# Experiment – 1

**Title:** Build Data Warehouse/Data Mart for a given problem statement (auto sales analysis)

i) Identifying the source tables and populating sample data

### ii) Design dimensional data model i.e. Star schema, Snowflake schema and Fact Constellation schema (if applicable)

Perform following for the above warehouse:

1. Maximum/minimum sale in first quarter w.r.t. location
2. Maximum/minimum sale of item “vehicles” throughout the year
3. Maximum/minimum sale of item throughout the year
4. Maximum/minimum sale during the second & third quarter w.r.t. location and item
5. List out the items in increasing order w.r.t. sales amount & quantity sold
6. List out the suppliers who supply maximum number of “bikes/cars” during year
7. Find out the customer who purchase maximum number of items and also find out all the details of customer along with region.

### Notes:

Design a data cube which contain one fact table and design item, time, supplier, location, customer dimension table , also identify measures for sales. Insert minimum 4 items like bikes, small cars, mid segment cars, car consumables items etc. Also enter minimum 10, 12 records Region/location, enter minimum 2 cities from each state also enter minimum 2 states. Keep track of sales quarter wise.

Perform and implementabovefact&dimensiontablesinoracle10gwhicharesameas relational table of database, perform analyze above with the help of SQL tool.

You have to use concepts of OLAP operation like slice, dice, roll-up, drill-down etc.

### Objective:

* To learn fundamental of data warehousing
* To learn concepts of dimensional modeling
* To learn star, snowflake & Galaxy schema

### Reference:

* SQL-PL/SQL by Ivan Bayrose
* Data Mining Concept and Technique By Han & Kamber
* Data Warehousing Fundamentals By Paulraj
* Data warehousing & Mining By Reema Thereja

### Pre-requisite:

* Fundamental Knowledge of Database Management

**.** Fundamental Knowledge of SQL

### Theory:

Dimensional modeling (DM) is the name of a logical design technique often used for data warehouses. Dimensional modeling always uses the concepts of facts, measures, and dimensions.

Facts are typically (but not always) numeric values that can be aggregated,

Dimensions are groups of hierarchies and descriptors that define the facts. For example, sales amount is a fact; timestamp, product, register#, store#, etc. are elements of dimensions.

Dimensional models are built by business process area, e.g. store sales, inventory, claims, etc.

### Fact table

The fact table is not a typical relational database table as it is de-normalized on purpose, to Enhance query response times. The fact table typically contains records that are ready to explore, usually with adhoc queries. Records in the fact table are often referred to as events, due to the time-variant nature of a data warehouse environment. The primary key for the fact table is a composite of all the columns except numeric values/scores (like QUANTITY, TURN OVER, exact invoice date and time).Typical fact tables in a global enterprise at a ware house are (usually there may be additional company or business specific fact tables):

Sales fact table-contains all details regarding sales

Orders fact table-in some cases the table can be split into open orders and historical orders. Sometimes the values for historical orders are stored in a sales fact table.

Budget fact table-usually grouped by month and loaded once at the end of a year.

Forecast fact table : usually grouped by month and loaded daily, weekly or monthly. Inventory fact table : report stocks, usually refreshed daily.

### 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:

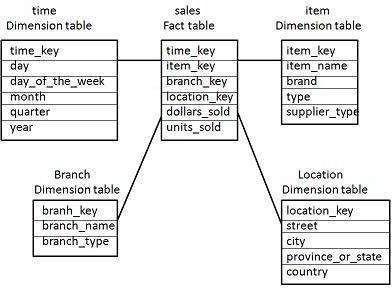
### Time dimension table Customers dimension table Products dimension table

Key account managers (KAM) dimension table Sales office dimension

### Star schema architecture

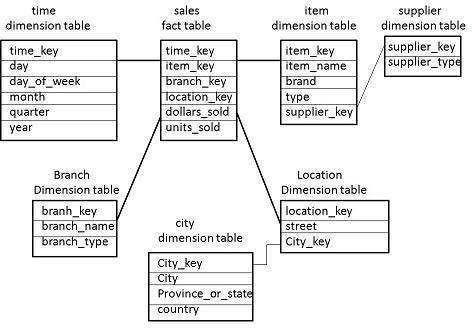
Star schema architecture is the simplest data warehouse design. The main feature of a star schema is a table at the center, called the fact table and the dimension tables which allow browsing of specific categories, summarizing, drill-downs and specifying criteria. Typically, most of the fact tables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



### Snowflake Schema architecture

Snow flake schema architecture is a more complex variation of a star schema design. The main difference is that dimensional tables in a snow flake schema are normalized, so they have a typical relational database design. Snow flake schemas are generally used when a dimensional table becomes very big and when a star schema can’t represent the complexity of a data structure.For example if a PRODUCT dimension table contains millions of rows, the use of snow flake schemas should significantly improve performance by moving out some data to other table(with BRANDS for instance). The problem is that the more normalized the dimension table is, the more complicated SQL joins must be issued to query them. This is because in order for a query to be answered, many tables need to be joined and aggregates generated.



### Fact constellation/Galaxy schema Architecture

For each star schema or snow flake schema it is possible to construct a fact constellation schema. This schema is more complex than star or snowflake architecture, which is because it contains multiple fact tables. This allows dimension tables to be shared amongst many fact tables.

In a fact constellation schema, different fact tables are explicitly assigned to the dimensions, which are for given facts relevant. This may be useful in cases when some facts are associated with a given dimension level and other facts with a deeper dimension level.

Use of that model should be reasonable when for example, there is a sales fact table (with

Details down to the exact date and invoice header id) and a fact table with sales forecast which is calculated based on month, client id and product id.

### Conclusion:

A schema is a logical description of database where fact and dimension tables are joined in a logical manner. Data Warehouse is maintained in the form of Star, Snow flakes, and Fact Constellation schema.

# Experiment – 2

## **Title**: Experiment to execute different OLAP operation on OLAP Cube and compare result with OLAP queries o/p. (Roll up, Drill down, Slice, Dice, pivot operations).

**Theory :**

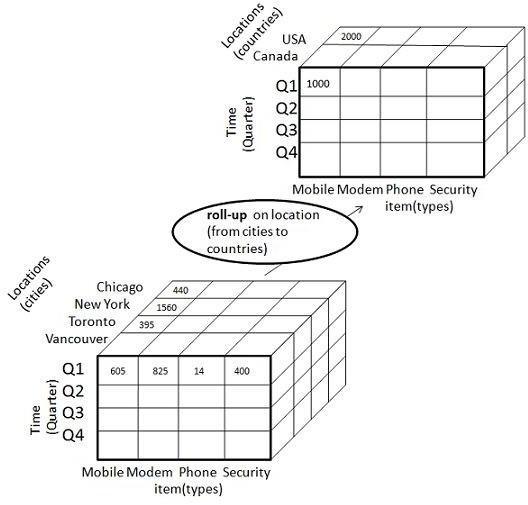
Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information.

### 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.

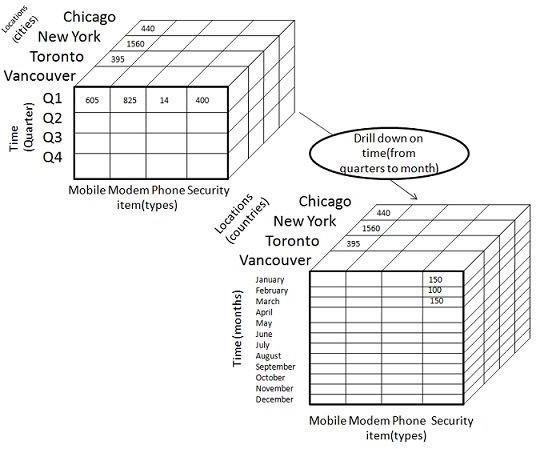


**Drill-down**

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

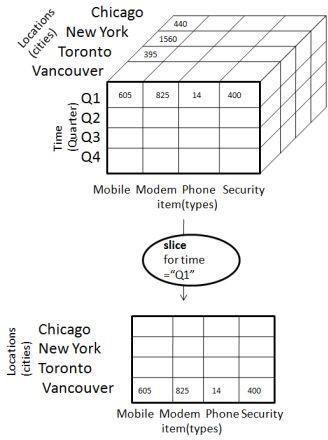
* + By stepping down a concept hierarchy for a dimension
  + By introducing a new dimension.

The following diagram illustrates how drill-down works –



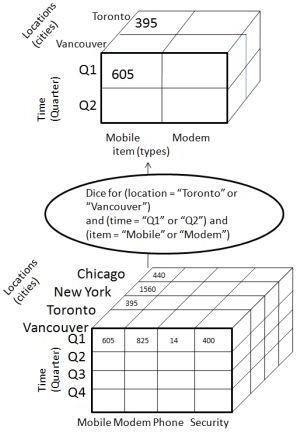
## Slice

The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.



**Dice**

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



**Pivot**

The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentation of data. Consider the following diagram that shows the pivot operation.

