DATA MINING ASSIGNMENT

Association Rule Mining –

Hash Tree Implementation

FP Tree Implementation

GROUP MEMBERS

ABHILASH K MIRJI – 2011AAPS049H

ASHWIN RAGHAVAN – 2011B5A7550H

PRAVEEN VENKATESWARAN – 2011B4A7640H

Introduction: Association Rule Mining

Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on the concept of strong rules, Rakesh Agrawal et al. introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule **{onions, potatoes} -> {burger}** found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, they are likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from market basket analysis association rules are employed today in many application areas including Web usage mining, intrusion detection, Continuous production, and bioinformatics. In contrast with sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

We implemented 2 popular rule mining algorithms as a part of this assignment – Hash Tree based and FP Tree based Association Rule Mining.

The Dataset: Car Evaluation Dataset

The dataset used in this assignment which is a collection of the records on specific attributes on cars donated by Marco Bohanec in 1997 was obtained from the UCI dataset repository. The car evaluation dataset as described in the UCI dataset repository was derived from simple hierarchical decision.

Number of Instances: 1728 (instances completely cover the attribute space)

Number of Attributes: 6

Attribute Values:

buying	v-high, high, med, low
maint	v-high, high, med, low
doors	2, 3, 4, 5-more
persons	2, 4, more
lug_boot	small, med, big
safety	low, med, high

Missing Attribute Values: none

Class Distribution (number of instances per class)

Class	N	%
unacc	1210	70.023
acc	384	22.222
good	69	3.993
v-good	65	3.762

Reason for choosing:

- It is a categorical dataset with a relatively less number of attributes. This helped us in binarizing the data while ensuring that dimensionality of the input did not blow up.
- The dataset gives a chance to physically interpret and make sense of the rules generated.
- The dataset has 1728 instances which seemed like a fair enough number to ensure that the code gave sensible results.

Objective

We hope to find some meaningful rules regarding car features and the corresponding buy worthiness.

Pre-processing

Pre-processing was done in the following way –

1. The attributes were all binarized so that each attribute value had a corresponding column which showed 1 or 0 for an instance depending on its presence or absence. For our dataset, this resulted in 25 different columns.

2. Now a letter was assigned to each column and if an instance had 1 for a particular column then the corresponding letter was used. For example a 1 in the first, third and fifth column would correspond to ace. A file containing this output was used as input for the hash tree code.

3. Another file was generated which contains the count of number of 1s in a row followed by a number corresponding to each column that had a 1. For example 7 1 4 9 13 17 20 22 would mean 7 1s and these 1s are in the 1^{st}, 4^{th}, 9^{th} … columns.

Results

Because of the skewed nature of the dataset, we had to settle for really low support and confidence values. Almost 70% of the instances had the value unacceptable attribute which favoured most of the rules to include it.

We experimented with different values and combinations of support and confidence values. The dataset was run on both hash tree and frequent pattern mining codes. We had some slight variations in the input parameters required for both the algorithms possibly due to some implementation deficiencies. The range of input parameters were as follows

Measure	Range
Min Support	0.05 - 0.10
Min Confidence	0.40 - 0.80

Some of the rules which resulted from our investigation include:

- { lug_boot_small & safety_low } → { value_unacc }
- { safety_low } → { value_unacc }
- { persons_2 } → { value_unacc }
- { value_acc } → { safety_high }
- { value_acc } → { persons_4 }
- { value_acc } → { persons_more }
- { persons_more & safety_high } → { value_acc }

Concluding Remarks

These rules seem to make sense. A car with low safety, or one with very less carrying capacity is unacceptable whereas one which ensures high safety and/or can carry many passengers is acceptable to buyers. These rules can be used by the seller to accordingly price his cars as he knows which one is more preferable to the buyer and thus would be willing to pay more for.