Soft Computing

CSE-6002

Final Project Report

Identifying Interesting Association Rules with Genetic Algorithms

Elnaz Delpisheh

Winter 2011

Table of Content

1. Introduction

DATA mining aims at discovering interesting patterns from large-volume data [1]. Association mining is one of the many data mining tasks that has recently attracted more and more attention. It discovers the inherent relationships among objects in an application domain. A good application example of these application domains could be the basket analysis in supermarkets, where one tries to discover the relationships among commodities in baskets. In association mining, interesting patterns are represented as association rules. For example, in the basket analysis, {milk, eggs}→{bread} is an association rule, implying that if milk and eggs are bought together, bread is likely to be bought as well [2].

One of the non-trivial issues in association mining considered in literature [3], [4], [5], [6], [7], [8], [9] is the large number of association rules discovered. While many of the rules convey non-trivial and interesting information, some of them are trivial or even irrelevant [5], [7], [10]. Thus, they are too difficult to be interpreted and comprehended by users. Generally, two types of users are considered to be involved in association mining process. An end user is an expert of the data to be mined in a particular application domain, while an analyst is a specialist in data mining. The end user should provide the analyst with sufficient domain knowledge, represented as requirements, in the domain, in order to extract interesting association rules [11]. Throughout this paper, the term user is meant to be end user. To deal with the large number of association rules, a variety of interestingness measures to assess them as how beneficial and interesting they are have been proposed [3], [6], [10], [12]. These measures rank association rules according to a set of criteria, some of which are as follows. Reliability asks whether the relationship the association rule reveals occurs frequently. Peculiarity checks whether an association rule largely differs from other discovered association rules. Other criteria include diversity, novelty, surprisingness, utility, and comprehensibility [6], [9], [10], [13].

It is easy to see that some of the above criteria such as novelty and surprisingness are dependent on each other, while some others such as conciseness and reliability are contradictory [10]. Requiring them by the user at the same time introduces inconsistency in measuring association rules [9], [10], [14]. In addition, the user's requirements could be multi-facet, desiring multiple-criteria to be put into consideration when measuring association rules. Moreover, the user's requirements may change, reflecting the current preferences of them in an application domain and as a result in association rule evaluations.

In this work, I attempt to develop an algorithm that is independent from the users as well as the application domains to extract interesting association rules. This method, a revised version of the method proposed in [19], utilizes genetic algorithms to produce rules and then a fitness function to evaluate interestingness of the rules. In Section II, I briefly review mining frequent itemsets. In Section III, I discuss how association rules are mined from frequent itemsets. In Section IV, a

genetic algorithm used to extract interesting association rules is discussed. Section V discusses the results of my algorithm and compares my approach to the previous ones. Finally, Section VI summarizes my discussions.

2. Mining Frequent Itemsets

Let $I = \{I_1, I_2, I_3, ..., I_m\}$ be a set of n items and $D = \{t_1, t_2, t_3, ..., t_m\}$ be a transactional database, where each transaction $t_i \subseteq D$ is a nonempty subset of I. An association rule is of the form $A \rightarrow B$, where A and B are called the itemsets and $A \subset I$ and $B \subset I$. It is required that $A \cap B = \Phi$. A is called the *antecedence* of the rule while B is called its *consequence*. We say that the rule $A \rightarrow B$ holds for the database D with *support s*, where s is the percentage of transactions in D that contain both A and B, and with *confidence c*, where c is the percentage of transactions containing A that also contain B. A typical example of association rule mining, as previously mentioned, is the basket analysis. It analyzes customers' shopping habits by finding associations among the different items that customers place in their shopping baskets, as shown in Figure 1.

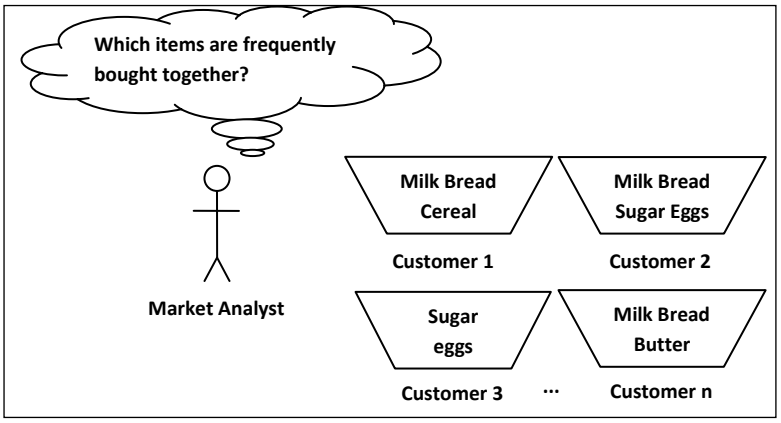


Figure 1: Market basket analysis

The discovery of such associations can help retailers develop marketing strategies, i.e., selective marketing, arranging shelf spaces, etc., by gaining insight into which items are frequently purchased together by customers [1].

The algorithm is explained informally by applying it to the transactional database example shown in Table 1. The minimum support for finding frequent itemsets is set to 22%. This is equivalent to requiring that a frequent itemset must appear in at least $\lceil 22\% * 9 \rceil = 2$ transactions, where 9 is the total number of transactions in Table 1.

TID	List of item IDs
T100	I1,I2,I3
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1,I3
T800	I1,I2,I3,I5
T900	I1,I2,I3

Table 1: A transactional database [1].

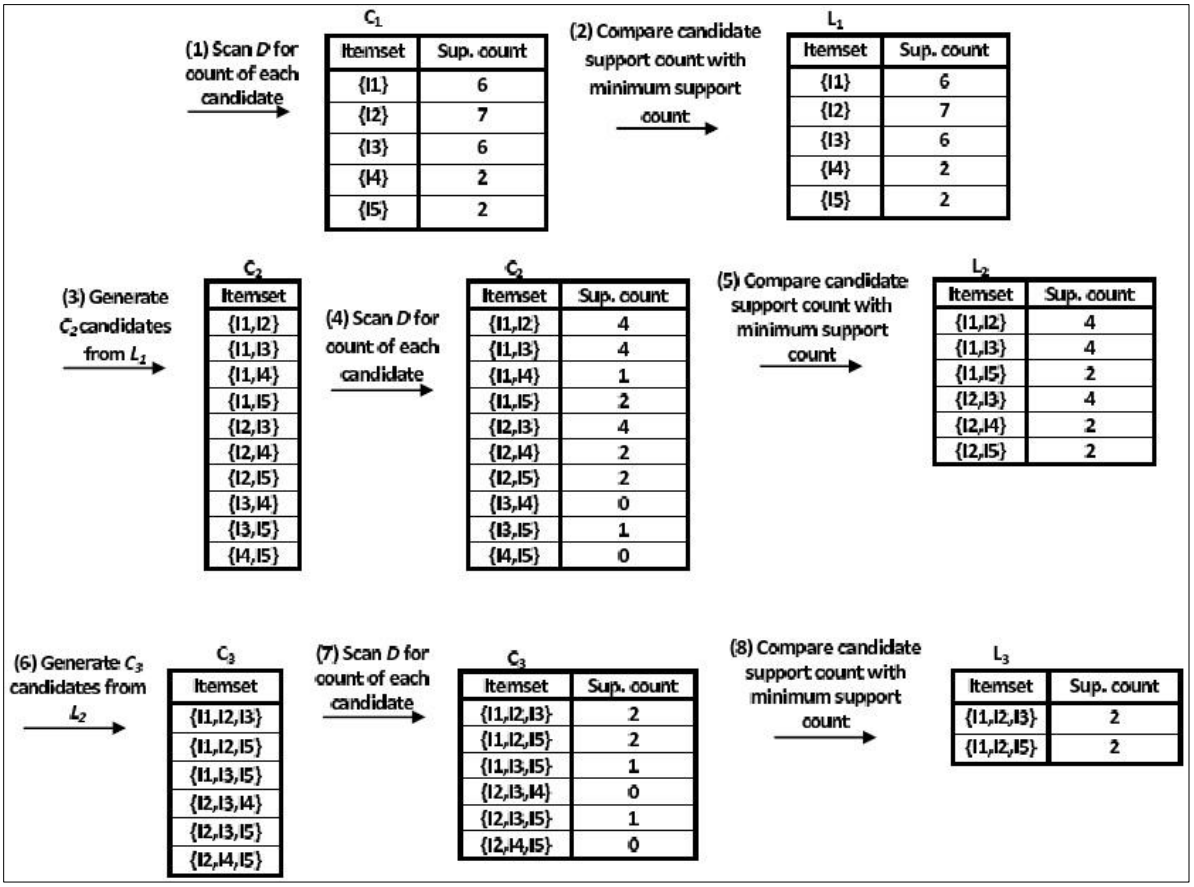


Figure 2: Frequent itemsets generation [1].

The first step for generating association rules for a transactional database is to find the so-called frequent itemsets, whose supports are more than a user-specified minimum support. In the literature, many algorithms for finding different frequent itemsets have been proposed and studied [15]. The one I have chosen to discuss is the basic Apriori algorithm. The Apriori algorithm is first proposed by Agrawal and Srikant [18]. It uses the prior knowledge of itemsets to find future itemsets. To improve the efficiency of the algorithm, the so-called *Apriori property* is used, which states that all nonempty subsets of a frequent itemset must also be frequent. Thus, if an itemset is found not frequent, it will not be used to generate larger itemsets, i.e., any larger itemsets generated from it will definitely fail to satisfy the minimum support.

