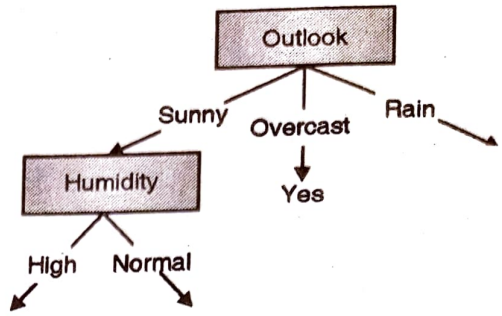


Fig. P. 5.2.2(c)

Step 4 :

Consider temperature and wind for outlook = Rain and count the number of tuples from the original given training set

T_{rain} = {4, 5, 6, 10, 14}

= 5 (From Table. P. 5.2.2, outlook = rain)

Day	Outlook	Temperature	Humidity	Wind	Play ball
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Consider the above table as the new training set and calculate the Gain for temperature and Wind.

Class P : Playball = "yes"

Class N : Playball = "no"

Total number of records 5

Count the number of records with "yes" class and "no" class.

So number of records with "yes" class = 3 and "no" class = 2

So Information gain = $I(p, n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$

$I(p, n) = I(3, 2) = -(3/5)\log_2(3/5) - (2/5)\log_2(2/5) = 0.970$

(iv) Compute the entropy for Wind

For Wind = Weak

p_i = with "yes" class = 3 and n_i = with "no" class = 0

Therefore, $I(p_i, n_i) = I(3,0) = 0$.

For Wind = strong

p_i = with "yes" class = 0 and n_i = with "no" class = 2

Therefore, $I(p_i, n_i) = I(0, 2) = 0$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Wind	p_i	n_i	$I(p_i, n_i)$
Weak	3	0	0
Strong	0	2	0

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$$E(\text{Wind}) = \frac{3}{5} I(3, 0) + \frac{2}{5} I(0, 2) = 0$$

Hence $\text{Gain}(T_{rain}, \text{Wind}) = I(p, n) - E(\text{Wind})$

$$= 0.970 - 0 = 0.970$$

(v) Compute the entropy for Temperature : (Hot, mild , cool)

For Temperature = Hot,

p_i = with "yes" class = 0 and n_i = with "no" class = 0

Therefore, $I(p_i, n_i) = I(0,0) = 0$

Similarly for different outlook ranges $I(p_i, n_i)$ is calculated as given below :

Temperature	p_i	n_i	$I(p_i, n_i)$
Hot	0	0	0
Mild	2	1	0.918
Cool	1	1	1

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$E(\text{Temperature}) = 0/5 * I(0, 0) + 3/5 * I(2, 1) + 2/5 * I(1, 1)$

$$= 0.951$$

Note : T_{Rain} is the total training set.

Hence

$\text{Gain}(T_{Rain}, \text{temperature}) = I(p, n) - E(\text{temperature})$

$$= 0.970 - 0.951 = 0.019$$

Therefore,

$\text{Gain}(T_{rain}, \text{Temperature}) = 0.019$

$\text{Gain}(T_{rain}, \text{Wind}) = 0.970$

Wind has the highest gain; therefore, it is below outlook = "rain".

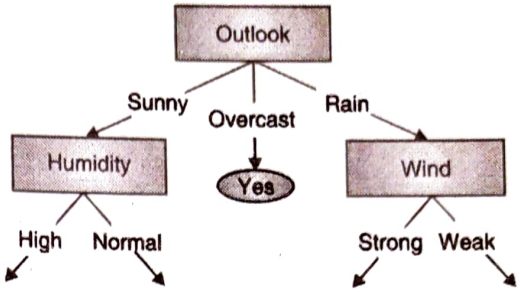


Fig. P. 5.2.2(d)

Therefore the final decision tree is :

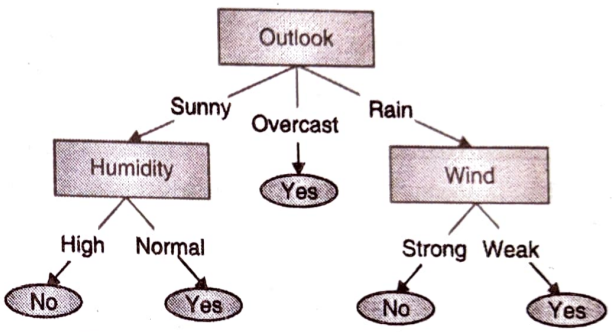


Fig. P. 5.2.2(e) : Decision tree for play tennis

The decision tree can also be expressed in rule format :

IF outlook = sunny AND humidity = high THEN playball = no

IF outlook = Sunny AND humidity = normal THEN playball = yes

IF outlook = overcast THEN playball = yes

IF outlook = rain AND wind = strong THEN playball = no

IF outlook = rain AND wind = weak THEN playball = yes

Ex. 5.2.3 : A sample training dataset for stock market is given below. Profit is the class attribute and value is based on age, contest and type.

Age	Contest	Type	Profit
Old	Yes	Swr	Down
Old	No	Swr	Down
Old	No	Hwr	Down
Mid	Yes	Swr	Down
Mid	Yes	Hwr	Down
Mid	No	Hwr	Up
Mid	No	Swr	Up
New	Yes	Swr	Up
New	No	Hwr	Up
New	No	Swr	Up

Soln. : In the stock market case the decision tree is :

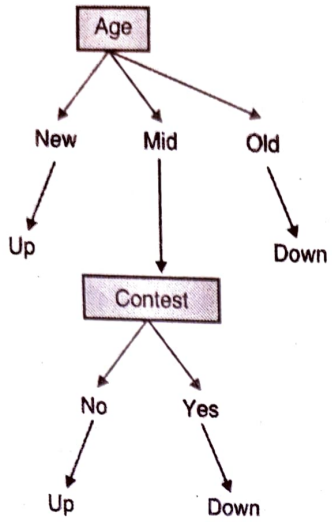


Fig. P. 5.2.3

Ex. 5.2.4 : Using the following training data set. Create classification model using decision-tree and hence classify following tuple.

Tid	Income	Age	Own House
1.	Very High	Young	Yes
2.	High	Medium	Yes
3.	Low	Young	Rented
4.	High	Medium	Yes
5.	Very high	Medium	Yes
6.	Medium	Young	Yes
7.	High	Old	Yes
8.	Medium	Medium	Rented
9.	Low	Medium	Rented
10.	Low	Old	Rented
11.	High	Young	Yes
12.	medium	Old	Rented

Soln. :

Class P : Own house = "yes"

Class N : Own house = "rented"

Total number of records 12

Count the number of records with "yes" class and "rented" class.

So number of records with "yes" class = 7 and "no" class = 5

So Information gain = I (p, n)

$$= -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$$I(p, n)=I(7, 5) = -(7/12)\log_2(7/12) - (5/12)\log_2(5/12) = 0.979$$

Step 1 : Compute the entropy for Income : (Very high, high, medium, low)

For Income =Very high,

p_i = with "yes" class = 2 and n_i = with "no" class = 0

Therefore, $I(p_i, n_i) = I(2,0) = 0$

Similarly for different Income ranges $I(p_i, n_i)$ is calculated as given below :

Income	p_i	n_i	$I(p_i, n_i)$
Very high	2	0	0
High	4	0	0
Medium	1	2	0.918
Low	0	3	0

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

E (Income) = 2/12 * I(2,0)+4/12*I(4,0)+ 3/12*I(0,3)=0.229

Note : S is the total training set.

Hence

Gain(S, Income) = I (p, n) – E (Income)

= 0.979 – 0.229 = 0.75

Step 2 : Compute the entropy for Age : (Young , medium, old)

Similarly for different age ranges $I(p_i, n_i)$ is calculated as given below :

Age	p_i	n_i	$I(p_i, n_i)$
Young	3	1	0.811
Medium	3	2	0.971
Old	1	2	0.918

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

E (Age) = 4/12 * I(3,1)+5/12*I(3,2)+3/12*I(1,2)

= 0.904

Note : S is the total training set.

Hence

$$\text{Gain}(S, \text{age}) = I(p, n) - E(\text{age})$$

$$= 0.979 - 0.904 = 0.075$$

Income attribute has the highest gain, therefore it is used as the decision attribute in the root node.

