G.S. Mandal's
Maharashtra Institute of Technology, Aurangabad
Department of Computer Science and Engineering

LAB MANUAL

CSE 403: Data Warehousing and Data Mining

(2019-20 Part 1)

Maharashtra Institute of Technology, Aurangabad
NH-211, MIT Campus, Satara Village Road, Aurangabad- 431 010 (M.S.); India.
Phone: (0240) 2375222; Fax: (0240) 2376618, E-mail: principalmitt@mit.asia
Website: www.btech.mit.asia

Department of Computer Science and Engineering

Vision

To develop the department as a center of excellence in the field of computer science and engineering by imparting knowledge & training to the students for meeting growing needs of the industry & society.

Mission

Providing quality education through a well-designed curriculum in tune with the challenging needs of software industry by providing state of the art facilities and to impart knowledge in the thrust areas of computer science and engineering.

Department of Computer Science and Engineering

Program Educational Objectives

PEO1: To prepare the students to achieve success in Computing Domain to create individual careers, innovations or to work as a key contributor to the private or Government sector and society.

PEO2: To develop the ability among the students to understand Computing and mathematical fundamentals and apply the principles of Computer Science for analyzing, designing and testing software for solving problems.

PEO3: To empower the students with ability to quickly reflect the changes in the new technologies in the area of computer software, hardware, networking and database management.

PEO4: To promote the students with awareness for lifelong learning, introduce them to professional practice, ethics and code of professionalism to remain continuous in their profession and leaders in technological society.

Program Specific Objectives

PSO1: Identify appropriate data structures and algorithms for a given contextual problem and develop programs to design and implement applications.

PSO3: Design and manage the large databases and develop their own databases to solve real world problems and to design, build, manage networks and apply wireless techniques in mobile based applications.

PSO3: Design a variety of computer-based components and systems using computer hardware, system software, systems integration process and use standard testing tools for assuring the software quality.

Program Outcomes

PO1: Apply knowledge of mathematics, science, and engineering fundamentals to solve problems in Computer science and Engineering.

PO2: Identify, formulate and analyze complex problems.

PO3: Design system components or processes to meet the desired needs within realistic constraints for the public health and safety, cultural, societal and environmental considerations.

PO4: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data for valid conclusions.

PO5: Select and apply modern engineering tools to solve the complex engineering problem.

PO6: Apply knowledge to assess contemporary issues.

PO7: Understand the impact of engineering solutions in a global, economic, environmental, and societal context.

PO8: Apply ethical principles and commit to professional ethics and responsibilities.

PO9: Work effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

PO10: Communicate effectively in both verbal and written form.

PO11: Demonstrate knowledge and apply engineering and management principles to manage projects and in multi-disciplinary environment.

PO12: To engage in life-long learning to adopt to the technological changes.

Department of Computer Science and Engineering

Course: CSE 403: Data warehousing and Data Mining

Course Outcomes:

After Completing the course students will be able to

CO1: Compare Informational and Operational Data

CO2: Apply pre-processing statistical methods for any given raw data

CO3: Modelling Multidimensional data using star and snowflake schemas

CO4: Discover interesting patterns from databases to analyse and extract pattern to make predictions of outcome

CO5:Select and apply appropriate data mining algorithm for Classification

CO6:Categorise and differentiate between situations for applying different clustering algorithms

Mapping

Experiment No.	Blooms Level	Mapping To CO	Mapping To PO
1	Apply	CO1	PO2
2	Apply	CO1	PO2
3	Apply	CO1	PO2
4	Apply	CO3	PO2
5	Apply	CO2	PO2
6	Apply	CO4	PO2
7	Apply	CO5	PO2
8	Apply	CO6	PO2
9	Apply	CO1	PO2

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH PART: 1 (2018-19)	**LAB: write lab no**	**SUBJECT: subject code-subject name**

Index

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**

Experiment No 1

Aim: OLAP operations

Objective:

• To learn fundamentals of data warehousing
• To learn concepts of dimensional modeling
• To learn OLAP operations

Theory:

Description:

OLAP is an acronym for On Line Analytical Processing. An OLAP system manages large amount of historical data, provides facilitie

OLAP Operations

Since OLAP servers are based on multidimensional view of data, we will discuss OLAP operations in multidimensional data.

Here is the list of OLAP operations:

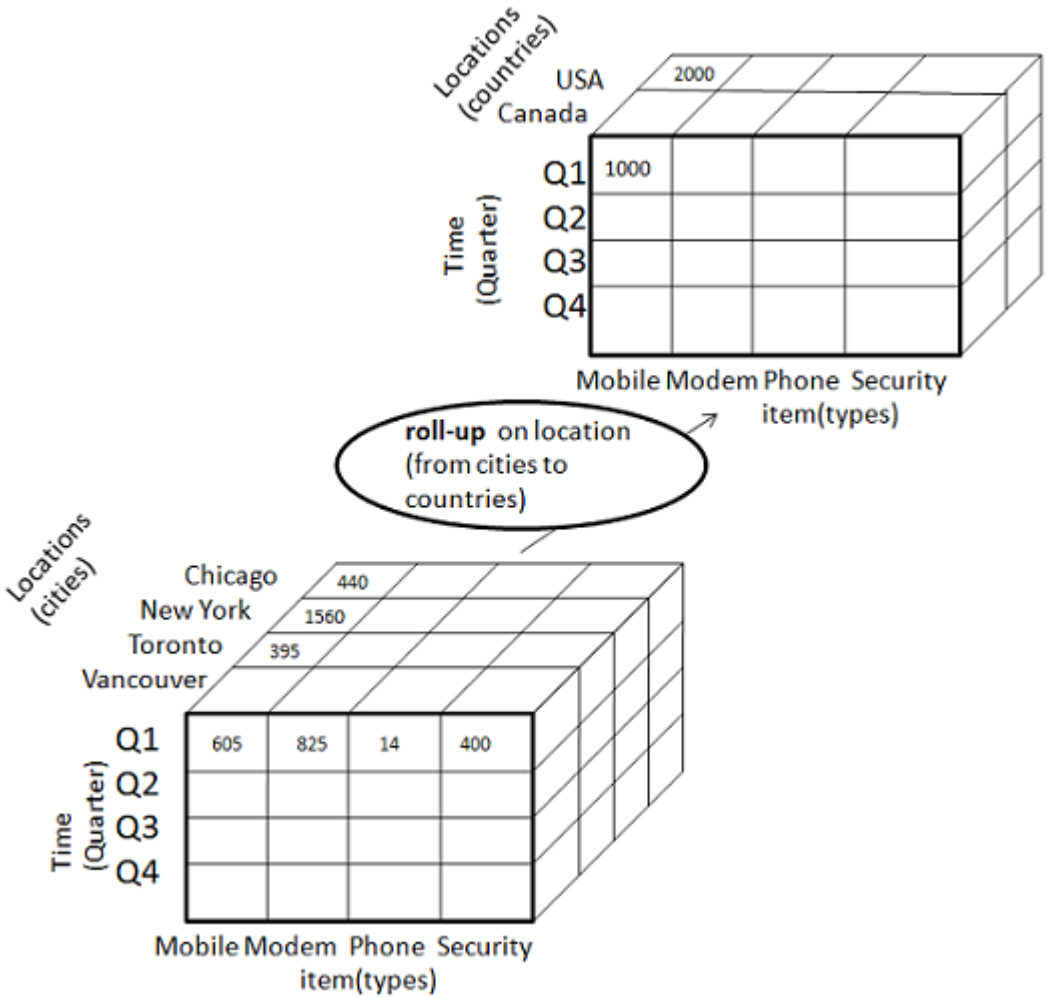
- Roll-up
- Drill-down
- Slice and dice
- Pivot (rotate)