Figure 2 illustrates generation of candidate itemsets and frequent itemsets, where the minimum support count is *2*. Steps (1) and (2) simply scan all the transactions in order to count the number of occurrences of each item. L_1 corresponds to the frequent 1-itemsets satisfying the minimum support; C_2 is generated using $L_1 \bowtie L_1$ in step (3); In step (4), the transactions are scanned and the support count of each candidate itemset in C_2 is accumulated; Then, in step (5), the set of frequent 2-itemsets, L_2, is determined, consisting of those candidate set of 2-itemsets in C_2 satisfying the minimum support count; In step (6), C_3 is generated using $L_2 \bowtie L_2$; In section (7), by scanning the transactions again, the support count for each 3-itemset in C_3 is collected; L_3 is obtained in step (8). Afterward, L_3 is used to generate the candidate set of 4-itemsets, $C_4 = \{I_1, I_2, I_3, I_5\}$. It is easy to check that $\{I_1, I_2, I_3, I_5\}$ is not frequent. Thus, $L_4 = \Phi$. The Apriori algorithm terminates after having found all of the frequent itemsets in $L_1 \cup L_2 \cup L_3$ [1].

3. Generating Association Rules

Association rules are generated from frequent itemsets that satisfy the minimum confidence requirements. For each frequent *k*-itemset *l*, where $k > 2$, we check every nonempty proper subset *s* of *l* as whether the rule $s \rightarrow (l - s)$ satisfies $\frac{\text{support count(l)}}{\text{support count(s)}} > min\_conf$, where *min_conf* is the minimum confidence threshold and *support count(x)* is the occurrence frequency of itemset *x* in a transactional database. For example, for the frequent 3-itemset $l = \{I_1, I_2, I_5\}$, obtained from Table 1, the nonempty subsets of *l* are $\{I_1, I_2\}$, $\{I_1, I_5\}$, $\{I_2, I_5\}$, $\{I_1\}$, $\{I_2\}$, and $\{I_5\}$. The resultant association rules are shown as below, each with its confidence:

$R_1 : \{I1, I2\} \rightarrow \{I5\}$	$2/4 = 50\%$
$R_2 : \{I1, I5\} \rightarrow \{I2\}$	$2/2 = 100\%$
$R_3 : \{I2, I5\} \rightarrow \{I1\}$	$2/2 = 100\%$
$R_4 : \{I1\} \rightarrow \{I2, I5\}$	$2/6 = 33\%$
$R_5 : \{I2\} \rightarrow \{I1, I5\}$	$2/7 = 29\%$
$R_6 : \{I5\} \rightarrow \{I1, I2\}$	$2/2 = 100\%$

If the minimum confidence is taken to be 40%, then only the association rules R_1, R_2, R_3, and R_6 are returned. While, if the minimum confidence is taken to be 20%, all the association rules are returned.

One of the non-trivial issues in association rule mining is to deal with the large number of association rules discovered [3], [4], [5], [6], [7], [8], [9]. While many of them convey non-trivial and interesting information, some are trivial or even irrelevant [10], [5], [7]. One solution to identify trivial or irrelevant information is to evaluate association rules in an association mining process, as how beneficial or interesting they are. This can take place during the mining process or in the post stage of the mining process. Doing so not only tackles the situation where a mining process usually would return a large number of association rules but also distinguishes the truly interesting association rules from those that are not. Thus, users should assign suitable requirements, i.e. minimum support threshold, to assess interestingness of association rules. However, in real-world applications, mining different databases requires different values of minimum support as shown in the case above. Depending on the database, setting the threshold to a small value returns a large number of association rules, while setting it in a large value returns a few number of association rules. This motivates us to design mining algorithms with database independent minimum support.

It is rare to see that genetic algorithm is used to mine association rules [19]. The following section further investigates the algorithm proposed to mine association rules using genetic algorithms.

4. Genetic Algorithm

This section describes a revised version of a genetic algorithm, called ARGMA for association rule mining [19].

Individual representation

An association rule of the form $\{A_1, A_2,...,A_j\} \rightarrow \{A_{j+1},A_{j+2},...,A_n\}$ can be represented by a variety of models. Here, I present two common individual representations[1].

Figure 3 represents an example of an individual representation and its corresponding decodified rule. In this representation, the first value of the vector represents the cut-point between the antecedence and consequence. There is a major problem regarding this representation. It highly depends on the length of the association rule. Thus, it results in equal length association rules. To overcome this issue, another representation is introduced in [21].

[1] For more models of representations, interested readers are referred to [20].

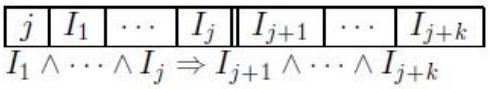


Figure 3: An individual representation [19].

In this representation, as illustrated in Figure 4, all attributes are considered in the chromosome. Thus, if we have *n* attributes, the chromosome will have *2n* alleles, the first two for the first attribute, second and third for the second attribute and so on. Then, each pair is decoded as: (*00*), the referred attribute is included in the rule in the antecedent; (*11*), the referred attribute is included in the rule in the consequent; (*10*) and (*01*), and the referred attribute is not included in the rule. One of the advantages of this representation is that we can have rules of different lengths [20]. Therefore, in this project, since I do not want to limit the length of association rules, this representation is selected.

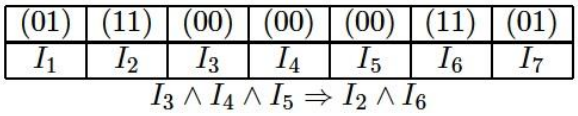


Figure 4: The individual representation used in this project [21].

Operators

Two genetic operators are used here: *crossover* and *mutation*.

Crossover

Function *crossover(pop,pc)* uses a two-point strategy to reproduce offspring chromosomes at a probability of *pc* from population *pop*, and returns a new population. These two crossover points are randomly generated, such that any segment of chromosome may be chosen, as illustrated in Figure 5[2].

Mutate

Function *mutate(c,pm)* occasionally changes genes of chromosome *c* at a probability of *pm*, while also considering the fitness of *c* as an additional weight. This function is illustrated in Figure 6.

[2] It should be noted that the newly produced offsprings will be added to the population.

```
population crossover(pop,pc)
begin
    pop-temp ←∅;
    for ∀c1={A1,1,...,A1,n}∋ pop  do
    begin
        for ∀c2={A2,1,...,A2,n}∋ pop ∧c2 ≠c1 do
        begin
          if (frand() <pc) then
          begin
            i ←irand(l+1);
            k ←irand(l+1);
            (i,k) ←(min(i,k),max(i,k));
            c3 ←(A1,1,...,A1,i,A2,i+1,...,A2,k-1,A1,k,...,A1,n);
            c4 ← (A2,1,...,A2,i,A1,i+1,...,A1,k-1,A2,k,...,A2,n);
          end
        end
    end
return pop-temp
end
```

Figure 5: Crossover function.

```
chromosome mutate(c,pm)
begin
    if (frand()*fit(c)<pm) then
    begin
        c.A0 ←irand(l-2)+1;
        i ←irand(l-1)+1;
        c.Ai ←irand(n-2);
    end
return c;
end
```

Figure 6: Mutate function.

Fitness function

Usually a large number of association rules are resulted from an association mining process. Fitness function aims at searching the most interesting association rules. Hence, it is crucial to choose an appropriate fitness function. Interestingness measures have been proposed and studied to evaluate and rank them based on their interestingness. Thus, they are used as fitness functions in genetic algorithms. For instance, given a rule, $A \rightarrow B$, for database D, support s and confidence c are two objective measures to respectively assess the frequency and coverage of an association rule. These measures state that $s\%$ of transactions in D contain $A \cap B$ and $c\%$ of transactions that contain A also contain B. Another interestingness measure is *Recall*. It is a measure assessing the conciseness of an association rule, which is defined as the ratio between the number of correctly returned rules to the total number of correct rules that should have been returned. This measure states that $r\%$ of relevant association rules are retrieved. For more discussions the interested readers are referred to [3], [5], [6], [7], [8], [9], [10], [11], [12], [13], [22], [23], [24], [25]. The following formula is the fitness function used in this project:

$$fitness(c) = \frac{supp(A_1 \dots A_k) - supp(A_1 \dots A_j)supp(A_{j+1} \dots A_k)}{supp(A_1 \dots A_j)(1 - supp(A_{j+1} \dots A_k))}$$

where for an association rule of the form $\{A_1, A_2,\dots,A_j\} \rightarrow \{A_{j+1},A_{j+2},\dots,A_n\}$, $supp(A_1, A_2,\dots,A_j)$ is the support of items $\{ A_1, A_2,\dots,A_j \}$.

Initialization

Given a seed chromosome, *mutate* is used to produce an initial population *pop[0]*, where we have *pm=1*. This initialization is shown in Figure 7. This function obtains a seed chromosome as its parameter and returns a population as the initial set of chromosomes.

```
population initialize(s)
begin
    pop[0]←s;
    while sizeof(pop[0])<popsize/2 do
    begin
        pop-temp ← Φ;
        for ∀c∋pop[0] do
        begin
            pop-temp ←pop-temp ∪mutate(c,1);
        end
        pop[0] ←pop[0] ∪pop−temp;
    return pop[0];
end
```

Figure 5: ARGMA initialization.

```
population ARGMA(s,ps,pc,pm)
begin
    i ←0;
    pop[i] ←initialize(s);
    while not terminate(pop[i]) do
    begin
        pop[i+1] ← Φ;
        pop-temp ← Φ;
        for ∀c∋pop[i] do
            if select(c,ps) then
              pop[i+1] ←pop[i+1] ∪c;
        pop-temp ←crossover(pop[i+1],pc);
        for ∀c∋pop-temp do
            pop[i+1] ←(pop[i+1]-c) ∪mutate(c,pm);
    end
    return pop[i]
end
```

Figure 8: ARGMA algorithm.

