Data Mining Lab Book

Name:- Pranav Balasaheb Bhosale
PRN:- 2019BTEIT00038
Class: Final Year-IT-Sem I (2022-2023)

Experiment No 1

Study and use different types of graphs and charts (use MS-XLS).

Walchand College of Engineering, Sangli.

Experiment No 1

Aim:- To study and use of different types of graphs and charts

Theory :- Visualization of data is possible with help of charts They condense large amount of data into an easy to understand formats.

① Pie charts :-

Pie charts is a type of graph that represents the data in circular graph. The slice of pie shows the relative size of data. Part-to whole comparison is the main purpose of this chart.

② Line chart :-

Line chart shows changes in values accross continous measurement Movement of line up and down helps bring out positive and negative changes.

It exposes overall trend & helps to make projections for future

③ Scatter plot

Displayes values on two numeric variables using point positioned on two axes. Scatter plot are greater for identifying outlier & gaps in data.

④ Bar chart:-

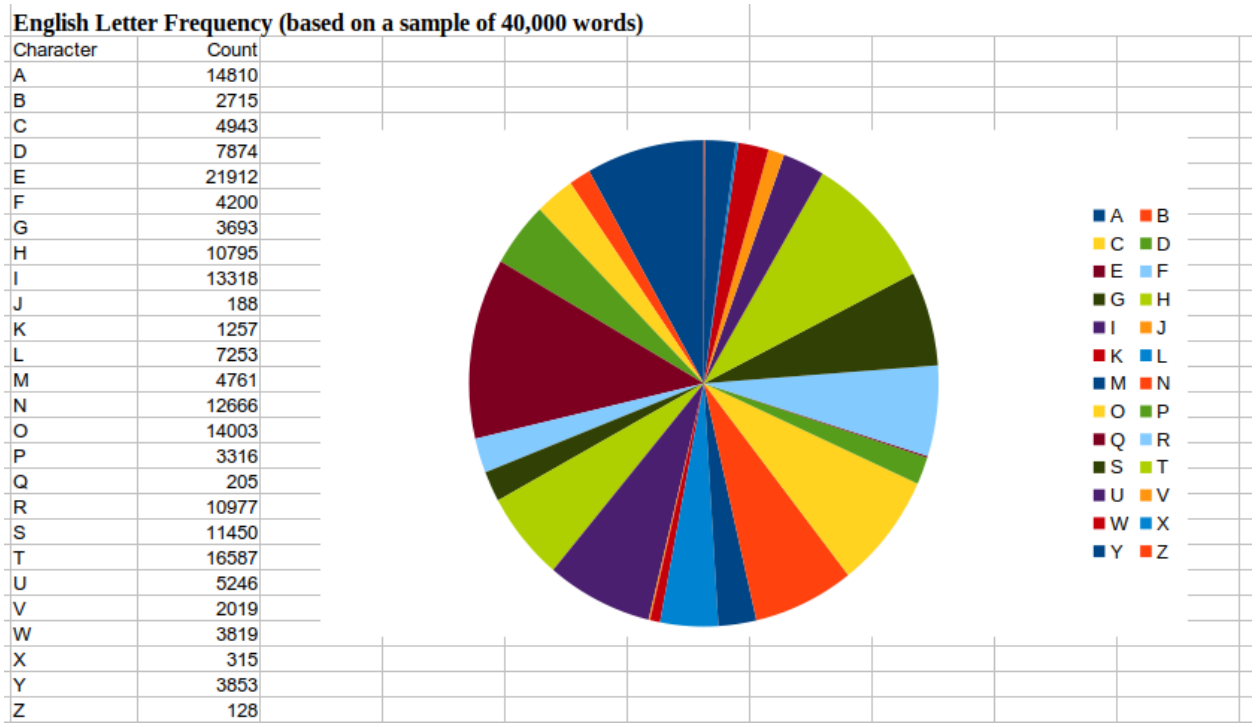
Values are indicated by length of bars, Bar charts can be oriented horizontally or vertically It gives statical analysis of value

conclusion:- study of the different types of charts and graph provide insights about data by condensing large information in visual formats.

Data set:- English Letter Frequency (based on a sample of 40,000 words)

Character	Count
A	14810
B	2715
C	4943
D	7874
E	21912
F	4200
G	3693
H	10795
I	13318
J	188
K	1257
L	7253
M	4761
N	12666
O	14003
P	3316
Q	205
R	10977
S	11450
T	16587
U	5246
V	2019
W	3819
X	315
Y	3853
Z	128

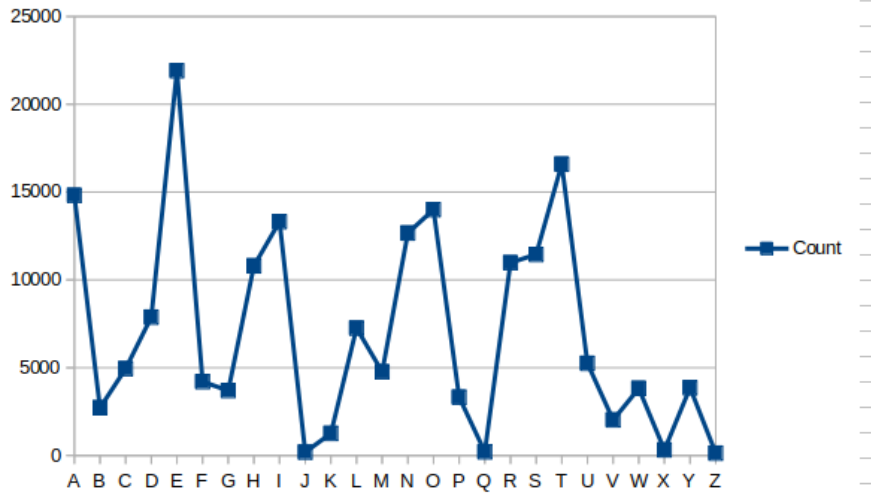
Pie chart:-



Line Chart:-

English Letter Frequency (based on a sample of 40,000 words)

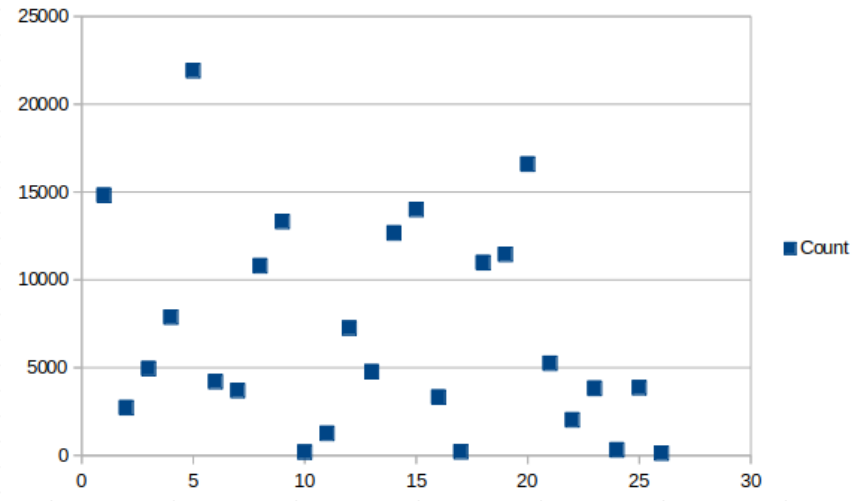
Character	Count
A	14810
B	2715
C	4943
D	7874
E	21912
F	4200
G	3693
H	10795
I	13318
J	188
K	1257
L	7253
M	4761
N	12666
O	14003
P	3316
Q	205
R	10977
S	11450
T	16587
U	5246
V	2019
W	3819
X	315
Y	3853
Z	128



Scatter Chart:-

English Letter Frequency (based on a sample of 40,000 words)

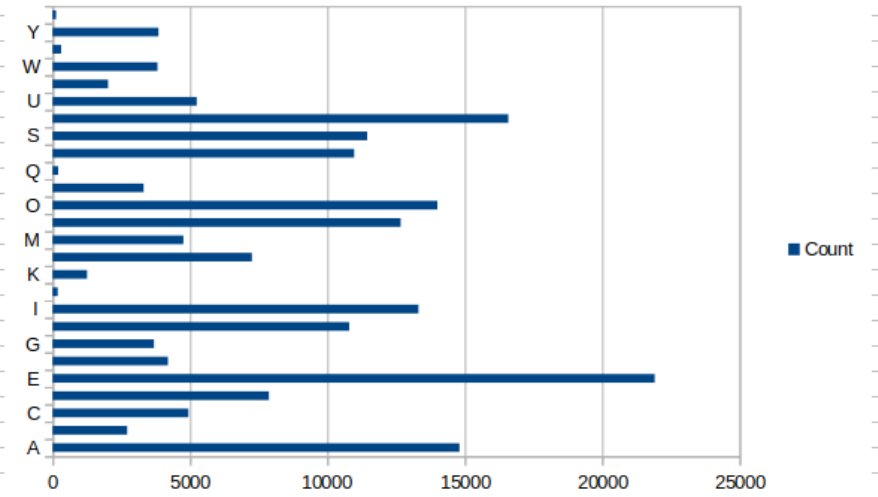
Character	Count
A	14810
B	2715
C	4943
D	7874
E	21912
F	4200
G	3693
H	10795
I	13318
J	188
K	1257
L	7253
M	4761
N	12666
O	14003
P	3316
Q	205
R	10977
S	11450
T	16587
U	5246
V	2019
W	3819
X	315
Y	3853
Z	128



Bar Chart:-

English Letter Frequency (based on a sample of 40,000 words)

Character	Count
A	14810
B	2715
C	4943
D	7874
E	21912
F	4200
G	3693
H	10795
I	13318
J	188
K	1257
L	7253
M	4761
N	12666
O	14003
P	3316
Q	205
R	10977
S	11450
T	16587
U	5246
V	2019
W	3819
X	315
Y	3853
Z	128

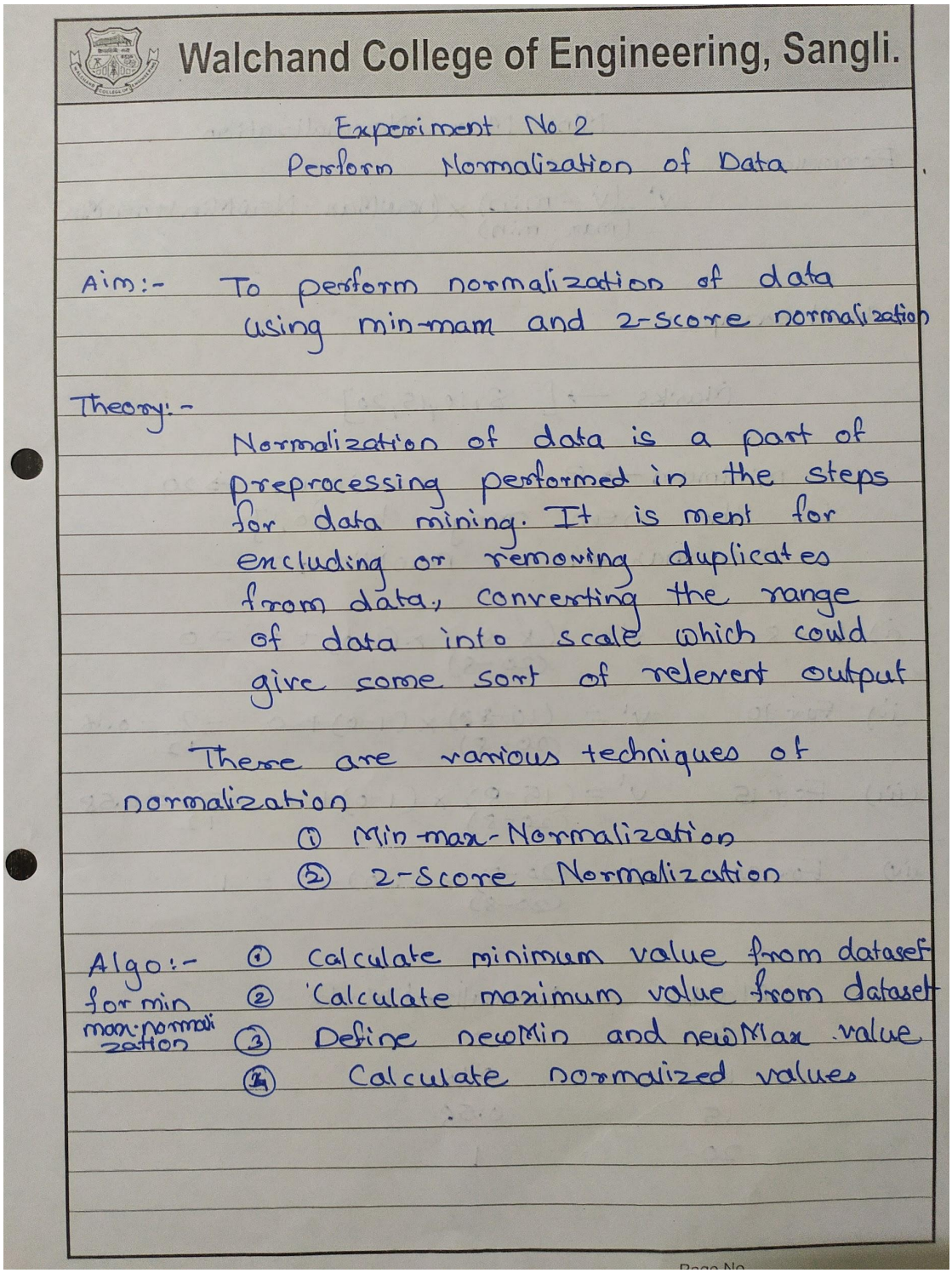


Conclusion:-

Graphs and charts are the best way to visualize large data. Different types of Graphs focus on particular feature comparison of data. They condense large information into easy-to-understand visual formats to provide important data insights.

Experiment No 2

Perform Normalization of data (Min-max and Z-score)



Walchand College of Engineering, Sangli.

Experiment No 2

Perform Normalization of Data

Aim:- To perform normalization of data using min-mam and 2-score normalization

Theory:-

Normalization of data is a part of preprocessing performed in the steps for data mining. It is ment for encluding or removing duplicates from data, converting the range of data into scale which could give some sort of relevent output

There are various techniques of normalization

① Min-max-Normalization
② 2-Score Normalization

Algo:- for min max-normalization

① Calculate minimum value from dataset
② Calculate maximum value from dataset
③ Define newMin and newMax value
④ Calculate normalized values

Min – Max Normalization

Formula :-

$$v' = \frac{(v - min)}{(max - min)} \times (newMax - NewMin) + neoMin$$

Solved example :-

Marks → [8, 10, 15, 20]

minimum → 8 maximum → 20

Let the neo range be [0, 1]

newmax → 1 neoMin → 0

(i) For 8 $v' = \frac{(8-8)}{(20-8)} \times (1-0) + 0 = 0$

(ii) For 10 $v' = \frac{(10-8)}{(20-8)} \times (1-0) + 0 = \frac{2}{12} = 0.16$

(iii) For 15 $v' = \frac{(15-8)}{(20-8)} \times (1-0) + 0 = \frac{7}{12} = 0.58$

(iv) For 20 $v' = \frac{(20-8)}{(20-8)} \times (1-0) + 0 = 1$

Marks	Normalized Marks
8	0
10	0.16
15	0.58
20	1

Z- Score Normalization

Algorithm:-

1. Calculate mean of given data column wise
2. Calculate standard deviation for particular column
3. Normalize the data using Z- score normalization
4. Display normalized data.

