Data Mining Laboratory

Part-B(Java Programs)

1.Write a java program to perform aggregation, discretization and stratified sampling on a given data set.

```
import java.io. *;

class Records
{
    int index,attr1;
    String attr2;
}
public class prog1
{
        static Records[] rc=new Records[10];
        public static void main(String args[]) throws
FileNotFoundException,IOException
        {
        String path = "/D:/csvfiles/data.csv";
        BufferedReader CSV=new BufferedReader(new FileReader(new
File(path)));
                String data=CSV.readLine();
                int i=0,min = Integer.MAX_VALUE,max = Integer.MIN_VALUE;
System.out.println("Dataset:");
                while(data!=null)
                {
                        rc[i]=new Records();
                        String[] dataArray=data.split(",");
                        rc[i].index=Integer.parseInt(dataArray[0]);
                        rc[i].attr1=Integer.parseInt(dataArray[1]);
                        rc[i].attr2=dataArray[2];
                        if(rc[i].attr1 > max)
                        max=rc[i].attr1; if(rc[i].attr1 < min)
                        min=rc[i].attr1;
        System.out.println(rc[i].index+" "+rc[i].attr1+" "+rc[i].attr2);
                        data=CSV.readLine();
                        i++;
                }

      int avg =0;
```

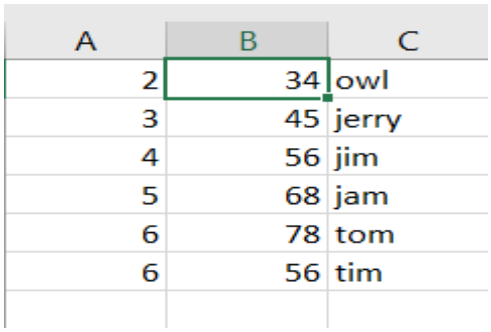
```
        for(int j=0;j<i;j++)
                avg+= rc[j].attr1;
        avg= avg/i;
                System.out.println("max value :"+max+"\tmin value:"+min);
                System.out.println("Average value is: "+avg);

            int mean = (min + max) / 2;
            int mid1 = (min + mean) / 2;
            int mid2 = (mean + max) / 2;
            int sampling[] = new int[4];
                for(int j=0; j<i; j++)
                {
      System.out.print(rc[j].index+" "+rc[j].attr1+" "+rc[j].attr2);
                if(rc[j].attr1 >= min && rc[j].attr1 < mid1)
                {
                System.out.println(" ["+min+"-"+mid1+")");
                sampling[0]=rc[j].attr1; }
                else if(rc[j].attr1 >= mid1 && rc[j].attr1 < mean)
                {
                System.out.println(" ["+mid1+"-"+mean+")");

                sampling[1] =rc[j].attr1; }
                else if(rc[j].attr1 >= mean && rc[j].attr1 < mid2)
                {
                System.out.println(" ["+mean+"-"+mid2+")");
                sampling[2] =rc[j].attr1; }
                else if(rc[j].attr1 >= mid2 && rc[j].attr1 <= max) {
                System.out.println(" ["+mid2+"-"+max+")");
                sampling [3] =rc[j].attr1; }
                            }

                System.out.println("----sampling-------");
                System.out.println(" ["+min+"-"+mid1+") -"+sampling [0]);
                System.out.println(" ["+mid1+"-"+mean+") -"+sampling [1]);
                System.out.println(" ["+mean+"-"+mid2+") -"+sampling [2]);
                System.out.println(" ["+mid2+"-"+max+") -"+sampling [3]);
                CSV.close();
        }
}
```

Input:



A	B	C
2	34	owl
3	45	jerry
4	56	jim
5	68	jam
6	78	tom
6	56	tim

Output:

Dataset:
2 34 owl
3 45 jerry
4 56 jim
5 68 jam
6 78 tom
6 56 tim
max value :78 min value:34
Average value is: 56
2 34 owl [34-45)
3 45 jerry [45-56)
4 56 jim [56-67)
5 68 jam [67-78)
6 78 tom [67-78)
6 56 tim [56-67)
----sampling-------
[34-45) -34
[45-56) -45
[56-67) -56
[67-78) -78

2.Write a java program to handle missing values

a. Replacing by the mean for numeric attributes.

b. Replace by the value that occurs the maximum number of times for categorical attribute

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class GFG {

	Scanner in=new Scanner(System.in);
	ArrayList<String[]> data;

	private void loadFile(String fileName) {
		try {
			data=new ArrayList<String[]>();