Roll-up

Roll-up performs aggregation on a data cube in any of the following ways:

- By climbing up a concept hierarchy for a dimension
- By dimension reduction

The following diagram illustrates how roll-up works



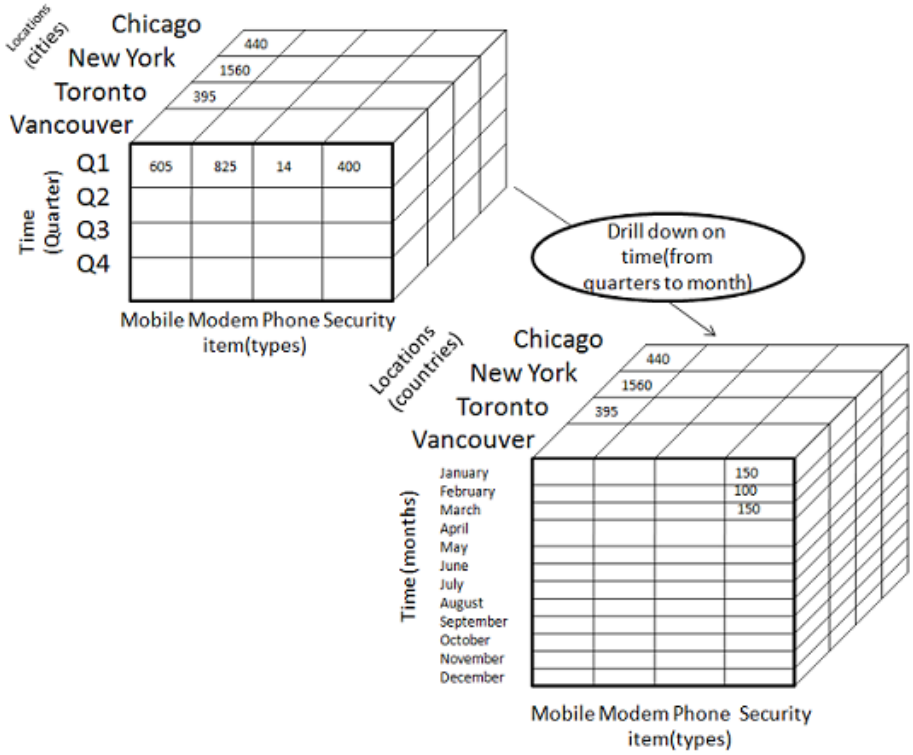
- Roll-up is performed by climbing up a concept hierarchy for the dimension location.
- Initially the concept hierarchy was "street < city < province < country".
- On rolling up, the data is aggregated by ascending the location hierarchy from the level of city to the level of country.
- The data is grouped into cities rather than countries.
- When roll-up is performed, one or more dimensions from the data cube are removed.

Drill-down

Drill-down is the reverse operation of roll-up. It is performed by either of the following ways:

- By stepping down a concept hierarchy for a dimension
- By introducing a new dimension.

The following diagram illustrates how drill-down works:



- Drill-down is performed by stepping down a concept hierarchy for the dimension time.
- Initially the concept hierarchy was "day < month < quarter < year."
- On drilling down, the time dimension is descended from the level of quarter to the level of month.
- When drill-down is performed, one or more dimensions from the data cube are added.
- It navigates the data from less detailed data to highly detailed data.

Slice

The slice operation selects one particular dimension from a given cube and provides a new sub-cube.

Dice

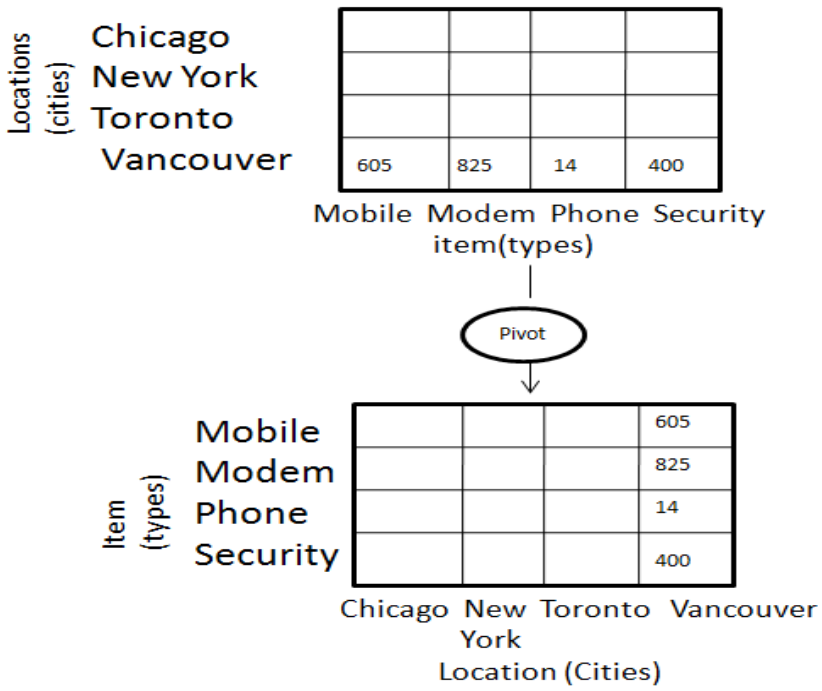
Dice selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram tha

The dice operation on the cube based on the following selection criteria involves three dimensions.