Since income has four possible values, the root node has four branches (very high, high, medium, low).

Fig. P. 5.2.4(a)

Step 3 :

Since we have used income at the root, now we have to decide on the age attribute.

Consider income = "very high" and count the number of tuples from the original given training set

$$S_{\text{very high}} = 2$$

Since both the tuples have class label = "yes" , so directly give "yes" as a class label below "very high".

Similarly check the tuples for income = "high" and income = "low" , are having the class label "yes" and "rented" respectively.

Now check for income = "medium", where number of tuples having "yes" class label is 1 and tuples having "rented" class label are 2.

So put the age label below income="medium".

So the final decision tree is :

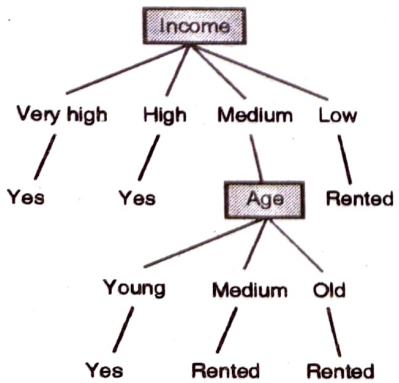
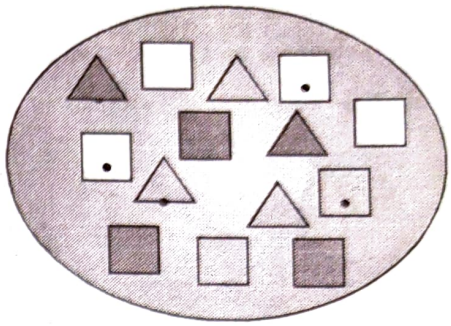


Fig. P. 5.2.4(b)

Ex. 5.2.5 : Data Set: A set of classified objects is given as below. Apply ID3 to generate tree.



	Attribute			
Sr. No	Colour	Outline	Dot	Shape
1	Green	Dashed	No	Triangle
2	Green	Dashed	Yes	Triangle
3	Yellow	Dashed	No	Square
4	Red	Dashed	No	Square
5	Red	Solid	No	Square
6	Red	Solid	Yes	Triangle
7	Green	Solid	No	Square
8	Green	Dashed	No	Triangle
9	Yellow	Solid	Yes	Square
10	Red	Solid	No	Square
11	Green	Solid	Yes	Square
12	Yellow	Dashed	Yes	Square
13	Yellow	Soild	No	Square
14	Red	Dashed	yes	Triangle

Soln. :

Class N : Shape = "Triangle"

Class P: Shape = "Square"

Total number of records 14

Count the number of records with "triangle" class and "square" class.

So number of records with "triangle" class = 5 and "square" class = 9

$$P(\text{square}) = 9/14$$

$$P(\text{triangle}) = 5/14$$

So information gain $= I(p, n)$

$$= -\frac{p}{p+n}\log 2\frac{p}{p+n} - \frac{n}{p+n}\log 2\frac{n}{p+n}$$

$I(p, n) = I(9,5)$

$= -(9/14)\log2(9/14) - (5/14)\log2(5/14)$

$= 0.940$

Step 1 : Compute the entropy for Color : (Red, green, yellow)

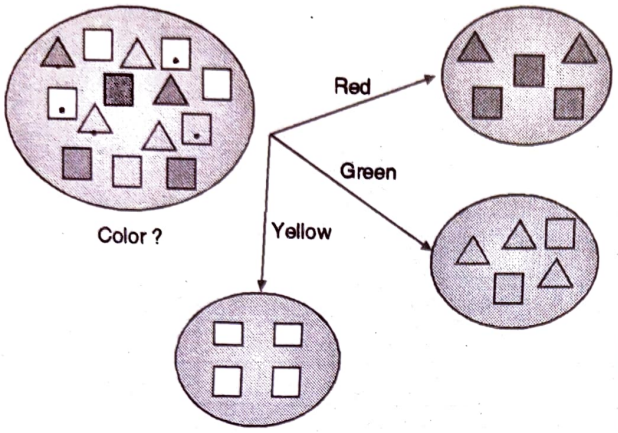


Fig. P. 5.2.5(a)

For color = Red,

p_i = with "square" class = 3 and n_i = with "triangle" class = 2

Therefore, $I(p_i, n_i) = I(3,2) = 0.971$

Similarly for different Color values, $I(p_i, n_i)$ is calculated as given below :

Color	p_i	n_i	$I(p_i, n_i)$
Red	3	2	0.971
Green	2	3	0.971
Yellow	4	0	0

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$E(Color) = 5/14 * I(3,2)+5/14*I(2,3)+4/14*I(4,0)$

$= 0.694$

Note : S is the total training set.

Hence $Gain(S, color) = I(p, n) - E(Color)$

$= 0.940 - 0.694 = 0.246$

Step 2 : Compute the entropy for outline : (Dashed, solid)

Similarly for different outline values, $I(p_i, n_i)$ is calculated as given below :

Outline	p_i	n_i	$I(p_i, n_i)$
Dashed	3	4	0.985
Solid	6	1	0.621

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$E(Outline) = 7/14 * I(3,4)+7/14*I(6,1)=0.803$

Note : S is the total training set.

Hence

$Gain(S, Outline) = I(p, n) - E(Outline)$

$= 0.940 - 0.803 = 0.137$

Step 3 : Compute the entropy for dot : (no, yes)

Similarly for different dot values, $I(p_i, n_i)$ is calculated as given below :

Outline	p_i	n_i	$I(p_i, n_i)$
No	6	2	0.811
Yes	3	3	1

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$E(Dot) = 8/14 * I(6,2)+6/14*I(3,3)$

$= 0.892$

Note : S is the total training set.

Hence $Gain(S, dot) = I(p, n) - E(dot)$

$= 0.940 - 0.892 = 0.048$

Therefore, $Gain(S, color) = 0.246$

$Gain(S, outline) = 0.137$

$Gain(S, dot) = 0.048$

As color has highest gain,it should be the root node.

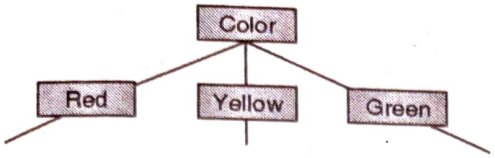


Fig. P. 5.2.5(b)

Step 4 : As attribute color is at the root, we have to decide on the remaining two attribute for red branch node.

Consider color =red and count the number of tuples from the original given training set

	Attribute			Shape
	Color	Outline	Dot	
1.	Red	Dashed	No	Square
2.	Red	Solid	No	Square
3.	Red	Solid	Yes	Triangle
4.	Red	Solid	No	Square
5.	Red	Dashed	Yes	Triangle

Note : Refer above table :

Total number of tuple with "square" class = 3 and total number of No tuple with "triangle" class = 2

$I(p, n) = I(3,2)$

$= -(3/5)\log_2(3/5) - (2/5)\log_2(2/5) = 0.971$

Compute the entropy for outline : (Dashed, solid)

Similarly for different outline values, $I(p_i, n_i)$ is calculated as given below.

Outline	p_i	n_i	$I(p_i, n_i)$
Dashed	1	1	1
Solid	2	1	0.918

Calculate Entropy using the values from the above table and the formula given as :

$$E(A) = \sum_{i=1}^{v} \frac{p_1 + n_1}{p + n} I(p_1, n_1)$$

$$E(\text{Outline}) = 2/5 * I(1,1) + 3/5 * I(2,1) = 0.951$$

Hence

$$\text{Gain}(S_{red}, \text{Outline}) = I(p, n) - E(\text{Outline})$$

$$= 0.971 - 0.951 = 0.02$$

Compute the entropy for Dot : (no,yes)

Similarly for different Dot values, $I(p_i, n_i)$ is calculated as given below :

Outline	p_i	n_i	$I(p_i, n_i)$
No	3	0	0
Yes	0	2	0

Calculate entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_1 + n_1}{p + n} I(p_1, n_1)$$

$$E(\text{Dot}) = 3/5 * I(3,0) + 2/5 * I(0,2) = 0$$

Hence $\text{Gain}(S_{red}, \text{Dot}) = I(p, n) - E(\text{Dot})$

$$= 0.971 - 0 = 0.971$$

Dot has the highest gain; therefore, it is below Color = "Red"

Check the tuples with Dot = "yes" from sample S_{red} , it has class triangle.