			String ipLine;
			BufferedReader br=new BufferedReader(new FileReader(new
File(fileName)));
			ipLine=br.readLine();
			while(ipLine!=null){

				String[] rowData=ipLine.split(",");
				ipLine=br.readLine();
				data.add(rowData);
			}

		} catch (FileNotFoundException e) {
			System.out.println("No such file found!");
			System.exit(0);
		} catch (IOException e) {
			e.printStackTrace();
		}


	}

	public void missingString(int col, String defValue) {
		HashMap<String, Integer> hm=new HashMap<String,Integer>();
		for(String[] row: data) {
			if(!row[col].equalsIgnoreCase(defValue)) {
				hm.put(row[col], hm.getOrDefault(row[col],0)+1);
```

```
                }
        }

        String maxValName = null;
        int maxVal=0;
        for(Map.Entry<String, Integer> me:hm.entrySet()) {
                if(me.getValue()>maxVal)
                        maxValName=me.getKey();
        }

        for(String[] row: data) {
                if(row[col].equalsIgnoreCase(defValue)) {
                        row[col]=maxValName;
                }
        }


}

public void missingInteger(int col, String defValue) {
        float avg=0;
        int count=0;
        for(String[] row: data) {
                if(!row[col].equalsIgnoreCase(defValue)) {
                        avg+=Float.parseFloat(row[col]);
                }
                count++;
        }
        avg=avg/count;
        for(String[] row: data) {
                if(row[col].equalsIgnoreCase(defValue)) {
                        row[col]=Integer.toString((int)avg);
                }
        }

}
public void generateFile() {
        try {
                FileWriter fw=new FileWriter(new File("output.csv"));
                for(String[] rowData:data) {
                        String newData=Arrays.toString(rowData);
                        newData=newData.substring(1,newData.length()-1);
                        fw.write(newData);
                        fw.write('\n');
                }

                fw.close();
                System.out.println("\nFile generated successfully!");

        } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }
```

```
	}

	public static void main(String[] args) {
		GFG mc=new GFG();
		mc.loadFile("/D:/csvfiles/data.csv");
		mc.missingString(5, "NA");
		mc.missingInteger(1, "NA");
		mc.generateFile();

	}

}
```

Input:

12	12	1157	female	491.143	SEP
2	16	1188	female	986.16	NOV
3	19	805	female	600.899	MAR
4	16	1240	male	779.786	NOV
5	NA	1240	male	456.12	MAR
6	18	609	male	887.106	JAN
7	15	558	male	969.761	OCT
8	NA	584	female	881.86	DEC
9	19	743	male	297.997	OCT
10	18	579	male	969.337	NA
11	13	762	male	621.146	FEB
12	16	865	female	416.887	JAN
13	12	622	male	407.317	NOV
14	10	671	female	538.424	APR
15	20	598	male	710.449	AUG
16	11	871	female	969.586	JUL
17	14	943	female	844.26	OCT
18	16	1133	male	371.034	SEP
19	16	569	male	200.232	MAY
20	10	901	male	170.367	MAR
21	NA	827	male	836.544	MAR
22	18	1004	male	309.653	DEC
23	16	752	male	898.725	MAR
24	18	528	female	427.456	NA
25	NA	1176	female	558.649	NA

Output:

1	12	1157	female	491.143	SEP
2	16	1188	female	986.16	NOV
3	19	805	female	600.899	MAR
4	16	1240	male	779.786	NOV
5	12	1240	male	456.12	MAR
6	18	609	male	887.106	JAN
7	15	558	male	969.761	OCT
8	12	584	female	881.86	DEC
9	19	743	male	297.997	OCT
10	18	579	male	969.337	MAR
11	13	762	male	621.146	FEB
12	16	865	female	416.887	JAN
13	12	622	male	407.317	NOV
14	10	671	female	538.424	APR
15	20	598	male	710.449	AUG
16	11	871	female	969.586	JUL
17	14	943	female	844.26	OCT

3.Write a java program to identify the frequent subsets, generate strong rules from a frequent 4-itemset