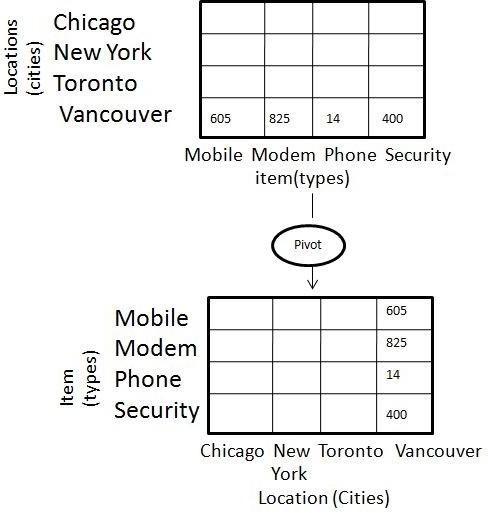


Table customer

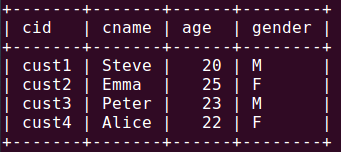


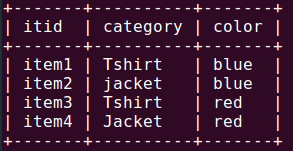
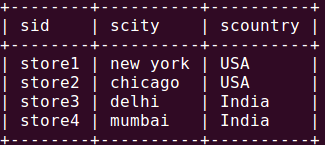
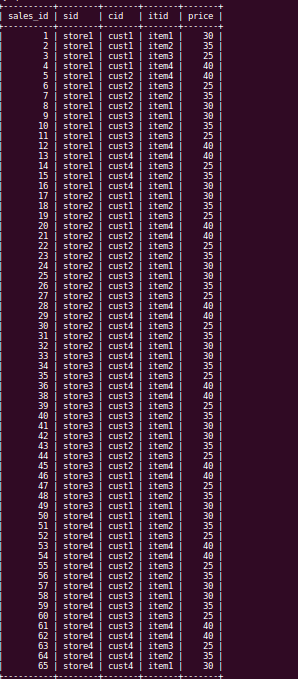
Table item  


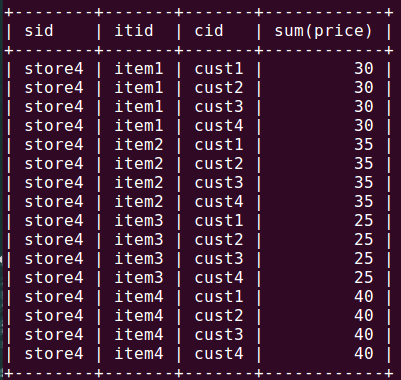
Table store





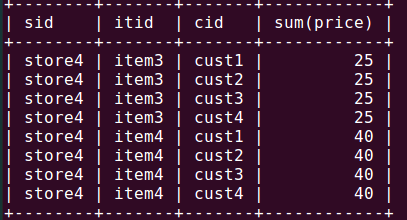
**Slicing** :

mysql> select Sl.sid, itid, cid ,sum(price) from sales Sl, store s where Sl.sid=s.sid and scity='mumbai' group by Sl.sid, itid, cid;



**Dicing :**

mysql> select Sl.sid, Sl.itid, cid ,sum(price) from sales Sl, store s, item i where Sl.sid=s.sid and Sl.itid=i.itid and scity='mumbai' and color='red' group by Sl.sid, Sl.itid, cid;



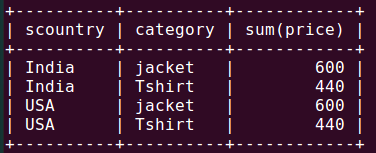
**Roll up:**

mysql> select scountry , category ,sum(price)

->from sales Sl, store s, item i

->whereSl.sid=s.sid and Sl.itid=i.itid

->group by scountry ,category;



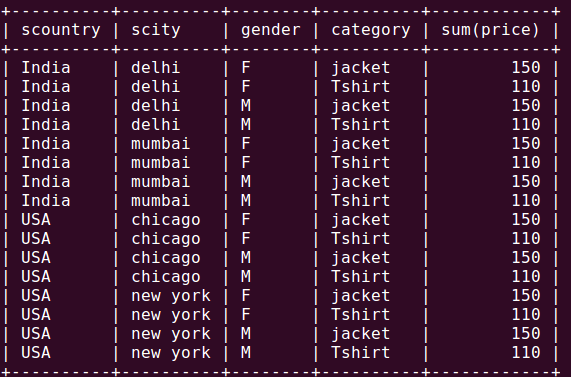
**Drill-down:**

mysql> select scountry , scity, gender, category ,sum(price)

->from sales Sl, store s, item i, customer c

->whereSl.sid=s.sid and Sl.itid=i.itid and Sl.cid=c.cid

->group by scountry ,scity ,gender,category;



## **Conclusion**:

OLAP enables a user to easily and selectively extract and view data from different points of view. It allows users to analyze database information from multiple database systems at one time.

# Experiment – 3

**Title:** Implementation of Classification algorithm (Decision Tree).

# Algorithm

Create node N

If all the samples are from same class, C then Return N as leaf node- labeled C

If attribute list is empty

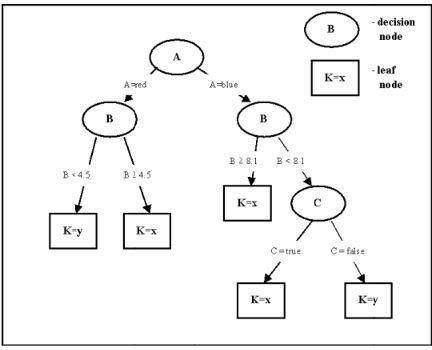
Return N as leaf node assign common class name

Select test- attribute from attribute list which having highest information Label N with test- attribute

(Determine the best splitting criteria)

For each value of attribute a of test –attribute Assign test- value to arch

Si is set of samples of test-attribute ai Repeat above



**Code:**

import java.util.\*;

class tree

{

public double hd;

public double p;

char ch;

String parent;

tree(char c,int x)

{

ch=c;

if(x==1)

parent="outlook";

else if(x==2)

parent="temperature";

else if(x==3)

parent="humidity";

else if(x==4)

parent="windy";

}

}

class d\_tree

{

static char outlook[]={'S','S','O','R','R','R','O','S','S','R','S','O','O','R'};

static char temperature[]={'B','B','B','M','C','C','C','M','C','M','M','M','B','M'};

static char humidity[]={'H','H','H','H','N','N','N','H','N','N','N','H','N','H'};

static char windy[]={'F','T','F','F','F','T','T','F','F','F','T','T','F','T'};

static char class1[]={'N','N','P','P','P','N','P','N','P','P','P','P','P','N'};

static double G1,G2,G3,G4;

static double HD;

static double play=9.0/14.0;

static double nplay=5.0/14.0;

static double temp1,temp2;

static int row=0,column=0;

static char classify[][]=new char[10][10];

static tree cal\_hd(tree t,int choice)

{

double count1=0,count2=0;

if(choice==1)

{for(int i=0;i<14;++i)

{if(t.ch==outlook[i] && class1[i]=='P')

++count1;

else

if(t.ch==outlook[i] && class1[i]=='N')

++count2;

}}

if(choice==2)

{for(int i=0;i<14;++i)

{if(t.ch==temperature[i] && class1[i]=='P')

++count1;

else

if(t.ch==temperature[i] && class1[i]=='N')

++count2;

}}

if(choice==3)

{for(int i=0;i<14;++i)

{if(t.ch==humidity[i] && class1[i]=='P')

++count1;

else

if(t.ch==humidity[i] && class1[i]=='N')

++count2;

}}

if(choice==4)

{for(int i=0;i<14;++i)

{if(t.ch==windy[i] && class1[i]=='P')

++count1;

else

if(t.ch==windy[i] && class1[i]=='N')

++count2;

}}

temp1=count1/(count1+count2);

temp2=count2/(count1+count2);

t.p=(count1+count2)/14;

if(temp1==0 || temp2==0)

t.hd=0;

else

t.hd=temp1\*(Math.log(1/temp1)/Math.log(10))+temp2\*(Math.log(1/temp2)/Math.log(10));

return t;

}

static int check\_if\_equal(char ca[][],int cnt)

{

int count1=0,count2=0,count3=0;

char c1=ca[0][0],c2=ca[0][1],c3=ca[0][2];

for(int i=0;i<cnt;++i)

{

if(ca[i][0]==c1)

++count1;

if(ca[i][1]==c2)

++count2;

if(ca[i][2]==c3)

++count3;

}

if(count1==cnt)

return 0;

else

if(count2==cnt)

return 1;

else

if(count3==cnt)

return 2;

else

return -1;

}

static void compute(char ch)

{

char ca1[][]=new char[10][10];

char ca2[][]=new char[10][10];

char ca3[][]=new char[10][10];

int cnt1=0;

int cnt2=0;

for(int i=0;i<14;++i)

{

if(outlook[i]==ch)

{

ca1[cnt1][0]=temperature[i];

ca1[cnt1][1]=humidity[i];

ca1[cnt1][2]=windy[i];

ca1[cnt1][3]=class1[i];

++cnt1;

}

}

int check1=0,check2=0;

for(int i=0;i<cnt1;++i)

{

if(ca1[i][3]=='P')

++check1;

else

if(ca1[i][3]=='N')

++check2;

if(check1==cnt1)

{

classify[row][column++]=ch;

classify[row][column++]='P';

System.out.print("--------->PLAY");

return;

}

else

if(check2==cnt1)

{

classify[row][column++]=ch;

classify[row][column++]='N';

System.out.print("--------->NO PLAY");

return;

}

}

//end of for loop

/\*System.out.println("The array containing outlook value ch");

for(int i=0;i<9;++i)

{

for(int j=0;j<9;++j)

{

System.out.print(" "+ca1[i][j]);

}

System.out.println();

}

\*/

//The array containing outlook value ch and class1=P

cnt1=0;

for(int i=0;i<10;++i)

{

if(ca1[i][3]=='P')

{

ca2[cnt1][0]=ca1[i][0];

ca2[cnt1][1]=ca1[i][1];

ca2[cnt1][2]=ca1[i][2];

ca2[cnt1][3]=ca1[i][3];

++cnt1;

}

}

classify[row][column++]=ch;

int z=check\_if\_equal(ca2,cnt1);

//System.out.println("The value of Z is "+z);

if(z==0)

{

System.out.print("\n--------->TEMPERATURE("+ca2[0][0]+")--------->PLAY\n");

classify[row][column++]=ca2[0][0];

classify[row][column++]='P';

}

else

if(z==1)

{

System.out.print("\n--------->HUMIDITY("+ca2[0][1]+")--------->PLAY\n");

classify[row][column++]=ca2[0][1];

classify[row][column++]='P';

}

else

if(z==2)

{

System.out.print("\n--------->WINDY("+ca2[0][2]+")--------->PLAY\n");

classify[row][column++]=ca2[0][2];

classify[row][column++]='P';

}

/\*

System.out.println("The array containing ch and play");

for(int i=0;i<9;++i)

{

for(int j=0;j<9;++j)

{

System.out.print(" "+ca2[i][j]);

}

System.out.println();

}

\*/

//The array containing outlook value ch and class1=N

cnt1=0;

for(int i=0;i<10;++i)

{

if(ca1[i][3]=='N')

{

ca3[cnt1][0]=ca1[i][0];

ca3[cnt1][1]=ca1[i][1];

ca3[cnt1][2]=ca1[i][2];

ca3[cnt1][3]=ca1[i][3];

++cnt1;

}}

++row;

column=0;

classify[row][column++]=ch;

z=check\_if\_equal(ca3,cnt1);

if(z==0)

