Experiment No.08

A.1 Aim: Implementation Association Mining-FPM (Frequent Pattern Mining) Algorithm using any programming language like JAVA, C++, Python or WEKA Tool.

PART B

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per the following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Blackboard access available)

Roll No. 50	**Name:** AMEY THAKUR
Class: Comps TE B	**Batch:** B3
Date of Experiment: 28/04/2021	**Date of Submission:** 28/04/2021
Grade:	

B.1 Software Code written by a student:

(Paste your problem statement related to your case study completed during the 2 hours of practice in the lab here)

- **Python Source Code**

```
import pandas as pd
data = pd. read_csv ("car.csv")

data = pd. read_csv ('car.csv', header=None)
data.head()

observations = []
for i in range (len (data)):
   observations.append([str (data.values[i,j]) for j in range (4)])

from apyori import apriori
associations = apriori (observations, min_support=0.01,
min_confidence=0.9, min_lift=3, min_length=2)
associations = list (associations)
print(associations [0])
```

```
for item in associations:
# first index of the inner list
# Contains base item and add item
   pair = item [0]
   items = [x for x in pair]
   print("Rule: " + items [0] + " -> " + items [1])

   print("Support: " + str(item[1]))

   print("Confidence: " + str(item [2] [0] [2]))
   print("Lift: " + str(item [2][0][3]))
   print("====================================")
```

- **FPM.java**

```
import java.util.*;
class F_P{
        public static void main(String args[])
        {
                char b[]={'l','m','n','o','p'};
                int b1[]={0,0,0,0,0,};
                String a[]={"lmno","ml","omnp","pon","ponm","nom"};
                int min=4,i,j,p,c=0;
                for(i=0;i<5;i++)
                {
                        for(j=0;j<a.length;j++)
                        {
                                for(p=0;p<a[j].length();p++)
                                {
                                        if(b[i]==a[j].charAt(p))
                                        {
                                                c=c+1;
                                        }
                                }
                        }
                        b1[i]=c;
                        c=0;
                }
                System.out.println("INPUT STRINGS:");
                for(j=0;j<a.length;j++)
                {
                        System.out.println(a[j]);
                }
```

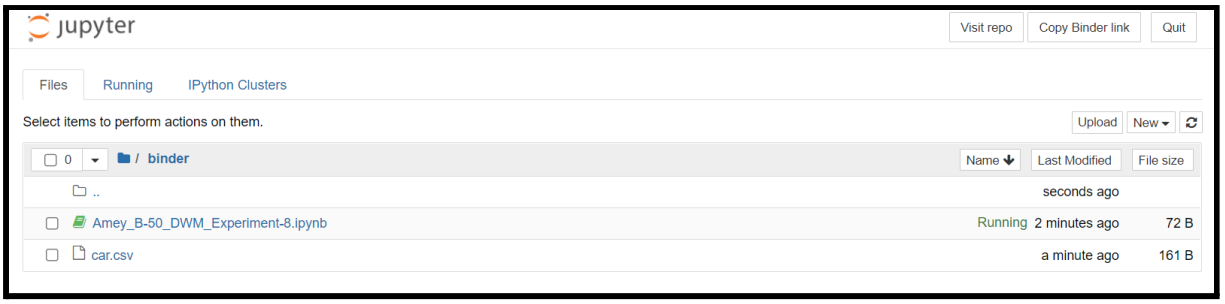
```
		for(i=0;i<b1.length;i++)
		{
			System.out.println(b[i]+" Support:"+b1[i]);
		}
		System.out.println("Minimum Support:"+min+"\n AFTER
ELIMINATING ELEMENT HAVING LESS SUPPORT THAN MINIMUM SUPPORT: ");
		for(i=0;i<b1.length;i++)
		{
			if(b1[i]>=min)
			{
			}
			else
			{
				b1[i]=0;
			}
		}
		for(i=0;i<b1.length;i++)
		{
			System.out.println(b[i]+" Support:"+b1[i]);
		}
		System.out.println("FP TREE PATHS: ");
		for(j=0;j<a.length;j++)
		{
			System.out.print("\n ROOT: ");
			for(p=0;p<a[j].length();p++)
			{
				for(i=0;i<5;i++)
				{
					if(b[i]==a[j].charAt(p) && b1[i]!=0)
					{
						System.out.print(a[j].charAt(p)+"->");
					}
				}
			}
			System.out.print("NULL");
		}
	}
}
```