```
import java.util.*;


    class p3 {
    static boolean check(String x1,String x2) {
    x2 = x2.replace("", ".*");
              if(x1.matches(x2))
                  return true;
                else
               return false;
              }
    public static void main(String[] args) throws
    IOException,FileNotFoundException
              {
    BufferedReader csv = new BufferedReader(new FileReader(new
    File("/D:/csvfiles/input.csv")));
      String data = csv.readLine();
      HashSet<String> hs = new HashSet<>();
      ArrayList<String> al = new ArrayList<>();
      ArrayList<String> bl = new ArrayList<>();
      ArrayList<String> cl = new ArrayList<>();
      double support = 0.4, confidence=0.5;
                  while(data != null)
                  {
```

```
                String dataarray [] = data.split(",");
                String temp1="";
                for(String x:dataarray)
                {
                        hs.add(x);
                    temp1=temp1+x;
                }
                bl.add(temp1);
                data = csv.readLine();
            }
            String d[] = hs.toArray(new String[hs.size()]);
            int n = d.length;
            // generate all possible subset
            for(int i=0;i < (1<<n); i++)
            {
                String temp="";
                for(int j=0;j<n;j++)
                   if(( i & (1<<j))>0)
                     temp = temp+d[j];
                al.add(temp);
            }

            for (int i=1;i<=4;i++)
            {
           System.out.println("\n Frequent "+i+"-itemset");
           for(String y:al)
                if (i == y.length())
                {
                    double count = 0;
                    for(String x:  b1)
                        if(check(x, y))
                              count++;
                    if(count/bl.size() >= support)
                    {
                        if (i == 4)
                        cl.add(y);
            System.out.println(y + " - >"+count/bl.size());
                    }
                }

                }
System.out.println("\n----------Strong rules--------\n");

                for(String p:cl)
                {
System.out.println("\n-----For string "+p+"-----------");
```

```
                    char[] c = p.toCharArray();
                    n = c.length;
                    for(int i=0; i < (1<<n); i++)
                    {
                        String temp3=” ”, temp4="";
                        for(int j=0; j<n;j++)
                          {
                            If (( i & (1<<j))>0)
                                temp3 = temp3+c[j];
                              else
                                temp4 = temp4+c[j];
                          }
     if (temp3.length() !=0 && temp3.length() != 4)
                {
     double count1=0, count2=0;
      for(String x: bl)
{
                                                                        if(check(x,
p))

        count1++;
                                                                        if(check(x,
temp3))

        count2++;
}
                                                                        if (count2
> 0 && (count1/count2) >= confidence) System.out.println(temp3+"-
>"+temp4+"confidence: "+count1/count2);
                              }
                     }
                }
        }
}
```

Input:

a	b	c		
a	b	c	d	e
a	c	d		
a	c	d	e	
a	b	c	d	

Output:

Frequent 1-itemset
a ->1.0
b ->0.6
c ->1.0
d ->0.8
e ->0.4

Frequent 2-itemset
ab ->0.6
ac ->1.0
bc ->0.6
ad ->0.8
bd ->0.4
cd ->0.8
ae ->0.4
ce ->0.4
de ->0.4

Frequent 3-itemset
abc ->0.6
abd ->0.4
acd ->0.8
bcd ->0.4
ace ->0.4
ade ->0.4
cde ->0.4

Frequent 4-itemset
abcd ->0.4
acde ->0.4

----------Strong rules----------

-----For string abcd----------
b->acd confidence: 0.6666666666666666
ab->cd confidence: 0.6666666666666666
bc->ad confidence: 0.6666666666666666
abc->d confidence: 0.6666666666666666
d->abc confidence: 0.5
ad->bc confidence: 0.5
bd->ac confidence: 1.0
abd->c confidence: 1.0
cd->ab confidence: 0.5
acd->b confidence: 0.5
bcd->a confidence: 1.0

-----For string acde----------
d->ace confidence: 0.5
ad->ce confidence: 0.5
cd->ae confidence: 0.5
acd->e confidence: 0.5
e->acd confidence: 1.0
ae->cd confidence: 1.0
ce->ad confidence: 1.0
ace->d confidence: 1.0
de->ac confidence: 1.0
ade->c confidence: 1.0
cde->a confidence: 1.0

4.Write a java program to implement the information gain and gini index measure to identify the best attribute to split.