Check the tuples with Dot = "no" from sample S_{red} , it has class square

So the partial tree for red color sample is as given in Fig. P. 5.2.5(c).

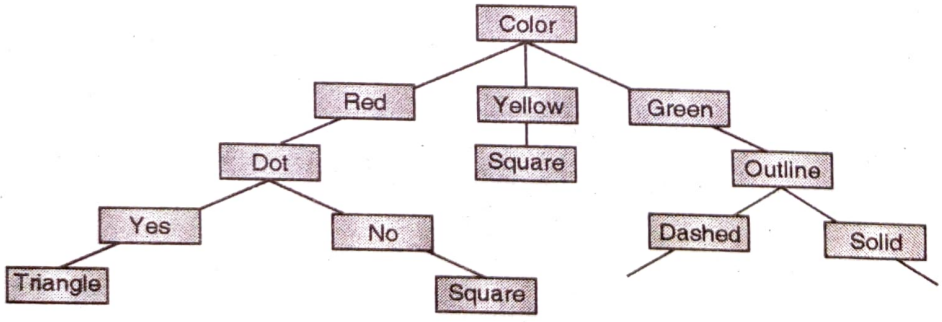


Fig. P. 5.2.5(c)

Step 5 : **Consider Color = Yellow and count the number of tuples from the original given training set.**

	Attribute			Shape
	Color	Outline	Dot	
1.	Yellow	Dashed	No	Square
2.	Yellow	Solid	Yes	Square
3.	Yellow	Dashed	Yes	Square
4.	Yellow	Solid	No	Square

As all the tuples belong to yellow color have class label square, so directly assign a class label below the node color = "yellow" as square.

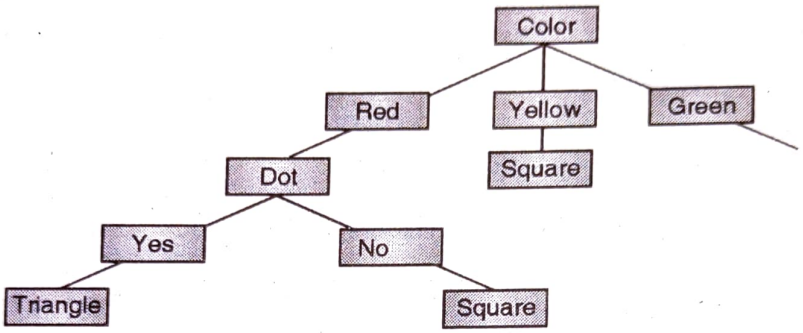


Fig. P. 4.2.5(d)

Step 6 : **Consider Color = green and count the number of tuples from the original given training set, as only attribute outline has left, it becomes a node below color ="green".**

	Attribute			Shape
	Color	Outline	Dot	
1.	green	dashed	no	Triangle
2.	green	dashed	Yes	triangle
3.	green	solid	No	square
4.	green	dashed	no	triangle
5.	green	solid	yes	Square

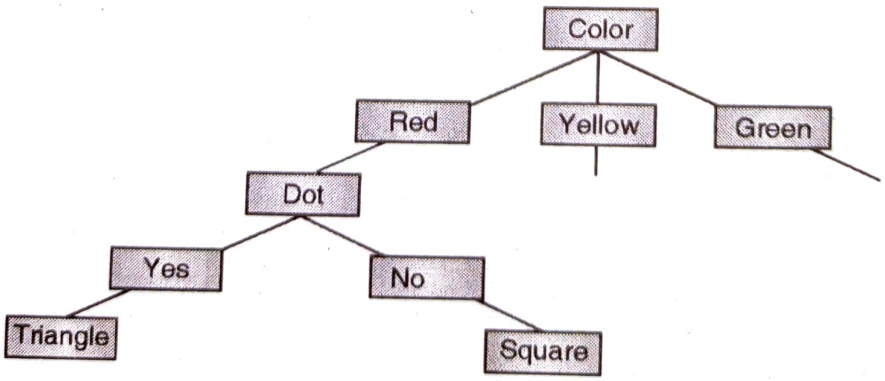


Fig. P. 5.2.5(e)

Check the tuples with Outline = "dashed" from sample S_{green} , it has class triangle.

Check the tuples with outline = "solid" from sample S_{green} , it has class square.

Therefore the final Decision Tree is

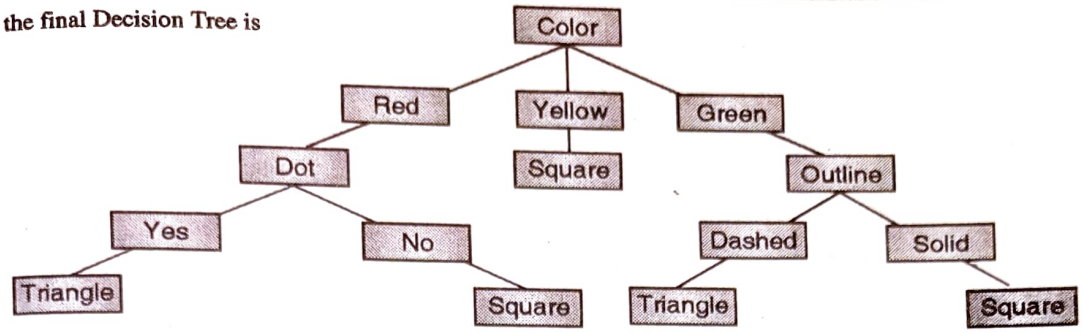


Fig. P. 5.2.5(f)

Ex. 5.2.6 : Apply statistical based algorithm to obtain the actual probabilities of each event to classify the new tuple as a tall. Use the following data

Person ID	Name	Gender	Height	Class
1	Kristina	Female	1.6m	Short
2	Jim	Male	2m	Tall
3	Maggie	Female	1.9m	Medium
4	Martha	Female	1.85m	Medium
5	John	Male	2.8m	Tall
6	Bob	Male	1.7m	Short
7	Clinton	Male	1.8m	Medium
8	Nyssa	Female	1.6m	Short
9	Kathy	Female	1.65m	Short

Soln. :

P(Short) = 4/9 P(Medium) = 3/9 P(Tall) = 2/9

Divide the height attribute into six ranges as given below :

[0,1.6] ,[1.6,1.7],[1.7,1.8],[1.8,1.9],[1.9,2.0],[2.0,infinity]

Gender attribute has only two values Male and Female.

Total Number of short person = 4, Medium = 3, Tall = 2

Prepare the probability table as given below :

Attribute	Value	Count			Probabilities		
		Short	Medium	Tall	Short	Medium	Tall
Gender	Male	1	1	2	1 / 4	1/3	2/2
	Female	3	2	0	3 /4	2/3	0/2
Height	[0,1.6]	2	0	0	2 /4	0	0
	[1.6,1.7]	2	0	0	2/4	0	0
	[1.7,1.8]	0	1	0	0	1/3	0
	[1.8,1.9]	0	2	0	0	2 /3	0
	[1.9,2.0]	0	0	1	0	0	1 /2
	[2.0,infinity]	0	0	1	0	0	1 /2

Use above values to classify new tuple as a tall :

Consider new tuple as t = {Adam, M, 1.95m}

$$P(t|Short) = 1/4 * 0 = 0$$

$$P(t|Medium) = 1/3 * 0 = 0$$

$$P(t|Tall) = 2/2 * 1/2 = 0.5$$

Therefore likelihood of being short

$$= p(t|short) * P(short) = 0 * 4/9 = 0$$

Likelihood of being Medium = 0 * 3/9 = 0

Likelihood of being Tall = 2/9 * 1/2 = 0.11

Then estimate P(t) by adding individual likelihood values since t will be either short or medium or tall.

$$P(t) = 0 + 0 + 0.11 = 0.11$$

Finally Actually probabilities of each event

$$P(Short | t) = (P(t|short) * p(short)) / P(t)$$

$$= (0 * 4/9)/0.11 = 0$$

Similarly $P(Medium|t) = (0 * 3/9)/0.11 = 0$

$$P(Tall|t) = (0.5 * 2/9)/0.11 = 1$$

New tuple is a Tall as it has the highest probability.