ARGMA algorithm

ARGMA algorithm, as illustrated in Figure 8, selects appropriate chromosomes and performs crossover and mutation to generate a population with high quality chromosomes. ARGMA stops under the following conditions:

1. The difference between the best and the worst chromosome is less than a given small value.
2. The number of iterations, *i*, is larger than a given maximum number.

5. Empirical Studies

The empirical studies are carried out on a real-life data set - *the car evaluation data set*, retrieved from the UCI Repository [26].

Implementation

Preparing data sets

The *car evaluation data set* includes 1728 records with 6 input attributes, such as *buying-price* (*buying* for short), *maintenance-price (maint), number-of-doors (doors), number-of-people (persons), size-of-lug-boot (lug_boot),* and *safety (safety)*. In order to make use of ARGMA algorithm [18] to extract association rules, the attributes and their respective values are combined and converted into 21 binary attributes. For instance, attribute buying has four values, namely *vhigh, high, med,* and *low*. Correspondingly, we have four binary attributes as *bp vhigh, bp high, bp med,* and *bp low*.

ARGMA algorithm is applied to the car dataset with probability of 50%, where 200 resultant association rules are reported.

Evaluation

In order to evaluate the outcome of ARGMA algorithm, I measured *Accuracy, Recall,* and *Support,* mentioned in Section IV, of the produced association rules. Moreover, I used Apriori algorithm to mine association rules. Then, I compared the results of both experiments

Five rules of the results of the experiments conducted are shown in Table 2. The complete set of rules is listed in Appendix II.

Discovered Rules	Accuracy	Recall	Support
buying=vhigh → maint=vhigh	0.625	0.250	0.062
persons=2 → maint=high	0.583	0.333	0.083
buying=vhigh→ lug_boot=small	0.583	0.250	0.083
lug_boot=med → maint=high	0.583	0.333	0.083
safety=high → persons=more	0.556	0.333	0.111

Table 2: Evaluation of the results of AGMA algorithm

Table 3 presents five rules using Apriori algorithm. The complete set of rules is listed in Appendix III.

Discovered Rules	Accuracy	Recall	Support
lug_boot=small → persons=2	0.556	0.333	0.111
persons=2 → safety=low	0.556	0.333	0.111
lug_boot=big → persons=4	0.556	0.333	0.111
lug_boot=small → safety=med	0.556	0.333	0.111
buying=med → persons=2	0.583	0.250	0.083

Table 3: Evaluation of the results of Apriori algorithm

Considering the aforementioned tables, ARGMA generally works better for the discovered association rules in terms of Accuracy, Recall, and Support.

6. Conclusions and Future Work

The main advantage of ARGMA algorithm is its database independence. The effectiveness of ARGMA algorithm highly depends on the seed chromosome and the fitness function. This algorithm is independent of users' interference. However, pursuing some subjective criteria, such as novelty and surprisingness, needs users' feedbacks. In order to evaluate subjective criteria, users' ideas should be taken into account. Moreover, users tend to be inconsistent in presenting their preferences. Thus, there should be a system that learns from users' preferences, removing their inconsistencies to mine association rules that are interesting. For this reason, neural networks could be used. In addition, this algorithm covers categorical data. It could be extended to cover continuous data.

References

[1] J. Han and M. Kamber, Data Mining: Concepts and Techniques, 2nd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

[2] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules in large databases," in VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 487-499.

[3] R. J. Hilderman and H. J. Hamilton, “Evaluation of interestingness measures for ranking discovered knowledge," in Lecture Notes in Computer Science. Springer-Verlag, 2001, pp. 247-259.

[4] P. Lenca, P. Meyer, B. Vaillant, and S. Lallich, “On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid,” European Journal of Operational Research, vol. 184, no. 2, pp. 610-626, 2008.

[5] B. Liu, W. Hsu, S. Chen, and Y. Ma, “Analyzing the subjective interestingness of association rules.” IEEE Intelligent Systems and their Applications, vol. 15, no. 5, pp. 47-55, 2000.

[6] O. Maimon and L. Rokach, Data Mining and Knowledge Discovery Handbook. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[7] B. Padmanabhan and A. Tuzhilin, “Unexpectedness as a measure of interestingness in knowledge discovery”, Decision Support Systems, vol. 27, no. 3, pp. 303-318, 1999.

[8] A. Silberschatz and A. Tuzhilin, “What makes patterns interesting in knowledge discovery systems”, IEEE Transactions on Knowledge and Data Engineering, vol. 8, no. 6, pp. 970-974, 1996.

[9] B. Vaillant, P. Lenca, and S. Lallich, “A clustering of interestingness measures”, in Lecture Notes in Computer Science. Springer Berlin/Heidelberg, 2004, pp. 290-297.

[10] L. Geng and H. J. Hamilton, “Choosing the right lens: finding what is interesting in data mining”, in Quality Measures in Data Mining. Springer Berlin / Heidelberg, 2007, vol. 43, pp. 3-24.

[11] P. Lenca, B. Vaillant, P. Meyer, and S. Lallich, Quality Measures in Data Mining, ser. Studies in Computational Intelligence. Springer Berlin/Heidelberg, 2007, vol. 43, ch. Association Rule Interestingness Measures: Experimental and Theoretical Studies, pp. 51-76.

[12] S. Sahar, “Interestingness via what is not interesting”, in KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. New York, NY, USA: ACM, 1999, pp. 332-336.

[13] A. A. Freitas, “On rule interestingness measures”, Elsevier Science, no. 3, pp. 309-315, 1999.

[14] P. Tan, V. Kumar, and J. Srivastava, “Selecting the right interestingness measure for association patterns”, in Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. Edmonton, Alberta, Canada: ACM, 2002, pp. 32-41.

[15] J. Hipp, U. Guntzer, and G. Nakhaeizadeh, “Algorithms for association rule mining - a general survey and comparison”, SIGKDD Exploration Newsletter, vol. 2, no. 1, pp. 58-64, 2000.

[16] B. Liao, “An improved algorithm of Apriori”, Computational intelligence and intelligent systems: fourth international symposium, ISICA, pp. 427-432, 2009.
[17] H. Wu, Z. Lu, L. Pan, R. Xu, and W. Jiang, “An improved apriori-based algorithm for association rules mining”, in FSKD'09: Proceedings of the 6th international conference on Fuzzy systems and knowledge discovery. Piscataway, NJ, USA: IEEE Press, 2009, pp. 51-55.
[18] R. Agrawal, T. Imielifinski, and A. Swami, “Mining association rules between sets of items in large databases”, in Proceedings of the International Conference on Management of Data. New York, NY, USA: ACM, 1993, pp. 207-216.
[19] X. Yan, C. Zhang, and S. Zhang, “ARGMA: Identifying interesting association rules with genetic algorithms”, Applied Artificial Intelligence, pp. 677-689, 2005.
[20] M. J. d. Jesus, J. A. Gamez, P. Gonzalez, and J. M. Puerta, “On the discovery of association rules by means of evolutionary algorithms”, John Wiley and Sons Inc., vol. 00, pp. 1-19, 2011.
[21] P. P. Wakabi-Waiswa and V. Baryamureeba, “Extraction of interesting association rules using genetic algorithms”, Advances in system modelling and ICT applications, pp. 1818-1139, 2008.
[22] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo, “Finding interesting rules from large sets of discovered association rules”, in Proceedings of the third international conference on Information and knowledge management, 1994.
[23] E. R. Omiecinski, “Alternative interest measures for mining associations in databases”, IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 1, pp. 57-69, 2003.
[24] P. Tan, V. Kumar, and J. Srivastava, “Selecting the right objective measure for association analysis”, Information System, vol. 29, no. 4, pp. 293-313, 2004.
[25] Y. Zhao, C. Zhang, and S. Zhang, “Discovering Interesting Association Rules by Clustering”. Secaucus, NJ, USA: Springer Berlin / Heidelberg, 2004.
[26] M. Bohanec and B. Zupan, UCI machine learning repository: Car evaluation data set, http://archive.ics.uci.edu/ml/datasets/car+evaluation, 1997.

Appendix I: User manual

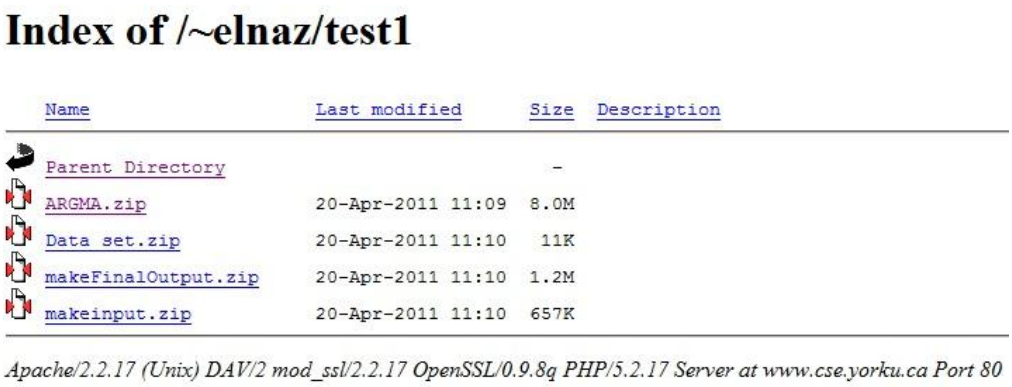
Running the program includes two steps:

- Step 1: Prepare the input.
- Step 2: Run the genetic algorithm.