```
import java.util.*;

import java.io. *;



        public class GiniIndexHandler {

        public ArrayList<String[]> csvLines;

        public void readCSV(String filename, String delimeter) {
                try {
                        csvLines = new ArrayList<>();
        BufferedReader br = new BufferedReader(new FileReader(new
        File(filename)));
                        String line;
                        while((line = br.readLine())!=null) {
                        String[] currentLine = line.split(delimeter);
                           csvLines.add(currentLine);
                           }
                           } catch (Exception e) {
                                   e.printStackTrace();
                           }

                }
                public void printCSV(){
                        for(String[] currentLine : csvLines) {
                                for(String s : currentLine) {
                                        System.out.print(s + ", ");
                                }
                                System.out.println();
```

```
		}
	}
	private double calculateGiniIndex(int col) {
		double giniIndexValue = 0;
		int numberOfColumns = csvLines.get(0).length;
	HashMap<String, ArrayList<String[]>> hm = new
HashMap<>();

		for(String[] currentLine : csvLines) {
			String key = currentLine[col];
			if(hm.containsKey(key)) {
				ArrayList<String[]> arr = hm.get(key);
				arr.add(currentLine);
				hm.put(key, arr);
			}
			else {
			ArrayList<String[]> arr = new ArrayList<>();
				arr.add(currentLine);
				hm.put(key, arr);
			}
		}

	ArrayList<Double> nodesGiniIndexValues = new ArrayList<>();
for(Map.Entry<String, ArrayList<String[]>> m : hm.entrySet()) {
			ArrayList<String[]> currentNode = m.getValue();
			int size = currentNode.size();

			double numberOfYes=0, numberOfNo=0;
			for(String[] currentLine : currentNode) {
			if(currentLine[numberOfColumns-1].equals("Yes")) {
					numberOfYes++;
				}
				else {
					numberOfNo++;
				}
			}

			double currentNodeGiniValue;
currentNodeGiniValue = 1 - (numberOfYes*numberOfYes)/(size*size) -
(numberOfNo*numberOfNo)/(size*size);
			currentNodeGiniValue  =
currentNodeGiniValue*currentNode.size();
			nodesGiniIndexValues.add(currentNodeGiniValue);
		}

		for(Double d : nodesGiniIndexValues) {
```

```
                giniIndexValue += d/csvLines.size();
            }

            return giniIndexValue;
        }



        public void findBestSplit() {
            int numberOfColumns = csvLines.get(0).length;
            int columnToSplit = -1;
            double minGiniIndexValue = 999;
            for(int i=0; i<numberOfColumns-1; i++) {
                double giniValue = calculateGiniIndex(i);
            System.out.println("Column " + i + ": " + giniValue);
                if(giniValue < minGiniIndexValue) {
                    minGiniIndexValue = giniValue;
                    columnToSplit = i;
                }
            }

       System.out.println("Best column to split: " + columnToSplit);
        }
        public static void main(String[] args) {
    String filename = "/D:/csvfiles/data.csv";
            String delimiter = ",";
            GiniIndexHandler gh = new GiniIndexHandler();
            gh.readCSV(filename,delimiter);
            gh.printCSV();
            gh.findBestSplit();
        }

    }
```

Input:

A	C	E	Yes
B	C	E	No
A	C	E	Yes
B	C	F	No
A	C	F	Yes
B	C	F	No
A	D	E	Yes
B	D	E	No
A	D	E	Yes
B	D	F	No
A	D	F	Yes
B	D	F	No

Output:

A, C, E, Yes,
B, C, E, No,
A, C, E, Yes,
B, C, F, No,
A, C, F, Yes,
B, C, F, No,
A, D, E, Yes,
B, D, E, No,
A, D, E, Yes,
B, D, F, No,
A, D, F, Yes,
Column 0: 0.0
Column 1: 0.5
Column 2: 0.4444444444444444
Best column to split: 0

5. Write a java program to construct a Naïve Bayesianclassifier for a given dataset.

```
import java.util.*; import java.io.*;

class temp{

double p;

String q,r;

}

class naive

{

public static double probability(ArrayList<temp> al,String x) {

        double sum = 0;

        for(temp z:al)

           if((z.q).equals(x))
```

```
            sum++;
        return sum / al.size();
}
public static double mean(ArrayList<temp> al)
   {
       double sum = 0;
       for (temp z:al) {
           sum += z.p;
}
       return sum / al.size();
   }
public static double sd(ArrayList<temp> al,double mean) {
          double sum = 0;
          int x = al.size();
       for (temp z:al) {
           sum += Math.pow((z.p-mean),2);
       }
       return Math.sqrt(sum/(x*(x-1)));
    }
public static void main(String args[]) throws FileNotFoundException, IOException
{
BufferedReader csv = new BufferedReader(new FileReader(new
File("/Users/vishalprabhachandar/Documents/Programming/DataminingLab/Java-
Programs/B5/src/input.csv")));
String data = csv.readLine();
ArrayList<temp> y = new ArrayList<>();
ArrayList<temp> n = new ArrayList<>();
int i=0;
while(data != null)
{
String[] dataarray = data.split(",");
temp res = new temp();
res.p = Double.parseDouble(dataarray[0]); res.q = dataarray[1];
```