Ex. 5.2.7 : The training data is supposed to be a part of a transportation study regarding mode choice to select Bus, Car or Train among commuters along a major route in a city.

Attributes				Classes
Gender	Car ownership	Travel cost ($)/km	Income level	Transportation mode
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train

Attributes				Classes
Gender	Car ownership	Travel cost ($)/km	Income level	Transportation mode
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Male	0	Standard	Medium	Train
Female	1	Standard	Medium	Train
Female	1	Expensive	High	Car
Male	2	Expensive	Medium	Car
Female	2	Expensive	High	Car

Suppose we have new unseen records of a person from the same location where the data sample was taken. The following data are called *test data* (in contrast to *training data*) because we would like to examine the classes of these data.

Person name	Gender	Car ownership	Travel cost ($)/km	Income level	Transportation mode
Alex	Male	1	Standard	High	?
Buddy	Male	0	Cheap	Medium	?
Cherry	Female	1	Cheap	High	?

The question is what transportation mode would Alex, Buddy and Cheery use?

Soln. :

Class P : Transportation mode= "Bus"

Class Q : Transportation mode= "Train"

Class N : Transportation mode= "Car"

Total no. of records: 10

No. of records with "Bus" class = 4

No. if records with "Train" class = 3

No. of records with "Car" class = 3

So,

Information Gain = I(p,q,n)

$$= -(p/(p+q+n))\log_2(p/(p+q+n)) - (q/(p+q+n))\log_2(q/(p+q+n)) - (n/(p+q+n))\log_2(n/(p+q+n))$$

$$I(p,q,n) = I(4,3,3) = -(0.4)(-1.322) - (0.3)(-1.737) - (0.3)(-1.737)$$

$$I(4,3,3) = 0.5288 + 0.5211 + 0.5211$$

$$\mathbf{I(4,3,3) = 1.571}$$

Step 1 : Compute the entropy of gender : (Male, Female)

For gender = Male $p_i = 3$

$q_i = 1$ $n_i = 1$

Therefore ,

$$I(p_i, q_i, n_i) = I(3,1,1) = -(3/5)\log_2(3/5) - (1/5)\log_2(1/5) - (1/5)\log_2(1/5)$$

$$= 1.371$$

Similarly for different gender $I(p_i, q_i, n_i)$ is calculated as given below :

Gender	p_i	q_i	n_i	$I(p_i, n_i)$
Male	3	1	1	1.371
Female	1	2	2	1.522

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

$$E(\text{gender}) = (5/10) * I(3,1,1) + (5/10) * I(1,2,2)$$

$$= 1.447$$

Note : S is the total training set.

Hence

$$\text{Gain}(S, \text{gender}) = I(p, q, n) - E(\text{gender}) = 1.571 - 1.447 = 0.124$$

Similarly,

Gain (S, Car Ownership) = 0.535

Gain (S, Travel Cost ($)/Km) = 1.21

Gain (S, Income Level) = 0.696

Travel cost ($)/Km attribute has the highest gain, therefore it is used as the decision attribute in the root node.

Since travel cost ($)/Km has three possible values, the root node has three branches (Cheap, Standard, Expensive).

Since for all the attributes of Travel Cost ($)/Km = expensive, Transportation mode = "Car" , so assign class 'Car' to expensive.

Since for all the attributes of Travel Cost ($)/Km = Standard, Transportation mode = "Train", so assign class 'Train' to standard.

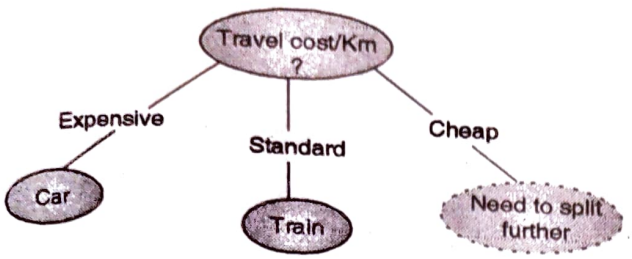


Fig. P. 5.2.7(a)

Consider travel cost ($)/Km = Cheap and count the number of tuples from the original given training set

S_{cheap} = 5

Attributes				Classes
Gender	Car ownership	Travel cost ($)/km	Income level	Transportation mode
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus

Note : Refer above table :
Total No. of Bus tuple = 4 and total no of Train tuple = 1, and total no of Car tuple = 0

$$I(p, q, n) = I(4, 1, 0) = -(4/5)\log_2(4/5) - (1/5)\log_2(1/5) - (0/5)\log_2(0/5)$$

$$= 0.722$$

(I) Compute the entropy for gender : (Male, female)

For gender = Male,

p_i = with "Bus" class = 3, q_i = with "Train" class = 0 and n_i with "car" class= 0

Therefore ,

$$I(p_i, q_i, n_i) = I(3,0,0)$$

$$= -(3/3)\log_2(3/3) - (0/3)\log_2(0/3) - (0/3)\log_2(0/3)$$

$$= 0$$

Similarly for different genders $I(p_i, q_i)$ is calculated as given below :

Gender	p_i	q_i	n_i	$I(p_i, q_i, n_i)$
Male	3	0	0	0
Female	1	1	0	1

Calculate entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$$E(\text{gender}) = 3/5 * I(3, 0,0) + 2/5 * I(1, 1,0)$$

$$= 0.4$$

Note : S_{cheap} is the total training set.

Hence

$$\text{Gain}(S_{cheap}, \text{gender}) = I(p,q, n) - E(\text{gender})$$

$$= 0.722 - 0.4 = 0.322$$

(II) Compute the entropy for Car ownership: (0, 1, 2)

For Car ownership = 0,

p_i = with "bus" class = 2 ,q_i = with "train" class = 0 and n_i with "car" class = 0

Therefore,

$$I(p_i, q_i, n_i) = I(2,0,0)$$

$$= -(2/2)\log_2(2/2) - (0/2)\log_2(0/2) - (0/2)\log_2(0/2) = 0.$$

Similarly for different outlook ranges $I(p_i, q_i, n_i)$ is calculated as given below :

Car ownership	p_i	q_i	n_i	$I(p_i, q_i, n_i)$
0	2	0	0	0
1	2	1	0	0.918
2	0	0	0	0

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$$E(\text{Car ownership}) = 2/5 * I(2, 0,0) + 3/5 * I(2, 1,0) + 0/5 * I(0,0,0)$$

$$= 0.551$$

Note : S_{cheap} is the total training set.

Hence

$$Gain(S_{cheap}, car\ ownership) = I(p,q,n) - E(car\ ownership)$$

$$= 0.722 - 0.551 = 0.171$$

(iii) Compute the entropy for Income level : (Low, medium, high)

For income level = Low,

p_i = with "Bus" class = 2 ,q_i = with "Train" class = 0 and $n_i$with "car" class = 0

Therefore,

$$I(p_i, q_i, n_i) = I(2,0,0)$$

$$= -(2/2)\log_2(2/2) - (0/2)\log_2(0/2) - (0/2)\log_2(0/2) = 0$$

Similarly for different outlook ranges $I(p_i, q_i, n_i)$ is calculated as given below :

Income level	p_i	q_i	n_i	$I(p_i, q_i, n_i)$
Low	2	0	0	0
Medium	2	1	0	0.918
High	0	0	0	0

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

$$E(\text{Income Level}) = 2/5 * I(2,0,0) + 3/5 * I(2,1,0) + 0/5 * I(0,0,0)$$

$$= 0.551$$

Note : S_{cheap} is the total training set.

Hence

$$Gain(S_{cheap}, \text{Income level}) = I(p,q,n) - E(\text{Income level})$$

$$= 0.722 - 0.551 = 0.171$$

Therefore, since gender has the highest gain, it comes below cheap.

For all gender = Male, Transportation mode= bus

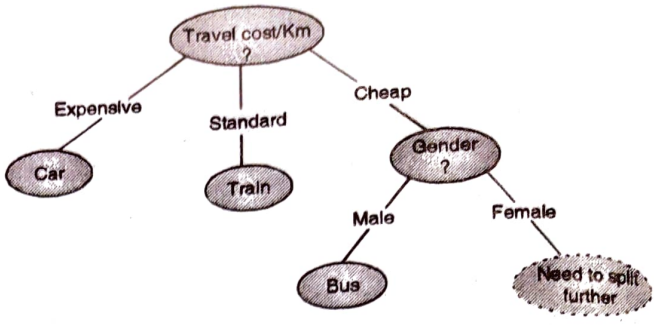


Fig. P. 5.2.7(b)

$S_{female} = 2$

Gender	Car ownership	Income level	Transportation mode
Female	1	Medium	Train
Female	0	Low	Bus

Suppose we select attribute car ownership, we can update our decision tree into the final version.