{

System.out.print("--------->TEMPERATURE("+ca3[0][0]+")--------->NO PLAY");

classify[row][column++]=ca3[0][0];

classify[row][column++]='N';

}

else

if(z==1)

{

System.out.print("--------->HUMIDITY("+ca3[0][1]+")--------->NO PLAY");

classify[row][column++]=ca3[0][1];

classify[row][column++]='N';

}

else

if(z==2)

{

System.out.print("--------->WINDY("+ca3[0][2]+")--------->NO PLAY");

classify[row][column++]=ca3[0][2];

classify[row][column++]='N';

}

/\*System.out.println("The array containing ch and not play");

for(int i=0;i<9;++i)

{

for(int j=0;j<9;++j)

{

System.out.print(" "+ca3[i][j]);

}

System.out.println();

}

\*/

}

public static void main(String args[])

{

Scanner scr=new Scanner(System.in);

HD=play\*(Math.log(1/play)/Math.log(10))+nplay\*(Math.log(1/nplay)/Math.log(10));

System.out.println("\nThe value of H(D) is "+HD);

tree sunny=new tree('S',1);

tree overcast=new tree('O',1);

tree rain=new tree('R',1);

tree hot=new tree('B',2);

tree mild=new tree('M',2);

tree cool=new tree('C',2);

tree high=new tree('H',3);

tree normal=new tree('N',3);

tree tru=new tree('T',4);

tree fal=new tree('F',4);

sunny=cal\_hd(sunny,1);

overcast=cal\_hd(overcast,1);

rain=cal\_hd(rain,1);

hot=cal\_hd(hot,2);

mild=cal\_hd(mild,2);

cool=cal\_hd(cool,2);

high=cal\_hd(high,3);

normal=cal\_hd(normal,3);

tru=cal\_hd(tru,4);

fal=cal\_hd(fal,4);

G1=HD-(sunny.p\*sunny.hd+overcast.p\*overcast.hd+rain.p\*rain.hd);

G2=HD-(hot.p\*hot.hd+mild.p\*mild.hd+cool.p\*cool.hd);

G3=HD-(high.p\*high.hd+normal.p\*normal.hd);

G4=HD-(tru.p\*tru.hd+fal.p\*fal.hd);

int gflag=0;

System.out.println("\nGain(D,Outlook)= "+G1 +"\nGain(D,Temperature)= "+G2+"\nGain(D,Humidity)= "+G3+"\nGain(D,Windy)= "+G4);

System.out.print("\nThe Splitting factor is ");

if(G1>G2 && G1>G3 && G1>G4)

System.out.println("Outlook");

else

if(G2>G1 && G2>G3 && G2>G4)

System.out.println("Temperature");

else

if(G3>G2 && G3>G1 && G3>G4)

System.out.println("Humidity");

else

if(G4>G2 && G4>G3 && G4>G1)

System.out.println("Windy");

System.out.println("\nThe Tree is as follows:-");

System.out.print("\nOUTLOOK(S)");

compute('S');

++row;

column=0;

System.out.print("\n\n\nOUTLOOK(O)");

compute('O');

++row;

column=0;

System.out.print("\n\n\nOUTLOOK(R)");

compute('R');

char input[]=new char[4];

String s;

System.out.println("\nMenu:\n\nOutlook: Sunny=S Overcast=O Rainy=R\n\nTemperature: Hot=B Medium=M Cool=C\n\nHumidity: High=H Normal=N\n\nWindy: True=T False=F");

System.out.println("\n\nEnter your new tuple to be classified ");

System.out.print("\nOutlook(S/O/R)= ");

s=scr.nextLine();

input[0]=s.charAt(0);

System.out.print("\nTemperature(B/M/C)= ");

s=scr.nextLine();

input[1]=s.charAt(0);

System.out.print("\nHumidity(H/N)= ");

s=scr.nextLine();

input[2]=s.charAt(0);

System.out.print("\nWindy(T/F)= ");

s=scr.nextLine();

input[3]=s.charAt(0);

/\* Display classify

for(int i=0;i<9;++i)

{

for(int j=0;j<9;++j)

{

System.out.print(" "+classify[i][j]);

}

System.out.println();

}

\*/

System.out.print("Your input is ");

for(int i=0;i<4;++i)

System.out.print(" "+input[i]);

int inc1=0;

do

{

if(input[0]==classify[inc1][0])

{

if(classify[inc1][1]=='P')

{

System.out.println("\n\nTuple classified as PLAY");

break;

}

else

if(input[1]==classify[inc1][1] || input[2]==classify[inc1][1] || input[3]==classify[inc1][1])

{

if(classify[inc1][2]=='P')

{

System.out.println("\n\nTuple classified as PLAY");

break;

}

else

if(classify[inc1][2]=='N')

{

System.out.println("\n\nTuple classified as NO PLAY");

break;

}

}

}

++inc1;

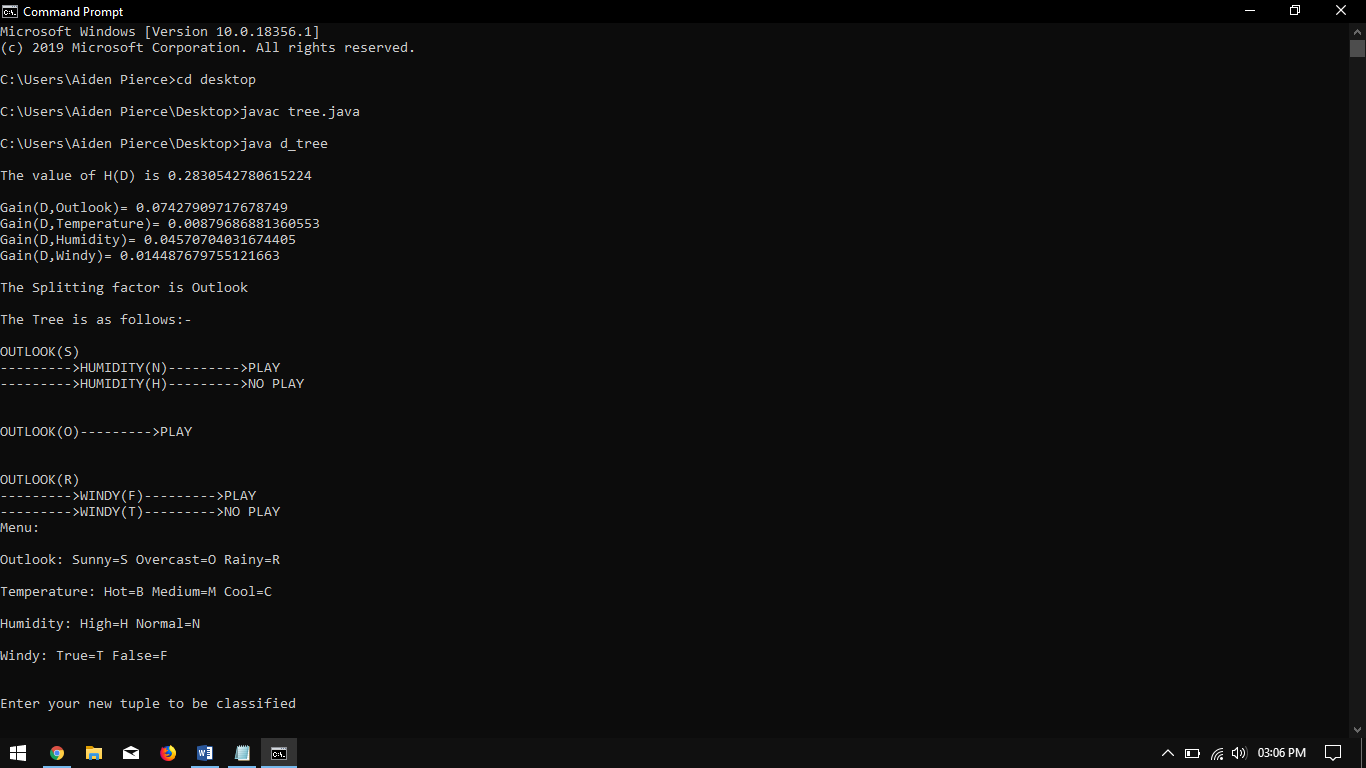
}

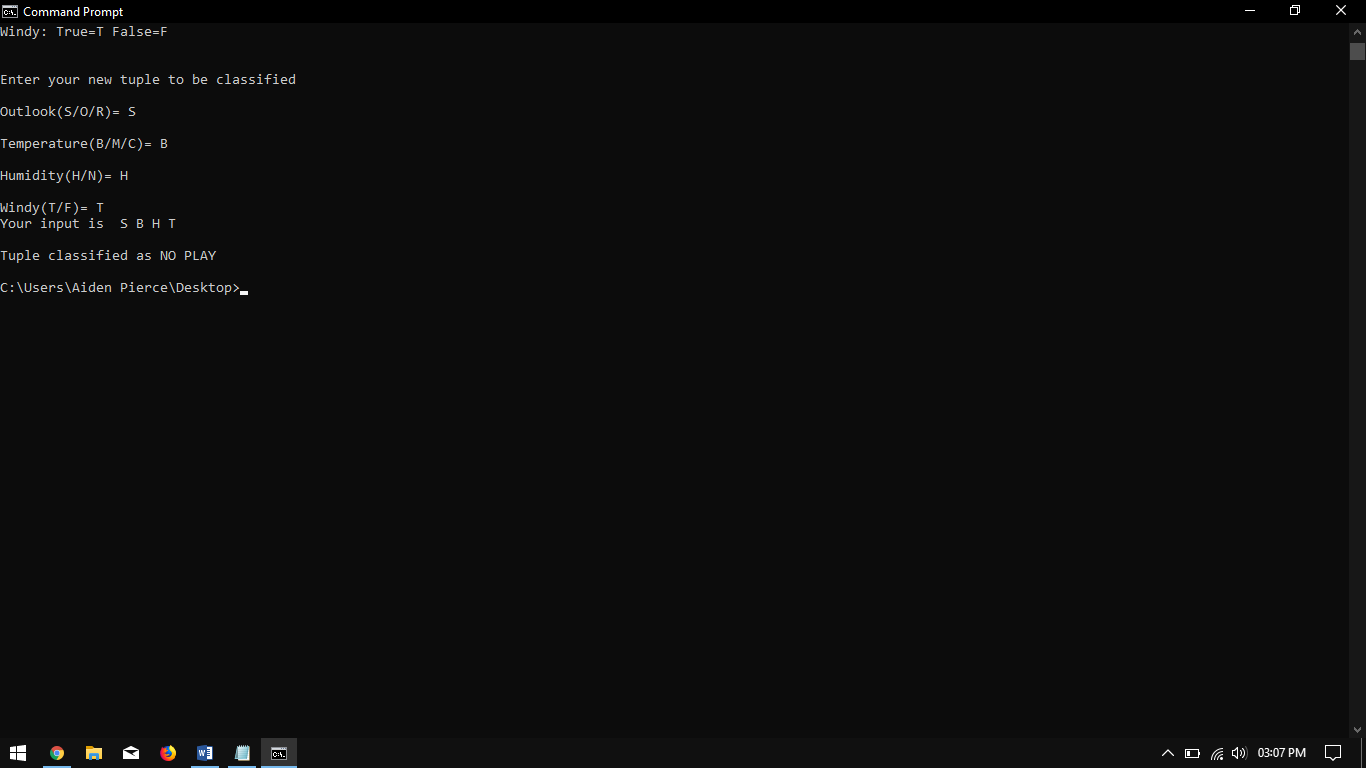
while(true);