```
    res.r = dataarray[2];
    if(dataarray[2].equals("Y"))
        y.add(res);
    else
        n.add(res);
    data = csv.readLine();
}
System.out.println("Enter weight(1.0 - 60.0) and shirt size(S,M,L) to find its
class");
Scanner in = new Scanner(System.in);
double x1 = in.nextDouble();
String x2 = in.next();
//initial probability
int t = y.size() + n.size();
double p_y = (double)y.size()/(double)t;
double p_n = (double)n.size()/(double)t;
//for numeric attributes
double p1,p2,p3,p4,exp;
double mean1 = mean(y);
double mean2 = mean(n);
double sd1 = sd(y,mean1);
double sd2 = sd(n,mean2);
//probability for numeric = (1/(root(2*PI)*sd))exponential(- square(value-mean)/
(2*square(sd))
//refer class notes for formula
exp = Math.pow((x1 - mean1), 2)/(2* Math.pow(sd1,2));
p3 = Math.exp(-1*exp)/(Math.sqrt(2*3.14)*sd1);
exp = Math.pow((x1 - mean2), 2)/(2 *Math.pow(sd2,2));
p4 = Math.exp(-1*exp)/(Math.sqrt(2*3.14)*sd2);
//for non-numeric atrributes
p1 = probability(y,x2);
p2 = probability(n,x2);
double res1,res2;
```

```
res1 = p1*p3*p_y;
res2 = p2*p4*p_n;

//results
System.out.println("--Class Y--");
System.out.println("Mean :"+mean1+"\tStandard deviation:  "+sd1);
System.out.println("size :S given class"+i+":"+probability(y,"S"));
System.out.println("size :M given class"+i+":"+probability(y,"M"));
System.out.println("size :L given class"+i+":"+probability(y,"L")+"\n");

System.out.println("--Class N--");
System.out.println("Mean :"+mean2+ "\tStandard deviation: "+sd2);
System.out.println("size :S given class"+i+":"+probability(n,"S"));
System.out.println("size :M given class"+i+":"+probability(n,"M"));
System.out.println("size :L given class"+i+":"+probability(n,"L")+"\n");
System.out.println(res1+"\n"+res2);
if(res1>res2)
System.out.println("Class for weight "+x1+" and shirt size"+x2+" is : Y");
else
System.out.println("Class for weight "+x1+" and shirt size "+x2+" is : N");
}
}
```

Input:

	Standard	Standard	Standard
1	20	S	Y
2	23	L	N
3	13	M	Y
4	24	S	Y
5	45	M	Y
6	35	M	N
7	23	S	N
8	45	M	Y
9	20	S	Y
10	23	S	N
11	13	M	Y
12	24	M	Y
13	45	S	Y
14	35	M	N
15	23	S	N

Output:

```
Enter weight(1.0 - 60.0) and shirt size(S,M,L) to find its class
3 M
--Class Y--
Mean :27.666666666666668          Standard deviation:  4.527692569068708
size :S given class0:0.4444444444444444
size :M given class0:0.5555555555555556
size :L given class0:0.0
--Class N--
Mean :29.571428571428573          Standard deviation: 3.3441999744913216
size :S given class0:0.42857142857142855
size :M given class0:0.2857142857142857
size :L given class0:0.2857142857142857
9.885896967258057E-9
2.916295163017795E-16
Class for weight 3.0 and shirt sizeM is : Y
```

6. Write a java program to construct a K-Nearest Neighbor classifier for a given dataset.