B.2 Input and Output:

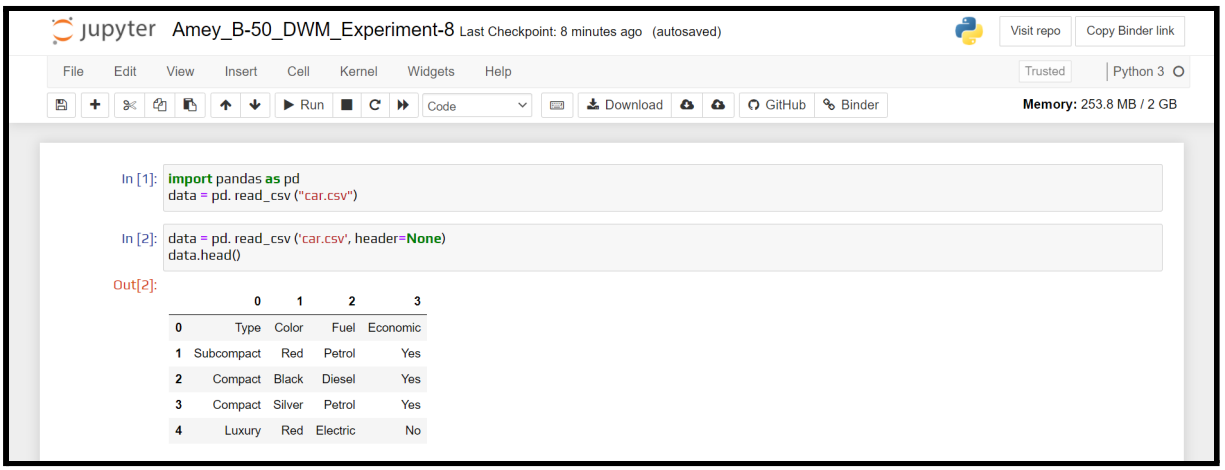
(Paste your program input and output in the following format, If there is an error then paste the specific error in the output part. In case of an error with the due permission of the faculty, an extension can be given to submit the error-free code with output in due course of time. Students will be graded accordingly.)