Formula:-

$$(\sigma)\ S.D = \sqrt{\frac{\Sigma\ (x - mean)^2}{n}}$$

$$Z\text{-score} = \frac{x - mean}{\sigma}$$

Example:-

Marks → [8, 10, 15, 20]

$$mean = \frac{8+10+15+20}{4} = 13.25$$

$$SD(\sigma) = \sqrt{\frac{(8-13.5)^2 + (10-13.5)^2 + (15-13.5)^2 + (20-13.5)^2}{4}}$$

$$= \sqrt{\frac{27.56 + 10.56 + 3.06 + 45.56}{4}}$$

$$= \sqrt{\frac{86.74}{4}} = \sqrt{21.6} = 4.6$$

Z-score

(i) for 8 $v' = \frac{8 - 13.25}{4.6} = -1.14$

(ii) for 10 $v' = \frac{10 - 13.25}{4.6} = -0.7$

(iii) for 15 $v' = \frac{15 - 13.25}{4.6} = 0.3$

(iv) for 20 $v' = \frac{20 - 13.25}{4.6} = 1.4$

Marks	Z-score Normalized marks
8	-1.14
10	-0.7
15	0.3
20	1.4

Knime Result:-

Marks	min-max	Z-score
8	0	-0.976
10	0.1666	-0.6743
15	0.5833	0.3243
25	0.9999	1.2552

Weka min-max normalization result

Marks	
8	0
10	1.66667
15	0.58333
25	1

Page No.

conclusion:-

With help of normalization attribute values are scaled to fall within specified range.

Normalization helps prevent attribute with initially large range from overweighting attribute with small range

Code:-

```
const prompt = require("prompt-sync")();
const xlsx = require("xlsx");

//Reading the input file
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["Student_Data"];

var data = xlsx.utils.sheet_to_json(ws);

var old_min = Number.POSITIVE_INFINITY,
 old_max = Number.NEGATIVE_INFINITY;

//Finding old_min and old_max values min max normalization
for (let i = 0; i < data.length; i++) {
 old_min = Math.min(old_min, data[i].Marks);
 old_max = Math.max(old_max, data[i].Marks);
}

//Taking the new range of scale formin max normalization
console.log("Enter New Range For Marks ");
var new_min = parseInt(prompt("From : "));
var new_max = parseInt(prompt("To : "));


//Calculating normalized data min max normalization

var newData = data.map(function (record) {
 record.min_max_normalized_marks =
   ((record.Marks - old_min) / (old_max - old_min)) * (new_max - new_min) +
   new_min;
 record.min_max_normalized_marks = record.min_max_normalized_marks.toFixed(2); //round
off to two decimal digits
 return record;
});

//Z -score Normalization
let n = data.length;
let sum = 0;
```

```
// calculating mean

for (let i = 0; i < data.length; i++) {
 sum += data[i].Marks;
}
var mean = sum / n;

// calculating standard deviation
var sd_sum = 0;
for (let i = 0; i < data.length; i++) {
 sd_sum += (data[i].Marks - mean) * (data[i].Marks - mean);
}
var std_dev = Math.sqrt(sd_sum / n);
std_dev = std_dev.toFixed(2);

// calculating z-score normalized value
var newData = data.map(function (record) {
 record.zscore_normalized_marks = ((record.Marks - mean) / std_dev).toFixed(2);
 return record;
});

console.log(newData);

//Creating and writing data to the result file for min max normalization and z score
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(newData);

xlsx.utils.book_append_sheet(newWB, newWS, "Normalized Student Data");

xlsx.writeFile(newWB, "Normalized_Data.xlsx");
```

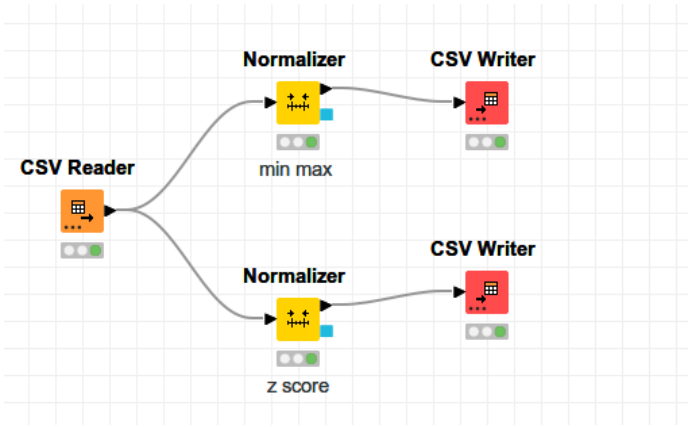
Input file:

Marks
8
10
15
20

Output file :

Marks	min_max_normalized_marks	zscore_normalized_marks
8	0.00	-1.13
10	0.17	-0.70
15	0.58	0.38
20	1.00	1.45

Knime result :



Knime Min-max normalization

Marks
0
0.166666666666667
0.583333333333333
1

Knime Z-score normalization

Marks
-0.976304270613646
-0.6043788341894
0.325434756871215
1.25524834793183

Weka Min-Max normalization

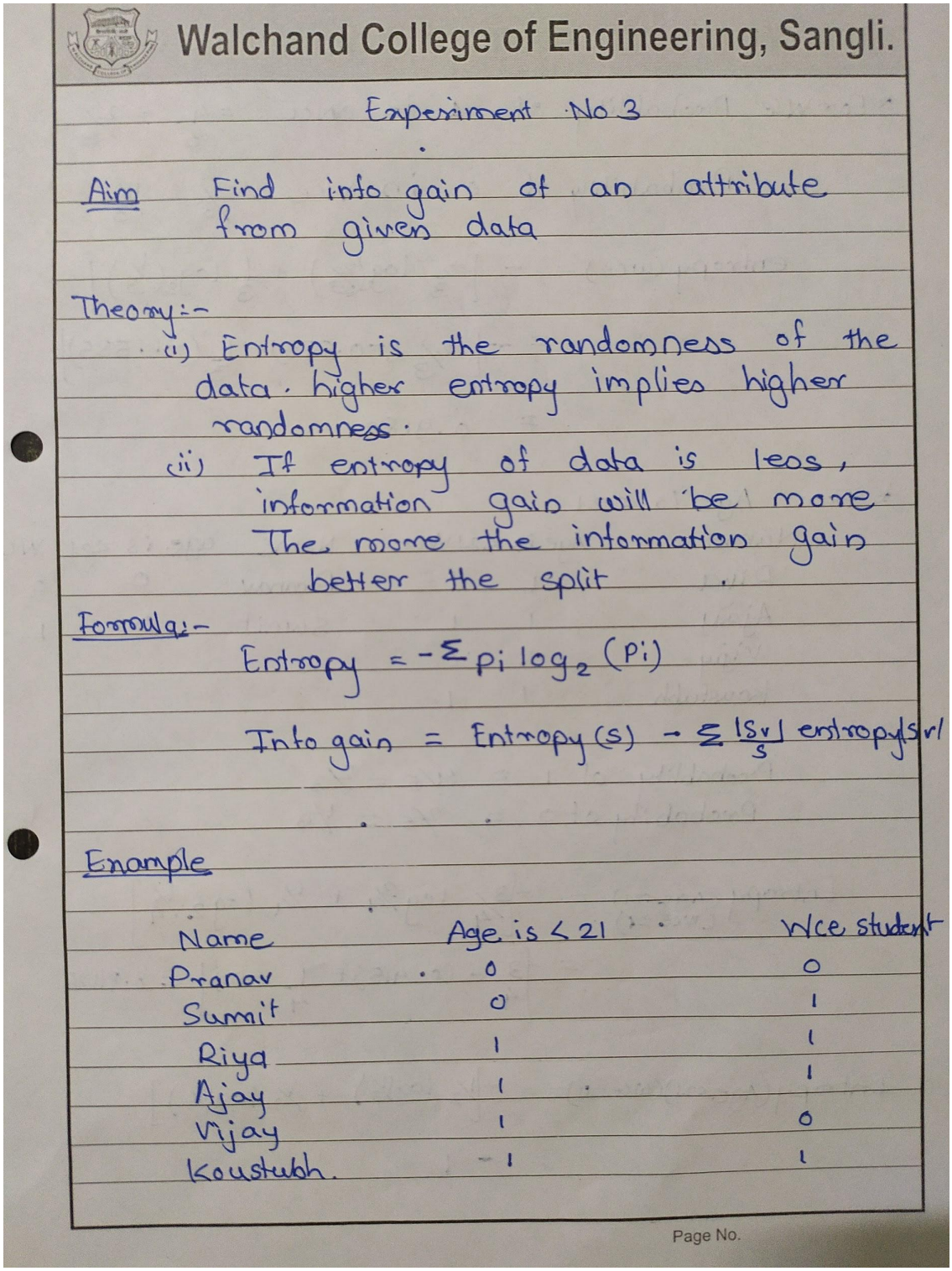
Marks
0
0.166667
0.583333
1

Conclusion:

Transformed values obtained by calculations and knime and weka are identical. Successfully performed normalization of given data using min-max and z-score normalization.

Experiment No 3

Find Info Gain of an attribute from given data.



Walchand College of Engineering, Sangli.

Experiment No 3

Aim Find info gain of an attribute from given data

Theory:-

(i) Entropy is the randomness of the data. higher entropy implies higher randomness.

(ii) If entropy of data is less, information gain will be more. The more the information gain better the split.

Formula:-

$$\text{Entropy} = -\sum p_i \log_2(p_i)$$

$$\text{Info gain} = \text{Entropy}(S) - \sum \frac{|S_v|}{S} \text{entropy}(S_v)$$

Example

Name	Age is < 21	Wce student
Pranav	0	0
Sumit	0	1
Riya	1	1
Ajay	1	1
Vijay	1	0
Koustubh	1	1

Page No.

* For Wce Probability of 1 in WCE $= \frac{4}{6} = \frac{2}{3}$

Probability of 0 in Wce $= \frac{2}{6} = \frac{1}{3}$

$$\text{entropy (wce)} = -\left[\frac{2}{3}\log_2\left(\frac{2}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right]$$

$$= -\left[\frac{2}{3}(-0.585) + \frac{1}{3}(-1.585)\right]$$

$$= 0.9183$$

* For Age is <21

Name	Age is <21	Wce
Riya	1	1
Ajay	1	1
Vijay	1	0
Koustubh	1	1

Name	age is <21	WCE
Pranav	0	0
Sumit	0	1

Probability of 1 $= 4/6 = 2/3$

Probability of 0 $= 2/6 = 1/3$

$$\text{Entropy}_{(\text{Age}<21)(\text{wce}=1)} = -\left[\frac{3}{4}\log\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right]$$

$$= -\left[\frac{3}{4}(-0.415) + \frac{1}{4}(-2)\right] = 0.8112$$

$$\text{Entropy}_{(\text{Age}<21)(\text{wce}=0)} = -\left[\frac{1}{2}\log\left(\frac{1}{2}\right) + \frac{1}{2}\log\left(\frac{1}{2}\right)\right]$$

$$= 1$$

Page No.

Info gain = Entropy(wce) − Σ (Sv/S) entropy(Sv)

= 0.9183 − [(4/6) × (0.81125) + (2/6) × (1)]

= 0.9183 − [0.5408 + 0.333]

Info gain = 0.0442

Knime output

size	entroply
4	0.81127
2	1
6	0.8741

Quality :- 0.126

Weka result

Info gain : 0.0441

Conclusion :- Info gain can be used to find Attribute subset. for more valid result. from a sub-set of attribute.

Code:

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("Games.xlsx");
var ws = wb.Sheets["PlayingInfo"];

var data = xlsx.utils.sheet_to_json(ws);
var parentattri = prompt("Enter Parent attribute : ");
var childattri = prompt("Enter Child attribute : ");
var posR = 0,
  negR = 0;
var arr = {};
// console.log(data);
for (let i = 0; i < data.length; i++) {
  if (data[i][parentattri] == 1) {
    posR++;
  } else if (data[i][parentattri] == 0) {
    negR++;
  }

  if (!(data[i][childattri] in arr)) {
    arr[data[i][childattri]] = 0;
  }

  arr[data[i][childattri]]++;
}

//
let arr2 = {};
for (let i = 0; i < data.length; i++) {
  if (!(data[i][childattri] in arr2)) {
    arr2[data[i][childattri]] = { 1: 0, 0: 0 };
  }

  var res = data[i][parentattri];
  // console.log(arr2[data[i].Wind].Yes);
  if (res == 0) {
    arr2[data[i][childattri]]["0"]++;
  } else if (res == 1) {
```

```
    arr2[data[i][childattri]]["1"]++;
  }
}
// console.log(arr2);

/// Parent Entropy

var totRecord = data.length;
var entropy = -(
  (posR / totRecord) * Math.log2(posR / totRecord) +
  (negR / totRecord) * Math.log2(negR / totRecord)
);

entropy = entropy.toFixed(4);
// console.log(entropy);
var ent_sum = 0;
for (const key in arr2) {
  var array = arr2[key];
  let posRes = array["0"],
    negRes = array["1"];
  let totRes = posRes + negRes;

  let ent =
    -((posRes / totRes) * Math.log2(posRes / totRes)) -
    (negRes / totRes) * Math.log2(negRes / totRes);
  ent_sum += arr2[key].ent = ent * (totRes / data.length);
}

// console.log(arr2);
// console.log(ent_sum);

var infoGain = entropy - ent_sum;
// console.log(infoGain);
var newdata = [];
newdata.push({
  Parent_ent: entropy,
  Child_ent: ent_sum,
  InfoGain: infoGain,
});
```

```
// console.log(newdata);

//Creating and writing data to the result file
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(newdata);

xlsx.utils.book_append_sheet(newWB, newWS, "InfoGain");

xlsx.writeFile(newWB, "InfoGain.xlsx");
```

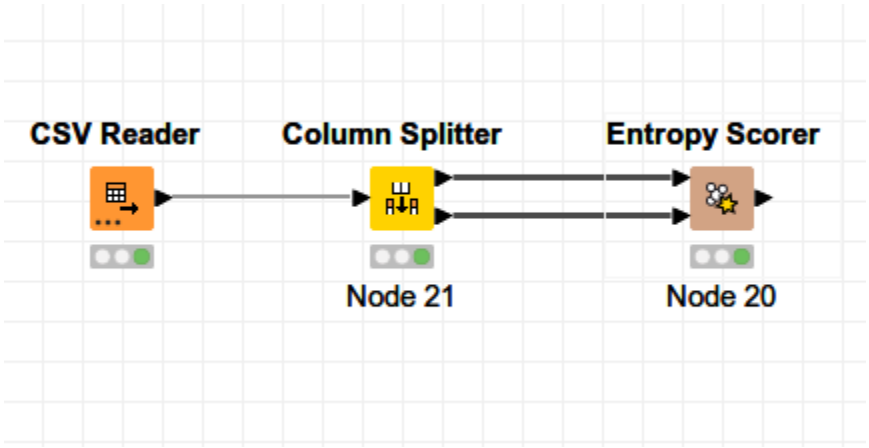
Input File :

Name	Age<21	Wce
Pranav	0	0
sumit	0	1
riya	1	1
ajay	1	1
viay	1	0
koustubh	1	1

Output file :

Wce entropy	Age<21 entropy	InfoGain
0.9183	0.874185416306088	0.0441145836939115

Knime result :



Statistics View - 3:20 - Entropy Scorer

File

Clustering statistics

Data Statistics

Statistics	Value
Number of clusters found:	2
Number of objects in clusters:	6
Number of reference clusters:	2
Total number of patterns:	6

Data Statistics

Score	Value
Entropy:	0.8742
Quality:	0.1258

Row ID	I Size	D Entropy	D Norma...	D Quality
1	4	0.811	0.811	?
0	2	1	1	?
Overall	6	0.874	0.874	0.126

Weka Result:

```
=== Run information ===

Evaluator:    weka.attributeSelection.InfoGainAttributeEval
Search:       weka.attributeSelection.Ranker -T -1.7976931348623157E308 -N -1
Relation:     Games
Instances:    6
Attributes:   2
              Age<21
              Wce
Evaluation mode:    evaluate on all training data



=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 2 Wce):
        Information Gain Ranking Filter

Ranked attributes:
 0.0441 1 Age<21

Selected attributes: 1 : 1
```

Conclusion:

Info gain can be used for the process of attribute subset selection to produce more valid results from a subset of attribute

Experiment No 4

Find t and d weight of a data.

Walchand College of Engineering, Sangli.

Experiment no 4

Aim:- To find 't' and 'd' weight of data

Theory:-

① t- weight

A quantitative characteristic rule which can be basically represented It implies weight within class.

② d- weight

A quantitative discriminant rule which can be logically represented as d weight

It implies weight accross class

Algorithm:-

① Find row wise and colum wise total count for each attribute and transition

② For each value find t weight

③ For each value find d weight

④ Find 't & d' weight for total value.

Page No.

Example

Class	Type	Count
Comedy films	Bollywood	150
	Hollywood	120
Horror films	Bollywood	40
	Hollywood	60

Film Type	Bollywood			Hollywood			Total		
	count	t	d	count	t	d	count	t	d
comedy	150	56%	78%	120	44%	67%	270	100%	73%
Horror	40	40%	22%	60	60%	33%	100	100%	22%
Total	90	52%	100%	180	48%	100%	370	100%	100%

Conclusion

① d-weight helps to find the weight of an attribute accross the class

② t-weight helps to find the weight of an attribute within the class

Code:-

```
const prompt = require("prompt-sync")();
const xlsx = require("xlsx");
//Reading the input file
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["Movies"];
var data = xlsx.utils.sheet_to_json(ws);
console.log(data);
var tot = {};
var classNames = {};
var type = {};
for (let i = 0; i < data.length; i++) {
 if (!(data[i].class in tot)) {
   tot[data[i].class] = 0;
   classNames[data[i].class] = 0;
 }
 tot[data[i].class] += data[i].count;

 if (!(data[i].type in tot)) {
   tot[data[i].type] = 0;
   type[data[i].type] = 0;
 }
 tot[data[i].type] += data[i].count;
}
console.log(tot);
for (let i = 0; i < data.length; i++) {
 data[i].t_weight = (data[i].count / tot[data[i].class]) * 100;
 data[i].d_weight = (data[i].count / tot[data[i].type]) * 100;
}
var grandTot = 0;
for (var key in tot) {
 grandTot += tot[key];
}
grandTot = grandTot / 2;
for (var key in tot) {
 newdata = {};
 newdata.class = "tot";
 newdata.type = key;
 newdata.count = tot[key];
 if (key in classNames) {
```

```
  newdata.t_weight = 100;
  newdata.d_weight = (tot[key] / grandTot) * 100;
 } else {
  newdata.t_weight = (tot[key] / grandTot) * 100;
  newdata.d_weight = 100;
 }
 data.push(newdata);
}
console.log(data);
//Creating and writing data to the result file
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(data);
xlsx.utils.book_append_sheet(newWB, newWS, "T_D_Weight");
xlsx.writeFile(newWB, "T_D_Weight.xlsx");
```

Input:-

class	type	count
12th	science	150
12th	arts	120
11th	science	40
11th	arts	60

Output:-

class	type	count	t_weight	d_weight
12th	science	150	55.5555555555556	78.9473684210526
12th	arts	120	44.4444444444444	66.6666666666667
11th	science	40	40	21.0526315789474
11th	arts	60	60	33.3333333333333
tot	12th	270	100	72.972972972973
tot	science	190	51.3513513513514	100
tot	arts	180	48.6486486486487	100
tot	11th	100	100	27.027027027027

Conclusion :

D- weight helps to find the weight of an attribute across the class.

T- weight helps to find the weight of the attribute within the class.

Experiment No 5

Find 5 no summary of a dataset.

Walchand College of Engineering, Sangli.

Experiment no 5

Aim:- Find 5 no summery of dataset

Theory:-

5 no summery is used measure spread of data. It includes following values

1. minimum :- smallest element of dataset
2. Q_1 :- first Quartile (median betn min & median)
3. Q_2 : median of whole dataset
4. Q_4 :- third Quartile (median of data betn median and max)
5. maximum:- maximum of dataset

Algorithm:-

1. Sort dataset in ascending order
2. Find min & max from sorted data
3. If number of elements are odd then middle element is median (Q_2). else median is avg of middle 2 elements
4. Divide the numbers into two groups through median
5. Using 3rd step find Q_1, Q_2 that is median of subgroups

Page No.

example:- Data → [8, 16, 9, 18, 21, 21, 24, 31, 36, 27, 30, 35]

min → 8 max → 35

median = $\frac{21+24}{2} = 22.5$

$Q_1 \Rightarrow$ 8, 16, 9, 18, 21, 21

$Q_1 = \frac{16+18}{2} = \frac{34}{2} = 17$

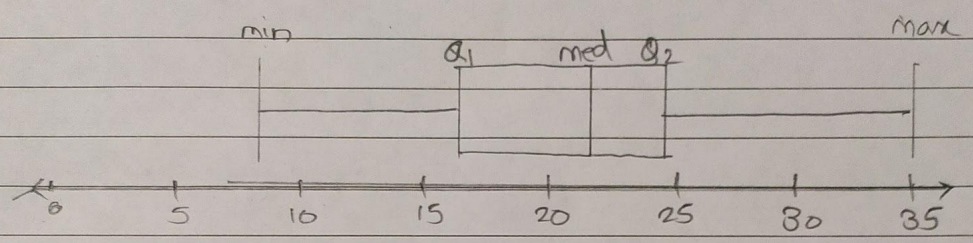
$Q_2 \Rightarrow$ 24, 26, 27, 30, 31, 35

$Q_3 = \frac{27+30}{2} = 28.5$

InterQuartile range $= Q_3 - Q_1$
$= 28.5 - 17$
$= 11.5$

Lower whiskers $= Q_1 - (1.5 \times IQ)$
$= 17 - (1.5 \times 11.5)$
$= -0.25$

Upper whiskers $= Q_3 + (1.5 \times IQ)$
$= 28.5 + (1.5 \times 11.5)$
$= 45.75$



Lower whiskers = -0.25
minimum = 8
Q_1 = 17
Q_2 = 22.5
Q_3 = 28.5
maximum = 35
upper whiskers = 45.75

Knime output
min = 8
Q_1 = 17
Q_2 = 22.5
Q_3 = 28.5
max = 35

Conclusion:-
→ 5 no summery can be used to obtain measure for spread of data.

Page No.

Code:-

```
const prompt = require("prompt-sync")();
const xlsx = require("xlsx");
//Reading the input file
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["Student_Data"];
var data = xlsx.utils.sheet_to_json(ws);

var min_val;
var max_val;
data.sort((a, b) => (a.Marks > b.Marks ? 1 : -1));
min_val = data[0].Marks;
max_val = data[data.length - 1].Marks;
let n = data.length;
let q1Term = Math.round((25 / 100) * (n + 1));
let q1 = data[q1Term - 1].Marks;
let q2;
if (n % 2 == 1) {
 q2 = data[(n - 1) / 2].Marks;
} else {
 q2 = (data[n / 2 - 1].Marks + data[n / 2].Marks) / 2;
}
let q3term = Math.round((75 / 100) * (n + 1));
let q3 = data[q3term - 1].Marks;
var IQ = q3 - q1;
var lowerwhisers = q1 - 1.5 * IQ,
 upperWhisers = q3 + 1.5 * IQ;
var newdata = [];
newdata.push({
 Lowerwhisers: lowerwhisers,
 MinimumValue: min_val,
 LowerQuartile: q1,
 MedianValue: q2,
 UpperQuartile: q3,
 MaximumValue: max_val,
 UpperWhisers: upperWhisers,
});

console.log(newdata);
```

```
//Creating and writing data to the result file
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(newdata);

xlsx.utils.book_append_sheet(newWB, newWS, "FiveNOSummery");

xlsx.writeFile(newWB, "FiveNOSummery.xlsx");
```

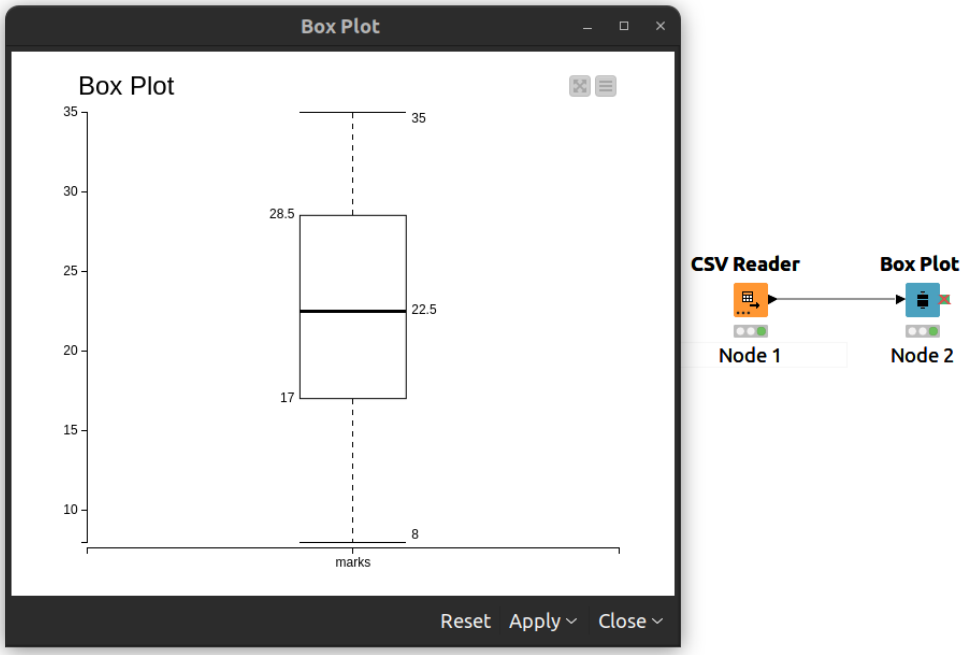
Input:-

Marks
8
16
9
18
21
21
24
31
26
27
30
35

Output:-

Lowerwhisers	-0.25
MinimumValue	8
LowerQuartile	16
MedianValue	22.5
UpperQuartile	28.5
MaximumValue	35
UpperWhisers	4575

Knime result :

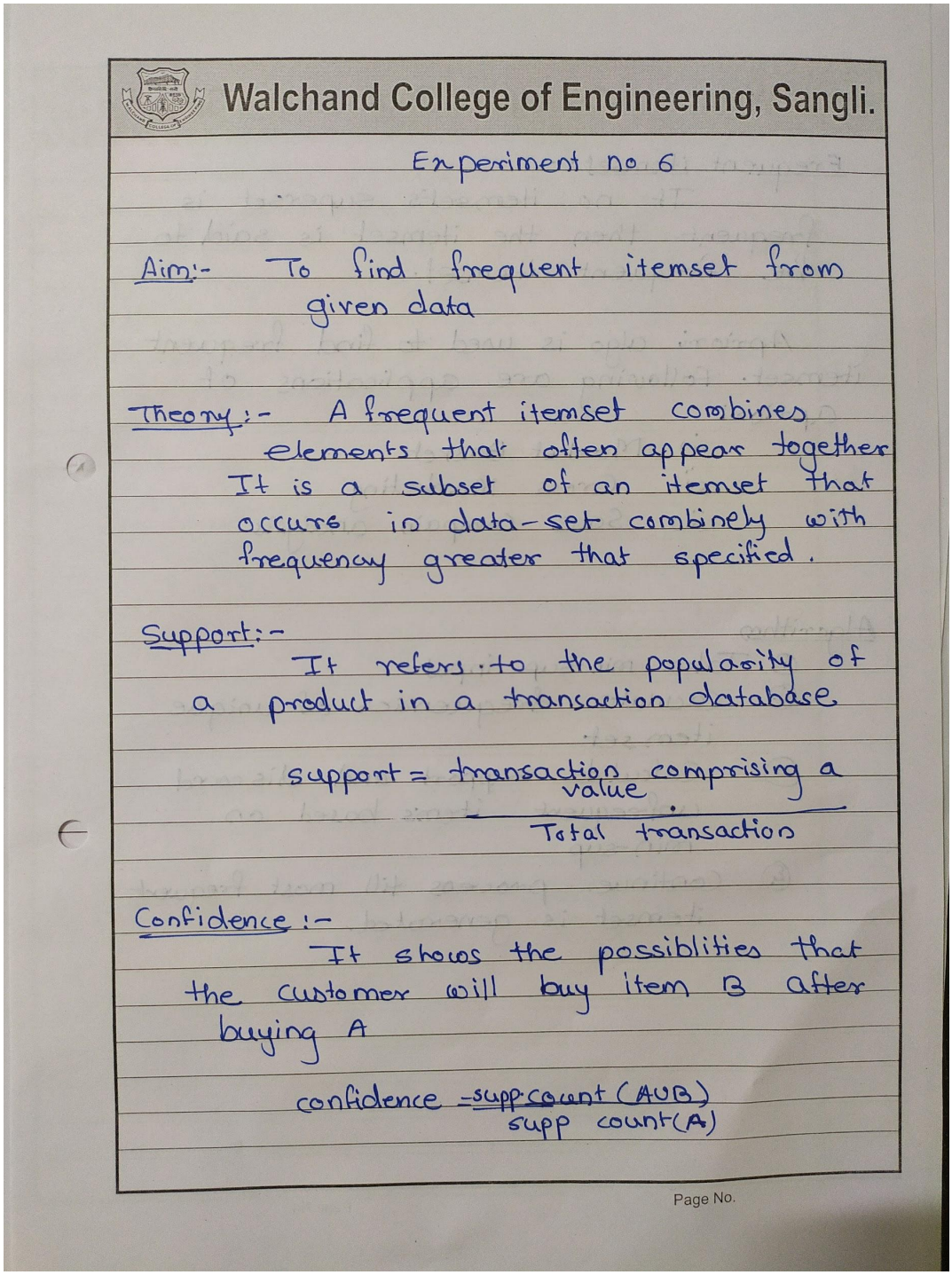


Conclusion:

5 number summary can be used to obtain the degree of spread of numeric data.

Experiment 6

Find frequent itemset from given transaction data



Walchand College of Engineering, Sangli.

Experiment no 6

Aim:- To find frequent itemset from given data

Theory:- A frequent itemset combines elements that often appear together It is a subset of an itemset that occurs in data-set combinely with frequency greater that specified.

Support:-

It refers to the popularity of a product in a transaction database.

$$\text{support} = \frac{\text{transaction comprising a value}}{\text{Total transaction}}$$

Confidence :-

It shows the possiblities that the customer will buy item B after buying A

$$\text{confidence} = \frac{\text{supp count}(A \cup B)}{\text{supp count}(A)}$$

Page No.

Frequent itemset :-

If no itemset's superset is frequent then the itemset is said to be frequent itemset.

Apriori algo is used to find frequent itemset. following are applications of apriori

(i) Market Basket
(ii) Cross Marketing
(iii) Sale Compain analysis

Algorithm

1. Take min-sup input
2. Caculate frequencies of unique item set
3. Calculate support and discard unfrequent items based on min-sup
4. Continue process till most frequent itemset is generated.

Dataset

1. a b c
2. a c
3. a d
4. b e f

Let min-sup be 20%. ⇒ 0.2
∴ min-freq = 0.2 × 4 = 0.8

Level 1

a → 3
b → 2
c → 2
d → 1
e → 1
f → 1

Level 2

ab → 1
ac → 2
ad → 1
ae → 0
af → 0
bc → 1
bd → 0
be → 1
bf → 1
cd → 0
cf → 0
de → 0
df → 0
ef → 1

Ignoring items whose frequency is < min-freq.

Page No.

∴ we have

ab, ac, ad, bc, be, bf, ef

Level 3.

We will get only two combinations whose freq is more than min-freq

abc → 1

bef → 1

∴ frequent itemset are

{abc}, {bef}

* Knime output

- [a,b,c] support of 25%.
- [b,e,f] support of 25%.

* Weka output

Largest itemset {a,b,c} {b,e,f}

Conclusion :- Frequent itemset mining is market basket analysis methodology that helps to find patterns in shopping behaviours. This frequent itemset mining can be used to find association rules.

Code :