}

}

**Output:**





**Conclusion**: Thus we have studied Decision Tree for classification.

**Experiment – 4**

**Title: Implementation of Classification Algorithm (Bayesian).**

### Algorithm–

1. START
2. Store the training dataset
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

Calculate actual probabilitie

### Input :

Training data set

|  |  |  |  |
| --- | --- | --- | --- |
| pid | Name | Gender | Height |
| 1 | A | Female | 1.2 |
| 2 | B | Male | 1.3 |
| 3 | C | Female | 1.65 |
| 4 | P | Male | 1.95 |
| 5 | Q | Male | 1.8 |

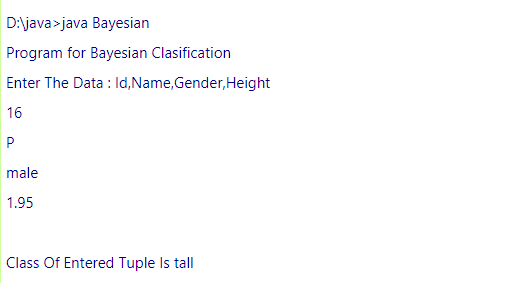
### Code:

|  |
| --- |
|  |

### import java.io.\*; import java.sql.\*; import java.math.\*; class Bayesian  { public static void main(String[] args)  { String name,egender,gender,temph,classn; classn = null;name=egender=null; float height=0;int eid=0; System.out.println("Program for Bayesian Clasification"); System.out.println("Enter The Data : Id,Name,Gender,Height"); try { BufferedReader in = new BufferedReader(new InputStreamReader(System.in)); temph = in.readLine(); eid = Integer.parseInt(temph); name = in.readLine(); egender = in.readLine(); temph = in.readLine(); height = Float.parseFloat(temph); }catch (Exception e ){} try { Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); Connection connect = DriverManager.getConnection("jdbc:odbc:person"); Statement sest = connect.createStatement(); float countms,countmm,countmt,countfs,countfm,countft; countms=countmm=countmt=countfs=countfm=countft=0; gender= "male"; ResultSet rs = sest.executeQuery("SELECT \* FROM personal where gender like '"+gender+"'"); while(rs.next()) { float temp = rs.getFloat(4); if(temp<=1.8) countms++; else if(temp>1.99) countmt++; else countmm++; } gender="female"; ResultSet rs1 = sest.executeQuery("SELECT \* FROM personal where gender like '"+gender+"'"); while(rs1.next()) { float temp = rs1.getFloat(4); if(temp<=1.71) countfs++; else if(temp>1.99) countft++; else countfm++; } float probms,probmm,probmt,probfs,probfm,probft; probms=probmm=probmt=probfs=probfm=probft=0; probms = countms/(countms+countfs);// use this  probmm = countmm/(countmm+countfm); probmt = countmt/(countmt+countft); probfs = countfs/(countms+countfs); probfm = countfm/(countmm+countfm); probft = countft/(countmt+countft); ResultSet rs2 = sest.executeQuery("SELECT \* FROM personal "); float s1,m1,t1,s2,m2,t2,s3,m3,t3,s4,m4,t4,s5,m5,t5,s6,m6,t6; s1=m1=t1=s2=m2=t2=s3=m3=t3=s4=m4=t4=s5=m5=t5=s6=m6=t6=0; while(rs2.next()) { float temp1=rs2.getFloat(4); if(temp1<=1.61) s1++; else if(temp1>1.61&&temp1<=1.71) s2++; else if(temp1>1.71&&temp1<=1.81) m3++; else if(temp1>1.81&&temp1<=1.91) m4++; else if(temp1>1.91&&temp1<=1.96) m5++; else if(temp1>1.96&&temp1<=2.0) t5++; else if(temp1>2.0) t6++; } float ps1,pm1,pt1,ps2,pm2,pt2,ps3,pm3,pt3,ps4,pm4,pt4,ps6,pm6,pt6,ps5,pm5,pt5; ps1=pm1=pt1=ps2=pm2=pt2=ps3=pm3=pt3=ps4=pm4=pt4=ps6=pm6=pt6=ps5=pm5=pt5=0; ps1 = s1/(s1+s2+s3+s4+s5+s6); ps2 = s2/(s1+s2+s3+s4+s5+s6); ps3 = s3/(s1+s2+s3+s4+s5+s6); ps4 = s4/(s1+s2+s3+s4+s5+s6); ps5 = s5/(s1+s2+s3+s4+s5+s6); ps6 = s6/(s1+s2+s3+s4+s5+s6); pm1 = m1/(m1+m2+m3+m4+m5+m6); pm2 = m2/(m1+m2+m3+m4+m5+m6); pm3 = m3/(m1+m2+m3+m4+m5+m6); pm4 = m4/(m1+m2+m3+m4+m5+m6); pm5 = m5/(m1+m2+m3+m4+m5+m6); pm6 = m6/(m1+m2+m3+m4+m5+m6); pt1 = t1/(t1+t2+t3+t4+t5+t6); pt2 = t2/(t1+t2+t3+t4+t5+t6); pt3 = t3/(t1+t2+t3+t4+t5+t6); pt4 = t4/(t1+t2+t3+t4+t5+t6); pt5 = t5/(t1+t2+t3+t4+t5+t6); pt6 = t6/(t1+t2+t3+t4+t5+t6); float pshort,ptall,pmedium; pshort=pmedium=ptall=0; gender = "short"; ResultSet rs4 = sest.executeQuery("SELECT \* FROM personal where class like '"+gender+"'"); while(rs4.next()){ pshort++; } gender = "medium"; rs4 = sest.executeQuery("SELECT \* FROM personal where class like '"+gender+"'"); while(rs4.next()){ pmedium++; } gender="tall"; rs4 = sest.executeQuery("SELECT \* FROM personal where class like '"+gender+"'"); while(rs4.next()){ ptall++; } float total; total=pshort+pmedium+ptall; pshort = pshort/total;pmedium = pmedium/total;ptall = ptall/total; float ptgivens,ptgivenm,ptgivent; ptgivens=ptgivenm=ptgivent=0; if(egender.equals("male")) {  if(height<=1.61) { ptgivens = probms\*ps1; ptgivenm = probmm\*pm1; ptgivent = probmt\*pt1; } else if(height>1.61&&height<=1.71) { ptgivens =probms\*ps2; ptgivenm = probmm\*pm2; ptgivent = probmt\*pt2

}  
else  
if(height>1.71&&height<=1.81)  
{  
ptgivens = probms\*ps3;   
ptgivenm = probmm\*pm3;  
ptgivent = probmt\*pt3;  
}  
else  
if(height>1.81&&height<=1.91)  
{  
ptgivens = probms\*ps4;  
ptgivenm = probmm\*pm4;  
ptgivent = probmt\*pt4;  
}  
else  
if(height>1.91&&height<=2.0)  
{  
ptgivens = probms\*ps5;  
ptgivenm = probmm\*pm5;  
ptgivent = probmt\*pt5;  
}  
else  
if(height>2.0)  
{  
ptgivens = probms\*ps6;  
ptgivenm = probmm\*pm6;  
ptgivent = probmt\*pt6;  
}  
float pls,plm,plt,ptotal;  
pls=plm=plt=ptotal=0;  
pls=ptgivens\*pshort;  
plm=ptgivenm\*pmedium;  
plt=ptgivent\*ptall;  
ptotal= pls+plm+plt;  
float psgivent,pmgivent,ptagivent;  
psgivent=pmgivent=ptagivent=0;  
psgivent = pls/ptotal;  
pmgivent = plm/ptotal;  
ptagivent= plt/ptotal;  
  
if(psgivent>pmgivent&&psgivent>ptagivent)  
classn="short";  
else  
if(pmgivent>psgivent&&pmgivent>ptagivent)  
classn="medium";  
else  
classn="tall" ;}  
else  
if(egender.equals("female"))  
{   
if(height<=1.61)  
{  
ptgivens = probfs\*ps1;  
ptgivenm = probfm\*pm1;  
ptgivent = probft\*pt1;  
}  
else  
if(height>1.61&&height<=1.71)  
{  
ptgivens = probfs\*ps2;  
ptgivenm = probfm\*pm2;  
ptgivent = probft\*pt2;  
}  
else  
if(height>1.71&&height<=1.81)  
{  
ptgivens = probfs\*ps3;   
ptgivenm = probfm\*pm3;  
ptgivent = probft\*pt3;  
}  
else  
if(height>1.81&&height<=1.91)  
{  
ptgivens = probfs\*ps4;  
ptgivenm = probfm\*pm4;  
ptgivent = probft\*pt4;  
}  
else  
if(height>1.91&&height<=2.0)  
{  
ptgivens = probfs\*ps5;  
ptgivenm = probfm\*pm5;  
ptgivent = probft\*pt5;}  
else  
if(height>2.0)  
{  
ptgivens = probfs\*ps6;  
ptgivenm = probfm\*pm6;  
ptgivent = probft\*pt6;  
}  
float pls,plm,plt,ptotal;  
pls=plm=plt=ptotal=0;  
pls=ptgivens\*pshort;  
plm=ptgivenm\*pmedium;  
plt=ptgivent\*ptall;  
ptotal= pls+plm+plt;  
float psgivent,pmgivent,ptagivent;  
psgivent=pmgivent=ptagivent=0;  
psgivent = pls/ptotal;  
pmgivent = plm/ptotal;  
ptagivent= plt/ptotal;  
if(psgivent>pmgivent&&psgivent>ptagivent)  
classn="short";  
else  
if(pmgivent>psgivent&&pmgivent>ptagivent)  
classn="medium";  
else  
classn="tall" ;  
}  
System.out.println("\nClass Of Entered Tuple Is "+classn);  
Statement inst = connect.createStatement();  
inst.executeUpdate("insert into personal values ("+eid+",'" + name + "','" + egender + "'," + height + ",'" + classn + "')");  
connect.close();  
}catch (Exception e) {System.out.println(e);}}}

**Output:**



### Conclusion:

Bayes classification uses probability theory. Hence it is known as a statistical classifier.