```
import java.io.*;
import java.util.*;
class comp{
        double a;
        String b;
        int x,y,z;
        public comp(double p,String q,int r,int s,int t){
                a=p;b=q;x=r;y=s;z=t;
        }
}
class prog6
{
        public static void main(String args[]) throws FileNotFoundException,
IOException
```

```
	{
BufferedReader csv = new BufferedReader(new FileReader(new
File("input.csv")));
		String data = csv.readLine();
		int a[] = new int[3];
		int b[] = new int[3];
		Scanner in = new Scanner(System.in);
		System.out.println("Enter x,y,z to classify in Class A,B,C");
		a[0] = in.nextInt();
		a[1] = in.nextInt();
		a[2] = in.nextInt();
		System.out.println("enter number of nearest neighbours");
		int k = in.nextInt();
		int i;
		ArrayList<comp> al = new ArrayList<>();
		while(data != null)
		{
			String[] dataarray = data.split(",");
			int sum=0;
			for(i=0;i<3;i++)
				b[i] = Integer.parseInt(dataarray[i]);
			sum = (a[0]-b[0])*(a[0]-b[0]) + (a[1]-b[1])*(a[1]-b[1]) +
(a[2]-b[2])*(a[2]-b[2]);
			double res = Math.sqrt(sum);
			comp temp = new comp(res,dataarray[3],b[0],b[1],b[2]);
			al.add(temp);
			data = csv.readLine();
		}
		Collections.sort(al, new Comparator<comp>() {
		@Override public int compare(comp p1, comp p2) {
		if(p1.a == p2.a)
			return 0;
		else if(p1.a > p2.a)
```

```
                return 1;
        else
                return -1;
        } });
        int x[] = new int[3];
        System.out.println(k+" -nearest neighbour are:");
        for(comp temp:al)
        {
                if(k==0)
                        break;
                        System.out.println(temp.x+" "+temp.y+" "+temp.z+"
"+temp.b);
                        k--;
                        String c = temp.b;
                if(c.equals("A"))
                        x[0]++;
                else if(c.equals("B"))
                        x[1]++;
                else
                        x[2]++;
        }
        char res;
        int r = Math.max(x[0],Math.max(x[1],x[2]));
        if(x[0] == r)
                res = 'A';
        else if(x[1] == r)
                res = 'B';
        else
                res = 'C';
        System.out.println("classifer for "+a[0]+" "+a[1]+" "+a[2]+"
is :"+res);
        csv.close();
        in.close();
```

```
        }
}
```

Input:

	Standard	Standard	Standard	Standard
1	1	2	3	A
2	2	4	7	B
3	3	7	5	C
4	1	7	3	A
5	1	6	9	B
6	2	2	2	B
7	7	6	3	C
8	1	1	6	A
9	7	8	4	A
10	1	8	3	C

Output:

```
Enter x,y,z to classify in Class A,B,C
2 5 10
enter number of nearest neighbours
3
3 -nearest neighbour are:
1 6 9 B
2 4 7 B
3 7 5 C
classifer for 2 5 10 is :B
```

7. Write a java program to construct a single layer ANN perceptron for a given dataset.

```
import java.util.*;
import java.io.*;
class temp{
        double x,y,z;
}
class prog7
{
        public static void main(String args[]) throws FileNotFoundException,
IOException
        {
                BufferedReader csv = new BufferedReader(new
FileReader(new File("input.csv")));
                String data = csv.readLine();
                Random rand = new Random();
                ArrayList<temp> al = new ArrayList<>();
                double[] weight = new double[3];
                double rate = 0.02;
                while(data != null)
                {
                    String[] dataarray = data.split(",");
```

```
                temp v = new temp();
                v.x = Double.parseDouble(dataarray[0]);
                v.y = Double.parseDouble(dataarray[1]);
                v.z = Double.parseDouble(dataarray[2]);
                al.add(v);
                data = csv.readLine();
            }
            weight[0] = rand.nextInt(32767*2)-32767;
            weight[1] = rand.nextInt(32767*2)-32767;
            for(int i =0;i<10000;i++)
            {
                for(int j=0;j<al.size()-1;j++)
                {
                    int res=0;
                    double sum =
weight[0]+weight[1]*(al.get(j)).x+weight[2]*(al.get(j)).y;
                    if(sum>0)
                        {res=1;}
                    double error = (al.get(j)).z-res;
                    weight[0]=weight[0]+error*rate;
                    weight[1]=weight[1]+(rate*error*(al.get(j)).x);
                    weight[2]=weight[2]+(rate*error*(al.get(j)).y);
                }
            }
            System.out.println("enter x,y test data"); Scanner in = new
Scanner(System.in);
            double p = in.nextDouble();
            double q = in.nextDouble();
            double sum1 = weight[0]+weight[1]*p+weight[2]*q;
            System.out.println(sum1);
            if(sum1>0)
                System.out.println("predicted :1.0");

            else
                System.out.println("predicted :0.0");
            System.out.println("final weight[0] :"+weight[0]+"\nfinal
weight[1] :"+weight[1]+"\n final weight[2] :"+weight[2]);
        }
}
```

Input:

	Standard	Standard	Standard
1	2.7810836	2.550537003	0
2	1.465489372	2.362125076	0
3	3.396561688	4.400293529	0
4	1.38807019	1.850220317	0
5	7.673756466	3.508563011	1
6	8.675418651	-0.242068655	1
7	5.332441248	2.088626775	1
8	6.922596716	1.77106367	1
9	3.06407232	3.005305973	0

Output:

```
enter x,y test data
3 4
464.7484016088056
predicted :1.0
final weight[0] :-24084.00000000873
final weight[1] :10036.470942405751
final weight[2] :-1390.166106399929
```

8. Write a java program to perform linear regression on a given numeric dataset with numeric classattribute.

```
//y=Mx+b;
import java.util.*;

class prog8
{
	public static void main(String args[])
	{
		double M=-1000;
		double B=-1000;
		//y=5x+2
		double mean_x=0,mean_y=0,cov_xy=0,var_x=0;
		double[] x = { 1, 2, 3, 4, 5, 6, 7, 8 };
		double[] y = { 7,12,17,22,27,32,37,42};
		for(int i=0;i<x.length;i++)
		{
			mean_x = mean_x+x[i];
			mean_y = mean_y+y[i];
		}
		mean_x = mean_x/x.length;
		mean_y = mean_y/x.length;
		for(int i=0;i<x.length;i++)
		{
			var_x += Math.pow(x[i] - mean_x, 2);
			cov_xy += (x[i] - mean_x) * (y[i] - mean_y);
		}
		var_x = var_x/x.length;
		cov_xy = cov_xy/x.length;
		M = cov_xy/var_x;
		B = mean_y-M*mean_x;
		System.out.println("enter the value for x");
		Scanner in = new Scanner(System.in);
		double r = in.nextInt();
		double res = r*M+B;
		System.out.println("Predicted output for "+r+" : "+res);
		in.close();
	}
}
```

Output:

enter the value for x
4
Predicted output for 4.0 : 22.0

9. Write a java program to perform k-means clustering on numeric dataset.

```
import java.util.*;
import java.io.*;
class record
{
	double attr1=0,attr2=0;
}
class prog9
{
	public static record calc(ArrayList<record> bl) {
	    record a = new record();
	    double x=0,y=0;
	    for(record b:bl){
	         x=x+b.attr1;
	         y=y+b.attr2;
	    }
	    a.attr1=x/bl.size();
	    a.attr2=y/bl.size();
	    return a;
	}
	public static double dist(double p,double q,double r,double s) {
	double res = Math.pow((p-r),2)+Math.pow((q-s),2);
	    return Math.sqrt(res);
	}
	public static void main(String args[]) throws FileNotFoundException,
IOException
	{
		BufferedReader csv = new BufferedReader(new FileReader(new
File("input.csv")));
		String data = csv.readLine(); ArrayList<record> al = new
ArrayList<>(); while(data != null)
		{
			String[] dataarray = data.split(",");
			record rc = new record();
			rc.attr1 = Double.parseDouble(dataarray[0]);
			rc.attr2 = Double.parseDouble(dataarray[1]);
			al.add(rc);
			data = csv.readLine();
		}
		ArrayList<record> centroid = new ArrayList<>();
		centroid.add(al.get(0));
		centroid.add(al.get(3));
		centroid.add(al.get(6));
		ArrayList<record> c1 = new ArrayList<>();
```