```
#include <bits/stdc++.h>
#include <map>
using namespace std;

ifstream fin;
double minfre;
vector<set<string>> datatable;
set<string> products;
map<string, int> freq;

vector<string> wordsof(string str)
{
  vector<string> tmpset;
  string tmp = "";
  int i = 0;
  while (str[i])
  {
     if (isalnum(str[i]))
        tmp += str[i];
     else
     {
        if (tmp.size() > 0)
           tmpset.push_back(tmp);
        tmp = "";
     }
     i++;
  }

  if (tmp.size() > 0)
     tmpset.push_back(tmp);

  return tmpset;
}

string combine(vector<string> &arr, int miss)
{
  string str;
  for (int i = 0; i < arr.size(); i++)
```

```
      if (i != miss)
         str += arr[i] + " ";
   str = str.substr(0, str.size() - 1);
   return str;
}

set<string> cloneit(set<string> &arr)
{
   set<string> dup;
   for (set<string>::iterator it = arr.begin(); it != arr.end(); it++)
      dup.insert(*it);
   return dup;
}

set<string> apriori_gen(set<string> &sets, int k)
{
   set<string> set2;
   for (set<string>::iterator it1 = sets.begin(); it1 != sets.end(); it1++)
   {
      set<string>::iterator it2 = it1;
      it2++;
      for (; it2 != sets.end(); it2++)
      {
         vector<string> v1 = wordsof(*it1);
         vector<string> v2 = wordsof(*it2);

         bool alleq = true;
         for (int i = 0; i < k - 1 && alleq; i++)
            if (v1[i] != v2[i])
               alleq = false;

         v1.push_back(v2[k - 1]);
         if (v1[v1.size() - 1] < v1[v1.size() - 2])
            swap(v1[v1.size() - 1], v1[v1.size() - 2]);

         for (int i = 0; i < v1.size() && alleq; i++)
         {
            string tmp = combine(v1, i);
            if (sets.find(tmp) == sets.end())
               alleq = false;
```

```
            }

            if (alleq)
                set2.insert(combine(v1, -1));
        }
    }
    return set2;
}

int main()
{
    fin.open("apriori.csv", ios::in);

    if (!fin.is_open())
    {
        perror("Error in opening file : ");
    }
    cout << "Frequency % :";
    cin >> minfre;

    string str;
    while (!fin.eof())
    {
        getline(fin, str);
        vector<string> arr = wordsof(str);
        set<string> tmpset;
        for (int i = 0; i < arr.size(); i++)
            tmpset.insert(arr[i]);
        datatable.push_back(tmpset);

        for (set<string>::iterator it = tmpset.begin(); it != tmpset.end(); it++)
        {
            products.insert(*it);
            freq[*it]++;
        }
    }
    fin.close();

    cout << "No of transactions: " << datatable.size() << endl;
    minfre = minfre * datatable.size() / 100;
```

```
cout << "Min frequency:" << minfre << endl;

queue<set<string>::iterator> q;
for (set<string>::iterator it = products.begin(); it != products.end(); it++)
   if (freq[*it] < minfre)
      q.push(it);

while (q.size() > 0)
{
   products.erase(*q.front());
   q.pop();
}

int pass = 1;
cout << "\nFrequent " << pass++ << " -item set : \n";
for (set<string>::iterator it = products.begin(); it != products.end(); it++)
   cout << "{" << *it << "} " << freq[*it] << endl;

int i = 2;
set<string> prev = cloneit(products);

while (i)
{
   set<string> cur = apriori_gen(prev, i - 1);

   if (cur.size() < 1)
   {
      break;
   }

   for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
   {
      vector<string> arr = wordsof(*it);

      int tot = 0;
      for (int j = 0; j < datatable.size(); j++)
      {
         bool pres = true;
         for (int k = 0; k < arr.size() && pres; k++)
            if (datatable[j].find(arr[k]) == datatable[j].end())
```

```
                pres = false;
            if (pres)
                tot++;
        }
        if (tot >= minfre)
            freq[*it] += tot;
        else
            q.push(it);
    }

    while (q.size() > 0)
    {
        cur.erase(*q.front());
        q.pop();
    }

    // cout << "Flag : " << flag << "\n";
    bool flag = true;

    for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
    {
        vector<string> arr = wordsof(*it);

        if (freq[*it] < minfre)
            flag = false;
    }

    if (cur.size() == 0)
        break;

    cout << "\n\nFrequent " << pass++ << " -item set : \n";
    for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
        cout << "{" << *it << "} " << freq[*it] << endl;

    prev = cloneit(cur);
    i++;
}

ofstream fw("ferquentItemsOutput.csv", ios::out);
```

```
  for (auto it = prev.begin(); it != prev.end(); it++)
  {
     fw << "{" << *it << "}" << endl;
  }

  return 1;
}
```

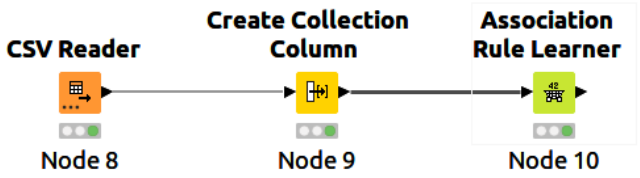
Input file :

	A	B	C
1	a	b	c
2	a	c	
3	a	d	
4	b	e	f
5			

Output :

```
Frequency % :20
No of transactions: 5
Min frequency:1

Frequent 1 -item set :
{a} 3
{b} 2
{c} 2
{d} 1
{e} 1
{f} 1


Frequent 2 -item set :
{a b} 1
{a c} 2
{a d} 1
{b c} 1
{b e} 1
{b f} 1
{e f} 1


Frequent 3 -item set :
{a b c} 1
{b e f} 1
```

Knime result :



Row ID	Suppo...	Items
item set 0	0.25	[a,d]
item set 1	0.25	[a,b,c]
item set 2	0.25	[b,e,f]
item set 3	0.5	[b]
item set 4	0.5	[a,c]
item set 5	0.75	[a]

Weka result:

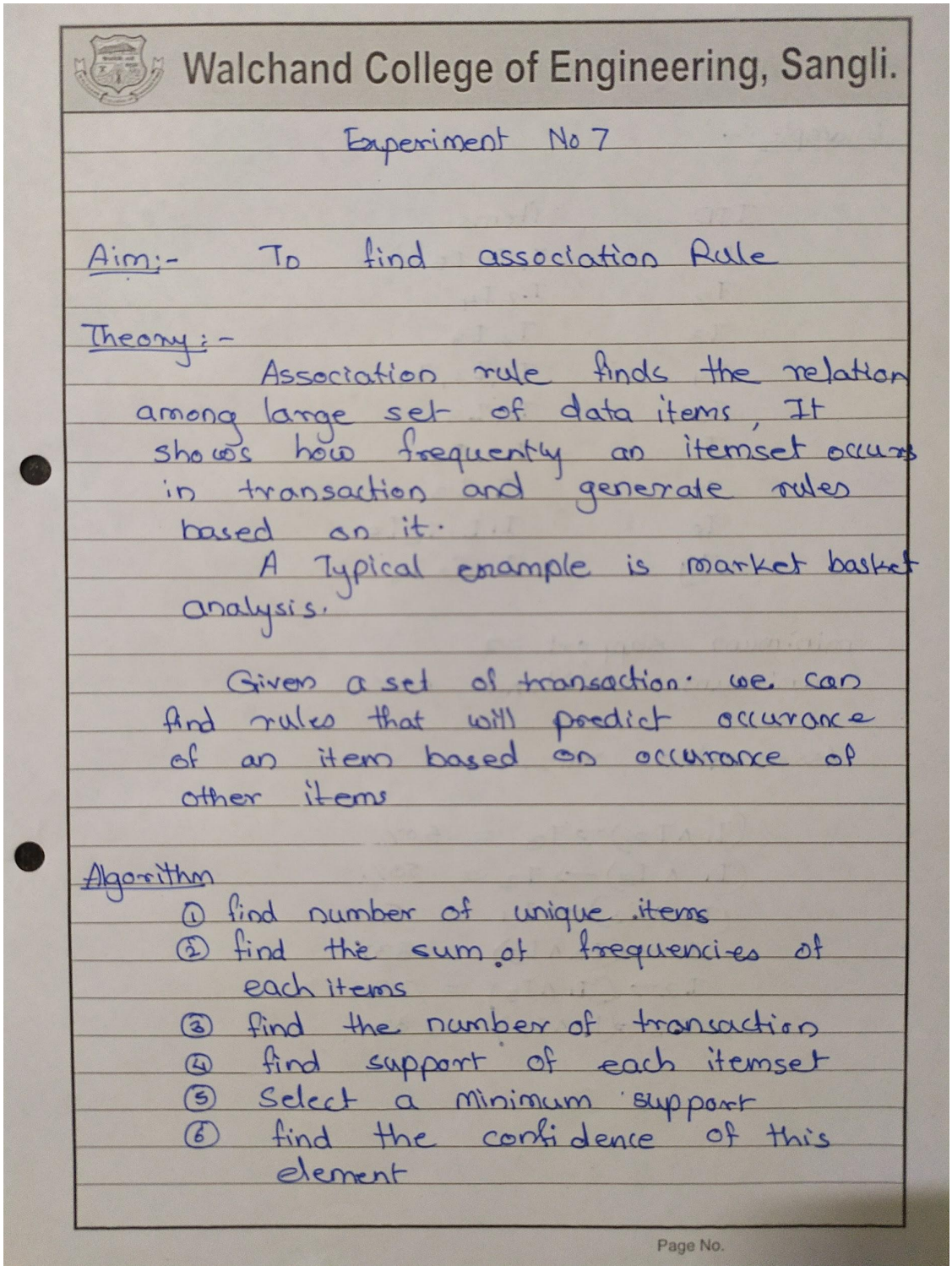
```
Generated sets of large itemsets:

Size of set of large itemsets L(1): 8

Large Itemsets L(1):
Attri1=a 3
Attri1=b 1
Attri2=b 1
Attri2=c 1
Attri2=d 1
Attri2=e 1
Attri3=c 1
Attri3=f 1

Size of set of large itemsets L(2): 8

Large Itemsets L(2):
Attri1=a Attri2=b 1
Attri1=a Attri2=c 1
Attri1=a Attri2=d 1
Attri1=a Attri3=c 1
Attri1=b Attri2=e 1
Attri1=b Attri3=f 1
Attri2=b Attri3=c 1
Attri2=e Attri3=f 1

Size of set of large itemsets L(3): 2

Large Itemsets L(3):
Attri1=a Attri2=b Attri3=c 1
Attri1=b Attri2=e Attri3=f 1
```

Conclusion:

Frequent itemset mining is a market basket analysis methodology that helps to find patterns in shopping behaviors. This frequent itemset mining can be used to find association rules.

Experiment 7

Extend program 6, to find association rules .



Walchand College of Engineering, Sangli.

Experiment No 7

Aim:- To find association Rule

Theory :-

Association rule finds the relation among large set of data items, It shows how frequently an itemset occurs in transaction and generate rules based on it.

A Typical example is market basket analysis.

Given a set of transaction. we can find rules that will predict occurance of an item based on occurance of other items

Algorithm

① find number of unique items
② find the sum of frequencies of each items
③ find the number of transaction
④ find support of each itemset
⑤ Select a minimum support
⑥ find the confidence of this element

Page No.

Example

Generating association rule for frequent itemset generated in 6th assignment

$\{a,b,c\}$ Let minConf = 60%.

(i) $\{a,b\} \rightarrow \{c\}$

$$conf = \frac{sup(abc)}{sup(ab)} = \frac{1}{1} \times 100 = 100\%.$$

(ii) $\{a,c\} \rightarrow \{b\}$

$$conf = \frac{sup(abc)}{sup(ac)} = \frac{1}{2} \times 100 = 50\%.$$

(iii) $\{b,c\} \rightarrow \{a\}$

$$conf = \frac{sup(abc)}{sup(bc)} = \frac{1}{1} \times 100 = 100\%.$$

(iv) $\{a\} \rightarrow \{b,c\}$

$$conf = \frac{sup(abc)}{sup(a)} = \frac{1}{3} \times 100 = 33\%.$$

(v) $\{b\} \rightarrow \{a,c\}$

$$conf = \frac{sup(abc)}{sup(a)} = \frac{1}{2} \times 100 = 50\%.$$

(vi) $\{\epsilon\} \rightarrow \{a,b\}$

$$conf = \frac{sup(abc)}{sup(\epsilon)} = \frac{1}{2} \times 100 = 50\%.$$

∴ based on minimum confidence = 60%.

$\{a,b\} \rightarrow c$ are strong rules

$\{b,c\} \rightarrow a$

Knime output.

confidence ≥ 60%.

[d] → [a]	conf → 100%.
[b,c] → a	conf → 100%.
[a,b] → c	conf → 100%.
[e,f] → b	conf → 100%.
[b,f] → e	conf → 100%.
[b,e] → f	conf → 100%.
[c] → a	conf → 100%.
[a] → c	conf → 66.7%.

conclusion:-

Association rules helps to find interesting relationships among the dataset It can be useful to predict customer behaviours and predict results.

Code :

```
#include <bits/stdc++.h>
#include <map>
using namespace std;

ifstream fin;
double minfre;
vector<set<string>> datatable;
set<string> products;
map<string, int> freq;
double confidence;

vector<string> wordsof(string str)
{
  vector<string> tmpset;
  string tmp = "";
  int i = 0;
  while (str[i])
  {
     if (isalnum(str[i]))
        tmp += str[i];
     else
     {
        if (tmp.size() > 0)
           tmpset.push_back(tmp);
        tmp = "";
     }
     i++;
  }

  if (tmp.size() > 0)
     tmpset.push_back(tmp);

  return tmpset;
}

string combine(vector<string> &arr, int miss)
{
  string str;
```

```
  for (int i = 0; i < arr.size(); i++)
     if (i != miss)
        str += arr[i] + " ";
  str = str.substr(0, str.size() - 1);
  return str;
}

set<string> cloneit(set<string> &arr)
{
  set<string> dup;
  for (set<string>::iterator it = arr.begin(); it != arr.end(); it++)
     dup.insert(*it);
  return dup;
}

set<string> apriori_gen(set<string> &sets, int k)
{
  set<string> set2;
  for (set<string>::iterator it1 = sets.begin(); it1 != sets.end(); it1++)
  {
     set<string>::iterator it2 = it1;
     it2++;
     for (; it2 != sets.end(); it2++)
     {
        vector<string> v1 = wordsof(*it1);
        vector<string> v2 = wordsof(*it2);

        bool alleq = true;
        for (int i = 0; i < k - 1 && alleq; i++)
           if (v1[i] != v2[i])
              alleq = false;

        v1.push_back(v2[k - 1]);
        if (v1[v1.size() - 1] < v1[v1.size() - 2])
           swap(v1[v1.size() - 1], v1[v1.size() - 2]);

        for (int i = 0; i < v1.size() && alleq; i++)
        {
           string tmp = combine(v1, i);
           if (sets.find(tmp) == sets.end())
```