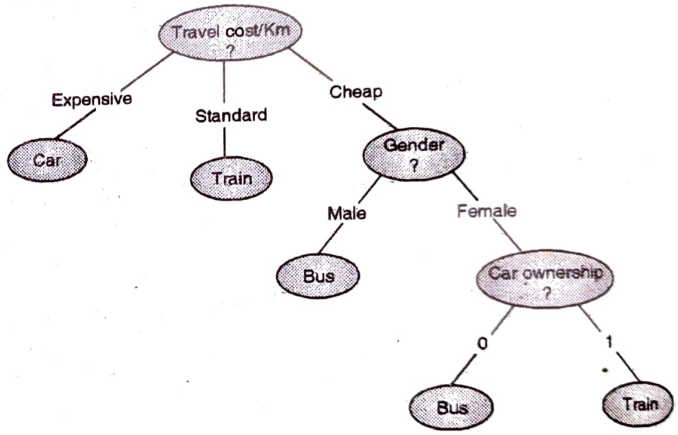


Fig. P. 5.2.7(c)

Ex. 5.2.8 : The table below shows a sample dataset of whether a customer responds to a survey or not. **"Outcome"** is the class label. Construct a Decision Tree Classifier for the dataset. For a new example **(Rural, semidetached, low, No)**, what will be the predicted class label?

District	House type	Income	Previous customer	Outcome
Suburban	Detached	High	No	Nothing
Suburban	Detached	High	Yes	Nothing
Rural	Detached	High	No	Responded
Urban	Semi-detached	High	No	Responded

District	House type	Income	Previous customer	Outcome
Urban	Semi-detached	Low	No	Responded
Urban	Semi-detached	Low	Yes	Nothing
Rural	Semi-detached	Low	Yes	Responded
Suburban	Terrace	High	No	Nothing
Suburban	Semi-detached	Low	No	Responded
Urban	Terrace	Low	No	Responded
Suburban	Terrace	Low	Yes	Responded
Rural	Terrace	High	Yes	Responded
Rural	Detached	Low	No	Responded
Urban	Terrace	High	Yes	Nothing

Soln. :

Sr. No.	District	House_Type	Income	Previous_Customer	Outcome
1	Suburban	Detached	High	No	Nothing
2	Suburban	Detached	High	Yes	Nothing
3	Rural	Detached	High	No	Responded
4	Urban	Terrace	High	No	Responded
5	Urban	Semi-detached	Low	No	Responded
6	Urban	Semi-detached	Low	Yes	Nothing
7	Rural	Semi-detached	Low	Yes	Responded
8	Suburban	Terrace	High	No	Nothing
9	Suburban	Semi-detached	Low	No	Responded
10	Urban	Terrace	Low	No	Responded
11	Suburban	Terrace	Low	Yes	Responded
12	Rural	Terrace	High	Yes	Responded
13	Rural	Detached	Low	No	Responded
14	Urban	Terrace	High	Yes	Nothing

Class P : Outcome = "Responded"

Class N : Outcome = "Nothing"

Total number of records 14.

Count the number of records with "Responded" class and "Nothing" class.

So number of records with "Responded" class = 9 and "Nothing" class = 5

So Information gain = I (p, n)

$$= -\frac{p}{p+n}\log 2\frac{p}{p+n} - \frac{n}{p+n}\log 2\frac{n}{p+n}$$

$$I(p, n) = I(9, 5)$$

$$= -(9/14)\log 2(9/14) - (5/14)\log 2(5/14)$$

$$= (-0.643) * (-0.637) + (-0.357) * (-1.485)$$

$$I(p, n) = 0.409 + 0.530 = 0.940$$

Step 1 : Compute the entropy for District : (Suburban, Rural, Urban)

For District = Suburban,

Pi = with "Responded" class = 2 and ni

= with "Nothing" class = 3

Therefore, I(pi , ni) = I(2,3)

$$= -(2/5)\log 2(2/5) - (3/5)\log 2(3/5) = 0.971.$$

Similarly for different District ranges I(pi , ni) is calculated as given below :

District	pi	ni	I(pi, ni)
Suburban	2	3	0.971
Rural	4	0	0
Urban	3	2	0.971

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(pi, ni)$$

$$E(\text{District}) = \frac{5}{14} I(2, 3) + \frac{4}{14} I(4, 0) + \frac{5}{14} I(3, 2) = 0.694$$

T is the total training set.

Hence Gain(T, District) = I (p, n) – E (District)

= 0.940 – 0.694 = 0.246

Similarly,Gain (T, House_Type) = 0.029

Gain (T, Income) = 0.151

Gain (T, Previous_Customer) = 0.048

District shows the highest gain, so it is used as the decision attribute in the root.

As District has only values "Suburban, Rural, Urban", the root node has three branches

Step 2 :

As attribute District at root, we have to decide on the remaining three attribute for Suburban branch

Consider District = Suburban and count the number of tuples from the original given training set

SSuburban = {1, 2, 8, 9, 11} = 5

Sr. No.	District	House_Type	Income	Previous_Customer	Outcome
1	Suburban	Detached	High	No	Nothing
2	Suburban	Detached	High	Yes	Nothing
8	Suburban	Terrace	High	No	Nothing
9	Suburban	Semi-detached	Low	No	Responded
11	Suburban	Terrace	Low	Yes	Responded

Total number of Responded tuple = 2 and total number of Nothing tuple = 3

$I(p, n) = I(2, 3) = -(2/5)\log2(2/5) - (3/5)\log2(3/5) = 0.971$

(i) Compute the entropy for House_Type : (Detached, Terrace, Semi-detached)

For House_Type = Detached,

pi = with "Responded" class = 0 and ni = with "Nothing" class = 2

Therefore , I(pi , ni) = I(0,2)

$= -(0/2)\log2(0/2) - (2/2)\log2(2/2)$

$= 0$

Similarly for different District ranges I(pi , ni) is calculated as given below :

House_Type	pi	ni	I(pi, ni)
Detached	0	2	0
Terrace	1	1	1
Semi-detached	1	0	0

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(pi, ni)$$

$E(House\_Type) = 2/5 * I(0, 2) + 2/5 * I(1, 1) + 1/5 * I(1, 0) = 0.4$

Note : TSuburban is the total training set.

Hence,

Gain(TSuburban, House_Type) = I (p, n) – E (House_Type)

$= 0.971 - 0.4 = 0.571$

(ii) Compute the entropy for Income : (High, Low)

For Income = High,

pi = with "Responded" class = 0 and ni = with "Nothing" class = 3

Therefore ,

$I(pi, ni) = I(0,3) = -(0/3)\log2(0/3) - (3/3)\log2(3/3) = 0$

Similarly for different District ranges I(pi , ni) is calculated as given below :

Income	pi	ni	I(pi, ni)
High	0	3	0
Low	2	0	0

Calculate Entropy using the values from the above table and the formula given below

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(pi, ni)$$

$E(Income) = 3/5 * I(0, 3) + 2/5 * I(2, 0) = 0$

Note : TSuburban is the total training set.

Hence,

Gain(TSuburban, Income) = I (p, n) – E (Income)

$= 0.971 - 0 = 0.971$

(iii) Compute the entropy for Previous_Customer : (No, Yes)

For Previous_Customer = No,

pi = with "Responded" class = 1 and ni = with "Nothing" class = 2

Therefore,

$I(pi , ni) = I(1,2) = -(1/3)\log2(1/3) - (2/3)\log2(2/3) = 0.918$

Similarly for different District ranges I(pi , ni) is calculated as given below :

Previous_Customer	pi	ni	I(pi, ni)
No	1	2	0.918
Yes	1	1	1

Calculate Entropy using the values from the above table and the formula given as:

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(p1, n1)$$

E (Previous_Customer) = 3/5 * I (1, 2) + 2/5* I (1, 1) = 0.951

Note : TSuburban is the total training set.