- (location = "Toronto" or "Vancouver")
- (time = "Q1" or "Q2")
- (item =" Mobile" or "Modem")

Pivot

The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentatio



Assessment Questions:

1. Star schema vs snowflake schema
2. Dimensional table Vs. Relational Table
3. Advantages of snowflake schema

Conclusion:

Through OLAP operations the data can be extracted in different fashion. This helps further to ana

Experiment No. 2

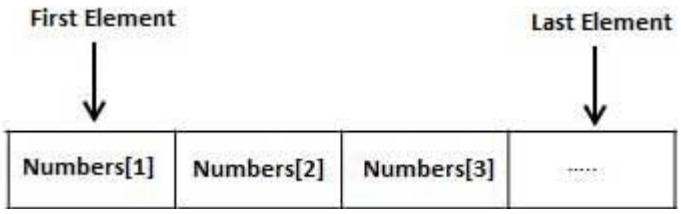
Aim: Implementation of Varying Arrays

Objective: • To learn fundamentals of var arrays

Theory:

PL/SQL programming language provides a data structure called the VARRAY, which can store a fixed-size sequential co variables of the same type.

All varrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest ad



Creating a Varray Type

A varray type is created with the CREATE TYPE statement. You must specify the maximum size and the type of elements

The basic syntax for creating a VRRAY type at the schema level is:

```
CREATE OR REPLACE TYPE varray_type_name IS VARRAY(n) of <element_type>
```

Where,

varray_type_name is a valid attribute name,
n is the number of elements (maximum) in the varray,
element_type is the data type of the elements of the array.

Maximum size of a varray can be changed using the ALTER TYPE statement.

For example,

```
CREATE Or REPLACE TYPE namearray AS VARRAY(3) OF VARCHAR2(10);
```

```
/

Type created.
```

The basic syntax for creating a VRRAY type within a PL/SQL block is:

```
TYPE varray_type_name IS VARRAY(n) of <element_type>
```

For example:

```
TYPE namearray IS VARRAY(5) OF VARCHAR2(10);
Type grades IS VARRAY(5) OF INTEGER;
```

Example 1

The following program illustrates using varrays:

```
DECLARE
   type namesarray IS VARRAY(5) OF VARCHAR2(10);
   type grades IS VARRAY(5) OF INTEGER;
   names namesarray;
   marks grades;
   total integer;
BEGIN
   names := namesarray('Kavita', 'Pritam', 'Ayan', 'Rishav', 'Aziz');
   marks:= grades(98, 97, 78, 87, 92);
   total := names.count;
   dbms_output.put_line('Total '|| total || ' Students');
   FOR i in 1 .. total LOOP
      dbms_output.put_line('Student: ' || names(i) || '
  Marks: ' || marks(i));
   END LOOP;
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Student: Kavita  Marks: 98
Student: Pritam  Marks: 97
```

```
Student: Ayan  Marks: 78
Student: Rishav  Marks: 87
Student: Aziz  Marks: 92

PL/SQL procedure successfully completed.
```

Note:

In oracle environment, the starting index for varrays is always 1.

You can initialize the varray elements using the constructor method of the varray type, which has the same name as

Varrays are one-dimensional arrays.

A varray is automatically NULL when it is declared and must be initialized before its elements can be referenced.

Post lab assignment:

1. Advantages of varrays

Conclusion:

I have understood the process of creating and handling the varying arrays.

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**

Experiment no 3

Aim: Implementation of Nested Tables

Objective: • To learn fundamentals of Nested Arrays

Theory:

A collection is an ordered group of elements having the same data type. Each element is identified by a ur

PL/SQL provides three collection types:

Index-by tables or Associative array

Nested table

Variable-size array or Varray

Oracle documentation provides the following characteristics for each type of collections:

Collection Type	Number of Elements	Subscript Type	Dense or Sparse	Where Created	Can Be Object Type Attribute
Associative array (or index-by table)	Unbounded	String or integer	Either	Only in PL/SQL block	No
Nested table	Unbounded	Integer	Starts dense, can become sparse	Either in PL/SQL block or at schema level	Yes
Variable-size array (Varray)	Bounded	Integer	Always dense	Either in PL/SQL block or at schema level	Yes

We have already discussed varray in the chapter 'PL/SQL arrays'. In this chapter, we will discuss PL/SQL tables.

Both types of PL/SQL tables, i.e., index-by tables and nested tables have the same structure and their rows are accessed us cannot.

Index-By Table

An **index-by** table (also called an associative array) is a set of **key-value** pairs. Each key is unique and is used to locate th
An index-by table is created using the following syntax. Here, we are creating an index-by table named**table_name** whos

```
TYPE type_name IS TABLE OF element_type [NOT NULL] INDEX BY subscript_type;

table_name type_name;
```

Example:

Following example shows how to create a table to store integer values along with names and later it prints the same list of

```
DECLARE
   TYPE salary IS TABLE OF NUMBER INDEX BY VARCHAR2(20);
   salary_list salary;
   name   VARCHAR2(20);
BEGIN
   -- adding elements to the table
   salary_list('Rajnish')  := 62000;
   salary_list('Minakshi')  := 75000;
   salary_list('Martin') := 100000;
   salary_list('James') := 78000;

   -- printing the table
   name := salary_list.FIRST;
   WHILE name IS NOT null LOOP
      dbms_output.put_line
      ('Salary of ' || name || ' is ' || TO_CHAR(salary_list(name)));
      name := salary_list.NEXT(name);
   END LOOP;
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Salary of Rajnish is 62000
Salary of Minakshi is 75000
Salary of Martin is 100000
Salary of James is 78000

PL/SQL procedure successfully completed.
```

Example:

Elements of an index-by table could also be a %ROWTYPE of any database table or %TYPE of any database table field. T

```
Select * from customers;

+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
+----+----------+-----+-----------+----------+
DECLARE
   CURSOR c_customers is
      select  name from customers;

   TYPE c_list IS TABLE of customers.name%type INDEX BY binary_integer;
   name_list c_list;
   counter integer :=0;
BEGIN
   FOR n IN c_customers LOOP
      counter := counter +1;
      name_list(counter)  := n.name;
      dbms_output.put_line('Customer('||counter|| '):'||name_list(counter));
  END LOOP;
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Customer(1): Ramesh
Customer(2): Khilan
Customer(3): kaushik
Customer(4): Chaitali
Customer(5): Hardik
Customer(6): Komal
```

```
PL/SQL procedure successfully completed
```

Nested Tables

A **nested table** is like a one-dimensional array with an arbitrary number of elements. However, a nested table differs from

An array has a declared number of elements, but a nested table does not. The size of a nested table can increase dy

An array is always dense, i.e., it always has consecutive subscripts. A nested array is dense initially, but it can beco

A **nested table** is created using the following syntax:

```
TYPE type_name IS TABLE OF element_type [NOT NULL];

table_name type_name;
```

This declaration is similar to declaration of an **index-by** table, but there is no INDEX BY clause.