```
                alleq = false;
            }
            if (alleq)
                set2.insert(combine(v1, -1));
        }
    }
    return set2;
}

int countOccurences(vector<string> v)
{

    int count = 0;

    for (auto s : datatable)
    {
        bool present = true;

        for (auto x : v)
        {
            if (s.find(x) == s.end())
            {
                present = false;
                break;
            }
        }

        if (present)
            count++;
    }

    return count;
}

ofstream fw1("assocationRulesOutput.csv", ios::out);

void subsets(vector<string> items, vector<string> v1, vector<string> v2, int idx)
{

    if (idx == items.size())
```

```
{
   if (v1.size() == 0 || v2.size() == 0)
      return;

   int count1 = countOccurences(items); // Total support
   int count2 = countOccurences(v1);

   double conf = (((double)count1) / count2) * 100;

   if (conf >= confidence)
   {

      fw1 << "{ ";
      for (auto s : v1)
      {
         fw1 << s << " ";
      }

      fw1 << "} ,"
         << "-> "
         << ", {";

      for (auto s : v2)
      {
         fw1 << s << " ";
      }

      fw1 << "} , " << conf << endl;
   }

   return;
}

v1.push_back(items[idx]);
subsets(items, v1, v2, idx + 1);

v1.pop_back();
v2.push_back(items[idx]);
subsets(items, v1, v2, idx + 1);
v2.pop_back();
```

```
}

void generateAssociationRules(set<string> freqItems)
{
  for (auto it = freqItems.begin(); it != freqItems.end(); it++)
  {
     vector<string> items = wordsof(*it);

     subsets(items, {}, {}, 0);
  }
}

int main()
{
  fin.open("apriori.csv", ios::in);

  if (!fin.is_open())
  {
     perror("Error in opening file : ");
  }
  cout << "Enter Support % :";
  cin >> minfre;

  cout << "Enter Confidence % : ";
  cin >> confidence;

  string str;
  while (!fin.eof())
  {
     getline(fin, str);
     vector<string> arr = wordsof(str);
     set<string> tmpset;
     for (int i = 0; i < arr.size(); i++)
        tmpset.insert(arr[i]);
     datatable.push_back(tmpset);

     for (set<string>::iterator it = tmpset.begin(); it != tmpset.end(); it++)
     {
        products.insert(*it);
        freq[*it]++;
```

```
    }
}
fin.close();

cout << "No of transactions: " << datatable.size() << endl;
minfre = minfre * datatable.size() / 100;
cout << "Min frequency:" << minfre << endl;

queue<set<string>::iterator> q;
for (set<string>::iterator it = products.begin(); it != products.end(); it++)
    if (freq[*it] < minfre)
        q.push(it);

while (q.size() > 0)
{
    products.erase(*q.front());
    q.pop();
}

int pass = 1;
cout << "\nFrequent " << pass++ << " -item set : \n";
for (set<string>::iterator it = products.begin(); it != products.end(); it++)
    cout << "{" << *it << "} " << freq[*it] << endl;

int i = 2;
set<string> prev = cloneit(products);

while (i)
{
    set<string> cur = apriori_gen(prev, i - 1);

    if (cur.size() < 1)
    {
        break;
    }

    for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
    {
        vector<string> arr = wordsof(*it);
```

```
    int tot = 0;
    for (int j = 0; j < datatable.size(); j++)
    {
        bool pres = true;
        for (int k = 0; k < arr.size() && pres; k++)
            if (datatable[j].find(arr[k]) == datatable[j].end())
                pres = false;
        if (pres)
            tot++;
    }
    if (tot >= minfre)
        freq[*it] += tot;
    else
        q.push(it);
}

while (q.size() > 0)
{
    cur.erase(*q.front());
    q.pop();
}

// cout << "Flag : " << flag << "\n";
bool flag = true;

for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
{
    vector<string> arr = wordsof(*it);

    if (freq[*it] < minfre)
        flag = false;
}

if (cur.size() == 0)
    break;

cout << "\n\nFrequent " << pass++ << " -item set : \n";
for (set<string>::iterator it = cur.begin(); it != cur.end(); it++)
    cout << "{" << *it << "} " << freq[*it] << endl;
```

```
    prev = cloneit(cur);
    i++;
  }

  generateAssociationRules(prev);

  return 1;
}
```

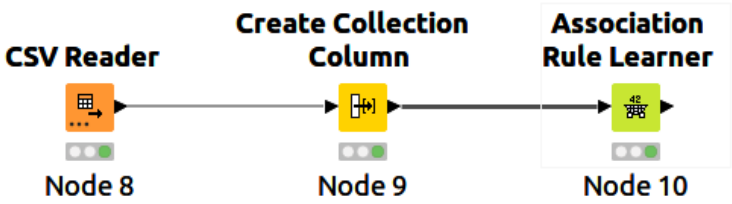
Input file :

	A	B	C
1	a	b	c
2	a	c	
3	a	d	
4	b	e	f
5			

Output :

{	a	b	}		->			{c	}		100
{	b	c	}		->			{a	}		100
{	b	e	}		->			{f	}		100
{	b	f	}		->			{e	}		100
{	e	f	}		->			{b	}		100
{	e	}		->			{b	f	}		100
{	f	}		->			{b	e	}		100

Knime result :



Row ID	Support	Confid...	Lift	Conse...	implies	Items
rule0	0.25	1	1.333	a	<---	[d]
rule1	0.25	1	1.333	a	<---	[b,c]
rule2	0.25	1	2	c	<---	[a,b]
rule3	0.25	1	2	b	<---	[e,f]
rule4	0.25	1	4	e	<---	[b,f]
rule5	0.25	1	4	f	<---	[b,e]
rule6	0.5	1	1.333	a	<---	[c]
rule7	0.5	0.667	1.333	c	<---	[a]

Weka Result:

```
Large Itemsets L(3):
Attri1=a Attri2=b Attri3=c 1
Attri1=b Attri2=e Attri3=f 1

Best rules found:

 1. Attri2=b 1 ==> Attri1=a 1    <conf:(1)> lift:(1.33) lev:(0.06) [0] conv:(0.25)
 2. Attri2=c 1 ==> Attri1=a 1    <conf:(1)> lift:(1.33) lev:(0.06) [0] conv:(0.25)
 3. Attri2=d 1 ==> Attri1=a 1    <conf:(1)> lift:(1.33) lev:(0.06) [0] conv:(0.25)
 4. Attri3=c 1 ==> Attri1=a 1    <conf:(1)> lift:(1.33) lev:(0.06) [0] conv:(0.25)
 5. Attri2=e 1 ==> Attri1=b 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
 6. Attri1=b 1 ==> Attri2=e 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
 7. Attri3=f 1 ==> Attri1=b 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
 8. Attri1=b 1 ==> Attri3=f 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
 9. Attri3=c 1 ==> Attri2=b 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
10. Attri2=b 1 ==> Attri3=c 1    <conf:(1)> lift:(4) lev:(0.19) [0] conv:(0.75)
```

Conclusion:

Association rules help to find interesting relationships among the dataset It can be used to predict customer behaviors under market basket analysis

Experiment No 8

Find correlation between items/entities.

Walchand College of Engineering, Sangli.

Experiment no 8

Aim:- To find co-relation between item entites

Theory:-

It is a statastical meathod to measure strength of a relationship betn two attribute

The occurance of item 'A' is independent of occurance of item B if

$$P(A \cup B) = P(A)\,(P(B))$$

otherwise items A & B are dependent on each other

$$\text{correlation}(A,B) = \frac{P(A \cup B)}{P(A)\,P(B)}$$

if $corr(A,B) < 1 \Rightarrow$ A & B are negatively co-related

if $corr(A,B) > 1 \Rightarrow$ A & B are positively co-related.

Page No.

Algorithm:

1. Calculate P(A), P(B)
2. Calculate P(A∪B)
3. Calculate

$$corr = \frac{P(A \cup B)}{P(A)P(B)}$$

Example

A	1	1	1	1	0	0
B	1	1	0	0	0	0

$P(A \cup B) = 2/6$

$P(A) = 4/6$

$P(B) = 2$

$$corr(A,B) = \frac{P(A \cup B)}{P(A)\,P(B)} = \frac{2/6}{4/6 \times 2/6} = \frac{6}{4} = 1.5$$

$corr(A,B) > 1 \Rightarrow$ A & B are positively correlated.

Conclusion:- With help of correlation the study of closeness of relationship i.e degree to which variables are associated with each other can be carried out.

Code:-

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["data"];
var data = xlsx.utils.sheet_to_json(ws);
var corr;
var a = 0,
 b = 0,
 a_b = 0;
for (var i = 0; i < data.length; i++) {
 if (data[i].A == 1 && data[i].B == 1) a_b++;
 if (data[i].A == 1) a++;
 if (data[i].B == 1) b++;
}
corr = a_b / (a * b);
console.log(corr);
var newdata = [];
var obj = { correaltion_Between_A_and_B: corr };
newdata.push(obj);
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(newdata);
xlsx.utils.book_append_sheet(newWB, newWS, "Correlation");
xlsx.writeFile(newWB, "Correlation.xlsx")
```

Input:-

A	B
1	1
1	1
1	0
1	0
1	1
0	0
0	1
0	1

Output:-

correaltion_Between_A_and_B
1.5

Conclusion:

With help of correlation, the study of the closeness of the relation between two variables can be carried out i.e degree to which two variables are associated with each other.

Experiment 9

Distance and cluster

Walchand College of Engineering, Sangli.

Experiment no 9 :-
Distance and cluster

Aim :- To obtain cluster by euclidean distance function

Theory :-

Distance matrix is symetric matrix with value on diagonal 0 because distance from itself is 0.

Significance of distance matrix is to indicate magnitude of dissimilarity bet^n two elements. Higher the distance than greater the dissimilarity

Algorithm :-

1. Asume one of the point to be the centre.
2. Calculate the distance of each point from this centre and also from each other points
3. Create the distance matrix from obtained values
4. Calculate the distance of all the points from this new centre.

Page No.

Example:-

X	Y
53	34
35	87
68	34

Initial centre (x) = $\frac{53+35+68}{3} = 52$

Initial centre (y) = $\frac{34+87+34}{3} = 51$

$\Rightarrow$ Centre (52, 51)

- Finding the distance of all the points with cluster centre

Dist of (53, 34) from (52, 51)
$= \sqrt{(53-52)^2 + (34-51)^2}$
$= 17.03$

Dist of (35, 87) from (52, 51)
$= \sqrt{(35-52)^2 + (87-51)^2}$
$= 39.812$

Dist of (68, 34) from (52, 51)
$= \sqrt{(68-52)^2 + (34-52)^2}$
$= 23.345$

Distance matrix

	A	B	C	centre
A	0			
B	55.97	0		
C	15	62.43	0	
centre	17.69	39.21	23.83	0

conclusion :-

Successfully formed the dissimilar matrix using euclidean distance formula. Clustering of data can be done by using euclidean distance function & distance matrix.

Points with close similarity can be merged in single cluster.

Code:

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["data"];

var data = xlsx.utils.sheet_to_json(ws);

var x_centre = 0,
  y_centre = 0;
for (var i = 0; i < data.length; i++) {
  x_centre += data[i].x;
  y_centre += data[i].y;
  data[i].isIncluded = 0;
}
x_centre = x_centre / data.length;
y_centre = y_centre / data.length;
console.log("Centre(" + x_centre + "," + y_centre + ")");
data.push({ point: "centre", x: x_centre, y: y_centre });
console.log(data);

var index = 0;
var matrix = [];
var j = 0;
for (var i = 0; i < data.length; i++) {
  var obj = {};

  for (j = 0; j <= i; j++) {
    var eucDist = Math.sqrt(
      (data[j].x - data[i].x) * (data[j].x - data[i].x) +
        (data[j].y - data[i].y) * (data[j].y - data[i].y)
    );
    obj["point"] = data[i].point;
```

```
    obj[data[j].point] = eucDist;
  }
  matrix.push(obj);
}

console.log(matrix);
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(matrix);
xlsx.utils.book_append_sheet(newWB, newWS, "matrix" + index);
xlsx.writeFile(newWB, "matrix" + index + ".xlsx");
```

Input:

point	x	y
A	53	34
B	35	87
C	68	34

Output:

point	A	B	C	centre
A	0			
B	55.97321	0		
C	15	62.43397	0	
centre	17.69495	39.21026	23.83508	0

Knime cluster center

Row ID	D x	D y
cluster_0	52	51.667

Row ID	x	y	Cluster
Row0	53	34	cluster_1
Row1	35	87	cluster_0
Row2	68	34	cluster_1

Weka output:

```
Number of clusters selected by cross validation: 1
Number of iterations performed: 2


            Cluster
Attribute         0
                (1)
====================
x
  mean            52
  std. dev.  13.4907

y
  mean       51.6667
  std. dev.  24.9844



Time taken to build model (full training data) : 0.02 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      3 (100%)


Log likelihood: -8.65813
```

Conclusion:

Clustering can be done using a distance matrix and distance function. Points close together can be merged to form a cluster. Points with closeness can be merged to gether to form a single cluster.

Experiment 10

Write a program for Agglomerative Hierarchical clustering using single linkage method

Walchand College of Engineering, Sangli.

Experiment No 10

Aim:- Agglomerative heirarchical clustering using single linkage method.

Theory:- This methods works by grouping data into a tree of clusters. Initially it treats every data as a seperate. Then it repeatively executes steps to identify close points together and merge in clusters

Algorithm:-

1. Consider every data point as a individual cluster
2. Merge cluster which are highly similar to each other.
3. Calculate prominity matrix for newly formed cluster set
4. Repeate step 3 & 4 till single cluster is formed.

Page No.

Example

	A	B	C	D	E	F
A	0					
B	16	0				
C	47	37	0			
D	72	57	40	0		
E	77	65	30	31	0	
F	79	66	35	23	10	0

Merging E&F (single linkage → min of distances)

	A	B	C	D	EF
A	0				
B	16	0			
C	47	37	0		
D	72	57	40	0	
EF	77	65	30	23	0

Merging A&B (single linkage)

	AB	C	D	EF
AB	0			
C	37	0		
D	57	40	0	
EF	65	30	23	0

Merging D and EF

	AB	C	DEF
AB	0		
C	37	0	
DEF	57	30	0

Merging DEF & C

$d(AB \rightarrow C) = \min(d(A,C), d(B,C))$

$= \min(47, 37)$

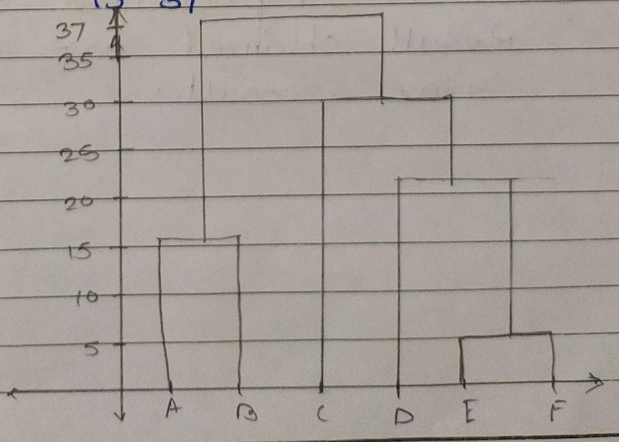
$= 37$

similarly compute for all

	AB	CDEF
AB	0	
CDEF	37	0

Now distance betn clusters AB and CDEF is 37

Dendogram



Page No.

Divisive:-

Divisive heirarchical clustering is opposite to ggolomerative. Here we take account iteration, we seperate dat points from the cluster

Knime output:-

Dendogram similar to above calculated.

conclusion:-

Data is forming cluster which is represented in dendogram Result obtained are similar to knime result.

This method starts placing each object in individually and merges them based on their closeness to form single cluster

A Bottom-up approach.

Can be visualized using dendograms

Code:

A.Generate distance matrix from given points

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["data"];

var data = xlsx.utils.sheet_to_json(ws);

console.log("data input\n" + data);
console.log(data);
console.log("\n");

var index = 1;
var matrix = [];
var j = 0;
for (var i = 0; i < data.length; i++) {
  var obj = {};

  for (j = 0; j <= i; j++) {
    var eucDist = Math.sqrt(
      (data[j].x - data[i].x) * (data[j].x - data[i].x) +
        (data[j].y - data[i].y) * (data[j].y - data[i].y)
    );
    obj["point"] = data[i].point;
    obj[data[j].point] = eucDist;
  }
  matrix.push(obj);
}

console.log("matrix" + index + "\n");
console.log(matrix);
```

```
console.log("\n");
var newWB = xlsx.utils.book_new();
var newWS = xlsx.utils.json_to_sheet(matrix);
xlsx.utils.book_append_sheet(newWB, newWS, "matrix" + index);
xlsx.writeFile(newWB, "matrix" + ".xlsx");
```