Step 1: Prepare the input:

In order to prepare the input, the following steps need to be done:

1. Install Visual studio 2010.
2. Click on the following link: http://www.cse.yorku.ca/~elnaz/test1. The following window will open:



Index of /~elnaz/test1

Name	Last modified	Size	Description
Parent Directory		-	
ARGMA.zip	20-Apr-2011 11:09	8.0M	
Data_set.zip	20-Apr-2011 11:10	11K	
makeFinalOutput.zip	20-Apr-2011 11:10	1.2M	
makeinput.zip	20-Apr-2011 11:10	657K	

Apache/2.2.17 (Unix) DAV/2 mod_ssl/2.2.17 OpenSSL/0.9.8q PHP/5.2.17 Server at www.cse.yorku.ca Port 80

Download and save the zip files.

3. Download the car evaluation dataset from UCI repository and convert it to an arff file. It can also be found in the dataset folder of the current submitted project.
4. Run the program, called "makeinput". It basically takes the input data and produces binary output. For example, for this dataset,

   input: vhigh,vhigh,2,2,small,low.

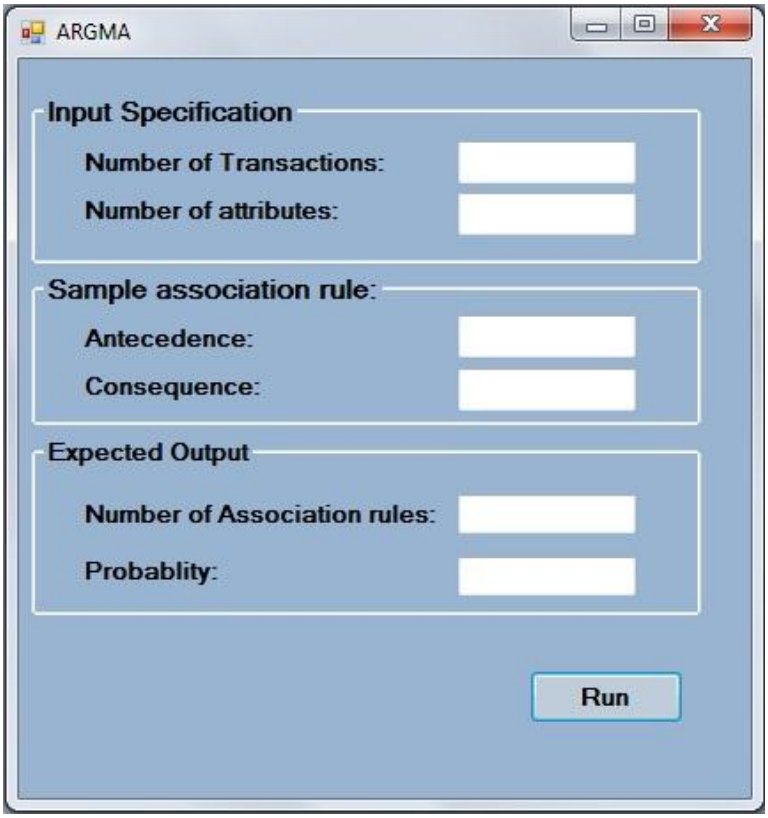
   output: 100010001000100100100

Step 2: Run the genetic algorithm:

In order to run the genetic algorithm, the following steps need to be done:

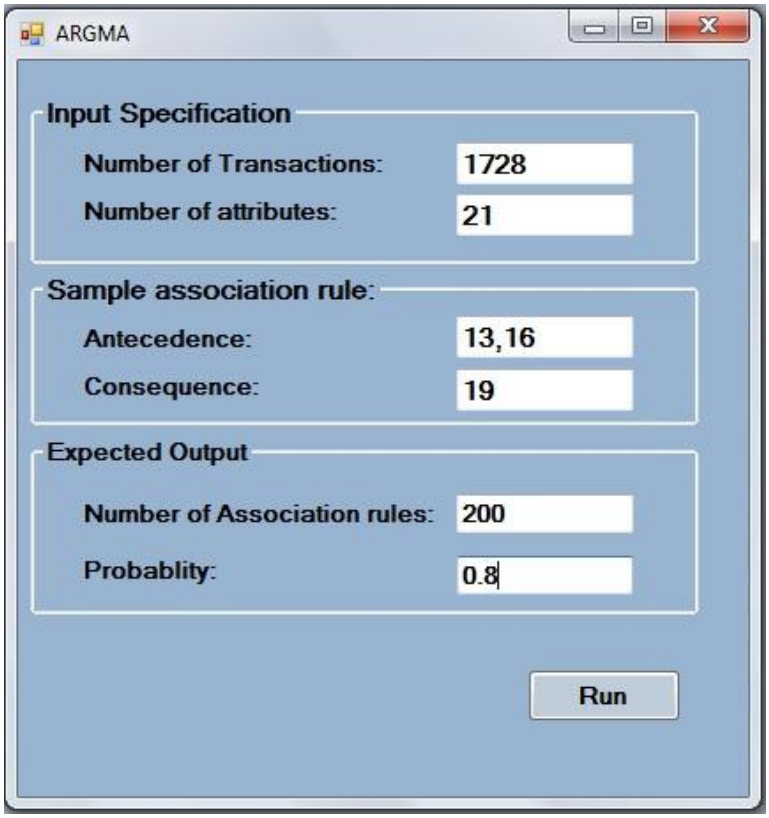
1. Install Visual studio 2010.

2. Complete the information in the following window.



Firstly, specify the number of transactions and attributes of the dataset. Then, input the seed association rule. Specify the antecedence and the consequence of the rule. Use ‘,’ to separate them. Finally, enter your expected output association rule, such as the number of association rule.

Here, I illustrated an example of an execution using the car evaluation dataset.



Once you are done with the inputs, click "Run". The output is produced and saved in a file named testoutput.txt. A sample output looks like this:

C, ,C, , , , , ,A, , , , , , , , , , , ,
C, ,C, , ,A, ,C,A, , , , , , , , , , , ,
C, ,C, , ,A,A,C, , , , , , , , , , , , ,
C, ,C, , ,A,A,C, , , , , , , , , , , , ,
C, ,C,C,A,A,A,C, , , , , , , , , , , , ,
C, ,C,C,A,A,A,C, , , , , , , , , , , , ,
C, ,C,C, ,A,A, , , , , , , , , , , , , ,
C, ,C,C, ,A,A, , , , , , , , , , , , , ,
C, ,C,C, ,A,A, , , , , , , , , , , , , ,
C, ,C,C, , ,A, , , ,A, , , , , , , , , ,

'A' stands for antecedence and 'C' stands for consequence.

3. Convert the previous output to readable association rules. To do so, run the code in folder ‘makeFinalOutput’. It takes the aforementioned output as its input and produces association rules of the form A→B. A complete set of output is listed in Appendix II.

Appendix II: Association rules resulted from ARGMA algorithm

Here is a sample output of my code, which is a revised version of AGMA algorithm. Numbers 1 to 21 represent successively attributes: buying_v-high, buying_high, buying_med, buying_low, maint_v-high, maint_high, maint_med, maint_low, doors_ 2, doors_3, doors_4, doors_more, persons_2, persons_4, persons_more, lug_boot_small, lug_boot_med, lug_boot_big, safety_low, safety_med, safety_high.

{9,}->{1,3,}
{6,9,}->{1,3,8,}
{6,7,}->{1,3,8,}
{6,7,}->{1,3,8,}
{5,6,7,}->{1,3,4,8,}
{5,6,7,}->{1,3,4,8,}
{6,7,}->{1,3,4,}
{6,7,}->{1,3,4,}
{6,7,}->{1,3,4,}
{7,11,}->{1,3,4,}
{7,11,}->{1,3,4,}
{11,}->{3,4,}
{11,}->{3,4,}
{11,}->{3,4,}
{11,}->{3,4,}
{11,}->{3,4,}
{11,}->{2,3,}
{11,}->{2,6,}
{10,11,}->{2,4,6,}
{10,11,}->{2,4,6,}
{10,11,}->{2,4,6,}
{10,11,}->{2,4,6,}
{10,11,}->{2,4,5,}
{10,11,}->{2,4,8,}
{10,11,}->{2,4,8,}
{3,10,11,}->{2,4,7,8,}
{3,10,11,}->{2,4,7,8,}
{5,10,11,}->{2,4,7,8,}
{5,10,11,}->{2,4,7,8,}
{5,11,}->{2,4,8,}
{5,6,11,}->{2,3,4,8,}
{5,6,11,}->{2,3,4,8,}
{5,6,11,}->{2,3,4,8,}
{5,6,11,}->{2,4,7,8,}
{5,6,11,}->{2,4,7,8,}
{5,11,}->{4,7,8,}
{3,11,}->{4,7,8,}
{3,11,}->{2,4,8,}
{3,11,}->{2,6,8,}
{4,11,}->{2,6,8,}
{4,11,}->{2,5,6,}
{11,}->{2,6,}
{11,}->{2,3,}
{11,}->{2,4,}
{10,11,}->{2,4,6,}
{10,11,}->{2,4,7,}

{10,11,}->{2,5,7,}
{10,11,}->{2,5,7,}
{10,11,}->{2,5,7,}
{10,11,}->{2,3,5,}
{10,}->{2,5,}
{10,}->{2,5,}
{10,}->{2,5,}
{10,}->{2,5,}
{7,10,}->{2,4,5,}
{7,10,}->{2,4,5,}
{6,7,}->{2,4,5,}
{6,}->{2,4,}
{6,}->{2,4,}
{6,}->{2,4,}
{6,}->{2,4,}
{6,}->{2,4,}
{6,}->{3,4,}
{6,}->{3,4,}
{6,}->{3,4,}
{6,}->{3,4,}
{6,7,}->{1,3,4,}
{6,}->{1,3,}
{4,6,}->{1,3,7,}
{4,6,}->{1,3,7,}
{6,}->{1,3,}
{6,}->{1,3,}
{6,}->{1,3,}
{8,}->{1,3,}
{8,}->{1,3,}
{8,10,}->{1,3,6,}
{8,10,}->{1,3,6,}
{8,10,}->{1,6,7,}
{8,}->{1,6,}
{8,}->{1,6,}
{8,}->{1,4,}
{8,}->{4,5,}
{}->{5,}
{}->{5,}
{}->{6,}
{}->{6,}
{5,}->{4,6,}
{5,}->{4,6,}
{5,}->{4,6,}
{3,5,}->{4,6,7,}
{2,3,}->{4,6,7,}
{2,3,}->{4,6,7,}
{2,3,}->{4,6,7,}
{2,}->{4,7,}
{2,}->{4,7,}
{2,}->{4,7,}
{2,5,}->{4,7,8,}
{2,}->{4,8,}
{2,}->{4,8,}
{2,}->{4,8,}
{2,}->{4,8,}
{2,9,}->{4,7,8,}