A nested table can be stored in a database column and so it could be used for simplifying SQL operations where you join a

Example:

The following examples illustrate the use of nested table:

```
DECLARE
   TYPE names_table IS TABLE OF VARCHAR2(10);
   TYPE grades IS TABLE OF INTEGER;

   names names_table;
   marks grades;
   total integer;
BEGIN
   names := names_table('Kavita', 'Pritam', 'Ayan', 'Rishav', 'Aziz');
   marks:= grades(98, 97, 78, 87, 92);
   total := names.count;
   dbms_output.put_line('Total '|| total || ' Students');
   FOR i IN 1 .. total LOOP
     dbms_output.put_line('Student:'||names(i)||', Marks:' || marks(i));
   end loop;
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Total 5 Students
Student:Kavita, Marks:98
Student:Pritam, Marks:97
Student:Ayan, Marks:78
Student:Rishav, Marks:87
Student:Aziz, Marks:92

PL/SQL procedure successfully completed.
```

Example:

Elements of a **nested table** could also be a %ROWTYPE of any database table or %TYPE of any database table field. The following

```
Select * from customers;

+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
+----+----------+-----+-----------+----------+
DECLARE
   CURSOR c_customers is
      SELECT  name FROM customers;

   TYPE c_list IS TABLE of customers.name%type;
   name_list c_list := c_list();
   counter integer :=0;
BEGIN
   FOR n IN c_customers LOOP
      counter := counter +1;
      name_list.extend;
      name_list(counter)  := n.name;
      dbms_output.put_line('Customer('||counter||'):'||name_list(counter));
   END LOOP;
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Customer(1): Ramesh
Customer(2): Khilan
Customer(3): kaushik
Customer(4): Chaitali
Customer(5): Hardik
Customer(6): Komal

PL/SQL procedure successfully completed.
```

Conclusion:

I have understood the process of creating and handling the Nested Tables. It is different from the tables we handled so far.

Experiment No 4

Aim: Develop a star schema defining subject area , fact Table and dimension table

Design schemas

star schema or dimensional model is a fairly popular design choice, as it enables a relational database to emulate the analytical functionality of a multidimensional database.

Star schema architecture

Star schema architecture is the simplest data warehouse design. The main feature of a star schema is a table at the center, called the fact table and the dimension tables which allow browsing of specific categories, summarizing, drill-downs and specifying criteria.

Typically, most of the fact tables in a star schema are in database third normal form, while dimensional tables are de-normalized (second normal form).

Despite the fact that the star schema is the simplest data warehouse architecture, it is most commonly used in the data warehouse implementations across the world today (about 90-95% cases).

Fact table

The fact table is not a typical relational database table as it is de-normalized on purpose - to enhance query response times scores (like QUANTITY, TURNOVER, exact invoice date and time). Typical fact tables in a global enterprise data wareh

1. sales fact table - contains all details regarding sales
2. orders fact table - in some cases the table can be split into open orders and historical orders. Sometimes the values for h
3. budget fact table - usually grouped by month and loaded once at the end of a year.

Dimension table

Nearly all of the information in a typical fact table is also present in one or more dimension tables. The main purpose of maintaining Dimension Tables is to allow browsing the categories quickly and easily.

The primary keys of each of the dimension tables are linked together to form the composite primary key of the fact table. In a star schema design, there is only one de-normalized table for a given dimension.

Typical dimension tables in a data warehouse are:

time dimension table

customers dimension table

products dimension table

key account managers (KAM) dimension table

sales office dimension table

Experiment 5

Aim: Demonstration of preprocessing on dataset student.arff

Objectives:

- To learn to use the Weka machine learning toolkit

Theory:

How do you load Weka?
1. What options are available on main panel?
2. What is the purpose of the the following in Weka:

1. The Explorer
2. The Knowledge Flow interface
3. The Experimenter
4. The command-line interface
5. Describe the arff file format.

Steps of execution:

Step1: Loading the data. We can load the dataset into weka by clicking on open button in preprocessing interface and se

Step2: Once the data is loaded, weka will recognize the attributes and during the scan of the data weka will compute son relation or table and the current working relation (which are same initially).

Step3: Clicking on an attribute in the left panel will show the basic statistics on the attributes for the categorical attribute

Step4: The visualization in the right button panel in the form of cross-tabulation across two attributes.

Note: we can select another attribute using the dropdown list

Step5: Selecting or filtering attributes

Removing an attribute- When we need to remove an attribute, we can do this by using the attribute filters in weka. In the

Scroll down the list and select the "weka filters unsupervised Attribute remove" filters.

Step 6: a) Next click the textbox immediately to the right of the choose button. In the resulting dialog box enter the inde

b) Make sure that invert selection option is set to false. The click OK now in the filter box you will see "Remove-R-7".

c) Click the apply button to apply filter to this data. This will remove the attribute and create new working relation.

d) Save the new working relation as an arff file by clicking save button on the top (button) panel(student.arff)

Dataset student .arff

@relation student

@attribute age {<30,30-40,>40}

@attribute income {low, medium, high}

@attribute student {yes, no}

@attribute credit-rating {fair, excellent}

@attribute buyspc {yes, no}

@data

%

<30, high, no, fair, no

<30, high, no, excellent, no

30-40, high, no, fair, yes

>40, medium, no, fair, yes

>40, low, yes, fair, yes

>40, low, yes, excellent, no

30-40, low, yes, excellent, yes

<30, medium, no, fair, no

<30, low, yes, fair, no

>40, medium, yes, fair, yes

<30, medium, yes, excellent, yes

30-40, medium, no, excellent, yes

30-40, high, yes, fair, yes

>40, medium, no, excellent, no

%

Conclusion:

Using Weka Tool is easier for processing the dataset in arff format.

Experiment No. 6
Aim: Implement Apriori algorithm for association rule

Objective: To learn Apriori algorithm for data association

Theory: **Theory:**
Association rule mining is to find out association rules that satisfy the predefined minimum support and confidence from a given database. The problem is usually decomposed into two sub problems.

- Find those item sets whose occurrences exceed a predefined threshold in the database; those item sets are called frequent or large item sets.
- Generate association rules from those large item sets with the constraints of minimal confidence.

Suppose one of the large item sets is $Lk = \{I1,I2,...,Ik\}$; association rules with this item sets are generated in the following way: the first rule is $\{I1,I2,...,Ik-1\} => \{Ik\}$. By checking the confidence this rule can be determined as interesting or not. Then, other rules are generated by deleting the last items in the antecedent and inserting it to the consequent, further the confidences of the new rules are checked to determine the interestingness of them. This process iterates until the antecedent becomes empty.

Since the second sub problem is quite straight forward, most of the research focuses on the first sub problem. The Apriori algorithm finds the frequent sets L in Database D.

- Find frequent set L_{k-1}.
- Join Step.

- C_k is generated by joining L_{k-1} with itself

Prune Step.

- Any $(k-1)$ -itemset that is not frequent cannot be a subset of a frequent k -itemset, hence should be removed.

where

- (C_k: Candidate itemset of size k)

- (L_k: frequent itemset of size k)

Input :
A large supermarket tracks sales data by ***SKU(Stoke Keeping Unit)*** (item), and thus is able to know what items are typically purchased together. Apriori is a moderately efficient way to build a list of frequent purchased item pairs from this data.