Agglomirative :

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();

var index = 1;
var wb = xlsx.readFile("matrix.xlsx");
var ws = wb.Sheets["matrix" + index];

var datamain = xlsx.utils.sheet_to_json(ws);

while (index < datamain.length - 1) {
  var outputobj = {};
  var ws = wb.Sheets["matrix" + index];
  var data = xlsx.utils.sheet_to_json(ws);
  var minfirst = Number.MAX_VALUE;
  var minptsecond;
  var minptfirst;

  for (var i = 0; i < data.length; i++) {
    for (var j = 1; j <= i; j++) {
      const key = Object.keys(data[i])[j];
      if (data[i][key] != 0 && minfirst > data[i][key]) {
        minfirst = data[i][key];
        minptfirst = i;
        minptsecond = j - 1;
      }
    }
```

```
}

var matrix = [];
var j = 0;
for (var i = 0; i < data.length; i++) {
  var obj = {};
  for (j = 0; j <= i + 1; j++) {
    const key = Object.keys(data[i])[j];
    if (key != "point") {
      if (
        (data[i].point == data[minptfirst].point ||
          data[i].point == data[minptsecond].point) &&
        j != minptsecond + 1 &&
        j != minptfirst + 1
      ) {
        var d1 = Number.MAX_VALUE;
        var d2 = Number.MAX_VALUE;
        var d3 = Number.MAX_VALUE;
        var d4 = Number.MAX_VALUE;
        if (data[minptfirst][key]) {
          d1 = data[minptfirst][key];
        }
        if (data[minptsecond][key]) {
          d2 = data[minptsecond][key];
        }
        const key1 = Object.keys(data[j - 1])[minptfirst + 1];
        if (data[j - 1][key1]) {
          d3 = data[j - 1][key1];
        }
        const key2 = Object.keys(data[j - 1])[minptsecond + 1];
        if (data[j - 1][key2]) {
          d4 = data[j - 1][key2];
        }
        obj["point"] = data[minptfirst].point + data[minptsecond].point;
```

```
  obj[key] = Math.min(d1, d2, d3, d4);
 } else if (
  (data[i].point == data[minptfirst].point ||
    data[i].point == data[minptsecond].point) &&
  (j == minptsecond + 1 || j == minptfirst + 1)
 ) {
  if (
    !Number.isNaN(
      Math.min(data[minptfirst][key], data[minptsecond][key])
    )
  ) {
    obj["point"] = data[minptfirst].point + data[minptsecond].point;
    obj[data[minptfirst].point + data[minptsecond].point] = 0;
  }
 } else if (
  data[i].point != data[minptfirst].point &&
  data[i].point != data[minptsecond].point &&
  (j == minptfirst + 1 || j == minptsecond + 1)
 ) {
  const key1 = Object.keys(data[data.length - 1])[minptfirst + 1];
  const key2 = Object.keys(data[data.length - 1])[minptsecond + 1];
  if (Math.min(data[i][key1], data[i][key2])) {
    obj["point"] = data[i].point;
    obj[data[minptfirst].point + data[minptsecond].point] = Math.min(
      data[i][key1],
      data[i][key2]
    );
  } else if (
    !Number.isNaN(
      Math.min(data[minptfirst][key], data[minptsecond][key])
    )
  ) {
    obj["point"] = data[i].point;
    obj[data[minptfirst].point + data[minptsecond].point] = Math.min(
```

```
            data[minptfirst][key],
            data[minptsecond][key]
          );
        }
      } else {
        const key = Object.keys(data[i])[j];
        obj["point"] = data[i].point;
        obj[key] = data[i][key];
      }
    }
  }
  matrix.push(obj);
}
var obj1 = Object.keys(matrix[minptfirst]).size;
var obj2 = Object.keys(matrix[minptsecond]).size;
if (obj1 < obj2) {
  matrix.splice(minptfirst, 1);
} else {
  matrix.splice(minptsecond, 1);
}

var newWS = xlsx.utils.json_to_sheet(matrix);
index++;
xlsx.utils.book_append_sheet(wb, newWS, "matrix" + index);
xlsx.writeFile(wb, "matrix" + ".xlsx");
}
```

Input:

point	A	B	C	D	E	F
A	0					
B	16	0				
C	47	37	0			
D	72	57	40	0		
E	77	65	30	31	0	
F	79	66	35	23	10	0

Output:

Matrix 1

point	A	B	C	D	FE
A	0				
B	16	0			
C	47	37	0		
D	72	57	40	0	
FE	77	65	30	23	0

Matrix 2

point	BA	C	D	FE
BA	0			
C	37	0		
D	57	40	0	
FE	65	30	23	0

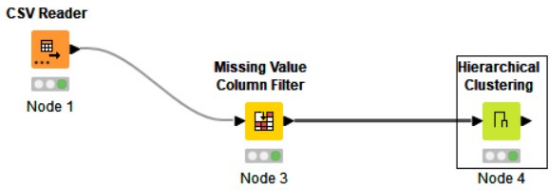
Matrix 3

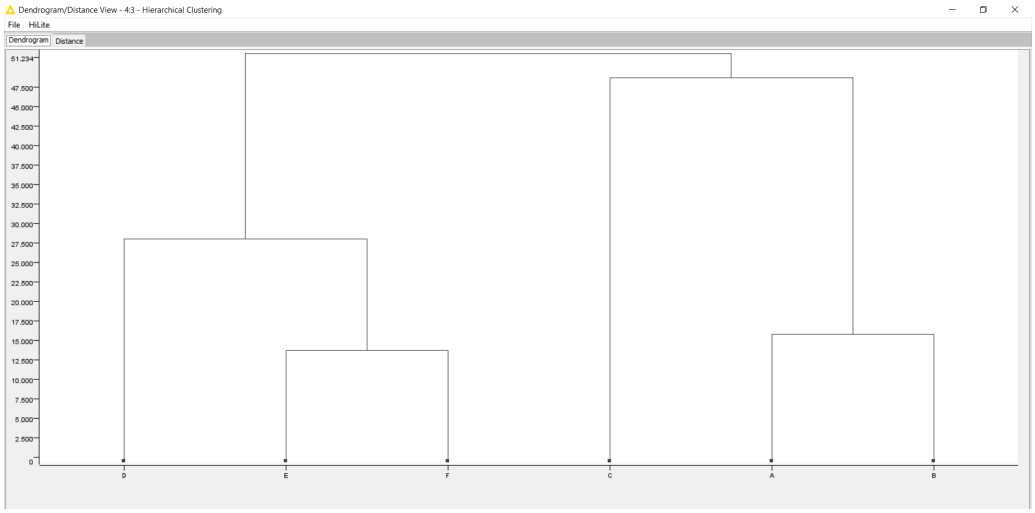
point	BA	C	FED
BA	0		
C	37	0	
FED	57	30	0

Matrix 4

point	BA	FEDC					
BA	0						
FEDC	37	0					

Knime Dendogram:





Conclusion:

This bottom-up approach places each object in the individual cluster and gradually merges them based on their closeness to form a single cluster. Can be visualized by using Dendrogram.

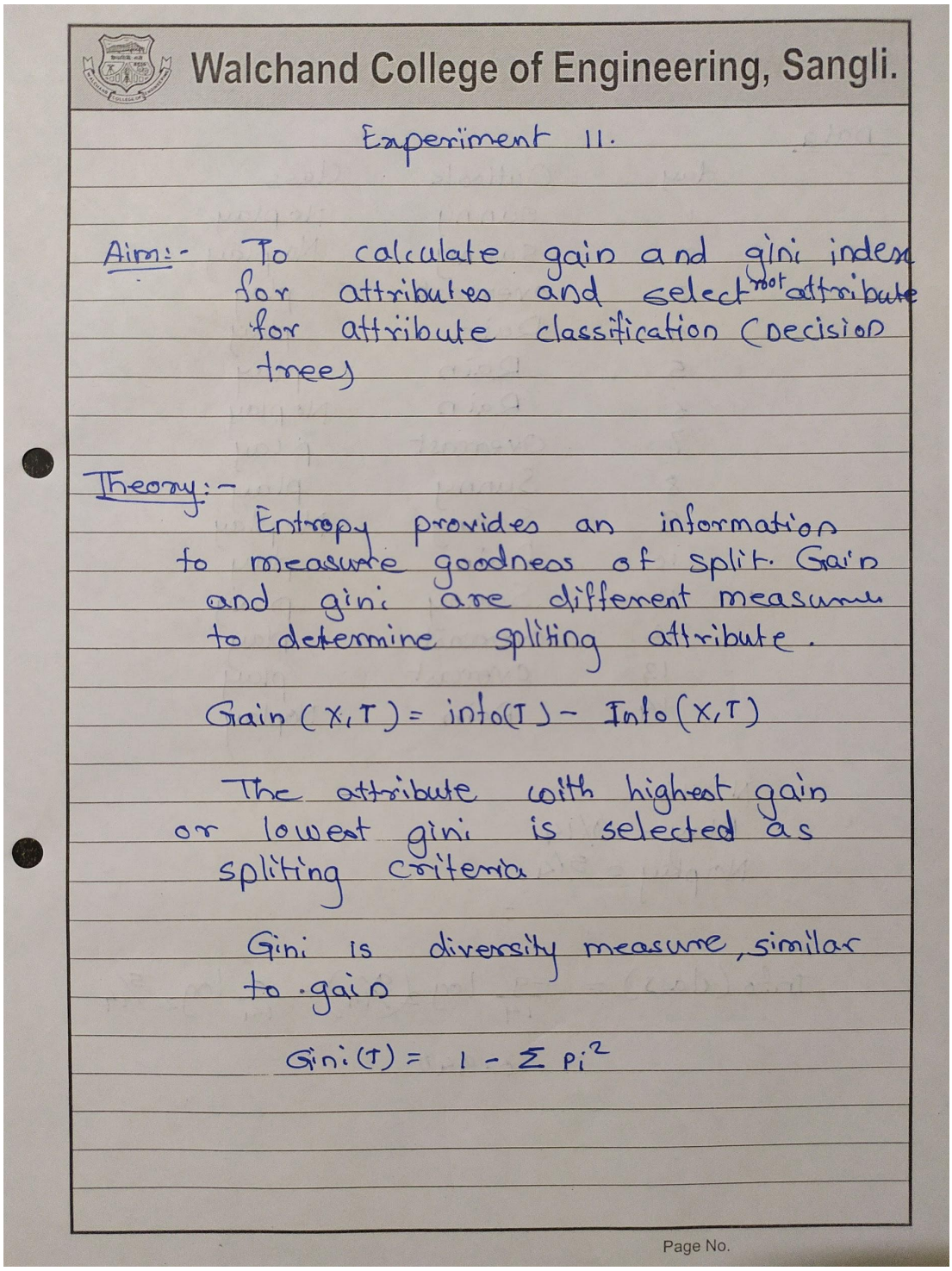
Experiment 11

Attribute for classification, Write a program to find

A. Gain

B. Gini index

For categorical and numerical values



Walchand College of Engineering, Sangli.

Experiment 11.

Aim:- To calculate gain and gini index for attributes and select root attribute for attribute classification (Decision trees)

Theory:-

Entropy provides an information to measure goodness of split. Gain and gini are different measures to determine spliting attribute.

$Gain(X,T) = info(T) - Info(X,T)$

The attribute with highest gain or lowest gini is selected as spliting criteria

Gini is diversity measure, similar to gain

$Gini(T) = 1 - \Sigma p_i^2$

Page No.

Data

day	Outlook	Class
1	Sunny	No play
2	Sunny	No play
3	Overcast	play
4	Rain	play
5	Rain	play
6	Rain	No play
7	Overcast	play
8	Sunny	play
9	Sunny	No play
10	Rain	play
11	Sunny	play
12	Overcast	play
13	Overcast	play
14	Rain	No play

N = 14

Play = 9/14

No play = 5/14

$$\text{Info(class)} = \frac{-9}{14}\log_2(9/14) - \frac{5}{14}\log_2 5/14$$

$$= 0.940$$

outlook	n	play	noplay
Sunny	5/14	2/5	3/5
overcast	4/14	4/4	0/4
Rain	5/14	3/5	2/5

$$\text{Infogain} = \sum \frac{T_i}{T} \text{info}(T_i)$$

$$= \frac{5}{14}\left[\text{info}\left(\frac{3}{5}, \frac{3}{5}\right)\right] + \frac{4}{14}\left[\text{info}\left(\frac{4}{4}, \frac{0}{4}\right)\right] + \frac{5}{14}\left[\text{info}\,\frac{3}{5}, \frac{2}{5}\right]$$

$$= \frac{+5}{14}\left[-\frac{2}{3}\log\frac{2}{3} - \frac{3}{5}\log\frac{3}{5}\right] + \frac{4}{14}\left[\frac{-4}{4}\log\frac{4}{4} + 0\right] + \frac{5}{14}\left[-\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5}\right]$$

$$= 0.692$$

Info (outlook, T) = 0.692

Gain (outlook = Info (T) − Info (outlook, T)

= 0.940 − 0.692

= 0.248

Gain (outlook) = 0.248

$$Gini(outlook) = \frac{5}{14} gini\left(\frac{2}{3}, \frac{3}{5}\right) + \frac{4}{14} gini\left(\frac{4}{4}, \frac{0}{4}\right) + \frac{5}{14} gini\left(\frac{3}{5}, \frac{2}{5}\right)$$

$$= \frac{5}{14}\left(1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2\right) + \frac{4}{14}\left(1 - 1^2 - 0^2\right) + \frac{5}{14}\left(1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2\right)$$

$$= 0.3428$$

Gini (outlook) = 0.3428

conclusion:- with help of gini, gain we can identify attribute for best split. Decision tree can be generated using these values.

Page No.

Code:

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["data"];

var data = xlsx.utils.sheet_to_json(ws);
var attri = new Map();
var count = new Map();
for (var i = 0; i < data.length; i++) {
  for (var key in data[i]) {
    var name = data[i][key] + data[i].class;
    if (!count.has(name)) {
      count.set(name, 1);
    } else {
      count.set(name, count.get(name) + 1);
    }
    if (!attri.has(key)) {
      attri.set(key, {});
    }
    var obj = attri.get(key);
    if (!(data[i][key] in obj)) obj[data[i][key]] = 0;
    obj[data[i][key]]++;
    attri.set(key, obj);
  }
}
console.log(count);
console.log(attri);

for (let [key, value] of attri.entries()) {
  if (key != "class") {
    var gini = 0;
    for (attribute in value) {
      var temp = value[attribute] / data.length;
      var temp2 = 1;
      var classObj = attri.get("class");
      for (classname in classObj) {
        var name = attribute + classname;
        if (count.get(name)) {
          temp2 =
```

```
          temp2 -
          (count.get(name) / value[attribute]) *
            (count.get(name) / value[attribute]);
        }
      }
      gini = gini + temp * temp2;
    }
    console.log("gini index for " + key + " is " + gini);
  }
}
```