```
                ArrayList<record> c2 = new ArrayList<>();
                ArrayList<record> c3 = new ArrayList<>();
                for(int i=0;i<100;i++)
                {
                        c1.clear();c2.clear();c3.clear();
                        for(record r:al)
                        {
                            ArrayList<Double> t = new ArrayList<>();
                                        double p = dist(r.attr1,r.attr2,
(centroid.get(0)).attr1,(centroid.get(0)).attr2 );
                                        double q = dist(r.attr1,r.attr2,
(centroid.get(1)).attr1,(centroid.get(1)).attr2 );
                                        double s = dist(r.attr1,r.attr2,
(centroid.get(2)).attr1,(centroid.get(2)).attr2 );
                                t.add(p);
                                t.add(q);
                                t.add(s);
                                double res = Collections.min(t);
                                if(res == p){c1.add(r);}
                                if(res == q){c2.add(r);}
                                if(res == s){c3.add(r);}
                        }
                        centroid.clear();
                        centroid.add(calc(c1));
                        centroid.add(calc(c2));
                        centroid.add(calc(c3));
                }
                System.out.println("------0th cluster--------"); for(record r:c1)
                System.out.println(r.attr1 +" "+r.attr2); System.out.println("------1th
cluster--------"); for(record r:c2)
                System.out.println(r.attr1 +" "+r.attr2); System.out.println("------2th
cluster--------"); for(record r:c3)
                System.out.println(r.attr1 +" "+r.attr2); System.out.println("-----
centroid------"); for(int i=0;i<3;i++)
                System.out.println((centroid.get(i)).attr1+","+
(centroid.get(i )).attr2);
        }
}
```

Input:

	Standard	Standard
1	2	10
2	2	5
3	8	4
4	5	8
5	7	5
6	6	4
7	1	2
8	4	9
9	3	9

Output:

```
------0th cluster--------
2.0 10.0
5.0 8.0
4.0 9.0
3.0 9.0
------1th cluster--------
8.0 4.0
7.0 5.0
6.0 4.0
------2th cluster--------
2.0 5.0
1.0 2.0
-----centroid------
3.5,9.0
7.0,4.333333333333333
1.5,3.5
```

10. Write a java program to compute sensitivity, specificity, precision, recall, weighted accuracy using confusion matrix

Demo.java

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.ArrayList;


public class Demo {

	public static ArrayList<DataPoint> getInput(String filename, String
delimiter){
		ArrayList<DataPoint> data = new ArrayList<>();
		try {
			BufferedReader br = new BufferedReader(new
FileReader(new File(filename)));
			String line;
			boolean rv, pv;
			while((line=br.readLine())!=null) {
				String[] currentLine = line.split(delimiter);
				if(currentLine[0].equals("T")) {
					rv = true;
				}
				else {
					rv = false;
				}

				if(currentLine[1].equals("T")) {
					pv = true;
```

```
                        }
                        else {
                                pv = false;
                        }

                        DataPoint d = new DataPoint(rv, pv);
                        data.add(d);

                }
        } catch (Exception e) {
                e.printStackTrace();
        }

        return data;
}



public static void main(String[] args) {
        String filename, delimiter;
        filename = "data.csv";
        delimiter = ",";
        ArrayList<DataPoint> data = getInput(filename, delimiter);
        double truePositive=0, falsePositive=0, trueNegetive=0
,falseNegetive=0;
        double w1 = 2, w2 = 1, w3 = 1, w4 = 2;
        for(DataPoint d : data) {
                if(d.realValue && d.predictedValue) {
                        truePositive++;
                }
                if(!d.realValue && d.predictedValue) {
                        falsePositive++;
                }
                if(!d.realValue && !d.predictedValue) {
                        trueNegetive++;
                }
                if(d.realValue && !d.predictedValue) {
                        falseNegetive++;
                }
        }

        System.out.println(truePositive + " " + falseNegetive + " " +
falsePositive + " " + trueNegetive);

        double sensitivity = truePositive/(truePositive + falseNegetive);
        double specificity = trueNegetive/(trueNegetive + falsePositive);
        double recall = truePositive/(truePositive + falseNegetive);
        double precision = truePositive/(truePositive+falsePositive);
        double weightedAccuracy = (w1*truePositive + w4*trueNegetive)/
(w1*truePositive + w2*falseNegetive + w3*falsePositive + w4*trueNegetive);
```

```
                System.out.println("Sensitivity: " + sensitivity);
                System.out.println("Specificity: " + specificity);
                System.out.println("Precision: " + precision);
                System.out.println("Recall: " + recall);
                System.out.println("Weighted Accuracy: " + weightedAccuracy);
        }

}
```

DataPoint.java

```
public class DataPoint {
        public boolean realValue, predictedValue;
        public DataPoint(boolean rv, boolean pv) {
                this.realValue = rv;
                this.predictedValue = pv;
        }
}
```

Input:

	Standard	Standard
1	T	T
2	T	T
3	F	F
4	T	F
5	F	T
6	T	F
7	F	F
8	F	T
9	T	T
10	T	F

Output:

```
3.0 3.0 2.0 2.0
Sensitivity: 0.5
Specificity: 0.5
Precision: 0.6
Recall: 0.5
Weighted Accuracy: 0.6666666666666666
```