HenceGain(TSuburban, Previous_Customer)

= I (p, n) – E (Previous_Customer)

= 0.971 – 0.951 = 0.02

Therefore,

Gain(TSuburban, Income) = 0.970

Gain(TSuburban, House_Type) = 0.570

Gain(TSuburban, Previous_Customer) = 0.02

Income has the highest gain; therefore, it is below District = "Suburban".

Step 3 :

Consider only House_Type and Previous_Customer for District = Rural and count the number of tuples from the original given tUrbaning set

TRural = {3,7,12,13}

= 4

Sr. No.	District	House_Type	Income	Previous_Customer	Outcome
3	Rural	Detached	High	No	Responded
7	Rural	Semi-detached	Low	Yes	Responded
12	Rural	Terrace	High	Yes	Responded
13	Rural	Detached	Low	No	Responded

Since for the attributes House_Type and Previous_Customer, Outcome = Responded, so assign class 'Responded' to Rural.

Step 4 :

Consider House_Type and Previous_Customer for District = Urban and count the number of tuplesfrom the original given training set

TUrban = {4, 5, 6, 10, 14}

= 5

Sr. No.	District	House_Type	Income	Previous_Customer	Outcome
4	Urban	Terrace	High	No	Responded
5	Urban	Semi-detached	Low	No	Responded
6	Urban	Semi-detached	Low	Yes	Nothing
10	Urban	Terrace	Low	No	Responded
14	Urban	Terrace	High	Yes	Nothing

Consider the above table as the new training set and calculate the Gain for House_Type and Previous_Customer.

Class P : Outcome = "Responded"

Class N : Outcome = "Nothing"

Total number of records 5

Count the number of records with "Responded" class and "Nothing" class.

So number of records with "Responded" class = 3 and "Nothing" class = 2

So Information gain =I (p, n)

$$= -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

I (p, n) =I (3, 2)

=– (3/5) log2 (3/5) – (2/5)log2(2/5)

=0.970

(iv) Compute the entropy for Previous_Customer

For Previous_Customer = No

pi = with "Responded" class = 3 and ni = with "Nothing" class = 0

Therefore, I(pi , ni) = I(3,0) = 0.

For Previous_Customer = Yes

pi = with "Responded" class = 0 and ni = with "Nothing" class = 2

Therefore, I(pi, ni) = I(0, 2) = 0

Similarly for different District ranges I(pi , ni) is calculated as given below :

Previous_Customer	pi	ni	I(pi, ni)
No	3	0	0
Yes	0	2	0

Calculate entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(pi, ni)$$

$$E(\text{Previous\_Customer}) = \frac{3}{5} I(3, 0) + \frac{2}{5} I(0, 2) = 0$$

Hence

Gain(TUrban, Previous_Customer) = I (p, n) – E (Previous_Customer)

= 0.970 – 0 = 0.970

(v) Compute the entropy for House_Type : (Detached, Terrace , Semi-detached)

For House_Type = Detached,

pi = with "Responded" class = 0 and ni = with "Nothing" class = 0

Therefore, I(pi , ni) = I(0,0) = 0

Similarly for different District ranges I(pi , ni) is calculated as given below :

House_Type	pi	ni	I(pi, ni)
Detached	0	0	0
Terrace	2	1	0.918
Semi-detached	1	1	1

Calculate Entropy using the values from the above table and the formula given below :

$$E(A) = \sum_{i=1}^{v} \frac{pi + ni}{p + n} I(pi, ni)$$

E (House_Type)= 0/5 * I (0, 0) + 3/5* I(2, 1) + 2/5 *I(1, 1)

= 0.951

Note : TUrban is the total training set.

Hence

Gain(TUrban, House_Type) = I (p, n) – E (House_Type)

= 0.970 – 0.951 = 0.019

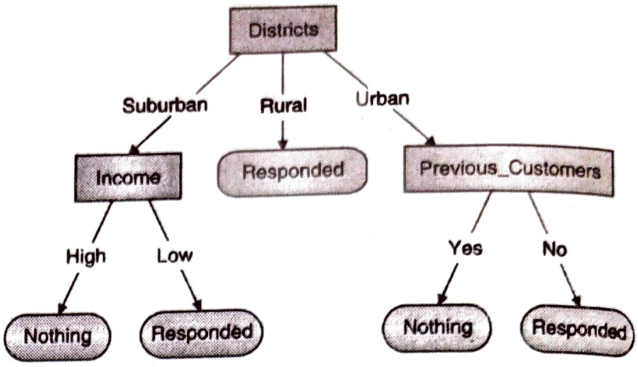
Therefore,

Gain(TUrban, House_Type) = 0.019

Gain(TUrban, Previous_Customer) = 0.970

Previous_Customer has the highest gain; therefore, it is below District = "Urban".

Therefore the final decision tree is :



The decision tree can also be expressed in rule format :

IF District = Suburban AND Income

= high THEN Outcome = Nothing

IF District = Suburban AND Income

= Low THEN Outcome = Responded

IF District = Rural THEN Outcome = Responded

IF District = Urban AND Previous_Customer

= Yes THEN Outcome = Nothing

IF District = Urban AND Previous_Customer

= No THEN Outcome = Responded

Syllabus Topic : Rule-Based Classification : using IF-THEN Rules for Classification

5.3 Rule-Based Classification : using IF-THEN Rules for Classification

A set of IT-THEN rules is used for classification in Rule Based Classification. It classifies the record based on collection of IF-THEN rules. The syntax for rules is

"IF condition THEN conclusion"

Example

- If Rule is X → Y where X is condition.
- X is conjunctions of attributes and Y the class label of the rule
- LHS of rule is rule antecedent and RHS is consequent.

Example

- (Refund=No) ∧(Status=Married) → (Cheat=N0)
- (Status = Student) ∧ (age< 30) → (buys_coomputer = YES)

5.3.1 Rule Coverage and Accuracy

- Coverage of a rule: Percentage of tuples that satisfies the antecedent of a rule.
- Accuracy of a rule: Percentage of tuples that satisfy both the antecedent and consequent of a rule. i.e. percentage of tuples which are correctly classified.

☞ **Formulae**

Coverage (Rule) = Number of tuples covered by Rule / number of tuples in dataset D

Accuracy (Rule) = Number of tuples correctly classified by Rule / Number of tuples covered by Rule

Example

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

(Status=Single) → No

Coverage = 4/10 = 0.4 = 40%

Accuracy = 2/4 = 0.5 = 50%

5.3.2 Characteristics of Rule-Based Classifier

1. **Mutually exclusive rules :** Every record of dataset is covered by at most one rule of classifier and the rules are independent of each other
2. **Exhaustive rules :** Classifier generates rules for every possible combination of attribute values and each record is covered by at least one rule.

Example

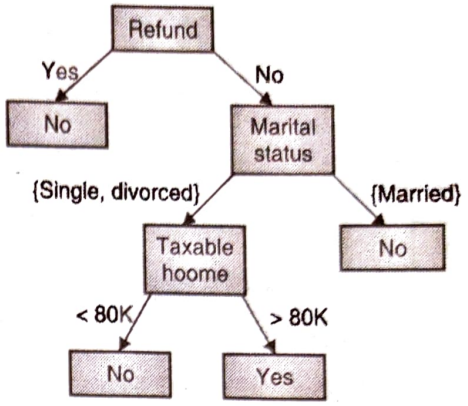


Fig. 5.3.1

☞ **Classification rules**

(Refund=Yes)==> No

(Refund = No, Marital Status = {Single, Divorced}, Taxable Income<80K) ==> No

(Refund = No, Marital Status = {Single, Divorced}, Taxable Income>80K)==> Yes

(Refund=No, Marital Status={Married}) ==> No

In the above example

- Rules are mutually exclusive and exhaustive.
- Rule set contains as much information as the tree.

☞ **Extract the rules from decision tree**

- If more than one rule is triggered then it need **conflict resolution.**
- Based on size, it has to order. So give highest priority to that triggering rule which has the maximum attribute test.
- Make the decision list based on the ordering of the rules. Rules are organized based on some measure of rule quality or by taking expert opinion.
- Once the decision tree is created, list the rules which are easy to understand than big and complex tree.
- For every path of the tree, create a rule from root node to a leaf node.
- The last node or leaf node gives the class label.