{2,9,}->{4,7,8,}
{2,9,}->{4,7,8,}
{2,6,9,}->{4,7,8,11,}
{6,9,}->{7,8,11,}
{6,9,}->{7,8,11,}
{6,9,}->{7,8,11,}
{4,6,}->{7,8,11,}
{4,6,}->{7,8,11,}
{4,6,}->{7,8,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{4,}->{7,11,}
{5,}->{7,11,}
{4,5,}->{2,7,11,}
{4,5,}->{2,7,11,}
{4,5,}->{2,7,11,}
{4,5,}->{2,7,11,}
{4,5,}->{2,7,11,}
{5,}->{2,7,}
{5,}->{2,3,}
{5,}->{2,3,}
{5,}->{2,11,}
{5,}->{2,11,}
{5,}->{3,11,}
{5,}->{3,11,}
{6,}->{3,11,}
{6,}->{3,11,}
{6,}->{5,11,}
{6,}->{5,11,}
{6,}->{3,11,}
{6,10,}->{3,4,11,}
{6,10,}->{3,4,11,}
{6,10,}->{3,4,11,}
{6,10,}->{3,4,11,}
{6,10,}->{3,4,11,}
{10,}->{3,4,}
{10,}->{2,4,}
{10,}->{2,4,}
{10,}->{4,6,}
{10,}->{4,6,}
{10,}->{4,6,}
{10,}->{4,6,}
{10,}->{4,6,}
{10,}->{2,4,}
{10,}->{2,4,}
{10,}->{2,4,}
{10,}->{2,9,}
{10,}->{2,9,}
{10,}->{5,9,}
{10,}->{1,9,}
{10,}->{1,9,}
{8,10,}->{1,3,9,}

{8,10,}->{1,3,9,}
{8,10,}->{1,6,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{8,10,}->{1,4,9,}
{7,10,}->{1,4,9,}
{7,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{2,5,10,}->{1,3,4,9,}
{2,5,10,}->{1,3,4,9,}
{2,5,10,}->{1,3,4,9,}
{2,10,}->{1,4,9,}
{2,10,}->{1,8,9,}
{2,10,}->{1,8,9,}
{2,10,}->{1,8,9,}
{2,10,}->{1,8,9,}
{2,3,10,}->{1,7,8,9,}
{2,3,5,10,}->{1,4,7,8,9,}
{2,3,5,10,}->{1,4,7,8,9,}
{2,5,10,}->{4,7,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{3,8,9,}
{10,}->{8,9,}
{10,}->{8,9,}
{5,10,}->{4,8,9,}
{5,10,}->{4,8,9,}
{5,10,}->{4,8,9,}
{3,5,10,}->{4,6,8,9,}
{3,10,}->{4,8,9,}
{3,7,10,}->{4,8,9,11,}
{3,7,10,}->{4,8,9,11,}
{3,7,10,}->{4,8,9,11,}
{7,10,}->{1,4,8,9,11,}
{3,10,}->{8,9,11,}
{3,10,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{8,9,11,}
{3,5,}->{4,9,11,}
{3,5,10,}->{4,7,9,11,}
{3,5,10,}->{4,7,9,11,}
{3,5,10,}->{4,7,9,11,}
{3,10,}->{4,9,11,}

{3,10,}->{4,9,11,}
{3,10,}->{4,9,11,}
{10,}->{9,11,}
{4,10,}->{3,9,11,}
{4,10,}->{3,9,11,}
{4,6,10,}->{3,8,9,11,}
{4,6,10,}->{3,8,9,11,}
{6,10,}->{8,9,11,}
{6,10,}->{8,9,11,}
{10,}->{9,11,}
{10,}->{9,11,}
{10,}->{3,9,}
{10,}->{3,9,}
{10,}->{3,9,}
{10,}->{3,9,}
{6,10,}->{3,7,9,}
{10,}->{3,9,}
{10,}->{1,9,}
{10,}->{1,9,}
{4,10,}->{1,6,9,}
{4,10,}->{1,6,9,}
{4,10,}->{1,6,9,}
{4,10,}->{1,6,9,}
{4,}->{1,9,}
{4,8,}->{1,3,9,}
{4,8,11,}->{1,3,5,9,}
{4,8,11,}->{1,3,5,9,}
{4,8,11,}->{1,2,3,9,}
{4,8,10,11,}->{1,2,3,5,9,}
{4,8,10,11,}->{1,2,3,5,9,}
{4,8,10,11,}->{1,2,3,5,9,}
{4,8,10,11,}->{1,2,3,5,9,}
{8,10,11,}->{1,2,5,9,}
{4,8,11,}->{1,2,5,9,}
{4,8,11,}->{1,2,5,9,}
{4,8,11,}->{1,2,5,9,}
{1,2,7,8,9,10,11,}->{4,5,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,7,8,9,10,11,}->{1,6,}
{2,5,7,8,9,10,}->{1,6,}
{2,5,7,8,9,10,}->{1,6,}
{2,3,7,8,9,10,}->{1,6,}
{3,7,8,9,10,}->{1,}
{3,7,8,9,10,}->{1,}
{3,7,8,9,10,}->{1,}
{3,5,8,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}

{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,5,9,10,}->{1,}
{3,4,7,9,10,}->{1,}
{3,4,7,9,10,}->{1,}
{3,4,7,9,10,}->{1,}
{3,4,7,9,10,}->{1,}
{3,4,7,9,10,}->{1,}
{3,7,9,10,}->{}
{2,3,7,9,10,}->{4,}
{2,3,7,10,}->{}
{2,3,5,10,}->{}
{2,3,5,10,}->{}
{2,3,5,10,11,}->{4,}
{2,3,5,10,11,}->{4,}
{2,3,5,10,11,}->{4,}
{2,3,5,10,11,}->{4,}
{2,3,5,10,11,}->{4,}
{2,3,5,10,11,}->{7,}
{3,5,10,11,}->{}
{3,9,10,11,}->{}
{3,4,9,10,11,}->{6,}
{3,4,9,10,11,}->{6,}
{3,4,9,10,11,}->{6,}
{3,4,9,10,11,}->{6,}
{3,9,10,11,}->{}
{3,9,10,11,}->{}
{2,}->{1,3,}
{2,6,}->{1,3,8,}
{2,6,}->{1,3,8,}
{2,6,}->{1,3,8,}
{2,5,6,}->{1,3,4,8,}
{2,5,6,}->{1,3,4,8,}
{2,6,}->{1,3,4,}
{2,6,}->{1,3,4,}
{2,6,}->{1,3,4,}
{2,11,}->{1,3,4,}
{2,7,11,}->{1,3,4,6,}
{2,11,}->{3,4,6,}
{2,11,}->{3,4,5,}
{2,11,}->{3,4,10,}
{2,11,}->{3,4,10,}
{2,11,}->{3,4,10,}
{11,}->{3,10,}
{11,}->{6,10,}
{11,}->{4,6,}
{11,}->{4,6,}
{11,}->{4,6,}
{11,}->{4,6,}
{11,}->{4,5,}
{11,}->{4,8,}
{11,}->{4,8,}
{3,11,}->{4,7,8,}
{3,11,}->{2,4,8,}

{5,11,}->{2,4,8,}
{5,11,}->{2,4,8,}
{5,7,11,}->{2,4,8,10,}
{5,6,7,11,}->{2,3,4,8,10,}
{5,6,7,11,}->{2,3,4,8,10,}
{5,6,7,11,}->{2,3,4,8,10,}
{5,6,11,}->{2,4,8,10,}
{5,11,}->{2,4,8,}
{5,11,}->{4,6,8,}
{3,11,}->{4,6,8,}
{3,11,}->{4,6,8,}
{3,11,}->{4,6,8,}
{11,}->{6,8,}
{11,}->{5,6,}
{11,}->{4,6,}
{11,}->{3,4,}
{11,}->{3,4,}
{10,11,}->{3,4,6,}
{10,11,}->{3,4,7,}
{10,11,}->{3,5,7,}
{10,11,}->{3,5,7,}
{10,11,}->{3,5,7,}
{10,11,}->{3,5,7,}
{10,}->{5,7,}
{10,}->{5,7,}
{10,}->{5,7,}
{10,}->{5,7,}
{10,}->{4,5,}
{10,}->{4,5,}
{6,}->{4,5,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,7,}
{6,}->{4,8,}
{6,7,}->{1,4,8,}
{6,}->{1,8,}
{4,6,}->{1,7,8,}
{4,6,}->{1,7,8,}
{6,}->{1,8,}
{6,}->{1,8,}
{6,7,}->{1,3,8,}
{7,}->{1,3,}
{7,}->{1,3,}
{7,10,}->{1,3,6,}
{5,10,}->{1,3,6,}
{5,10,}->{1,6,7,}
{5,}->{1,6,}
{5,}->{1,6,}
{5,}->{1,4,}
{}->{4,}
{}->{8,}