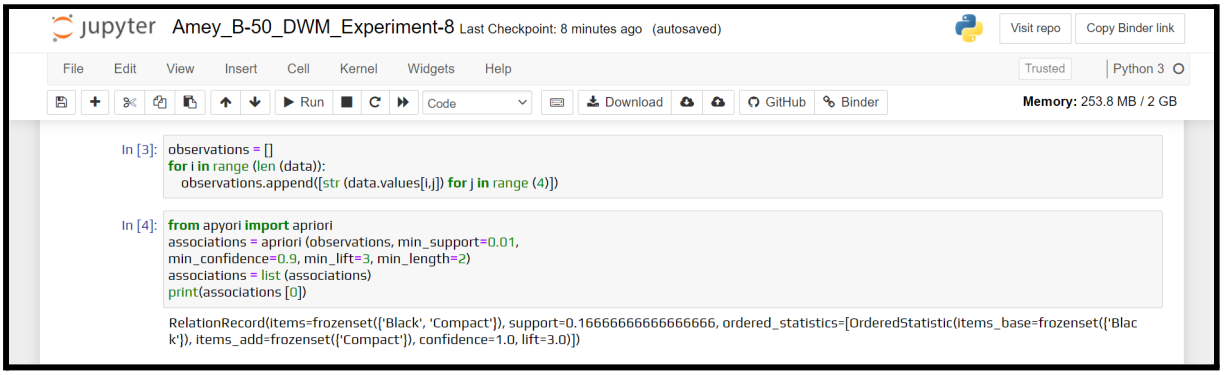
Jupyter Notebook

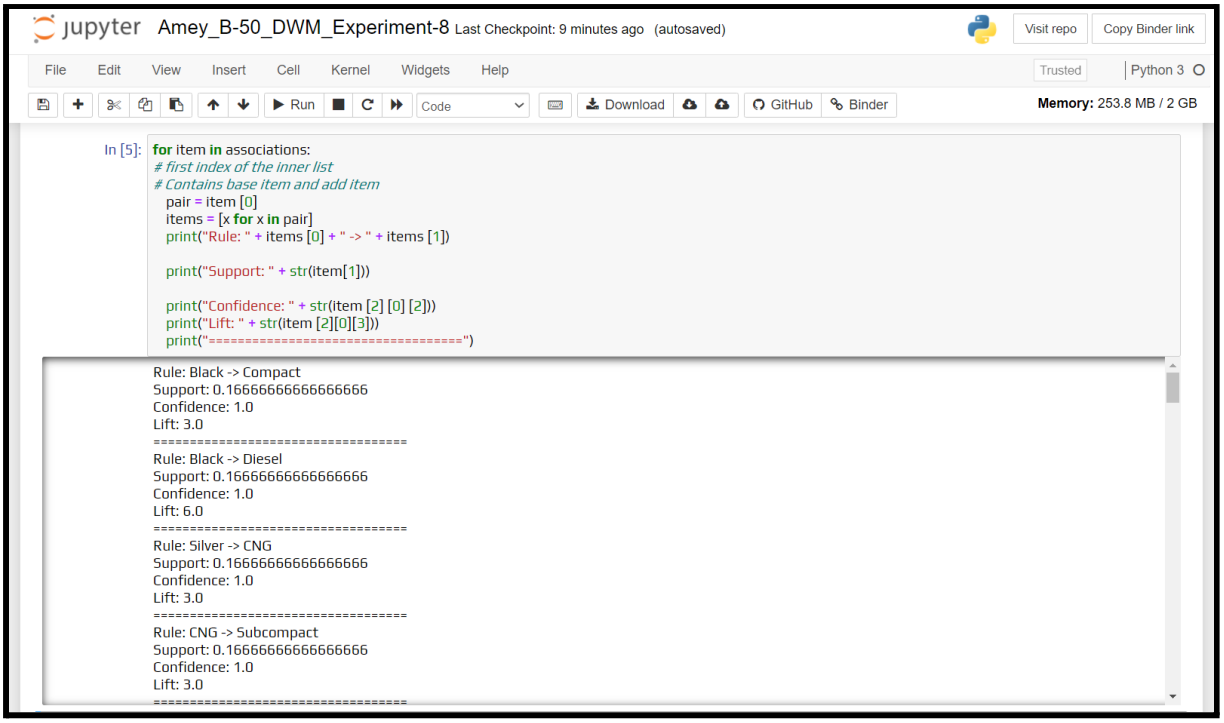
Jupyter/Binder:



Jupyter Notebook:



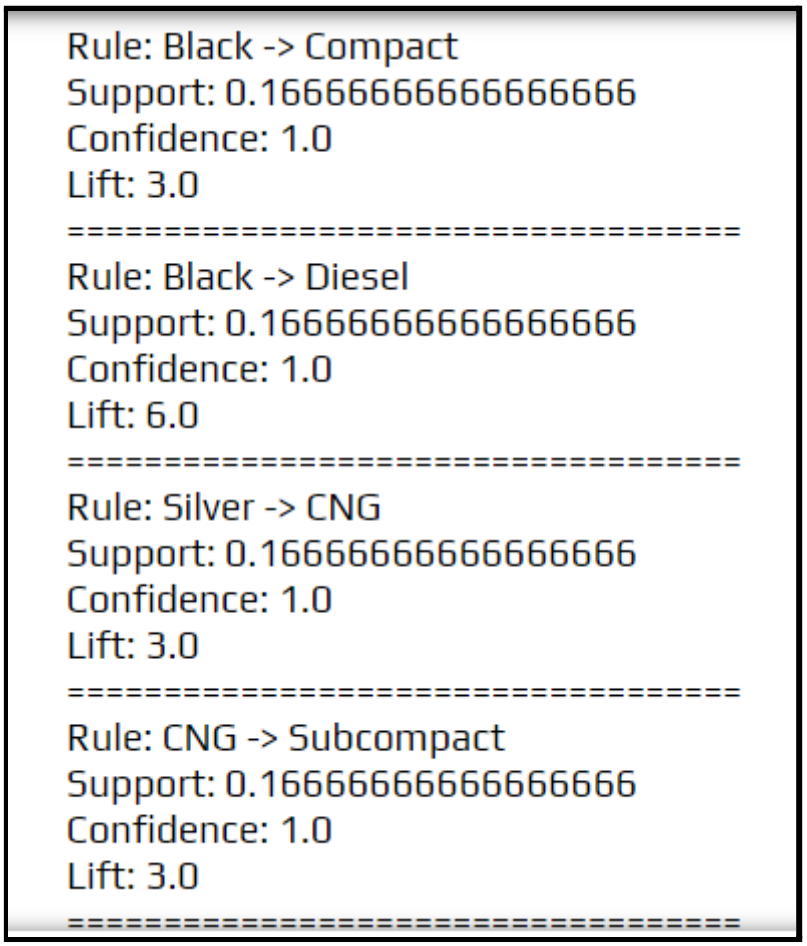




Sample Dataset:

	Type	Color	Fuel	Economic
0	Subcompact	Red	Petrol	Yes
1	Compact	Black	Diesel	Yes
2	Compact	Silver	Petrol	Yes
3	Luxury	Red	Electric	No
4	Subcompact	Silver	CNG	Yes

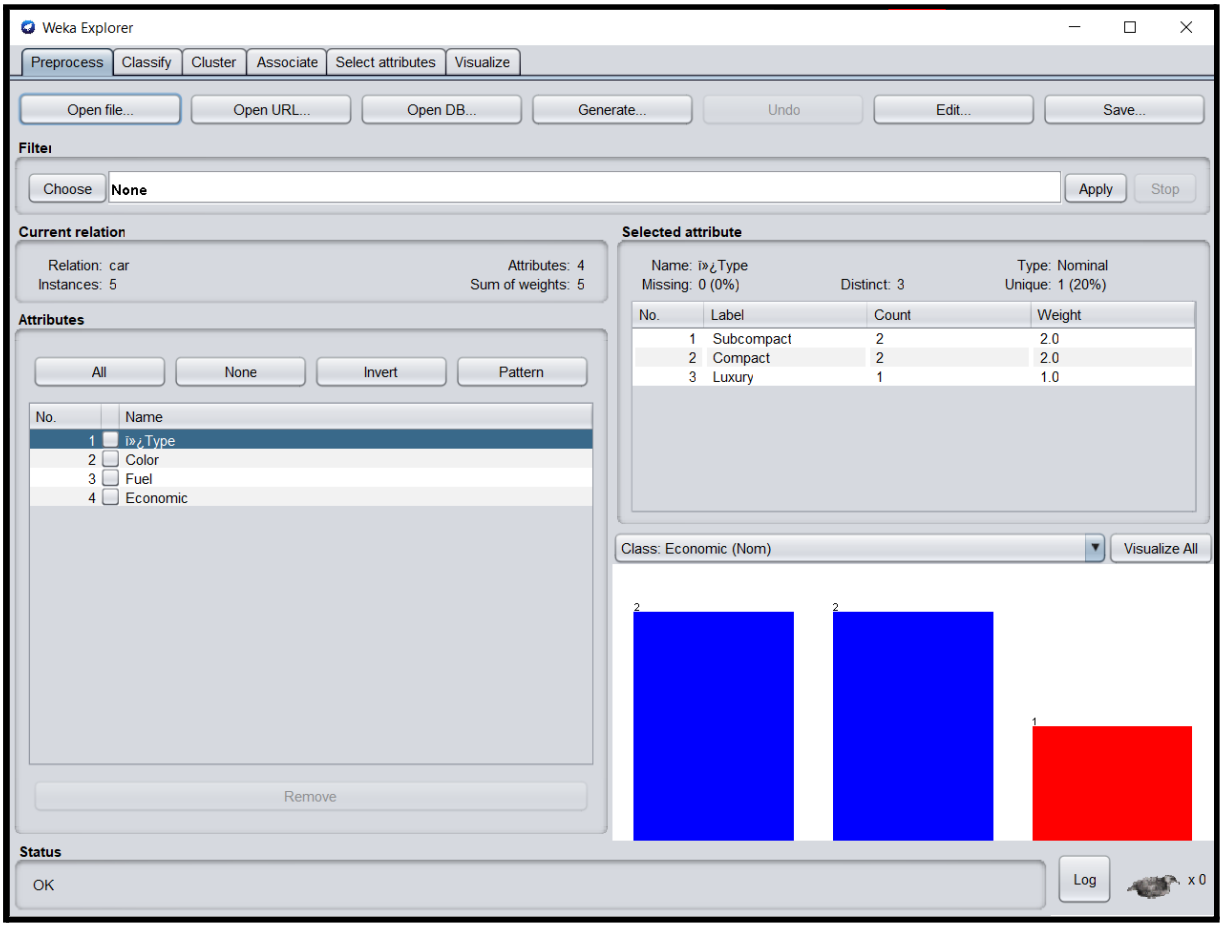
Python Output:



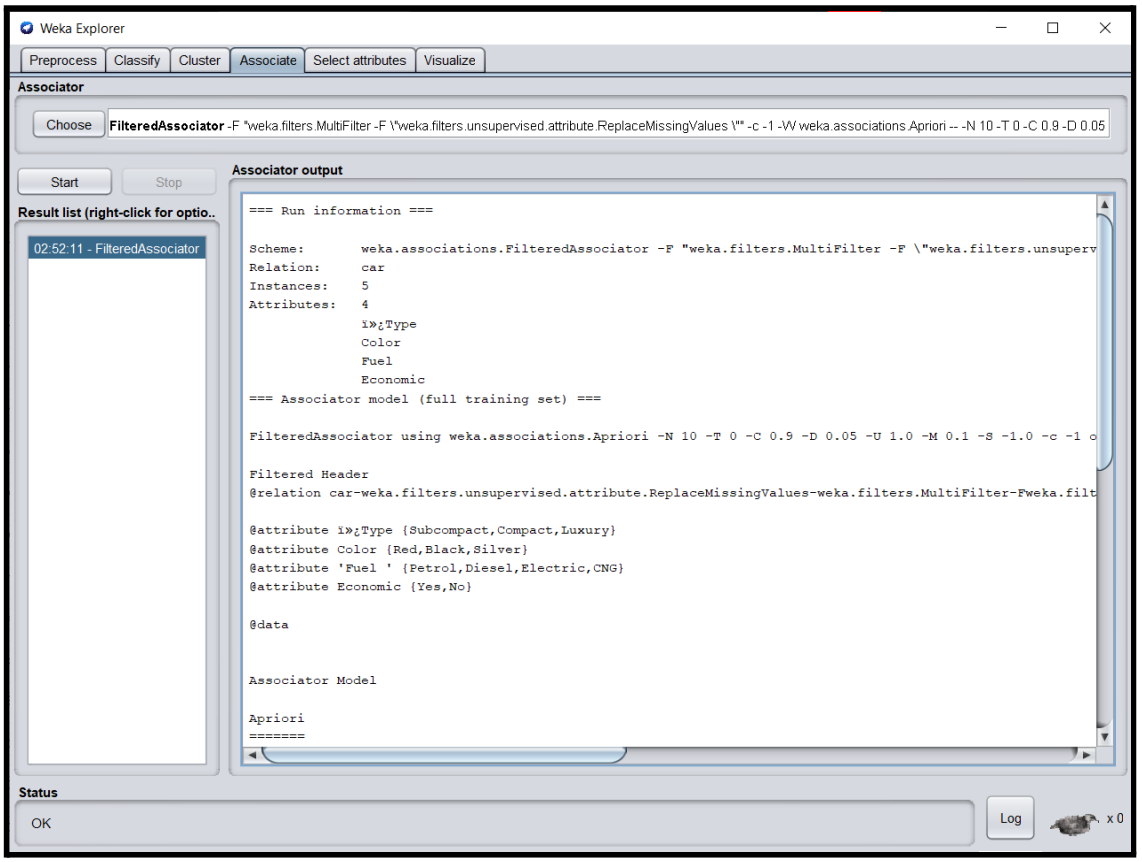
Java Output:

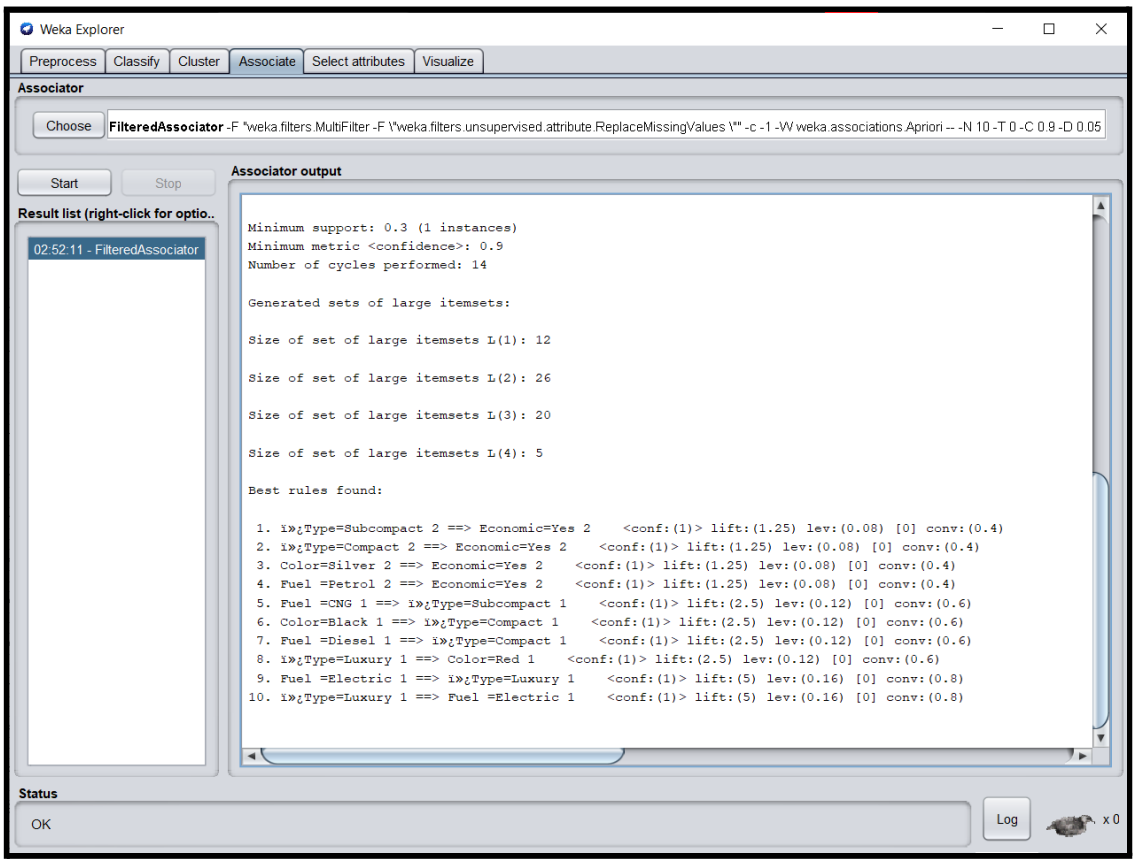
```
C:\Users\ameyt\Desktop>JAVA FPM.JAVA
INPUT STRINGS:
lmno
ml
omnp
pon
ponm
nom
l Support:2
m Support:5
n Support:5
o Support:5
p Support:3
Minimum Support:4
 AFTER ELIMINATING ELEMENT HAVING LESS SUPPORT THAN MINIMUM SUPPORT:
l Support:0
m Support:5
n Support:5
o Support:5
p Support:0
FP TREE PATHS:

 ROOT: m->n->o->NULL
 ROOT: m->NULL
 ROOT: o->m->n->NULL
 ROOT: o->n->NULL
 ROOT: o->n->m->NULL
 ROOT: n->o->m->NULL
```

Weka Tool


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: car
Instances: 5
Attributes: 4
Sum of weights: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 i»¿Type
2 Color
3 Fuel
4 Economic
Remove
Selected attribute
Name: i»¿Type
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 1 (20%)
No.
Label
Count
Weight
1 Subcompact 2 2.0
2 Compact 2 2.0
3 Luxury 1 1.0
Class: Economic (Nom)
Visualize All
2
2
1
Status
OK
Log
x 0

Weka Output:


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Associator
Choose
FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsupervised.attribute.ReplaceMissingValues \"" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05
Start
Stop
Result list (right-click for optio..
02:52:11 - FilteredAssociator
Associator output
=== Run information ===
Scheme: weka.associations.FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsuperv
Relation: car
Instances: 5
Attributes: 4
ï»¿Type
Color
Fuel
Economic
=== Associator model (full training set) ===
FilteredAssociator using weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1 o
Filtered Header
@relation car-weka.filters.unsupervised.attribute.ReplaceMissingValues-weka.filters.MultiFilter-Fweka.filt
@attribute ï»¿Type {Subcompact,Compact,Luxury}
@attribute Color {Red,Black,Silver}
@attribute 'Fuel ' {Petrol,Diesel,Electric,CNG}
@attribute Economic {Yes,No}
@data
Associator Model
Apriori
=======
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Associator
Choose
FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsupervised.attribute.ReplaceMissingValues \"" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05
Start
Stop
Result list (right-click for optio..
02:52:11 - FilteredAssociator
Associator output
Minimum support: 0.3 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14
Generated sets of large itemsets:
Size of set of large itemsets L(1): 12
Size of set of large itemsets L(2): 26
Size of set of large itemsets L(3): 20
Size of set of large itemsets L(4): 5
Best rules found:
1. ï»¿Type=Subcompact 2 ==> Economic=Yes 2 <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.4)
2. ï»¿Type=Compact 2 ==> Economic=Yes 2 <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.4)
3. Color=Silver 2 ==> Economic=Yes 2 <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.4)
4. Fuel =Petrol 2 ==> Economic=Yes 2 <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.4)
5. Fuel =CNG 1 ==> ï»¿Type=Subcompact 1 <conf:(1)> lift:(2.5) lev:(0.12) [0] conv:(0.6)
6. Color=Black 1 ==> ï»¿Type=Compact 1 <conf:(1)> lift:(2.5) lev:(0.12) [0] conv:(0.6)
7. Fuel =Diesel 1 ==> ï»¿Type=Compact 1 <conf:(1)> lift:(2.5) lev:(0.12) [0] conv:(0.6)
8. ï»¿Type=Luxury 1 ==> Color=Red 1 <conf:(1)> lift:(2.5) lev:(0.12) [0] conv:(0.6)
9. Fuel =Electric 1 ==> ï»¿Type=Luxury 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
10. ï»¿Type=Luxury 1 ==> Fuel =Electric 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
Status
OK
Log
x 0

B.3 Observations and learning:

(Students are expected to comment on the output obtained with clear observations and learning for each task/ subpart assigned)

I observed that the frequent itemset problem is to find all frequent itemset in a given transaction database. The first and most important solution for finding frequent itemsets is the Apriori algorithm and FP mining Algorithm.

B.4 Conclusion:

(Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)

Created algorithm for FP Tree mining algorithm and Implemented FP Tree Algorithm in java and python.

B.5 Question of Curiosity

(To be answered by the student based on the practical performed and learning/observations)

1. What is the association rule for Mining?

Ans:

➔ Association rules are if-then statements that help to show the probability of relationships between data items within large data sets in various types of databases. Association rule mining has a number of applications and is widely used to help discover sales correlations in transactional data or in medical data sets.

How association rules work:

➔ Association rule mining, at a basic level, involves the use of machine learning models to analyze data for patterns, or co-occurrence, in a database. It identifies frequent if-then associations, which are called association rules.

➔ An association rule has two parts: an antecedent (if) and a consequent (then). An antecedent is an item found within the data. A consequent is an item found in combination with the antecedent.

➔ Association rules are created by searching data for frequent if-then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the data. Confidence indicates the number of times the if-then statements are found true. A third metric, called lift, can be used to compare confidence with expected confidence.

➔ Association rules are calculated from itemsets, which are made up of two or more items. If rules are built from analyzing all the possible itemsets, there could be so many rules that the rules hold little meaning. With that, association rules are typically created from rules well-represented in data.

2. Explain the characteristics of the FPM algorithm.

Ans:

Characteristics of FPM algorithm:

- ➔ Find the most frequently occurring patterns satisfying various conditions.
- ➔ Extract features from the dataset by transforming them from the Item space into the Receipt space. These features can then be used for applications like clustering, classification, churn prediction, recommender systems etc.
- ➔ Make predictions based on new data using rules learned from sets of items that occur frequently together.
- ➔ Pattern mining algorithms can be applied to various types of data such as transaction databases, sequence databases, streams, strings, spatial data, graphs, etc.
- ➔ Pattern mining algorithms can be designed to discover various types of patterns: subgraphs, associations, indirect associations, trends, periodic patterns, sequential rules, lattices, sequential patterns, high-utility patterns, etc.
- ➔ There are two important characteristics-

  Discovering frequent itemsets

The most popular algorithm for pattern mining is without a doubt Apriori (1993). It is designed to be applied to a transaction database to discover patterns in transactions made by customers in stores. But it can also be applied in several other applications. A transaction has defined a set of distinct items.

Discovering sequential rules

A sequence database is defined as a set of sequences. A sequence is a list of transactions.