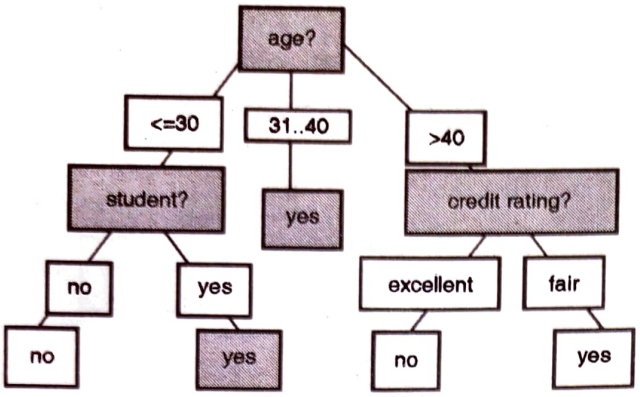


Fig. 5.3.2 : Decision Tree for "Buys_Computer"

Rule extraction from above buys_computerdecision-tree

1. IF age = "<=30" AND student = "no" THEN buys_computer = "no"
2. IF age = "<=30" AND student = "yes" THEN buys_computer = "yes"
3. IF age = "31…40" THEN buys_computer = "yes"
4. IF age = ">40" AND credit_rating = "excellent" THEN buys_computer= "no"
5. IF age = ">40" AND credit_rating = "fair" THEN buys_computer = "yes"

Syllabus Topic : Rule Induction Using a Sequential Covering Algorithm

5.4 Rule Induction Using a Sequential Covering Algorithm

- Using sequential covering algorithm , IF-THEN rules can be extracted without generating a decision tree from training data.
- The rules can be learned one at a time.
- Sequential covering algorithms are the most widely used approach to mining disjunctive sets of classification rules
- Some of sequential covering algorithms are
  - AQ,
  - CN2
  - more recent RIPPER
- A basic sequential covering algorithm given by MichelineKamber is given below :

Algorithm

Sequential covering. Learn a set of IF-THEN rule for classification.

Input

D, a data set of class-labeled tuples;

Att_vals, the set of all attributes and their possible values.

Output

A set of IF-THEN rules.

Method

(1) Rule_set = { }; // initial set of rules leaned is empty
(2) for each class c do
(3) repeat
(4) Rule = Learn_One_Rule (D, Att_vals, c) ;
(5) remove tuples covered by Rule from D;
(6) Rule_set = Rule_set + Rule; // add new rule to rule set
(7) until terminating condition ;
(8) endfor
(9) return Rule_Set;

This algorithm basic functionality is :

1. Start from an empty rule
2. Grow a rule using the Learn-One-Rule function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) until stopping criterion is met

Syllabus Topic : Bayesian Belief Networks

5.5 Bayesian Belief Networks

- In Bayesian Belief network, conditional independence is defined between subsets of variables
- The network provides a graphical model of causal relationship on which learning can be performed

- A trained network can be used for classification.
- Bayesian Belief networks are also known as belief networks, bayesian networks and probabilistic networks.
- A belief network is defined by following two components.
  - A directed acyclic graph.
  - A set of conditional probability tables.

A directed acyclic graph

- Each node represents a random variable.
- Variables may be discrete or continuous valued.
- Variables may correspond to actual attributes or to hidden variables.
- Each arc represents a probabilistic dependence.
- if an arc is drawn from a node A to node B, then A is a parent or immediate predecessor of B and B is a descendent of A.
- Each variable is conditionally independent of its non descendants in the graph, given its parents.
- An example of a portion of Belief network is shown in Fig. 5.5.1 :

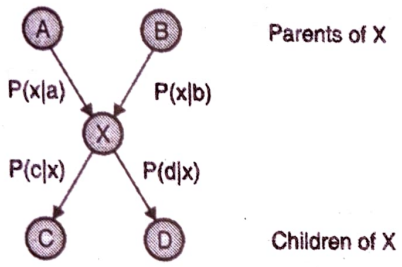


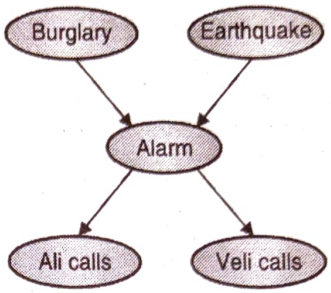
Fig. 5.5.1

- A portion of a belief network, consisting of a node X, having variable values (x1, x2, ...), its parents (A and B), and its children (C and D)

Example 1

- You have a new burglar alarm installed at home.
- It is fairly reliable at detecting burglary, but also sometimes responds to minor earthquakes.
- You have two neighbors, Ali and Veli, who promised to call you at work when they hear the alarm.
- Ali always calls when he hears the alarm, but sometimes confuses telephone ringing with the alarm and calls too.
- Veli likes loud music and sometimes misses the alarm.
- Given the evidence of who has or has not called, we would like to estimate the probability of a burglary.

The Bayesian network for the burglar alrarm example. Buglary (B) and earthquake (E) directly affect the probability of the alarm (A) going off, but whether or not Ali calls (AC) or Veli calls (VC) depends only on the alarm.



P(B = T)	P(B = F)
0.001	0.999

P(E = T)	P(E = F)
0.002	0.998

B	E	P(A = T)	P(A = F)
T	T	0.95	0.05
T	F	0.94	0.06
F	T	0.29	0.71
F	F	0.001	0.999

A	P(VC = T)	P(VC = F)
T	0.70	0.30
F	0.01	0.99

A	P(AC = T)	P(AC = F)
T	0.90	0.10
F	0.05	0.95

- What is the probability that the alarm has sounded but neither a burglary nor an earthquake has occurred, and both Ali and Veli Call ?

$$P(AC, VC, A, \neg B, \neg E)$$

$$= P(AC \backslash A)\, P(VC \backslash A)\, P(A \backslash \neg B, \neg E)\, P(\neg B)\, P(\neg E)$$

$$= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998$$

$$= 0.00062$$

(Capital letter represent variables having the value true and $\neg$ represents negation)

Example 2

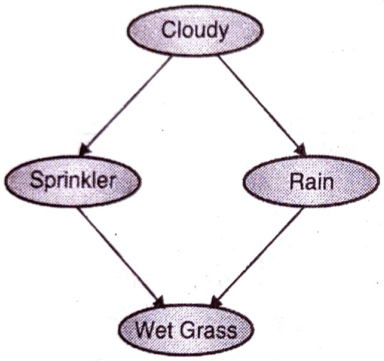
- Suppose we observe the fact that the grass is wet. There are two possible causes for this : either it rained, or the sprinkler was on. Which one is more likely ?

$$P(S|W) = \frac{P(S, W)}{P(W)} = \frac{0.2781}{0.6471} = 0.430$$

$$P(R|W) = \frac{P(R, W)}{P(W)} = \frac{0.4581}{0.6471} = 0.708$$

- We see that it is more likely that the grass is wet because it rained.

Another Bayesian network example. The event that the grass being wet (W = true) has two possible causes : either the water sprinkler was on (S = true) or it rained (R = true).



P(C = T)	P(C = F)
0.50	0.50

C	P(S = T)	P(S = F)
T	0.10	0.90
F	0.50	0.50

C	P(R = T)	P(R = F)
T	0.80	0.20
F	0.20	0.80

S	R	P(W = T)	P(W = F)
T	T	0.99	0.01
T	F	0.90	0.10
F	T	0.90	0.10
F	F	0.00	1.00

Applications of Bayesian Networks

- *Machine learning*
- *Statistics*
- *Computer vision*
- *Natural language Processing*
- Speech recognition
- Error-control codes
- Bioinformatics Medical diagnosis
- Weather forecasting

Syllabus Topic : Training Bayesian Belief Networks

5.6 Training Bayesian Belief Networks

- Training a network has a number of possible scenarios
  - The network topology (nodes and arcs) may be constructed by a human experts or inferred from the data
  - The network variables may be observable or hidden in all or some of the training tuples.
- Training the Network if the network topology is known and the variables are observable
  - Compute the CPT (Conditional Probability table) entries
- Training the Network if the network topology is known and some of the variables are hidden
  - Use gradient descent algorithm.
  - A gradient descent strategy performs a greedy hill climbing.
  - At each iteration, the weights are updated and will eventually converge to a local optimum solution.
  - Algorithms that follow this learning form are called Adaptive probabilistic networks.