{4,}->{5,8,}
{4,}->{6,8,}
{4,}->{6,8,}
{5,}->{6,8,}
{5,}->{6,8,}
{5,}->{4,8,}
{3,5,}->{4,7,8,}
{2,3,}->{4,7,8,}
{2,3,}->{4,7,8,}
{2,3,}->{4,6,8,}
{2,}->{4,8,}
{2,3,}->{4,7,8,}
{2,3,}->{4,7,8,}
{2,3,}->{4,7,8,}
{2,3,}->{4,5,8,}
{2,3,}->{1,4,8,}
{2,3,}->{1,4,8,}
{2,3,}->{1,4,8,}
{2,3,9,}->{1,4,7,8,}
{2,3,9,}->{1,4,7,8,}
{2,3,9,}->{4,5,7,8,}
{2,3,6,9,}->{4,5,7,8,11,}
{3,6,9,}->{5,7,8,11,}
{6,9,}->{7,8,11,}
{6,9,}->{7,8,11,}
{4,6,}->{7,8,11,}
{4,6,}->{7,8,11,}
{5,6,}->{7,8,11,}
{5,}->{7,11,}
{5,}->{7,11,}
{}->{11,}
{}->{7,}
{}->{7,}
{}->{7,}
{}->{7,}
{5,}->{4,7,}
{5,}->{2,7,}
{5,}->{2,7,}
{5,}->{2,7,}
{5,}->{2,7,}
{5,}->{2,7,}
{5,}->{2,7,}
{5,}->{2,3,}
{5,}->{2,3,}
{5,}->{2,3,}
{5,}->{2,3,}
{5,}->{2,3,}
{5,}->{2,3,}
{6,}->{2,3,}
{6,}->{2,3,}
{6,}->{2,5,}
{6,}->{2,5,}
{6,}->{2,3,}
{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}

{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}
{6,10,}->{2,3,4,}
{10,}->{3,4,}
{10,}->{3,4,}
{10,}->{3,4,}
{5,10,}->{3,4,6,}
{5,10,}->{3,4,6,}
{5,10,}->{2,3,4,}
{5,10,}->{2,4,7,}
{5,10,}->{2,4,7,}
{5,10,}->{2,7,9,}
{10,}->{2,9,}
{10,}->{5,9,}
{10,}->{1,9,}
{5,10,}->{1,7,9,}
{5,8,10,}->{1,3,7,9,}
{5,8,10,}->{1,3,7,9,}
{5,8,10,}->{1,6,7,9,}
{5,8,10,}->{1,4,7,9,}
{5,8,10,}->{1,4,7,9,}
{5,8,10,}->{1,3,4,9,}
{5,8,10,}->{1,3,4,9,}
{5,8,10,}->{1,3,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{5,10,}->{1,4,9,}
{2,5,10,}->{1,3,4,9,}
{2,5,10,}->{1,3,4,7,}
{2,5,10,}->{1,3,4,7,}
{2,10,}->{1,4,7,}
{2,10,}->{1,7,8,}
{2,10,}->{1,7,8,}
{2,10,}->{1,6,8,}
{2,10,}->{1,6,8,}
{2,3,10,}->{1,6,7,8,}
{2,3,5,10,}->{1,4,6,7,8,}
{2,3,5,10,}->{1,4,6,7,8,}
{2,5,10,}->{4,6,7,8,}
{2,10,}->{4,6,8,}
{2,10,}->{6,8,9,}
{2,10,}->{6,8,9,}
{2,10,}->{6,8,9,}
{2,10,}->{4,8,9,}
{2,10,}->{3,8,9,}
{10,}->{8,9,}
{10,}->{8,9,}
{5,10,}->{4,8,9,}
{5,10,}->{4,8,9,}
{5,10,}->{4,8,9,}

{5,10,}->{4,8,9,}
{5,10,}->{4,8,9,}
{5,7,10,}->{4,8,9,11,}
{7,10,}->{8,9,11,}
{13,}->{17,20,}

Appendix III: Association rules resulted Apriori algorithm

This Appendix includes the association rules resulted from Apriori algorithm ranked based on their interetingness. To extract these rules WEKA is used.

Min con:0
Min supp: 0.05
=== Run information ===

Scheme: weka.associations.Apriori -N 1000000 -T 0 -C 0.0 -D 0.05 -U 1.0 -M 0.05 -S -1.0 -c -1
Relation: car
Instances: 1728
Attributes: 6
buying
maint
doors
persons
lug_boot
safety
=== Associator model (full training set) ===

Apriori
=======

Minimum support: 0.05 (86 instances)
Minimum metric <confidence>: 0
Number of cycles performed: 19

Generated sets of large itemsets:

Size of set of large itemsets L(1): 21

Size of set of large itemsets L(2): 183

Best rules found:

1. lug_boot=small 576 ==> persons=2 192 conf:(0.33)
2. persons=2 576 ==> lug_boot=small 192 conf:(0.33)
3. lug_boot=med 576 ==> persons=2 192 conf:(0.33)
4. persons=2 576 ==> lug_boot=med 192 conf:(0.33)
5. lug_boot=big 576 ==> persons=2 192 conf:(0.33)
6. persons=2 576 ==> lug_boot=big 192 conf:(0.33)
7. safety=low 576 ==> persons=2 192 conf:(0.33)
8. persons=2 576 ==> safety=low 192 conf:(0.33)
9. safety=med 576 ==> persons=2 192 conf:(0.33)
10. persons=2 576 ==> safety=med 192 conf:(0.33)
11. safety=high 576 ==> persons=2 192 conf:(0.33)
12. persons=2 576 ==> safety=high 192 conf:(0.33)
13. lug_boot=small 576 ==> persons=4 192 conf:(0.33)
14. persons=4 576 ==> lug_boot=small 192 conf:(0.33)
15. lug_boot=med 576 ==> persons=4 192 conf:(0.33)
16. persons=4 576 ==> lug_boot=med 192 conf:(0.33)
17. lug_boot=big 576 ==> persons=4 192 conf:(0.33)
18. persons=4 576 ==> lug_boot=big 192 conf:(0.33)
19. safety=low 576 ==> persons=4 192 conf:(0.33)
20. persons=4 576 ==> safety=low 192 conf:(0.33)
21. safety=med 576 ==> persons=4 192 conf:(0.33)
22. persons=4 576 ==> safety=med 192 conf:(0.33)
23. safety=high 576 ==> persons=4 192 conf:(0.33)
24. persons=4 576 ==> safety=high 192 conf:(0.33)
25. lug_boot=small 576 ==> persons=more 192 conf:(0.33)
26. persons=more 576 ==> lug_boot=small 192 conf:(0.33)
27. lug_boot=med 576 ==> persons=more 192 conf:(0.33)
28. persons=more 576 ==> lug_boot=med 192 conf:(0.33)
29. lug_boot=big 576 ==> persons=more 192 conf:(0.33)
30. persons=more 576 ==> lug_boot=big 192 conf:(0.33)
31. safety=low 576 ==> persons=more 192 conf:(0.33)
32. persons=more 576 ==> safety=low 192 conf:(0.33)
33. safety=med 576 ==> persons=more 192 conf:(0.33)
34. persons=more 576 ==> safety=med 192 conf:(0.33)
35. safety=high 576 ==> persons=more 192 conf:(0.33)
36. persons=more 576 ==> safety=high 192 conf:(0.33)
37. safety=low 576 ==> lug_boot=small 192 conf:(0.33)
38. lug_boot=small 576 ==> safety=low 192 conf:(0.33)
39. safety=med 576 ==> lug_boot=small 192 conf:(0.33)
40. lug_boot=small 576 ==> safety=med 192 conf:(0.33)
41. safety=high 576 ==> lug_boot=small 192 conf:(0.33)
42. lug_boot=small 576 ==> safety=high 192 conf:(0.33)
43. safety=low 576 ==> lug_boot=med 192 conf:(0.33)
44. lug_boot=med 576 ==> safety=low 192 conf:(0.33)
45. safety=med 576 ==> lug_boot=med 192 conf:(0.33)
46. lug_boot=med 576 ==> safety=med 192 conf:(0.33)
47. safety=high 576 ==> lug_boot=med 192 conf:(0.33)
48. lug_boot=med 576 ==> safety=high 192 conf:(0.33)
49. safety=low 576 ==> lug_boot=big 192 conf:(0.33)
50. lug_boot=big 576 ==> safety=low 192 conf:(0.33)
51. safety=med 576 ==> lug_boot=big 192 conf:(0.33)
52. lug_boot=big 576 ==> safety=med 192 conf:(0.33)
53. safety=high 576 ==> lug_boot=big 192 conf:(0.33)
54. lug_boot=big 576 ==> safety=high 192 conf:(0.33)