Let the database of transactions consist of the sets {1,2,3,4}, {2,3,4}, {2,3}, {1,2,4}, {1,2,3,4}, and {2,4}.

Output
Each number corresponds to a product such as "butter" or "water". The first step of Apriori to count up the frequencies, called the supports, of each member item separately:

Item	Support
1	3
2	6
3	4
4	5

We can define a minimum support level to qualify as "frequent," which depends on the context. For this case, let min support = 3. Therefore, all are frequent. The next step is to generate a list of all 2-pairs of the frequent items. Had any of the above items not been frequent, they wouldn't have been included as a possible member of possible 2-item pairs. In this way, Apriori prunes the tree of all possible sets.

Item	Support
{1,2}	3
{1,3}	2
{1,4}	3
{2,3}	4
{2,4}	5
{3,4}	3

This is counting up the occurrences of each of those pairs in the database. Since minsup=3, we don't need to generate 3-sets involving {1,3}. This is because since they're not frequent, no supersets of them can possibly be frequent. Keep going:

Item	Support

{1,2,4} 3
{2,3,4} 3

Execution steps

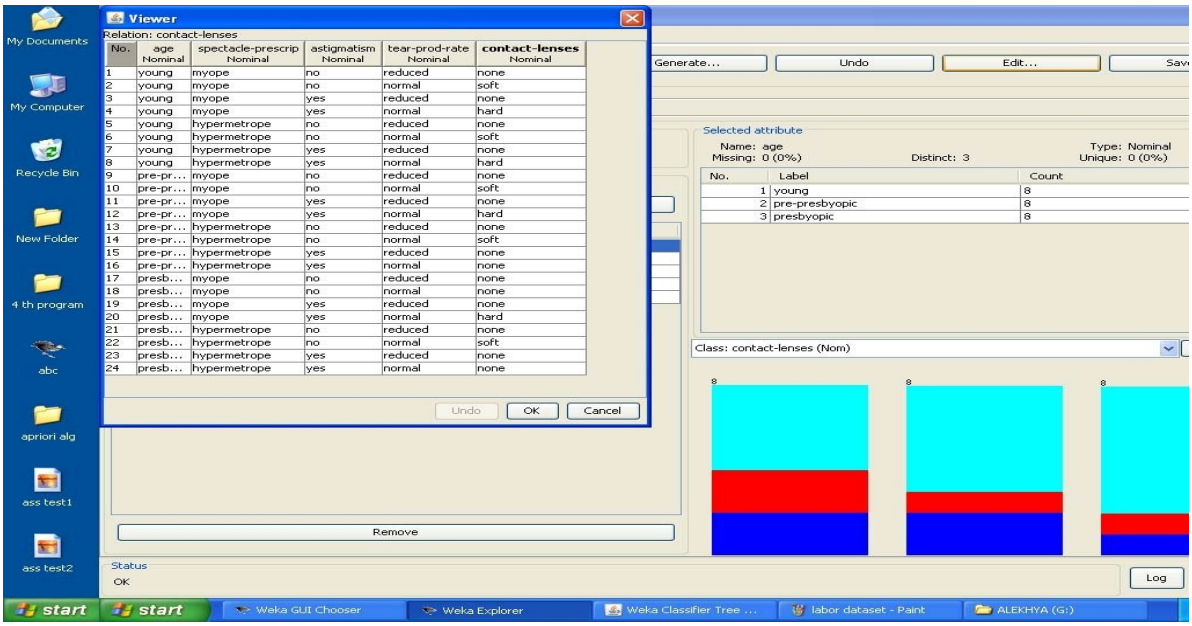
Step1: Open the data file in Weka Explorer. It is presumed that the required data fields have been discretized. In this ex

Step2: Clicking on the associate tab will bring up the interface for association rule algorithm.

Step3: We will use apriori algorithm. This is the default algorithm.

Step4: Inorder to change the parameters for the run (example support, confidence etc) we click on the text box imm