Syllabus Topic : Classification Using Frequent Patterns : Associative Classification

5.7 Classification Using Frequent Patterns : Associative Classification

- Frequent patterns generated from association can be used for classification is called **associative classification**. Initially association rules are generated from frequent patterns and used for classification.

Steps involved in associative classification :

1. **Find** frequent itemsets, i.e commonly occurring attribute–valuepairs in the data.
2. **Generate association rules by analysing** frequent itemsets as per the class by considering class confidence and support criteria.
3. Organize the rules to form a rule-based classifier.

5.7.1 CBA

- One of the earliest and simplest algorithms for associative classification is **CBA** (Classification Based on Associations).

Steps In CBA

- **Mine** for CARs satisfying support and confidence thresholds
- **Sort** all CARs based on confidence
- **Classify** using the rule that satisfies the query and has the highest confidence

5.7.2 CMAR

Classification based on Multiple ARs (CMAR)

Steps In CMAR

- **Mine** for CARs satisfying support and confidence thresholds
- **Sort** all CARs based on confidence
- **Find** all CARs which satisfy the given query
- **Group** them based on their class label
- **Classify** the query to the class whose group of CARs has the maximum **weight**.

Syllabus Topic : Lazy Learners

5.8 Lazy Learners : (or Learning from your Neighbors)

- Classification methods can be classifies as Eager Learners and Lazy Learners.
- Eager learners are those classification techniques in which a given set of training tuples contructs a generalised model and then uses the same to classify a previously unseen tuple.
- A learned model as being ready and eager to classify an unseen tuple.
- Examples of Eager learners are Decision tree induction, Bayesian Classification, Rule based classification, classification by backpropogation, support vector machines and classification based on Association rule Mining.
- In lazy Learner approach, the learner waits until the last minute before doing model construction to classify a given test tuple.
- A lazy learner approach performs generalization only when its sees a test tuple. Until then it only stores training tuple or does very little processing.
- Lazy learners do very less amount of processing when training tuples are presented and does more amount of work when a classification or numeric prediction is to be done.
- Since Lazy learners stores training tuples or instances, it is also known as instance -based learners.

Disadvantages of lazy learners

- Lazy Learners are computationally expensive.
- Require efficient storage techniques
- Offers less insight into the data structures

Advantages of lazy learners

- Well suited to implementation on parallel
- Supports incremental learning
- Able to model complex decision spaces having hyperpolygonal shapes that may not be describable by any other learning algorithms
- Two examples of Lazy learners , K-nearest neighbors and case based reasoning

Syllabus Topic : K-Nearest-Neighbor Classifiers

5.8.1 K-Nearest-Neighbor Classifiers

→ **(SPPU - May 16, May 17)**

Q. Explain with suitable example.

(i) K-Nearest-Neighbor Classifier

May 16, May 17, 4 Marks

- K-Nearest Neighbors is used in the field of Pattern Recognition
- It learns by analogy, i.e. by comparing a given test tuple with training tuples that are similar to it
- The training tuples have n attributes, every tuple represents a point in n-dimensional space
- All training tuples are stored in an n-dimensional pattern space
- K-nearest neighbors searches the pattern space for the k training tuples that are closest to the unknown test tuple.
- The k training tuples are the k "nearest neighbors" of the unknow tuple.
- Closeness is defined using distance metrics such as Euclidean distance
- The Manhattan (city block) distance or other distance measurements, may also be used.
- **To find the** Euclidean Distance between two points or tuples, the formula is given below.

Let Y1 = $\{y_{11}, y_{12}, y_{13}, \ldots\ldots y_{1n}\}$

and Y2 = $\{y_{21}, y_{22}, y_{23}, \ldots\ldots y_{2n}\}$

distance(Y1,Y2) = $(y_{11} - y_{21})^2$

- KNN classifiers can be extremely slow when classifying test tuples O(n).
- By simple presorting and arranging the stored tuples into search tree, the number of comparisons can be reduced to O(logN).
- Example : if k = 5, it selects the 5 nearest neighbor as shown in Fig. 5.8.1.

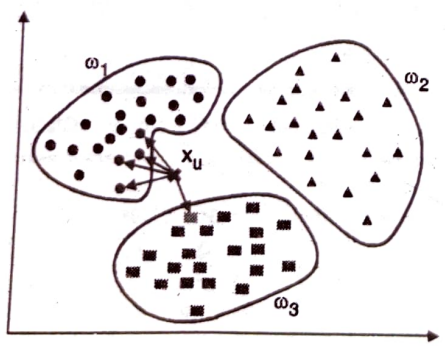


Fig. 5.8.1 : knn for k = 5

5.8.2 CBR (Case Based Reasoning)

- Case-based reasoning means using old experiences to understand and solve new problems. In case-based reasoning, a reasoner remembers a previous situation similar to the current one and uses that to solve the new problem.
- Case-based reasoning can mean adapting old solutions to meet new demands; using old cases to explain new situations; using old cases to critique new solutions; or reasoning from precedents to interpret a new situation (much like lawyers do) or create an equitable solution to a new problem (much like labor mediators do).
- To solve a current problem: the problem is matched against the cases in the case base, and similar cases are retrieved. The retrieved cases are used to suggest a solution which is reused and tested for success. If necessary, the solution is then revised. Finally, the current problem and the final solution are retained as part of a new case.
- All case-based reasoning methods have in common the following process :
  - retrieve the most similar case (or cases) comparing the case to the library of past cases;
  - reuse the retrieved case to try to solve the current problem;
  - revise and adapt the proposed solution if necessary;
  - retain the final solution as part of a new case.
- Retrieving a case starts with a (possibly partial) problem description and ends when a best matching case has been found. The subtasks involve :
  - identifying a set of relevant problem descriptors;
  - matching the case and returning a set of sufficiently similar cases (given a similarity threshold of some kind);
  - selecting the best case from the set of cases returned.
  - Some systems retrieve cases based largely on superficial syntactic similarities among problem descriptors, while advanced systems use semantic similarities.
- Reusing the retrieved case solution in the context of the new case focuses on: identifying the differences between the

retrieved and the current case; and identifying the part of a retrieved case which can be transferred to the new case. Generally, the solution of the retrieved case is transferred to the new case directly as its solution case.

- A CBR tool should support the four main processes of CBR: retrieval, reuse, revision and retention. A good tool should support a variety of retrieval mechanisms and allow them to be mixed when necessary. In addition, the tool should be able to handle large case libraries with retrieval time increasing linearly (at worst) with the number of cases.

Applications of CBR

Case based reasoning first appeared in commercial tools in the early 1990's and since then has been sued to create numerous applications in a wide range of domains :

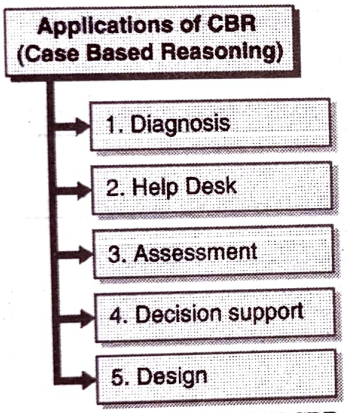


Fig. 5.8.2 : Applications of CBR

1. Diagnosis

Case-based diagnosis systems try to retrieve past cases whose symptom lists are similar in nature to that of the new case and suggest diagnoses based on the best matching retrieved cases. The majority of installed systems are of this type and there are many medical CBR diagnostic systems.

2. Help Desk

Case-based diagnostic systems are used in the customer service area dealing with handling problems with a product or service.

3. Assessment

Case-based systems are used to determine values for variables by comparing it to the known value of something similar. Assessment tasks are quite common in the finance and marketing domains.

4. Decision support

- In decision making, when faced with a complex problem, people often look for analogous problems for possible solutions. CBR systems have been developed to support in this problem retrieval process (often at the level of document retrieval) to find relevant similar problems. CBR is particularly good at querying structured, modular and non-homogeneous documents.

5. Design

- Systems to support human designers in architectural and industrial design have been developed. These systems assist the user in only one part of the design process, that of retrieving past cases, and would need to be combined with other forms of reasoning to support the full design process.

❑❑❑