# Experiment – 5

**Title:** Implementation of Clustering algorithm ( K-means/ Agglomerative).

**Code:**

|  |
| --- |
|  |

**/\* K-Means Clustering Algorithm \*/**

import java.sql.\*;  
 import java.lang.\*;  
 class clustering

{  
public clustering()  
{ }  
int item[]=new int[30];int fat[]=new int[30];//Main array  
Connection con=null;  
ResultSet rs=null;  
public void getdata()  
{  
int i=0;  
String s1,s2;  
try  
{  
System.out.println("\nAttempting to load JDBC Driver....");  
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
System.out.println("JDBC Driver loaded...");  
System.out.println("Connecting to database...");  
con=DriverManager.getConnection("jdbc:odbc:tester","","");  
System.out.println("Database connection established");  
}//end of try  
  
catch (Exception sqle)  
{System.out.println("Unable to load driver...");}  
try  
{  
String queryString=("SELECT \* FROM food");  
Statement stmt=con.createStatement();  
rs=stmt.executeQuery(queryString);  
while (rs.next())  
{  
s1=rs.getString("Item");  
s2=rs.getString("Fat");  
item[i]=Integer.parseInt(s1.trim());  
fat[i]=Integer.parseInt(s2.trim());  
i++;  
} //end of while  
}//end of try  
catch (SQLException sqle)  
{System.out.println("Some SQL error occured.");}  
try  
{if(con!=null)  
{con.close();}  
System.out.println("Connection to DB closed..Data Retrieved Successfully!");}  
catch(Exception e)  
{}  
}//end of function  
public void cluster()  
{int m1,m2,m3;int i;int d1=0;int d2=0;int d3=0;int a=1;int b=1;int c=1;  
int c11[]=new int[20];int c12[]=new int[20];//Cluster 1  
int c21[]=new int[20];int c22[]=new int[20];//Cluster 2  
int c31[]=new int[20];int c32[]=new int[20];//Cluster 3

c11[0]=item[0];c12[0]=fat[0]; //Randomly place one item in each cluster  
c21[0]=item[1];c22[0]=fat[1];  
c31[0]=item[2];c32[0]=fat[2];  
m1=c12[0];m2=c22[0];m3=c32[0];//Initial Mean value of each cluster  
for(i=3;i<20;i++)  
{  
d1=Math.abs(m1-fat[i]);  
d2=Math.abs(m2-fat[i]);  
d3=Math.abs(m3-fat[i]);  
if(d1<=d2 && d1<=d3)  
{c11[a]=item[i];  
c12[a]=fat[i];  
m1=(c12[a]+m1)/2;a++;}

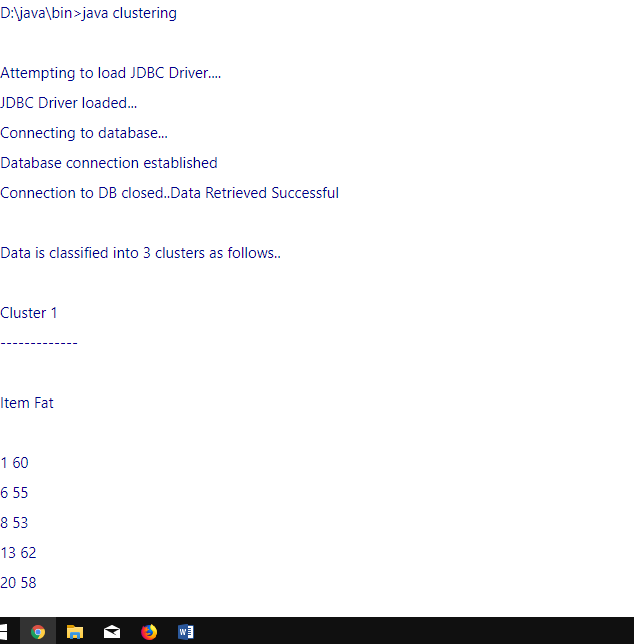
if(d2<=d1 && d2<=d3)  
{c21[b]=item[i];  
c22[b]=fat[i];  
m2=(c22[b]+m2)/2;  
b++;}  
if(d3<=d1 && d3<=d2)  
{c31[c]=item[i];  
c32[c]=fat[i];  
m3=(c32[c]+m3)/2;  
c++;}  
}//end of for...  
System.out.println("\n Data is classified into 3 clusters as follows..");  
System.out.println("\nCluster 1");  
System.out.println("----------");  
System.out.println("\nItem Fat\n");  
for(i=0;i<10;i++)

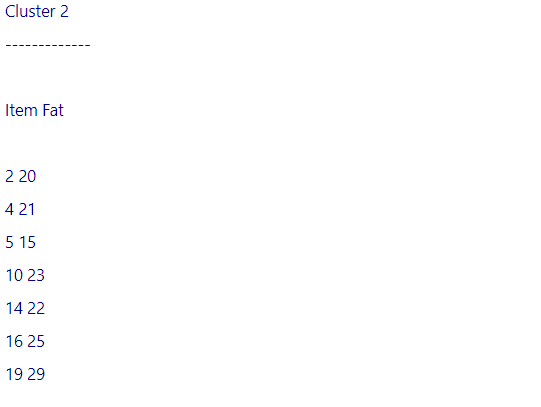
{  
if(c12[i]==0)  
{break;}  
System.out.print(c11[i]);  
System.out.print(" ");  
System.out.print(c12[i]);  
System.out.println("");  
}  
System.out.println("\nCluster 2");  
System.out.println("----------");  
System.out.println("\nItem Fat\n");  
for(i=0;i<10;i++)  
{  
if(c22[i]==0)  
{break;}  
System.out.print(c21[i]);  
System.out.print(" ");  
System.out.print(c22[i]);  
System.out.println("");  
}

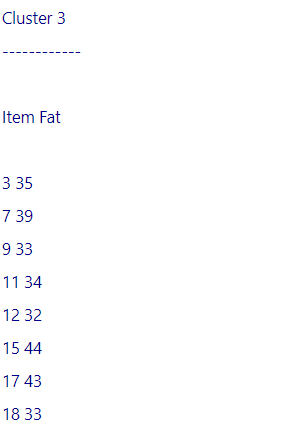
System.out.println("\nCluster 3");  
System.out.println("----------");  
System.out.println("\nItem Fat\n");  
for(i=0;i<10;i++)  
{  
if(c32[i]==0)  
{break;}  
System.out.print(c31[i]);  
System.out.print(" ");  
System.out.print(c32[i]);  
System.out.println("");  
}  
}//end of function  
  
public static void main(String args[])

{  
clustering t=new clustering();  
t.getdata();  
t.cluster();  
}//end of main  
}//end of class