Dataset contactlenses.arff



Dataset test.arff

@relation test

@attribute admissionyear {2005,2006,2007,2008,2009,2010}

@attribute course {cse,mech,it,ece}

@data

%

2005, cse

2005, it

2005, cse

2006, mech

2006, it

2006, ece

2007, it

2007, cse

2008, it

2008, cse

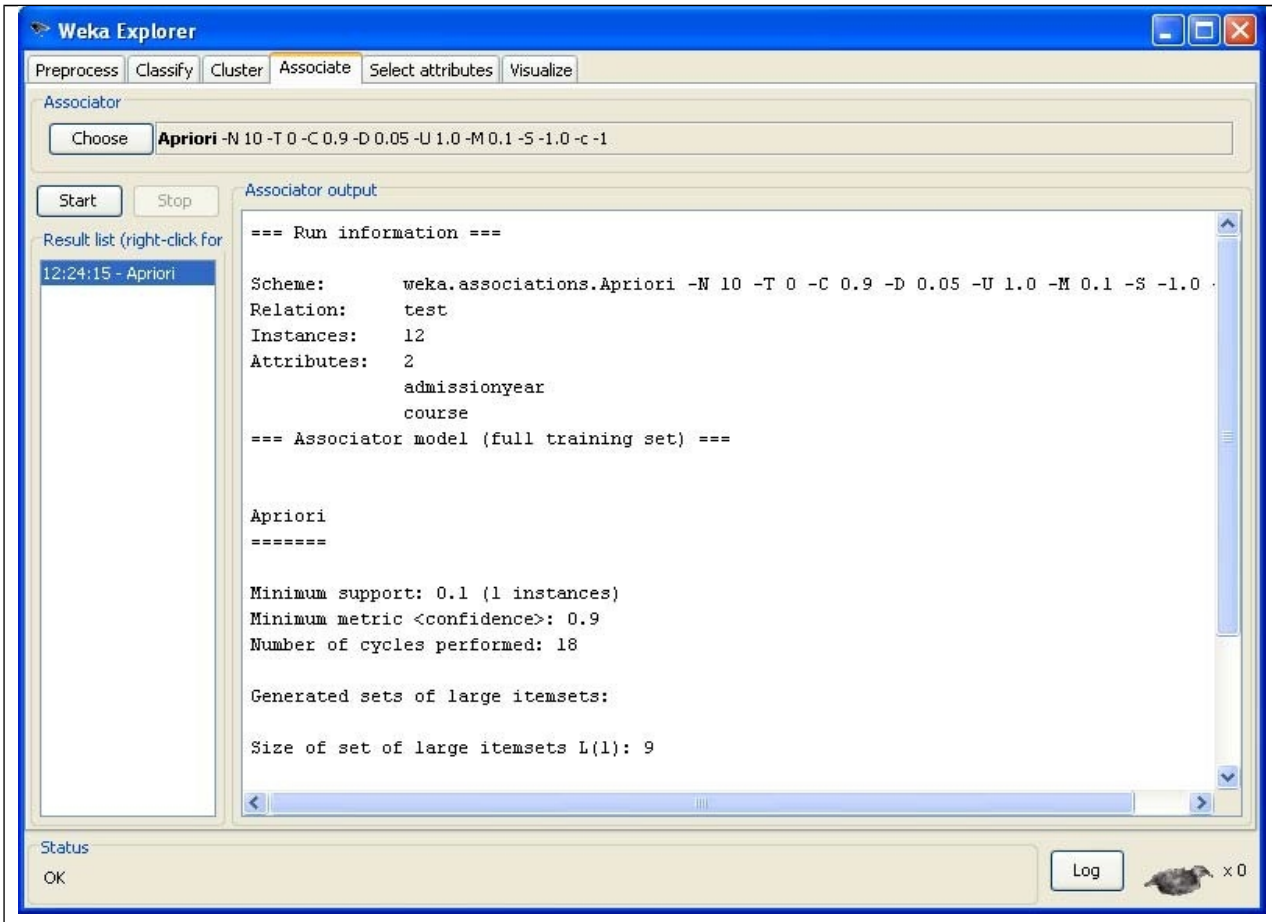
2009, it

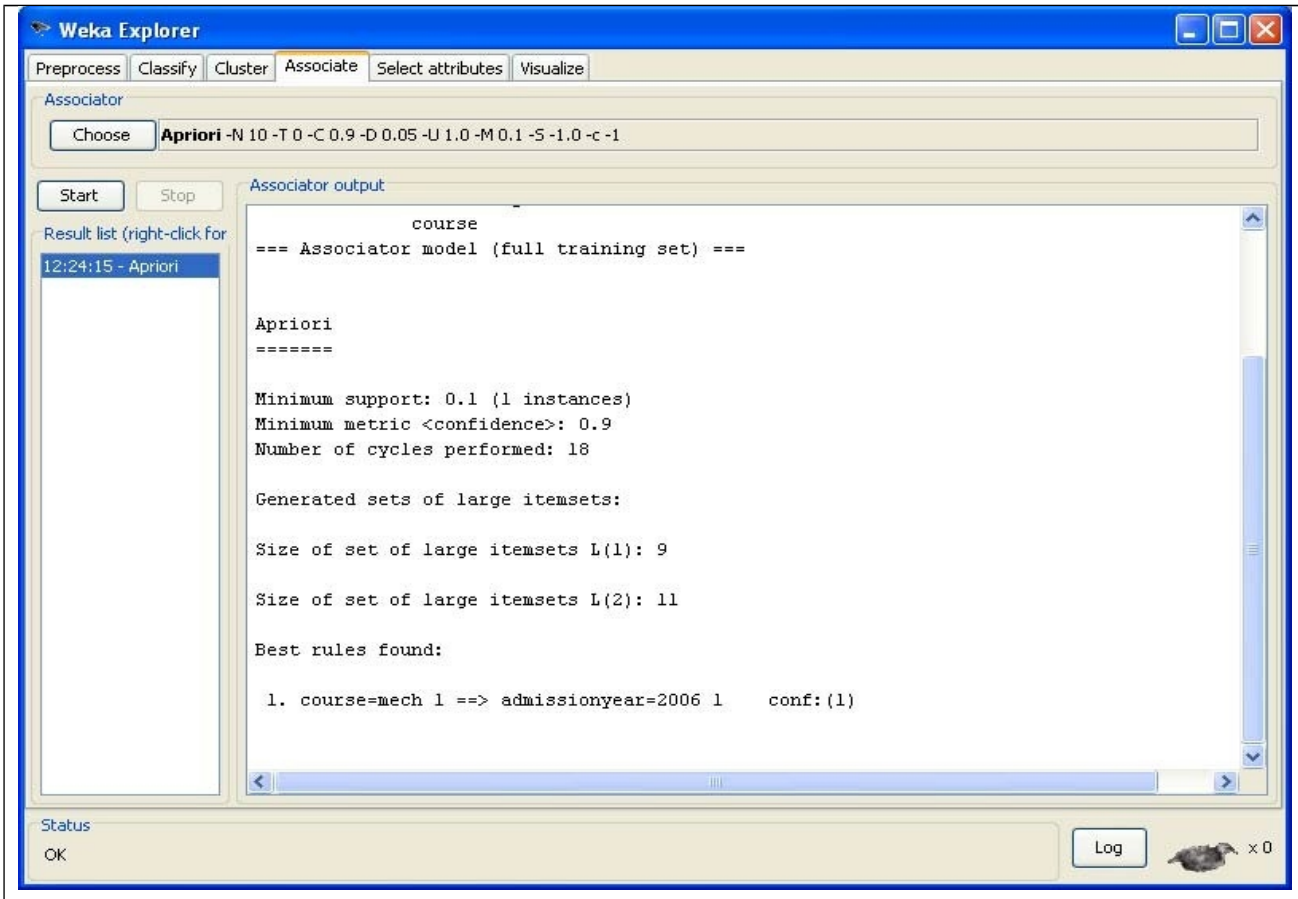
2009, ece

%

The following screenshot shows the association rules that were generated when Apriori algorithm is applied

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**





Post lab assignment:

1. Give an example for Apriori with transaction and explain Apriori-gen-algorithm

Conclusion:

The experiment displays Set of large itemsets, best rule found for the given support and the confidence values. We get the mine the basic data from a database.

Experiment No. 7

Aim: Bayesian Classification

Objective: • To implement classification using Bayes theorm.
The simple baysian classification assumes that the effect of an attribute value of a given class membership is independent of other attribute.

The Bayes theorm is as follows –
Let X be an unknown sample.
Let it be hypothesis such that X belongs to particular class C.
We need to determine P(H/X).

The probability that hypothesis it holds is given that all values of X are observed.

P(H/X) = P(X/H).P(H)/P(X)

In this program, we initially take the number of tuples in training data set in variable L. The string array's name, gender, hight, output to store the details and output respectfully. Therefore, the tuple details are taken from user using 'for' loops.

Bayesian classification has an expected classification. Now using the counter variables for various attributes i.e. (male/female) for gender and (short/medium/tall) for hight.

The tuples are scanned and the respective counter is incremented accordingly using ifelse-if structure.

Therefore variables pshort, pmed, plong are used to convert the counter variables to corresponding values.