55. buying=vhigh 432 ==> persons=2 144 conf:(0.33)
56. buying=vhigh 432 ==> persons=4 144 conf:(0.33)
57. buying=vhigh 432 ==> persons=more 144 conf:(0.33)
58. buying=vhigh 432 ==> lug_boot=small 144 conf:(0.33)
59. buying=vhigh 432 ==> lug_boot=med 144 conf:(0.33)
60. buying=vhigh 432 ==> lug_boot=big 144 conf:(0.33)
61. buying=vhigh 432 ==> safety=low 144 conf:(0.33)
62. buying=vhigh 432 ==> safety=med 144 conf:(0.33)
63. buying=vhigh 432 ==> safety=high 144 conf:(0.33)
64. buying=high 432 ==> persons=2 144 conf:(0.33)
65. buying=high 432 ==> persons=4 144 conf:(0.33)
66. buying=high 432 ==> persons=more 144 conf:(0.33)
67. buying=high 432 ==> lug_boot=small 144 conf:(0.33)
68. buying=high 432 ==> lug_boot=med 144 conf:(0.33)
69. buying=high 432 ==> lug_boot=big 144 conf:(0.33)
70. buying=high 432 ==> safety=low 144 conf:(0.33)
71. buying=high 432 ==> safety=med 144 conf:(0.33)
72. buying=high 432 ==> safety=high 144 conf:(0.33)
73. buying=med 432 ==> persons=2 144 conf:(0.33)
74. buying=med 432 ==> persons=4 144 conf:(0.33)
75. buying=med 432 ==> persons=more 144 conf:(0.33)
76. buying=med 432 ==> lug_boot=small 144 conf:(0.33)
77. buying=med 432 ==> lug_boot=med 144 conf:(0.33)
78. buying=med 432 ==> lug_boot=big 144 conf:(0.33)
79. buying=med 432 ==> safety=low 144 conf:(0.33)
80. buying=med 432 ==> safety=med 144 conf:(0.33)
81. buying=med 432 ==> safety=high 144 conf:(0.33)
82. buying=low 432 ==> persons=2 144 conf:(0.33)
83. buying=low 432 ==> persons=4 144 conf:(0.33)
84. buying=low 432 ==> persons=more 144 conf:(0.33)
85. buying=low 432 ==> lug_boot=small 144 conf:(0.33)
86. buying=low 432 ==> lug_boot=med 144 conf:(0.33)
87. buying=low 432 ==> lug_boot=big 144 conf:(0.33)
88. buying=low 432 ==> safety=low 144 conf:(0.33)
89. buying=low 432 ==> safety=med 144 conf:(0.33)
90. buying=low 432 ==> safety=high 144 conf:(0.33)
91. maint=vhigh 432 ==> persons=2 144 conf:(0.33)
92. maint=vhigh 432 ==> persons=4 144 conf:(0.33)
93. maint=vhigh 432 ==> persons=more 144 conf:(0.33)
94. maint=vhigh 432 ==> lug_boot=small 144 conf:(0.33)
95. maint=vhigh 432 ==> lug_boot=med 144 conf:(0.33)
96. maint=vhigh 432 ==> lug_boot=big 144 conf:(0.33)
97. maint=vhigh 432 ==> safety=low 144 conf:(0.33)
98. maint=vhigh 432 ==> safety=med 144 conf:(0.33)
99. maint=vhigh 432 ==> safety=high 144 conf:(0.33)
100. maint=high 432 ==> persons=2 144 conf:(0.33)
101. maint=high 432 ==> persons=4 144 conf:(0.33)
102. maint=high 432 ==> persons=more 144 conf:(0.33)
103. maint=high 432 ==> lug_boot=small 144 conf:(0.33)
104. maint=high 432 ==> lug_boot=med 144 conf:(0.33)
105. maint=high 432 ==> lug_boot=big 144 conf:(0.33)
106. maint=high 432 ==> safety=low 144 conf:(0.33)
107. maint=high 432 ==> safety=med 144 conf:(0.33)
108. maint=high 432 ==> safety=high 144 conf:(0.33)
109. maint=med 432 ==> persons=2 144 conf:(0.33)
110. maint=med 432 ==> persons=4 144 conf:(0.33)

111. maint=med 432 ==> persons=more 144 conf:(0.33)
112. maint=med 432 ==> lug_boot=small 144 conf:(0.33)
113. maint=med 432 ==> lug_boot=med 144 conf:(0.33)
114. maint=med 432 ==> lug_boot=big 144 conf:(0.33)
115. maint=med 432 ==> safety=low 144 conf:(0.33)
116. maint=med 432 ==> safety=med 144 conf:(0.33)
117. maint=med 432 ==> safety=high 144 conf:(0.33)
118. maint=low 432 ==> persons=2 144 conf:(0.33)
119. maint=low 432 ==> persons=4 144 conf:(0.33)
120. maint=low 432 ==> persons=more 144 conf:(0.33)
121. maint=low 432 ==> lug_boot=small 144 conf:(0.33)
122. maint=low 432 ==> lug_boot=med 144 conf:(0.33)
123. maint=low 432 ==> lug_boot=big 144 conf:(0.33)
124. maint=low 432 ==> safety=low 144 conf:(0.33)
125. maint=low 432 ==> safety=med 144 conf:(0.33)
126. maint=low 432 ==> safety=high 144 conf:(0.33)
127. doors=2 432 ==> persons=2 144 conf:(0.33)
128. doors=2 432 ==> persons=4 144 conf:(0.33)
129. doors=2 432 ==> persons=more 144 conf:(0.33)
130. doors=2 432 ==> lug_boot=small 144 conf:(0.33)
131. doors=2 432 ==> lug_boot=med 144 conf:(0.33)
132. doors=2 432 ==> lug_boot=big 144 conf:(0.33)
133. doors=2 432 ==> safety=low 144 conf:(0.33)
134. doors=2 432 ==> safety=med 144 conf:(0.33)
135. doors=2 432 ==> safety=high 144 conf:(0.33)
136. doors=3 432 ==> persons=2 144 conf:(0.33)
137. doors=3 432 ==> persons=4 144 conf:(0.33)
138. doors=3 432 ==> persons=more 144 conf:(0.33)
139. doors=3 432 ==> lug_boot=small 144 conf:(0.33)
140. doors=3 432 ==> lug_boot=med 144 conf:(0.33)
141. doors=3 432 ==> lug_boot=big 144 conf:(0.33)
142. doors=3 432 ==> safety=low 144 conf:(0.33)
143. doors=3 432 ==> safety=med 144 conf:(0.33)
144. doors=3 432 ==> safety=high 144 conf:(0.33)
145. doors=4 432 ==> persons=2 144 conf:(0.33)
146. doors=4 432 ==> persons=4 144 conf:(0.33)
147. doors=4 432 ==> persons=more 144 conf:(0.33)
148. doors=4 432 ==> lug_boot=small 144 conf:(0.33)
149. doors=4 432 ==> lug_boot=med 144 conf:(0.33)
150. doors=4 432 ==> lug_boot=big 144 conf:(0.33)
151. doors=4 432 ==> safety=low 144 conf:(0.33)
152. doors=4 432 ==> safety=med 144 conf:(0.33)
153. doors=4 432 ==> safety=high 144 conf:(0.33)
154. doors=5more 432 ==> persons=2 144 conf:(0.33)
155. doors=5more 432 ==> persons=4 144 conf:(0.33)
156. doors=5more 432 ==> persons=more 144 conf:(0.33)
157. doors=5more 432 ==> lug_boot=small 144 conf:(0.33)
158. doors=5more 432 ==> lug_boot=med 144 conf:(0.33)
159. doors=5more 432 ==> lug_boot=big 144 conf:(0.33)
160. doors=5more 432 ==> safety=low 144 conf:(0.33)
161. doors=5more 432 ==> safety=med 144 conf:(0.33)
162. doors=5more 432 ==> safety=high 144 conf:(0.33)
163. persons=2 576 ==> buying=vhigh 144 conf:(0.25)
164. persons=4 576 ==> buying=vhigh 144 conf:(0.25)
165. persons=more 576 ==> buying=vhigh 144 conf:(0.25)
166. lug_boot=small 576 ==> buying=vhigh 144 conf:(0.25)

167. lug_boot=med 576 ==> buying=vhigh 144 conf:(0.25)
168. lug_boot=big 576 ==> buying=vhigh 144 conf:(0.25)
169. safety=low 576 ==> buying=vhigh 144 conf:(0.25)
170. safety=med 576 ==> buying=vhigh 144 conf:(0.25)
171. safety=high 576 ==> buying=vhigh 144 conf:(0.25)
172. persons=2 576 ==> buying=high 144 conf:(0.25)
173. persons=4 576 ==> buying=high 144 conf:(0.25)
174. persons=more 576 ==> buying=high 144 conf:(0.25)
175. lug_boot=small 576 ==> buying=high 144 conf:(0.25)
176. lug_boot=med 576 ==> buying=high 144 conf:(0.25)
177. lug_boot=big 576 ==> buying=high 144 conf:(0.25)
178. safety=low 576 ==> buying=high 144 conf:(0.25)
179. safety=med 576 ==> buying=high 144 conf:(0.25)
180. safety=high 576 ==> buying=high 144 conf:(0.25)
181. persons=2 576 ==> buying=med 144 conf:(0.25)
182. persons=4 576 ==> buying=med 144 conf:(0.25)
183. persons=more 576 ==> buying=med 144 conf:(0.25)
184. lug_boot=small 576 ==> buying=med 144 conf:(0.25)
185. lug_boot=med 576 ==> buying=med 144 conf:(0.25)
186. lug_boot=big 576 ==> buying=med 144 conf:(0.25)
187. safety=low 576 ==> buying=med 144 conf:(0.25)
188. safety=med 576 ==> buying=med 144 conf:(0.25)
189. safety=high 576 ==> buying=med 144 conf:(0.25)
190. persons=2 576 ==> buying=low 144 conf:(0.25)
191. persons=4 576 ==> buying=low 144 conf:(0.25)
192. persons=more 576 ==> buying=low 144 conf:(0.25)
193. lug_boot=small 576 ==> buying=low 144 conf:(0.25)
194. lug_boot=med 576 ==> buying=low 144 conf:(0.25)
195. lug_boot=big 576 ==> buying=low 144 conf:(0.25)
196. safety=low 576 ==> buying=low 144 conf:(0.25)
197. safety=med 576 ==> buying=low 144 conf:(0.25)
198. safety=high 576 ==> buying=low 144 conf:(0.25)
199. persons=2 576 ==> maint=vhigh 144 conf:(0.25)
200. persons=4 576 ==> maint=vhigh 144 conf:(0.25)
201. persons=more 576 ==> maint=vhigh 144 conf:(0.25)
202. lug_boot=small 576 ==> maint=vhigh 144 conf:(0.25)
203. lug_boot=med 576 ==> maint=vhigh 144 conf:(0.25)
204. lug_boot=big 576 ==> maint=vhigh 144 conf:(0.25)
205. safety=low 576 ==> maint=vhigh 144 conf:(0.25)
206. safety=med 576 ==> maint=vhigh 144 conf:(0.25)
207. safety=high 576 ==> maint=vhigh 144 conf:(0.25)
208. persons=2 576 ==> maint=high 144 conf:(0.25)
209. persons=4 576 ==> maint=high 144 conf:(0.25)
210. persons=more 576 ==> maint=high 144 conf:(0.25)
211. lug_boot=small 576 ==> maint=high 144 conf:(0.25)
212. lug_boot=med 576 ==> maint=high 144 conf:(0.25)
213. lug_boot=big 576 ==> maint=high 144 conf:(0.25)
214. safety=low 576 ==> maint=high 144 conf:(0.25)
215. safety=med 576 ==> maint=high 144 conf:(0.25)
216. safety=high 576 ==> maint=high 144 conf:(0.25)
217. persons=2 576 ==> maint=med 144 conf:(0.25)
218. persons=4 576 ==> maint=med 144 conf:(0.25)
219. persons=more 576 ==> maint=med 144 conf:(0.25)
220. lug_boot=small 576 ==> maint=med 144 conf:(0.25)
221. lug_boot=med 576 ==> maint=med 144 conf:(0.25)
222. lug_boot=big 576 ==> maint=med 144 conf:(0.25)