**Output:**





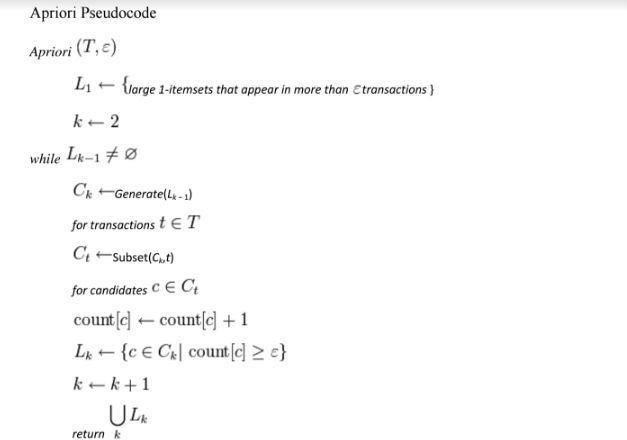


### Conclusion:

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

**Experiment No.6**

**Title:** Implementation of Association Rule Mining algorithm (**Apriori**).



### Program:

### import java.io.\*;

### class apriori

### {

### public static void main(String []arg)throws IOException

### {

### int i,j,m=0;

### int t1=0;

### BufferedReader b=new BufferedReader(new InputStreamReader(System.in));

### System.out.println("Enter the number of transaction :");

### int n=Integer.parseInt(b.readLine());

### System.out.println("items :1--Milk 2--Bread 3--Coffee 4--Juice 5--Cookies 6--Jam");

### int item[][]=new int[n][6];

### for(i=0;i<n;i++)

### for(j=0;j<6;j++)

### item[i][j]=0;

### String[] itemlist={"MILK","BREAD","COFFEE","JUICE","COOKIES","JAM"};

### int nt[]=new int[6];

### int q[]=new int[6];

### for(i=0;i<n;i++)

### { System.out.println("Transaction "+(i+1)+" :");

### for(j=0;j<6;j++)

### { //System.out.println(itemlist[j]);

### System.out.println("Is Item "+itemlist[j]+" present in this transaction(1/0)? :");

### item[i][j]=Integer.parseInt(b.readLine());

### }

### }

### for(j=0;j<6;j++)

### { for(i=0;i<n;i++)

### {if(item[i][j]==1)

### nt[j]=nt[j]+1;

### }

### System.out.println("Number of Item "+itemlist[j]+" :"+nt[j]);

### }

### for(j=0;j<6;j++)

### { if(((nt[j]/(float)n)\*100)>=50)

### q[j]=1;

### else

### q[j]=0;

### if(q[j]==1)

### {t1++;

### System.out.println("Item "+itemlist[j]+" is selected ");

### 

### }

### }

### for(j=0;j<6;j++)

### { for(i=0;i<n;i++)

### {

### 

### if(q[j]==0)

### {

### item[i][j]=0;

### }

### }

### }

### int nt1[][]=new int[6][6];

### for(j=0;j<6;j++)

### { for(m=j+1;m<6;m++)

### { for(i=0;i<n;i++)

### { if(item[i][j]==1 &&item[i][m]==1)

### { nt1[j][m]=nt1[j][m]+1;

### }

### }

### if(nt1[j][m]!=0)

### System.out.println("Number of Items of "+itemlist[j]+"& "+itemlist[m]+" :"+nt1[j][m]);

### }

### 

### }

### for(j=0;j<6;j++)

### { for(m=j+1;m<6;m++)

### {

### if(((nt1[j][m]/(float)n)\*100)>=50)

### q[j]=1;

### else

### q[j]=0;

### if(q[j]==1)

### {

### System.out.println("Item "+itemlist[j]+"& "+itemlist[m]+" is selected ");

### 

### }

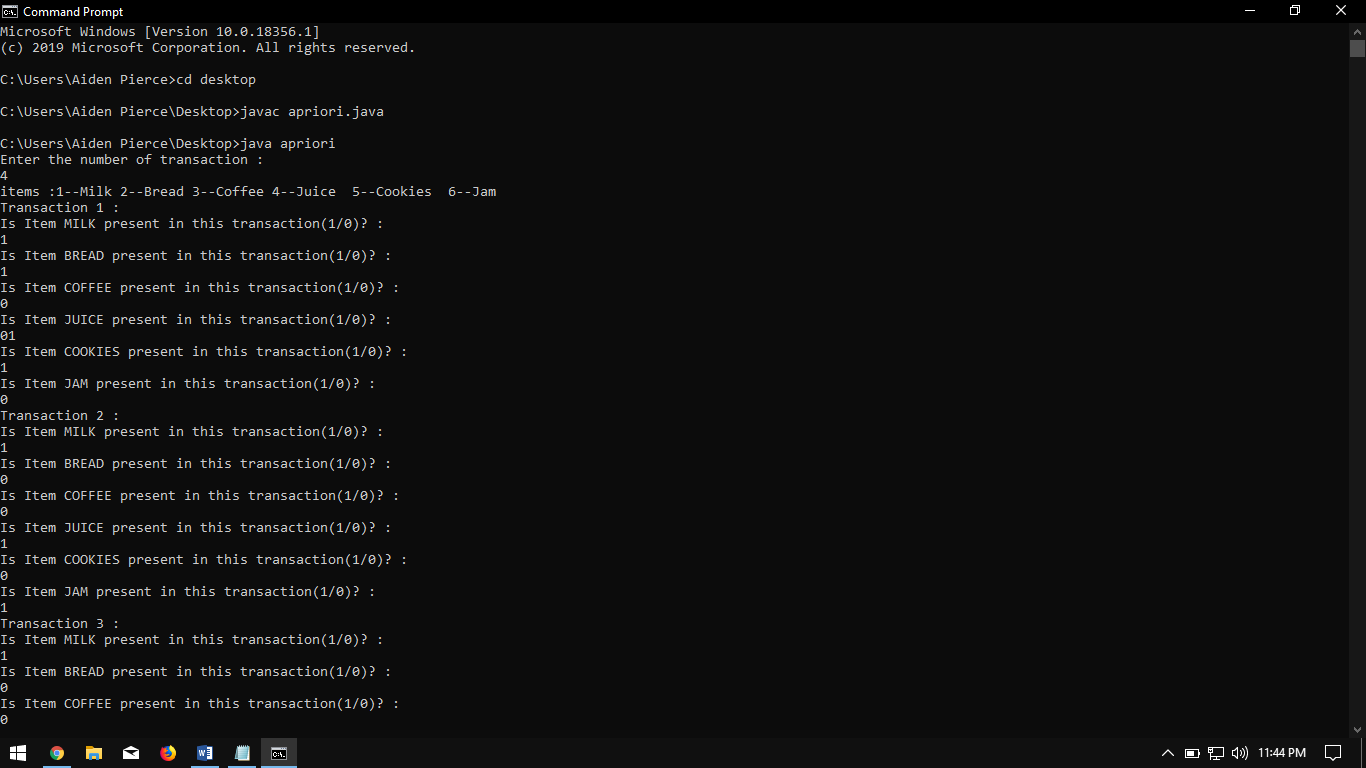
### }

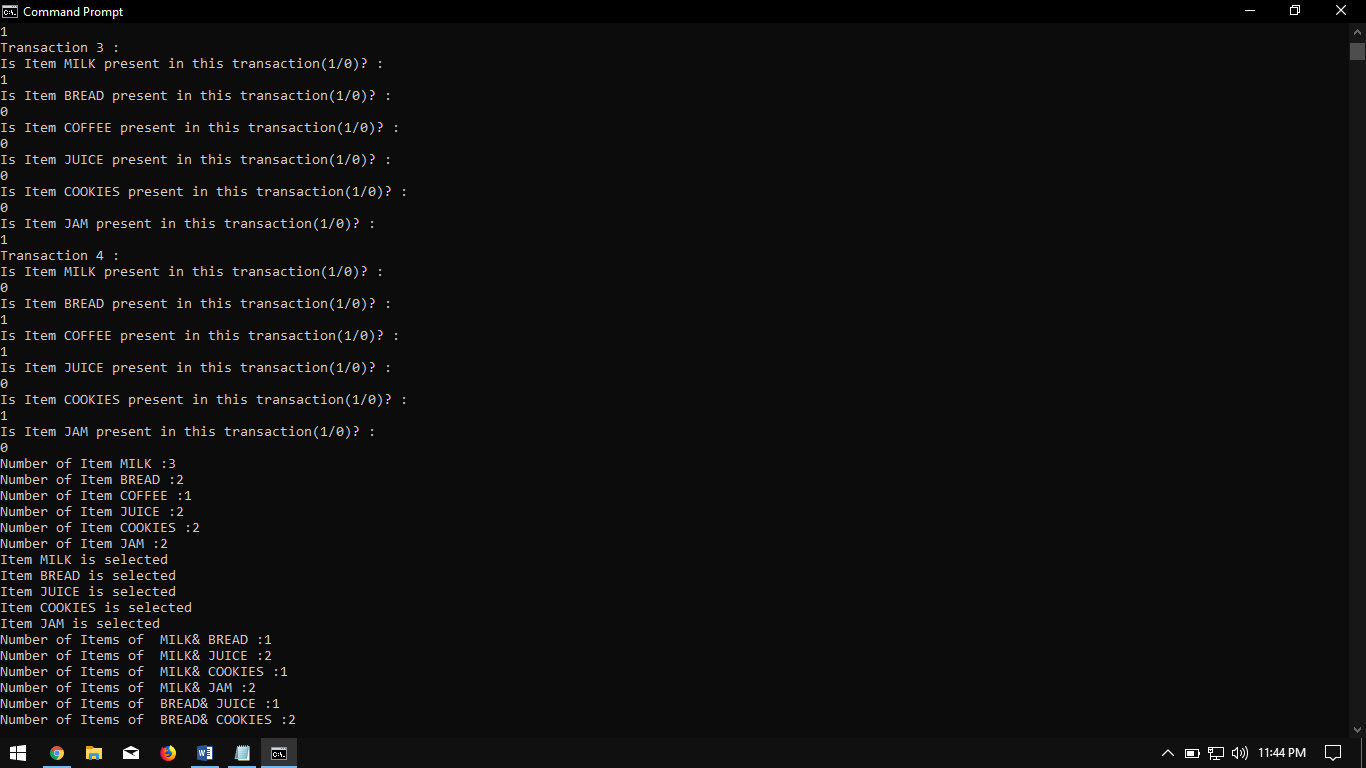
### }

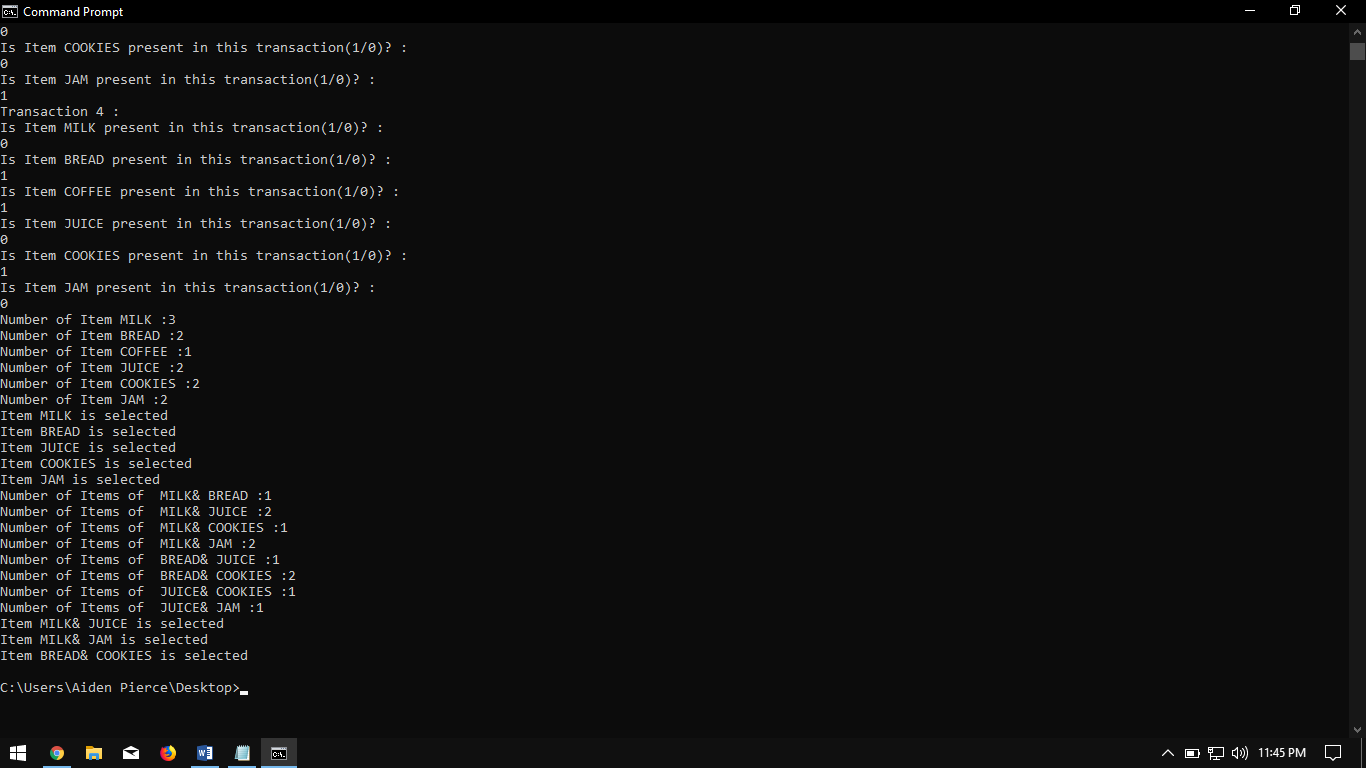
### }

### }

**OutPut**:







**Conclusion:**

Apriori Algorithm works on the principle of frequent item sets, support and confidence. We can generate association rules and mine the basic data from a database.