Algorithm –

1. START
2. Store the training data set
3. Specify ranges for classifying the data
4. Calculate the probability of being tall, medium, short
5. Also, calculate the probabilities of tall, short, medium according to gender and Classification ranges
6. Calculate the likelihood of short, medium and tall
7. Calculate P(t) by summing up of probable likelihood
8. Calculate actual probabilities

Input:

Training data set

Name	**Gender**	**Height**	**Output**
Christina	F	1.6m	Short
Jim	M	1.9m	Tall
Maggie	F	1.9m	Medium
Martha	F	1.88m	Medium
Stephony	F	1.7m	Medium
Bob	M	1.85m	Short
Dave	M	1.7m	Short
Steven	M	2.1m	Tall
Amey	F	1.8m	Medium

Output

The tuple belongs to the class having highest probability. Thus new tuple is classified.

Steps involved in this experiment:

Step-1: We begin the experiment by loading the data (student.arff)into weka.

Step2: Next we select the "classify" tab and click "choose" button t o select the "j48"classifier.

Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the chos

Step4: Under the "text" options in the main panel. We select the 10-fold cross validation as our evaluation approach.

Step-5: We now click "start" to generate the model .the Ascii version of the tree as well as evaluation statistic will app

Step-6: Note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either i

Step-7: Now weka also lets us a view a graphical version of the classification tree. This can be done by right clicking

Step-8: We will use our model to classify the new instances.

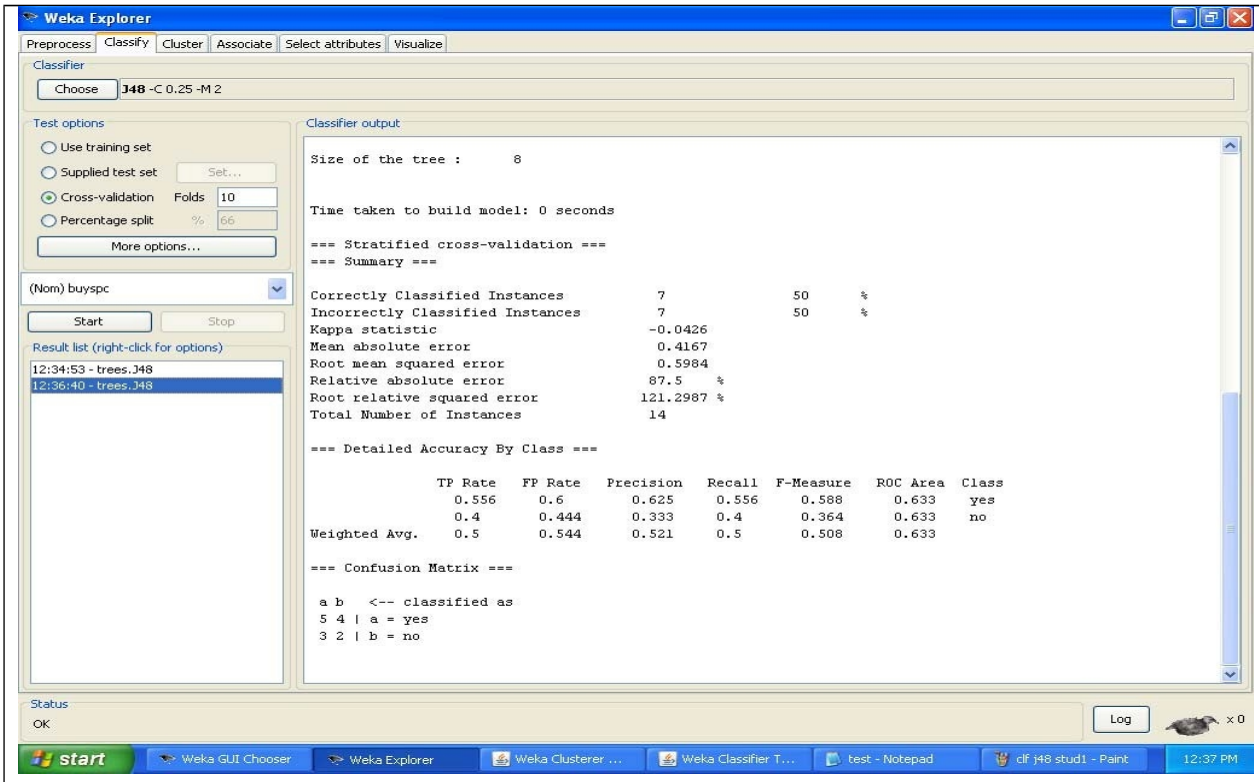
Step-9: In the main panel under "text" options click the "supplied test set" radio button and then click the "set" button. Th

Dataset student .arff

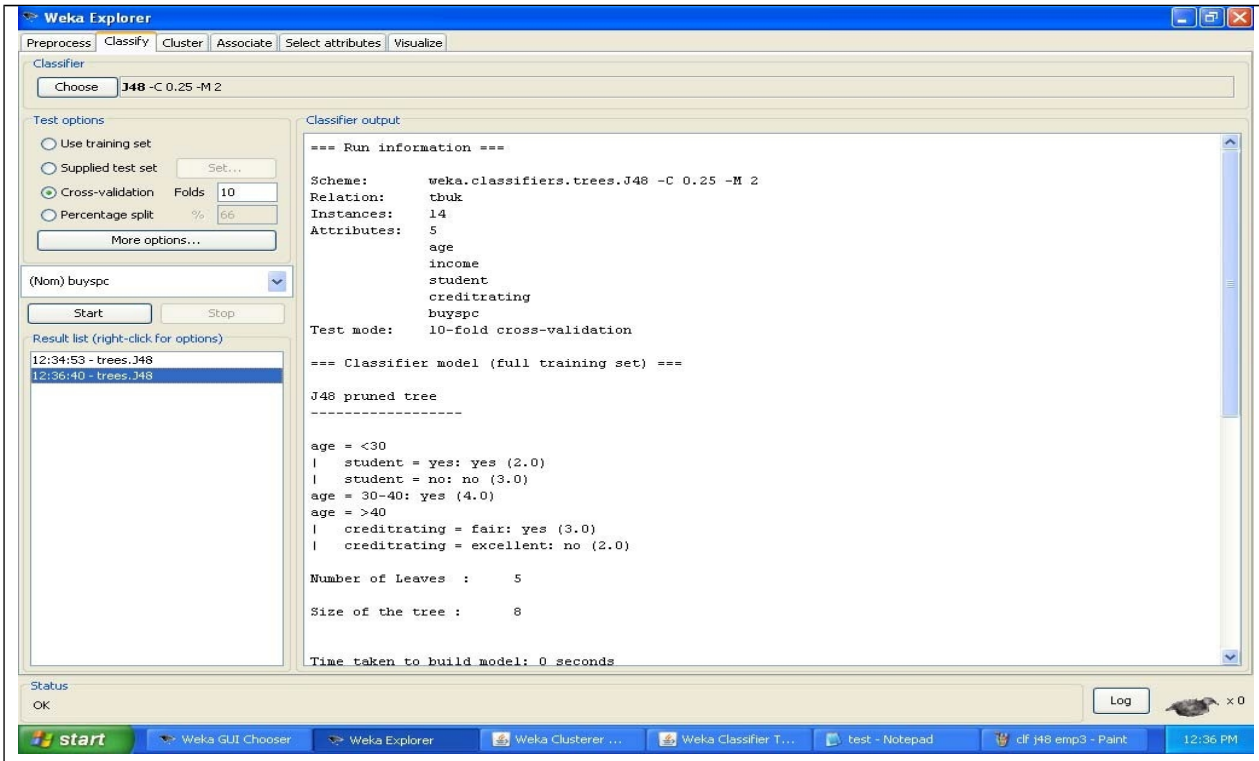
@relation student
@attribute age {<30,30-40,>40}
@attribute income {low, medium, high}
@attribute student {yes, no}
@attribute credit-rating {fair, excellent}
@attribute buyspc {yes, no}
@data
%
<30, high, no, fair, no
<30, high, no, excellent, no
30-40, high, no, fair, yes
>40, medium, no, fair, yes
>40, low, yes, fair, yes
>40, low, yes, excellent, no
30-40, low, yes, excellent, yes
<30, medium, no, fair, no
<30, low, yes, fair, no
>40, medium, yes, fair, yes
<30, medium, yes, excellent, yes
30-40, medium, no, excellent, yes
30-40, high, yes, fair, yes
>40, medium, no, excellent, no
%