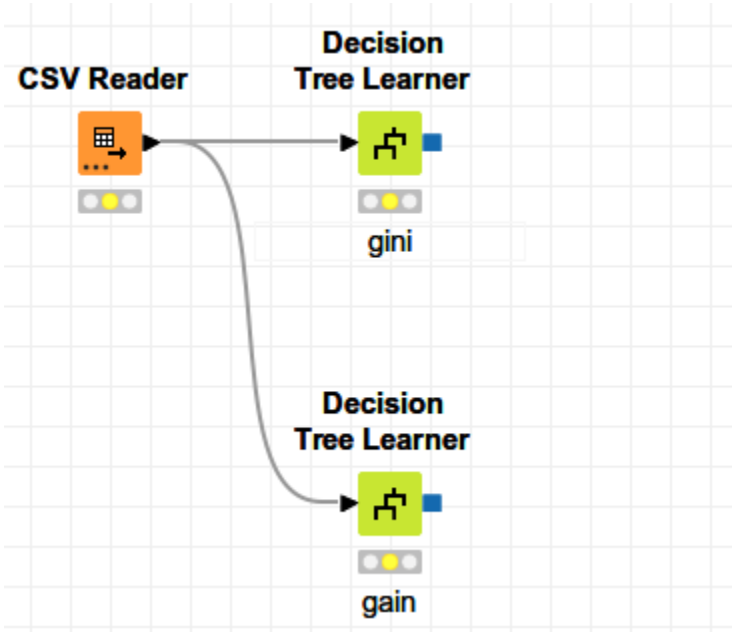
Input:

outlook	class
sunny	No play
sunny	No play
overcast	play
rainy	play
rainy	play
rainy	No play
overcast	play
sunny	No play
sunny	play
rainy	play
sunny	play
overcast	play
overcast	play
rainy	No play

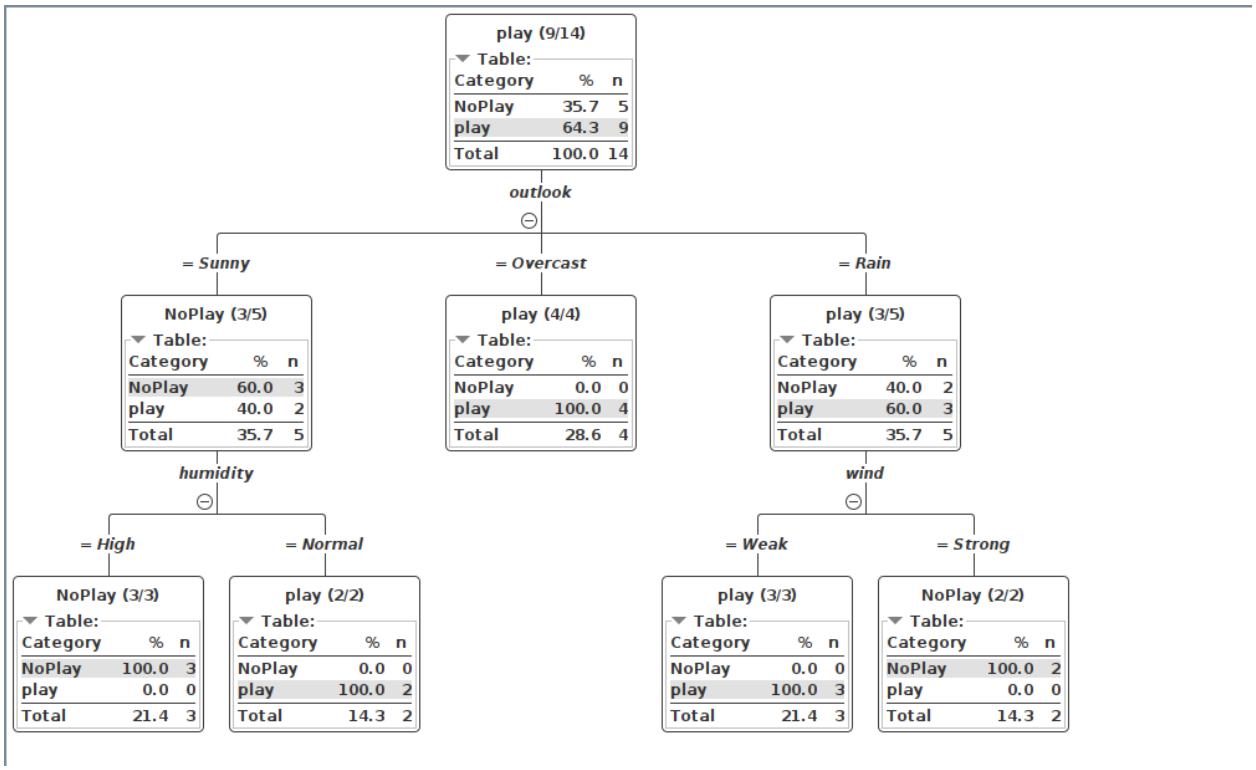
Output:

```
pranav-bhosale@pranav-bhosale:~/Desktop/Btech/DM LAB/DM Exp/Exp11$ node gini.js
Map(7) {
  'sunnyNo play' => 3,
  'No playNo play' => 5,
  'overcastplay' => 4,
  'playplay' => 9,
  'rainyplay' => 3,
  'rainyNo play' => 2,
  'sunnyplay' => 2
}
Map(2) {
  'outlook' => { sunny: 5, overcast: 4, rainy: 5 },
  'class' => { 'No play': 5, play: 9 }
}
gini index for outlook is 0.34285714285714286
pranav-bhosale@pranav-bhosale:~/Desktop/Btech/DM LAB/DM Exp/Exp11$
```

Knime:



Decision tree



Weka:

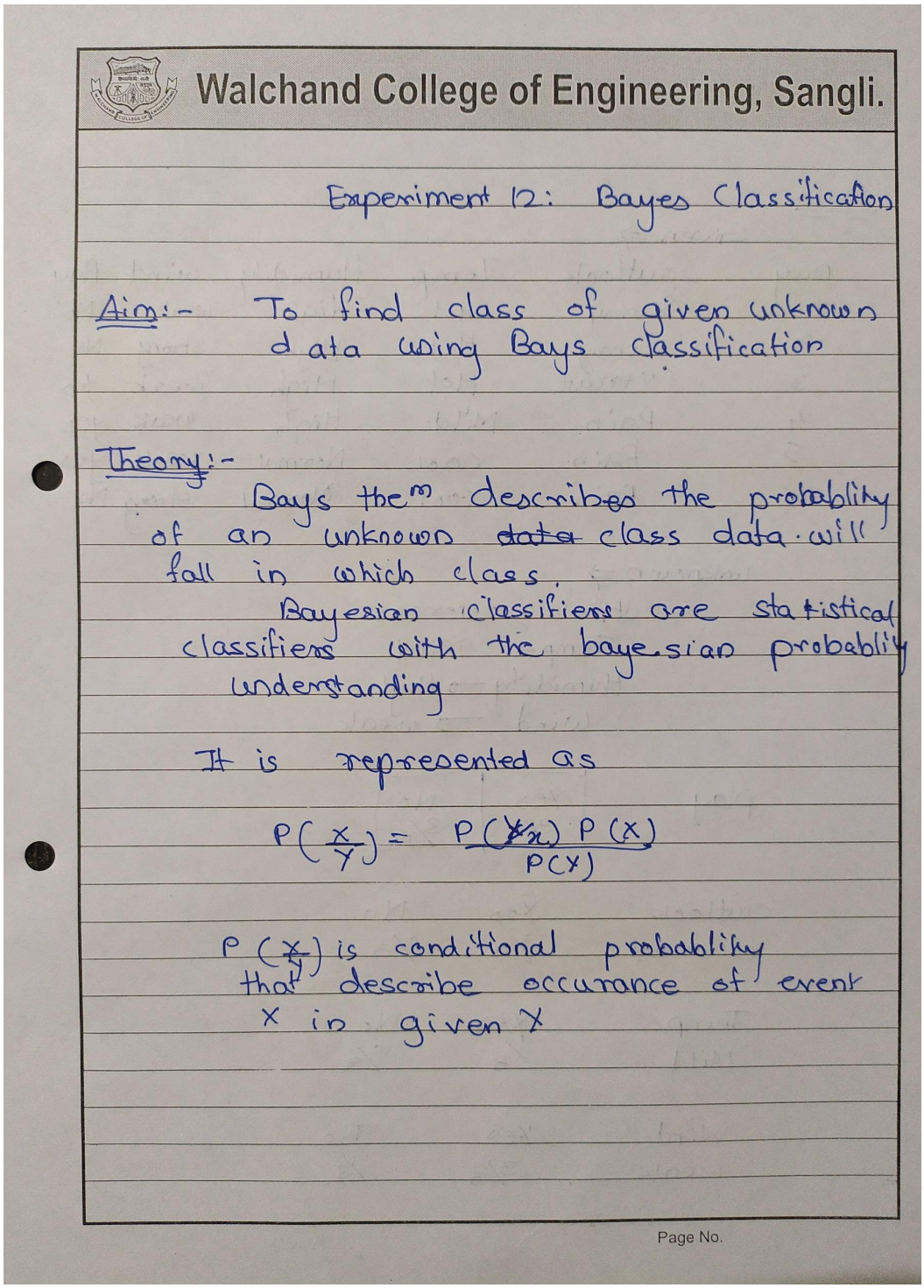
```
Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 5 playGame):
        Information Gain Ranking Filter

Ranked attributes:
 0.2467  1 outlook
 0.1518  3 humidity
 0.0481  4 wind
 0.0292  2 temp

Selected attributes: 1,3,4,2 : 4
```

Conclusion:

With help of Gini and gain we can identify attributes for the best split.
A decision tree can be generated using these values.

Experiment 12

WAP for Baye's classification



Walchand College of Engineering, Sangli.

Experiment 12: Bayes Classification

Aim:- To find class of given unknown data using Bays classification

Theory:-

Bays the^{m} describes the probability of an unknown ~~data~~ class data will fall in which class.

Bayesian classifiers are statistical classifiers with the bayesian probability understanding

It is represented as

$$P\left(\frac{X}{Y}\right) = \frac{P(X) P(X)}{P(X)}$$

P(X) is conditional probability that describe occurance of event X in given X

Page No.

Data

Given ⇒

Day	outlook	Temp	Humidity	wind	Play
1	Sunny	Hot	High	weak	No
2	sunny	Hot	High	strong	No
3	overcast	Hot	High	weak	yes
4	Rain	Mild	High	weak	yes
5	Rain	cool	Normal	weak	yes
6	Rain	cool	Normal	strong	No

unknown ⇒

outlook → overcast
Temp → Mild
Humidity → High
Wind → weak

play	Yes	No
	3/6	3/6

Outlook	Yes	No
overcast	1/3	0/3

Temp	Yes	No
Mild	1/3	0/3

Wind	Yes	No
weak	3/3	1/3

Page No.

Humidity	Yes	No
High	3/3	3/3

$$P_x(yes) = \frac{3}{6} \times \frac{1}{3} \times \frac{1}{3} \times \frac{3}{3} \times \frac{2}{3} = 0.037$$

$$P_x(No) = \frac{3}{6} \times \frac{6}{3} \times \frac{6}{3} \times \frac{1}{3} \times \frac{2}{3} = 0$$

Since $P_x(yes) > P_x(No)$
Unknown case will fall into
Playgame = Yes

Conclusion :- with help of Bayesian classifier we can classify unknown case by training over known data

Code:

```
const xlsx = require("xlsx");
const prompt = require("prompt-sync")();
var wb = xlsx.readFile("data.xlsx");
var ws = wb.Sheets["data"];

var data = xlsx.utils.sheet_to_json(ws);
var parentattri = "class";

var obj = {};
for (var i = 0; i < Object.keys(data[0]).length; i++) {
  let arr2 = {};
  const key = Object.keys(data[0])[i];
  for (let i = 0; i < data.length - 1; i++) {
    if (!(data[i][key] in arr2)) {
      arr2[data[i][key]] = { 1: 0, 0: 0 };
    }
    var res = data[i][parentattri];
    if (res == 0) {
      arr2[data[i][key]]["0"]++;
    } else if (res == 1) {
      arr2[data[i][key]]["1"]++;
    }
  }
  obj[key] = arr2;
}

console.log(obj);
var prob = {};
for (classkey in obj["class"]) {
  var value = 1;
  var classtot = obj["class"][classkey][classkey];
  for (var i = 0; i < Object.keys(data[data.length - 1]).length; i++) {
    const key = Object.keys(data[data.length - 1])[i];
    value = value * (obj[key][data[data.length - 1][key]][classkey] / classtot);
  }
  value = value * (classtot / (data.length - 1));
  prob[classkey] = value;
}
console.log("====================================");
```

```
console.log("probablities");
console.log("====================================");
console.log(prob);
```

Input:

outlook	temperature	humidity	windy	class
sunny	hot	high	no	0
sunny	hot	high	yes	0
overcast	hot	high	no	1
rain	mild	high	no	1
rain	cool	normal	no	1
rain	cool	normal	yes	0

Output:

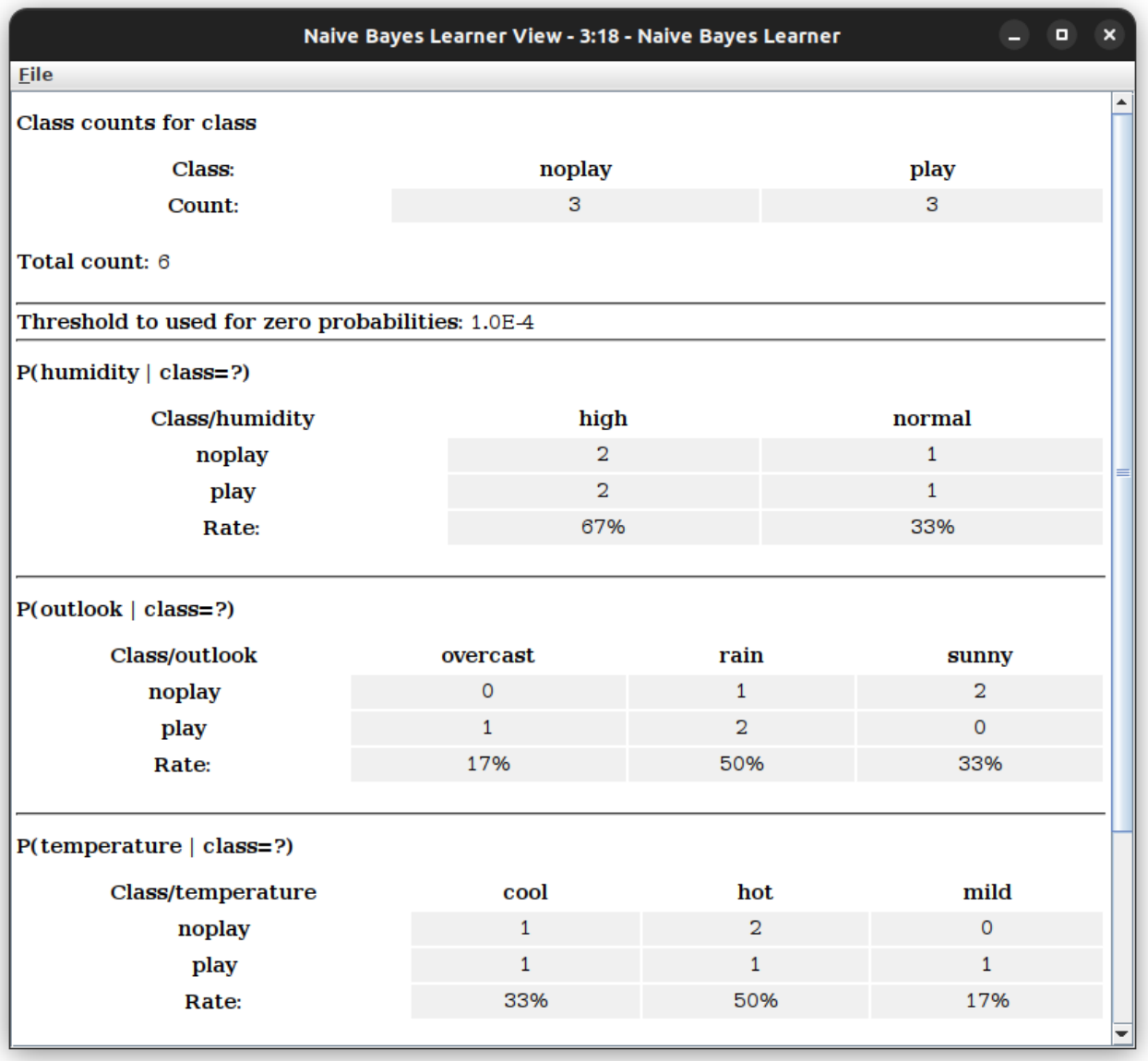
```
pranav-bhosale@pranav-bhosale:~/Desktop/Btech/DM LAB/DM Exp/Exp12$ node bayes.js
{
  outlook: {
    sunny: { '0': 2, '1': 0 },
    overcast: { '0': 0, '1': 1 },
    rain: { '0': 1, '1': 2 }
  },
  temperature: {
    hot: { '0': 2, '1': 1 },
    mild: { '0': 0, '1': 1 },
    cool: { '0': 1, '1': 1 }
  },
  humidity: { high: { '0': 2, '1': 2 }, normal: { '0': 1, '1': 1 } },
  windy: { no: { '0': 1, '1': 3 }, yes: { '0': 2, '1': 0 } },
  class: { '0': { '0': 3, '1': 0 }, '1': { '0': 0, '1': 3 } }
}
====================================
probablities
====================================
{ '0': 0, '1': 0.037037037037037035 }
```

Weka

```
Naive Bayes Classifier

                Class
Attribute      noplay   play
                (0.5)  (0.5)
============================
outlook
  sunny           3.0    1.0
  overcast        1.0    2.0
  rain            2.0    3.0
  [total]         6.0    6.0

temperature
  hot             3.0    2.0
  mild            1.0    2.0
  cool            2.0    2.0
  [total]         6.0    6.0

humidity
  high            3.0    3.0
  normal          2.0    2.0
  [total]         5.0    5.0

windy
  no              2.0    4.0
  yes             3.0    1.0
  [total]         5.0    5.0
```

```
Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances           6               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.2293
Root mean squared error                  0.2476
Relative absolute error                 45.8525 %
Root relative squared error             49.5197 %
Total Number of Instances                6

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     noplay
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     play
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

 a b   <-- classified as
 3 0 | a = noplay
 0 3 | b = play
```

Knime Output



P(windy | class=?)

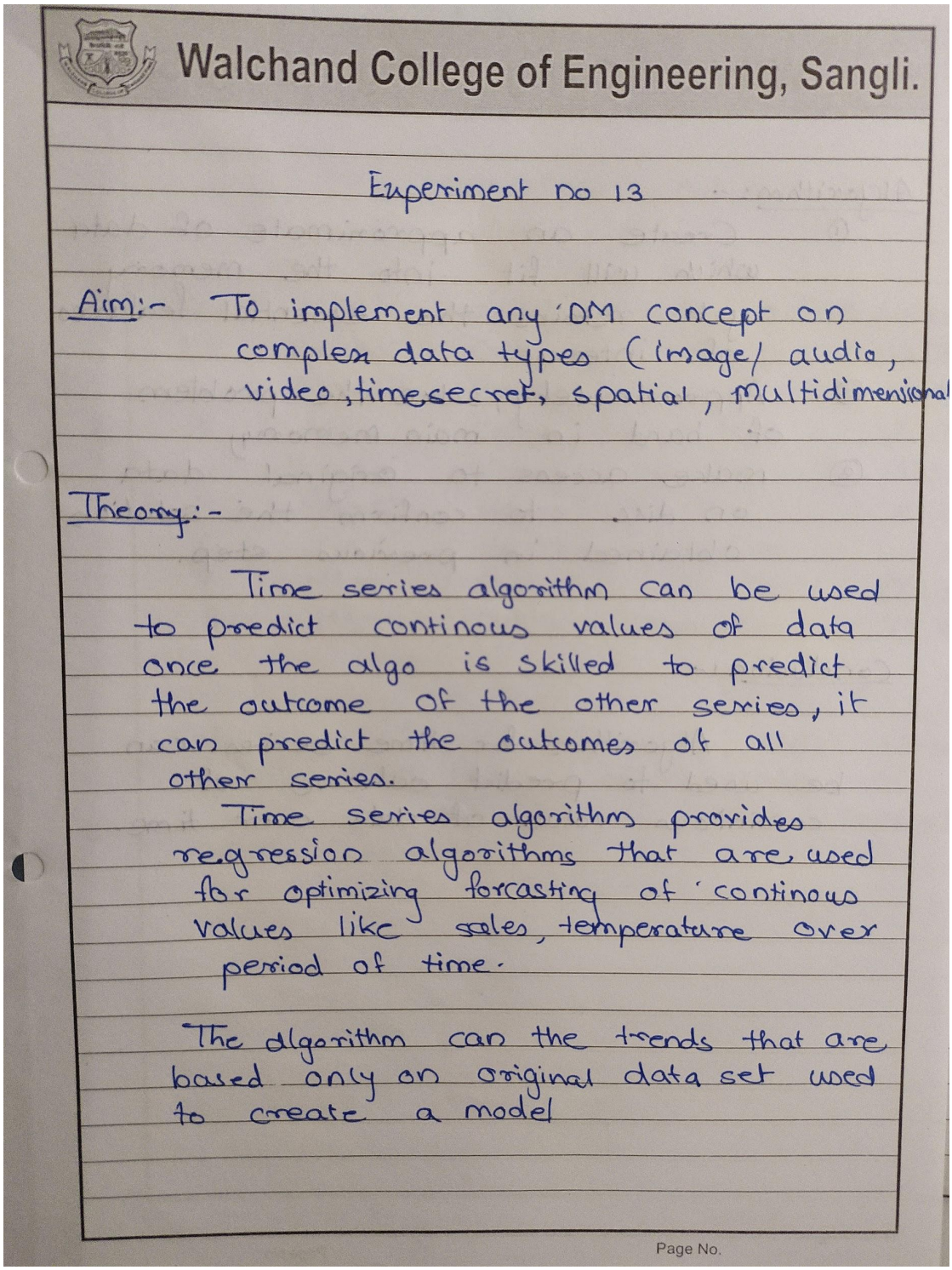
Class/windy	no	yes
noplay	1	2
play	3	0
Rate:	67%	33%

Conclusion:

With help of the bayesian classifier we can classify unknown cases by training over known data

Experiment 13

WAP a program to implement any DM concept on complex data type



Walchand College of Engineering, Sangli.

Experiment no 13

Aim:- To implement any DM concept on complex data types (image/ audio, video, timesecret, spatial, multidimensional

Theory:-

Time series algorithm can be used to predict continous values of data once the algo is skilled to predict the outcome of the other series, it can predict the outcomes of all other series.

Time series algorithms provides regression algorithms that are used for optimizing forcasting of continous values like sales, temperature over period of time.

The algorithm can the trends that are based only on original data set used to create a model

Page No.

Algorithm :-

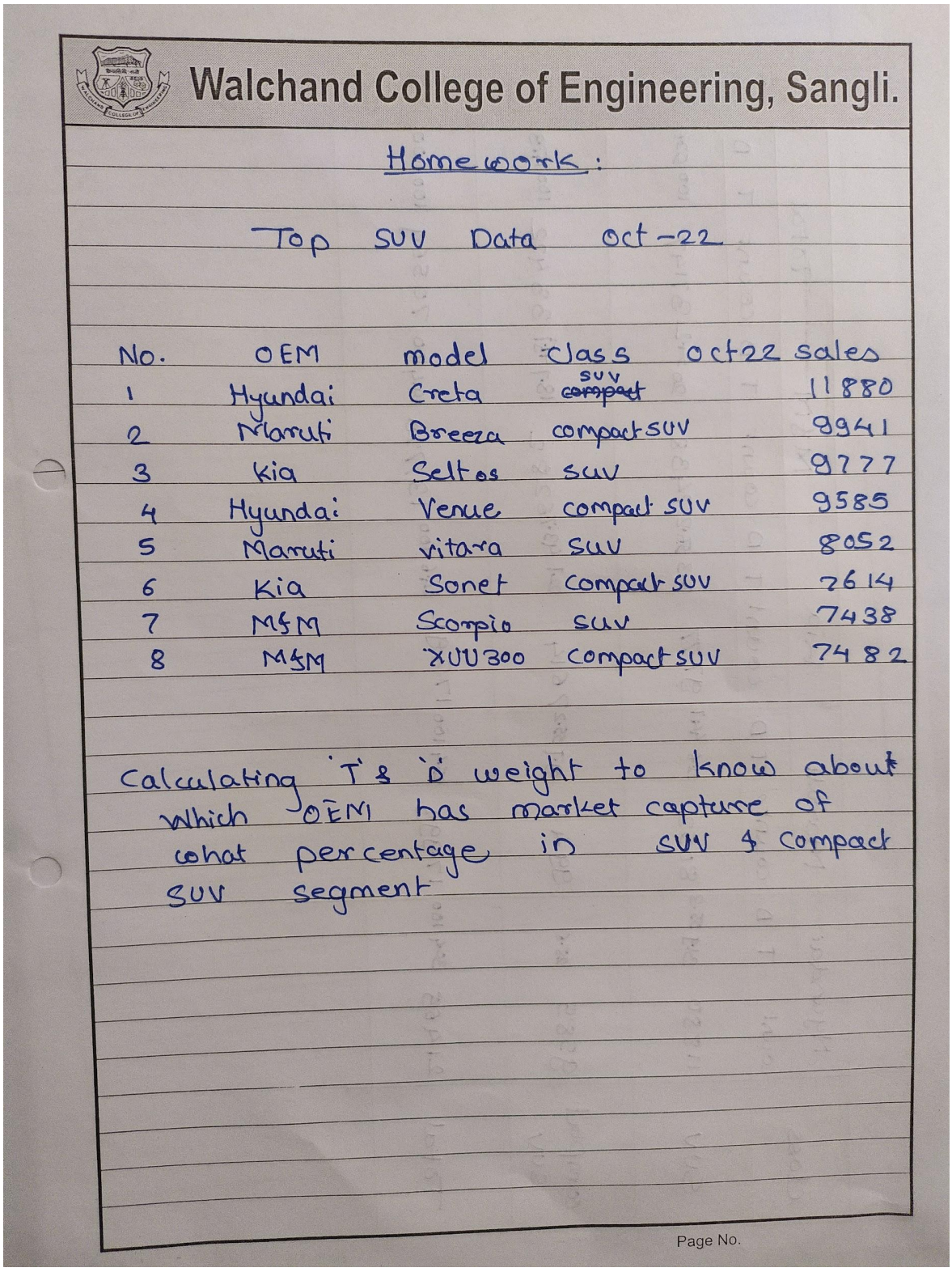
1. Create an approximate of data which will fit into the memory yet retains the essential features of interest
2. Approximately, solve the problem of hand in main memory
3. make access to original data on disc to confirm the solution obtained in previous step.

Conclusion :-

Algorithm for time series can be used to predict outcome of continious values of data over time.

Homework

Lab assignment: Top SUVs Data



Walchand College of Engineering, Sangli.

Homework :

Top SUV Data Oct-22

No.	OEM	model	class	oct22 sales
1	Hyundai	Creta	~~compact~~ SUV	11880
2	Maruti	Breeza	compact SUV	9941
3	Kia	Seltos	suv	9777
4	Hyundai	Venue	compact SUV	9585
5	Maruti	vitara	suv	8052
6	Kia	Sonet	compact SUV	7614
7	M&M	Scorpio	suv	7438
8	M&M	XUV300	compact SUV	7482

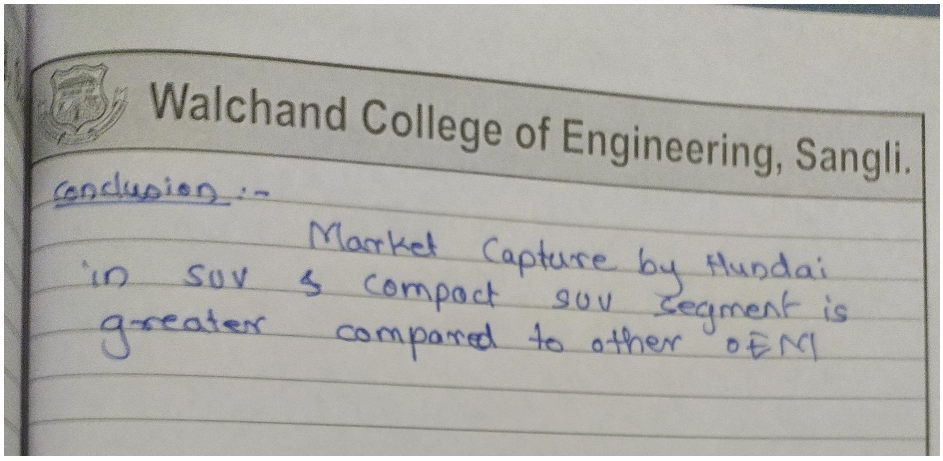
Calculating 'I' & 'D' weight to know about which OEM has market capture of what percentage in SUV & compact SUV segment

Page No.

Walchand College of Engineering, Sangli.

Class	Hyundai			Maruti			Kia			M&M			Total		
	count	T	D	count	T	D	count	T	D	count	T	D	count	T	D
SUV	11880	39	55.9	8052	24.4	44.7	9777	26.3	56.2	7438	20	54.2	37147	100	526
compact SUV	9585	25.6		9941	89.7	55.2	7614	22.1	43.7	6282	18.7	45.1	33422	100	473
Total	21465	30.4	100	17993	25.4	100	17391	24.6	100	13,720	19.4	100	70569	100	100

Page No.


Walchand College of Engineering, Sangli.
Conclusion :-
Market Capture by Hundai
in SUV & Compact SUV segment is
greater compared to other OEM