**Experiment – 7**

**Title:** Introduction to the Weka machine learning toolkit

**Objective:** To learn to use the Weak machine learning toolkit

### References

Witten, Ian and Eibe, Frank. Data Mining: Practical Machine Learning Tools and Techniques. Springer.

### Requirements

**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

### 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 selecting the appropriate file.

**Step2:** Once the data is loaded, weka will recognize the attributes and during the scan of the data weka will compute some basic strategies on each attribute. The left panel in the above figure shows the list of recognized attributes while the top panel indicates the names of the base 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 attributes the frequency of each attribute value is shown, while for continuous attributes we can obtain min, max, mean, standard deviation and deviation etc.,

**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 filter model panel, click on choose button, This will show a popup window with a list of available filters.

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 index of the attribute to be filtered out.

1. Make sure that invert selection option is set to false. The click OK now in the filter box you will see “Remove-R-7”.
2. Click the apply button to apply filter to this data. This will remove the attribute and create new working relation.
3. 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

%

### Press the Explorer button on the main panel and load the weather dataset and answer the following questions

* 1. How many instances are there in the dataset?
  2. State the names of the attributes along with their types and values.
  3. What is the class attribute?
  4. In the histogram on the bottom-right, which attributes are plotted on the X,Y-axes? How do you change the attributes plotted on the X, Y-axes?
  5. How will you determine how many instances of each class are present in the data
  6. What happens with the Visualize All button is pressed?
  7. How will you view the instances in the dataset? How will you save the changes?

### What is the purpose of the following in the Explorer Panel?

1. The Preprocess panel
   1. What are the main sections of the Preprocess panel?
   2. What are the primary sources of data in Weka?
2. The Classify panel
3. The Cluster panel
4. The Associate panel
5. The Select Attributes panel
6. The Visualize panel.

### Load the iris dataset and answer the following questions:

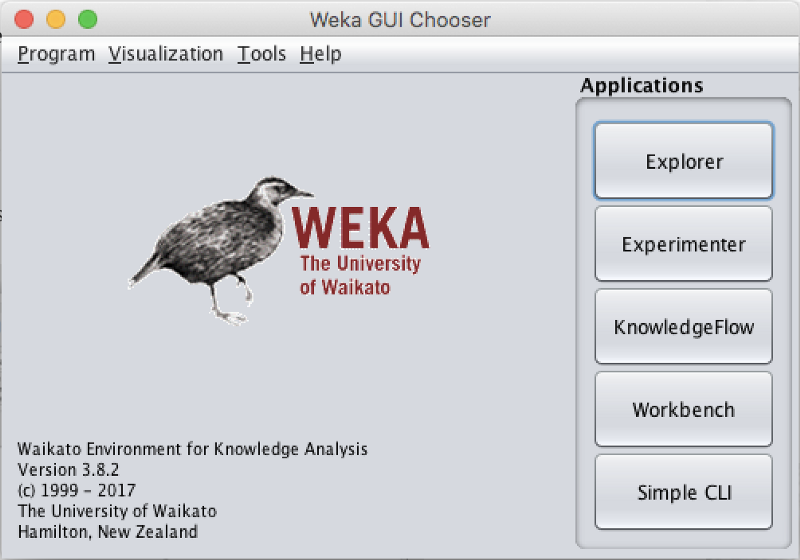
* 1. How many instances are there in the dataset?
  2. State the names of the attributes along with their types and values.
  3. What is the class attribute?
  4. In the histogram on the bottom-right, which attributes are plotted on the X,Y-axes? How do you change the attributes plotted on the X,Y-axes?
  5. How will you determine how many instances of each class are present in the data
  6. What happens with the Visualize All button is pressed?

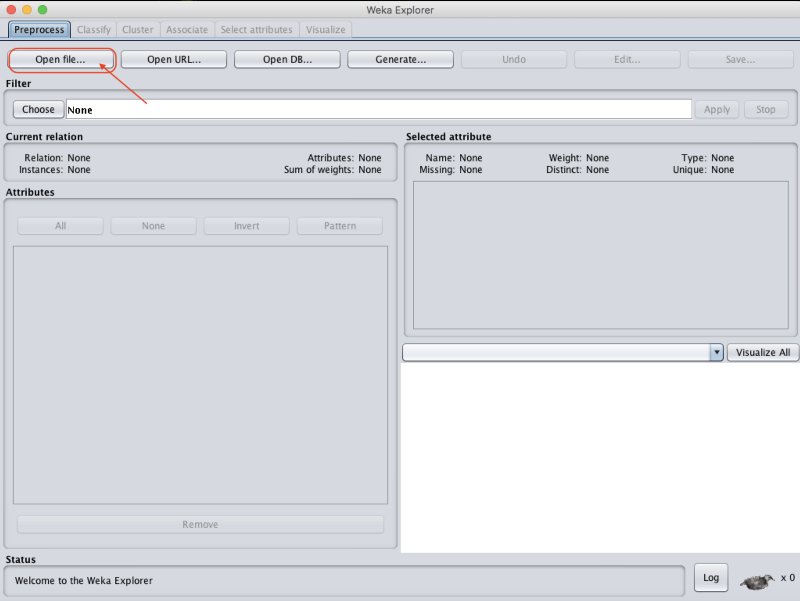
### Load the weather dataset and perform the following tasks:

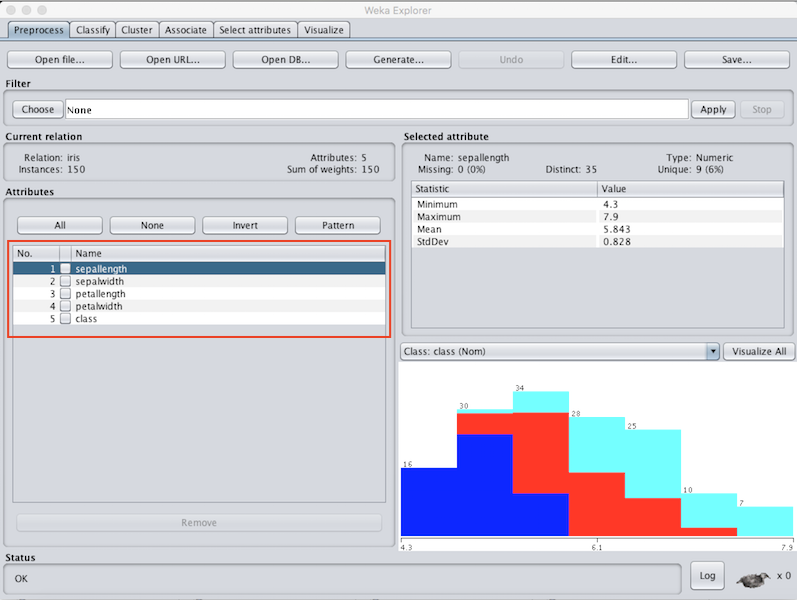
* + 1. Use the unsupervised filter Remove With Values to remove all instances where the attribute ‘humidity ’has the value ‘high’?
    2. Undo the effect of the filter.
    3. Answer the following questions:
       1. What is meant by filtering in Weka?
       2. Which panel is used for filtering a dataset?
       3. What are the two main types of filters in Weka?
       4. What is the difference between the two types of filters? What is the difference between and attribute filter and an instance filter?

### Load the iris dataset and perform the following tasks:

* + 1. Press the Visualize tab to view the Visualizer panel.
    2. What is the purpose of the Visualizer?
    3. Select one panel in the Visualizer and experiment with the buttons on the panel.







### Conclusion:

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

# Experiment – 8

**Title:** Perform data Pre-processing task and Demonstrate performing Classification, Clustering, Association algorithm on data sets using data mining tool (WEKA,R tool, XL Miner, etc.)

# DATA EXPLORATION:

# https://lh3.googleusercontent.com/F22yBBBIVABHZzXxRBhzClaTD6Odk4-bQR3i7Ou7IyFJ-5-HpoXLhP5HhA5khj3OceonQQwMrgmG7TVP5ZJN7_nNWoWKX3FNgS-XFcq694ygUMnHWfylTjI4DSnpR5xCRdaPh0Yd

# https://lh5.googleusercontent.com/0A9PInjd3dJq9Moci_rQEQWKrW9aJZmLNv40tjAVOWC0_oVExO2rB6YJ7U9QkyUT4FIVdspdSlSZxgT5jVq1CHV6HdQ70pZhUMpo9VTMRdTDjO9q1A9LBVce8SUnFsM4kQoh1fJT

**DATA PREPROCESSING:**

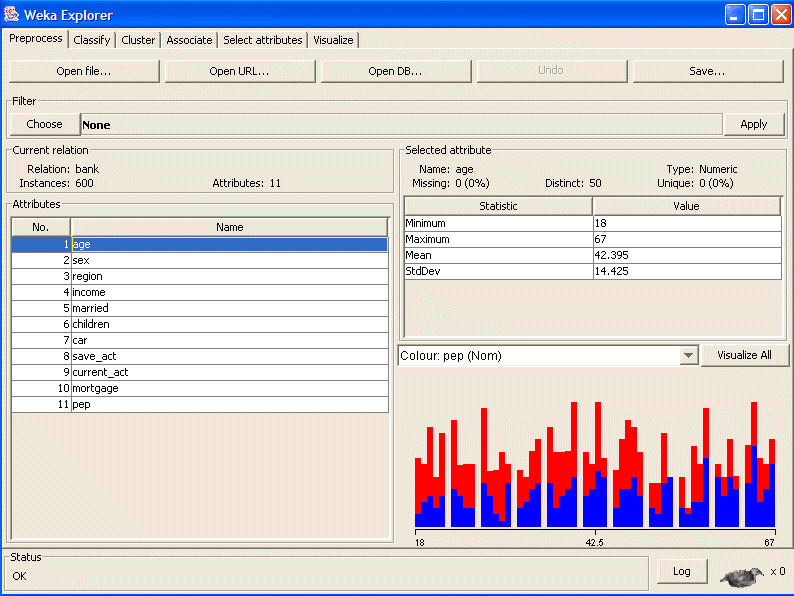
# https://lh5.googleusercontent.com/h2UQWMi60mU6B74hTjN4QozOvUlExvPmmgnP2pqAluNNOmzwAvxhf9ildVa7IOTS4za7x6zf304u4hG0xZ7GFvOLEGL125ps4LyvL9_48-hPaZ450MjV0O4Gs8zKORHXriPz5c31