The following screenshot shows the classification rules that were generated when j48 algorithm is applied

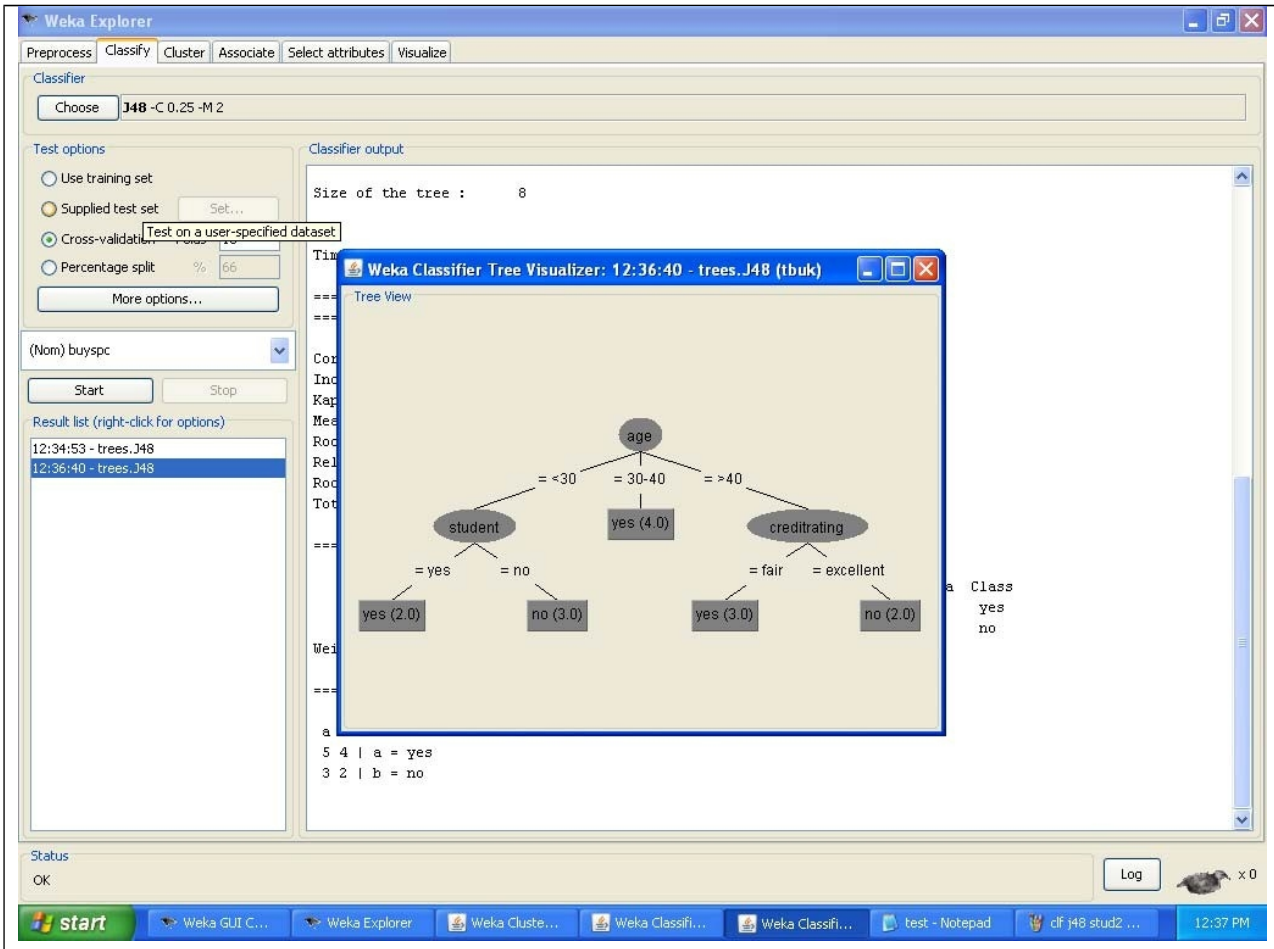
	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**



	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**



	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**



Conclusion:

The experiment displays decision tree, which is annotated (labeled). It also gives the time taken to build the tree and the co

Experiment 8

Aim: To understand principles of clustering
To Implement K-means algorithm for clustering

Theory:

In statistics and machine learning, k-means clustering is a method of cluster analysis which aims to partition n observation
Here is step by step k means clustering algorithm:

Step 1. Begin with a decision on the value of k = number of clusters

Step 2. Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly, or systematically as the following:

1. Take the first k training sample as single-element clusters

Assign each of the remaining (N-k) training sample to the cluster with the nearest centroid. After each assignment, recomputed the centroid of the gaining cluster.

Step 3. Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample.

Step 4. Repeat step 3 until coverage is achieved, that is until a pass through the training sample cause no new assignment

Conclusion:

K- means clustering is simplest method used for forming data clusters.

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**

Experiment 9

A case study of Business Intelligence in Government sector/Social Networking/Business

Objectives:

- To understand the concept of BI
- To understand the Role of Data Analyst in Business

Instructions:

Analyze the Case Study: Closure of Big Bazaar in Aurangabad

Points:

1. Potential of starting Big Bazar in Aurangabad
2. How people responded
3. What was the role of data analyst
4. How the data was maintained through daily transactions?
5. What were the weak points?

	G.S. MANDAL'S **MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**	
	LAB WORK INSTRUCTION SHEET	
	DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING	
CLASS: B. TECH (Final Year) PART: 1 (2019-20)	**LAB: write lab no**	**SUBJECT: subject code-subject name**