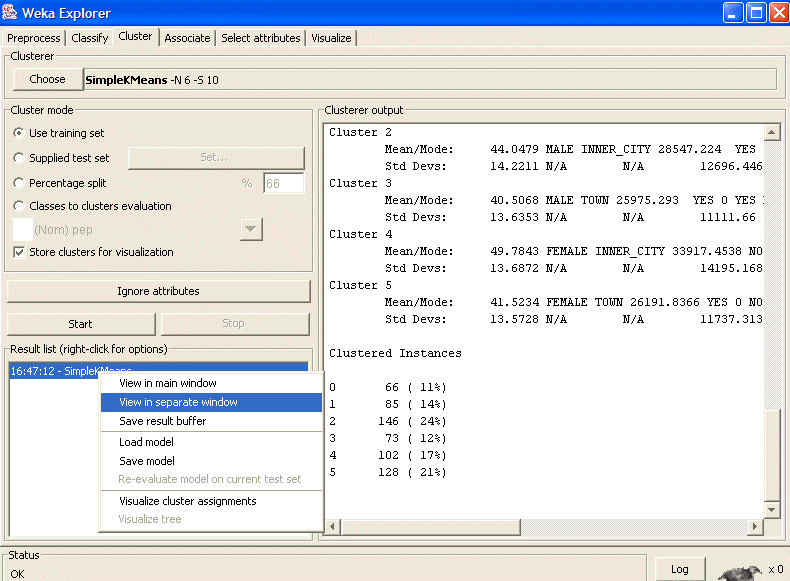
# https://lh6.googleusercontent.com/0X9VKkZvEkj2om-1QJ4ak19dAMmISB5x5VEBrPodnE-qlS8MENIYysfhq6OiYz1l2ZnN9Vdp4edTBzYPwUrcjdHvyf5aVMFcXIadurgZZl-AonVcFSNKFyhwGLJByAm0VCIY9ODw

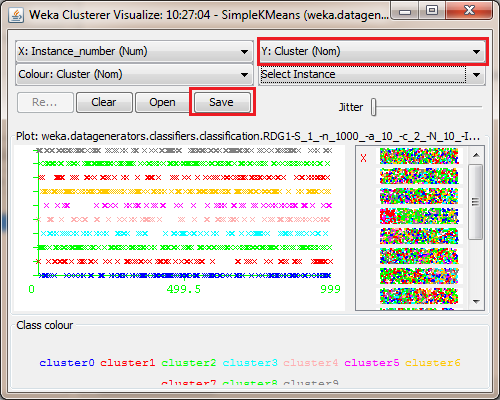
# Classification:

# C:\Users\ace\Desktop\1.pngC:\Users\ace\Desktop\2.pngC:\Users\ace\Desktop\3.pngC:\Users\ace\Desktop\4.png

# Clustering:







**Conclusion:**

Studied data Pre-processing task and Demonstrated performing Classification, Clustering algorithm on data sets using data mining WEKA tool.

**Experiment No.9**

**Title: To implement Page Rank Algorithm.**

**Code**:

import java.util.\*;

import java.io.\*;

public class PageRank {

public int path[][] = new int[10][10];

public double pagerank[] = new double[10];

public void calc(double totalNodes){

double InitialPageRank;

double OutgoingLinks=0;

double DampingFactor = 0.85;

double TempPageRank[] = new double[10];

int ExternalNodeNumber;

int InternalNodeNumber;

int k=1; // For Traversing

int ITERATION\_STEP=1;

InitialPageRank = 1/totalNodes;

System.out.printf(" Total Number of Nodes :"+totalNodes+"\t Initial PageRank of All Nodes :"+InitialPageRank+"\n");

// 0th ITERATION \_ OR \_ INITIALIZATION PHASE

for(k=1;k<=totalNodes;k++)

{

this.pagerank[k]=InitialPageRank;

}

System.out.printf("\n Initial PageRank Values , 0th Step \n");

for(k=1;k<=totalNodes;k++)

{

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

}

while(ITERATION\_STEP<=2) // Iterations

{

// Store the PageRank for All Nodes in Temporary Array

for(k=1;k<=totalNodes;k++)

{

TempPageRank[k]=this.pagerank[k];

this.pagerank[k]=0;

}

for(InternalNodeNumber=1;InternalNodeNumber<=totalNodes;InternalNodeNumber++)

{

for(ExternalNodeNumber=1;ExternalNodeNumber<=totalNodes;ExternalNodeNumber++)

{

if(this.path[ExternalNodeNumber][InternalNodeNumber] == 1)

{

k=1;

OutgoingLinks=0; // Count the Number of Outgoing Links for each ExternalNodeNumber

while(k<=totalNodes)

{

if(this.path[ExternalNodeNumber][k] == 1 )

{

OutgoingLinks=OutgoingLinks+1; // Counter for Outgoing Links

}

k=k+1;

}

// Calculate PageRank

this.pagerank[InternalNodeNumber]+=TempPageRank[ExternalNodeNumber]\*(1/OutgoingLinks);

}

}

}

System.out.printf("\n After "+ITERATION\_STEP+"th Step \n");

for(k=1;k<=totalNodes;k++)

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

ITERATION\_STEP = ITERATION\_STEP+1;

}

// Add the Damping Factor to PageRank

for(k=1;k<=totalNodes;k++)

{

this.pagerank[k]=(1-DampingFactor)+ DampingFactor\*this.pagerank[k];

}

// Display PageRank

System.out.printf("\n Final Page Rank : \n");

for(k=1;k<=totalNodes;k++)

{

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

}

}

public static void main(String args[])

{

int nodes,i,j,cost;

Scanner in = new Scanner(System.in);

System.out.println("Enter the Number of WebPages \n");

nodes = in.nextInt();

PageRank p = new PageRank();

System.out.println("Enter the Adjacency Matrix with 1->PATH & 0->NO PATH Between two WebPages: \n");

for(i=1;i<=nodes;i++)

for(j=1;j<=nodes;j++)

{

p.path[i][j]=in.nextInt();

if(j==i)

p.path[i][j]=0;

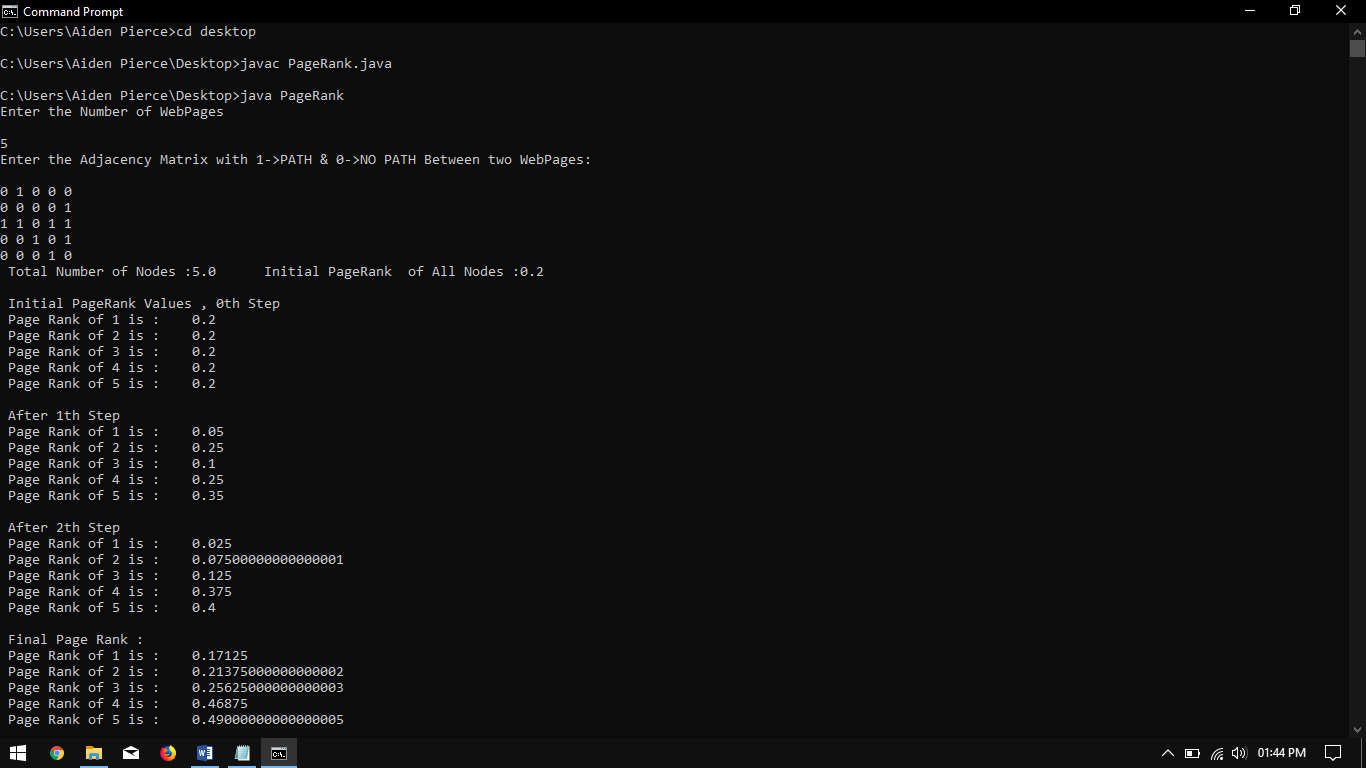
}

p.calc(nodes);

}

}

**Output:**



# Conclusion: Thus we have studied Page Rank Algorithm