223. safety=low 576 ==> maint=med 144 conf:(0.25)
224. safety=med 576 ==> maint=med 144 conf:(0.25)
225. safety=high 576 ==> maint=med 144 conf:(0.25)
226. persons=2 576 ==> maint=low 144 conf:(0.25)
227. persons=4 576 ==> maint=low 144 conf:(0.25)
228. persons=more 576 ==> maint=low 144 conf:(0.25)
229. lug_boot=small 576 ==> maint=low 144 conf:(0.25)
230. lug_boot=med 576 ==> maint=low 144 conf:(0.25)
231. lug_boot=big 576 ==> maint=low 144 conf:(0.25)
232. safety=low 576 ==> maint=low 144 conf:(0.25)
233. safety=med 576 ==> maint=low 144 conf:(0.25)
234. safety=high 576 ==> maint=low 144 conf:(0.25)
235. persons=2 576 ==> doors=2 144 conf:(0.25)
236. persons=4 576 ==> doors=2 144 conf:(0.25)
237. persons=more 576 ==> doors=2 144 conf:(0.25)
238. lug_boot=small 576 ==> doors=2 144 conf:(0.25)
239. lug_boot=med 576 ==> doors=2 144 conf:(0.25)
240. lug_boot=big 576 ==> doors=2 144 conf:(0.25)
241. safety=low 576 ==> doors=2 144 conf:(0.25)
242. safety=med 576 ==> doors=2 144 conf:(0.25)
243. safety=high 576 ==> doors=2 144 conf:(0.25)
244. persons=2 576 ==> doors=3 144 conf:(0.25)
245. persons=4 576 ==> doors=3 144 conf:(0.25)
246. persons=more 576 ==> doors=3 144 conf:(0.25)
247. lug_boot=small 576 ==> doors=3 144 conf:(0.25)
248. lug_boot=med 576 ==> doors=3 144 conf:(0.25)
249. lug_boot=big 576 ==> doors=3 144 conf:(0.25)
250. safety=low 576 ==> doors=3 144 conf:(0.25)
251. safety=med 576 ==> doors=3 144 conf:(0.25)
252. safety=high 576 ==> doors=3 144 conf:(0.25)
253. persons=2 576 ==> doors=4 144 conf:(0.25)
254. persons=4 576 ==> doors=4 144 conf:(0.25)
255. persons=more 576 ==> doors=4 144 conf:(0.25)
256. lug_boot=small 576 ==> doors=4 144 conf:(0.25)
257. lug_boot=med 576 ==> doors=4 144 conf:(0.25)
258. lug_boot=big 576 ==> doors=4 144 conf:(0.25)
259. safety=low 576 ==> doors=4 144 conf:(0.25)
260. safety=med 576 ==> doors=4 144 conf:(0.25)
261. safety=high 576 ==> doors=4 144 conf:(0.25)
262. persons=2 576 ==> doors=5more 144 conf:(0.25)
263. persons=4 576 ==> doors=5more 144 conf:(0.25)
264. persons=more 576 ==> doors=5more 144 conf:(0.25)
265. lug_boot=small 576 ==> doors=5more 144 conf:(0.25)
266. lug_boot=med 576 ==> doors=5more 144 conf:(0.25)
267. lug_boot=big 576 ==> doors=5more 144 conf:(0.25)
268. safety=low 576 ==> doors=5more 144 conf:(0.25)
269. safety=med 576 ==> doors=5more 144 conf:(0.25)
270. safety=high 576 ==> doors=5more 144 conf:(0.25)
271. maint=vhigh 432 ==> buying=vhigh 108 conf:(0.25)
272. buying=vhigh 432 ==> maint=vhigh 108 conf:(0.25)
273. maint=high 432 ==> buying=vhigh 108 conf:(0.25)
274. buying=vhigh 432 ==> maint=high 108 conf:(0.25)
275. maint=med 432 ==> buying=vhigh 108 conf:(0.25)
276. buying=vhigh 432 ==> maint=med 108 conf:(0.25)
277. maint=low 432 ==> buying=vhigh 108 conf:(0.25)
278. buying=vhigh 432 ==> maint=low 108 conf:(0.25)

279. doors=2 432 ==> buying=vhigh 108 conf:(0.25)
280. buying=vhigh 432 ==> doors=2 108 conf:(0.25)
281. doors=3 432 ==> buying=vhigh 108 conf:(0.25)
282. buying=vhigh 432 ==> doors=3 108 conf:(0.25)
283. doors=4 432 ==> buying=vhigh 108 conf:(0.25)
284. buying=vhigh 432 ==> doors=4 108 conf:(0.25)
285. doors=5more 432 ==> buying=vhigh 108 conf:(0.25)
286. buying=vhigh 432 ==> doors=5more 108 conf:(0.25)
287. maint=vhigh 432 ==> buying=high 108 conf:(0.25)
288. buying=high 432 ==> maint=vhigh 108 conf:(0.25)
289. maint=high 432 ==> buying=high 108 conf:(0.25)
290. buying=high 432 ==> maint=high 108 conf:(0.25)
291. maint=med 432 ==> buying=high 108 conf:(0.25)
292. buying=high 432 ==> maint=med 108 conf:(0.25)
293. maint=low 432 ==> buying=high 108 conf:(0.25)
294. buying=high 432 ==> maint=low 108 conf:(0.25)
295. doors=2 432 ==> buying=high 108 conf:(0.25)
296. buying=high 432 ==> doors=2 108 conf:(0.25)
297. doors=3 432 ==> buying=high 108 conf:(0.25)
298. buying=high 432 ==> doors=3 108 conf:(0.25)
299. doors=4 432 ==> buying=high 108 conf:(0.25)
300. buying=high 432 ==> doors=4 108 conf:(0.25)
301. doors=5more 432 ==> buying=high 108 conf:(0.25)
302. buying=high 432 ==> doors=5more 108 conf:(0.25)
303. maint=vhigh 432 ==> buying=med 108 conf:(0.25)
304. buying=med 432 ==> maint=vhigh 108 conf:(0.25)
305. maint=high 432 ==> buying=med 108 conf:(0.25)
306. buying=med 432 ==> maint=high 108 conf:(0.25)
307. maint=med 432 ==> buying=med 108 conf:(0.25)
308. buying=med 432 ==> maint=med 108 conf:(0.25)
309. maint=low 432 ==> buying=med 108 conf:(0.25)
310. buying=med 432 ==> maint=low 108 conf:(0.25)
311. doors=2 432 ==> buying=med 108 conf:(0.25)
312. buying=med 432 ==> doors=2 108 conf:(0.25)
313. doors=3 432 ==> buying=med 108 conf:(0.25)
314. buying=med 432 ==> doors=3 108 conf:(0.25)
315. doors=4 432 ==> buying=med 108 conf:(0.25)
316. buying=med 432 ==> doors=4 108 conf:(0.25)
317. doors=5more 432 ==> buying=med 108 conf:(0.25)
318. buying=med 432 ==> doors=5more 108 conf:(0.25)
319. maint=vhigh 432 ==> buying=low 108 conf:(0.25)
320. buying=low 432 ==> maint=vhigh 108 conf:(0.25)
321. maint=high 432 ==> buying=low 108 conf:(0.25)
322. buying=low 432 ==> maint=high 108 conf:(0.25)
323. maint=med 432 ==> buying=low 108 conf:(0.25)
324. buying=low 432 ==> maint=med 108 conf:(0.25)
325. maint=low 432 ==> buying=low 108 conf:(0.25)
326. buying=low 432 ==> maint=low 108 conf:(0.25)
327. doors=2 432 ==> buying=low 108 conf:(0.25)
328. buying=low 432 ==> doors=2 108 conf:(0.25)
329. doors=3 432 ==> buying=low 108 conf:(0.25)
330. buying=low 432 ==> doors=3 108 conf:(0.25)
331. doors=4 432 ==> buying=low 108 conf:(0.25)
332. buying=low 432 ==> doors=4 108 conf:(0.25)
333. doors=5more 432 ==> buying=low 108 conf:(0.25)
334. buying=low 432 ==> doors=5more 108 conf:(0.25)

335. doors=2 432 ==> maint=vhigh 108 conf:(0.25)
336. maint=vhigh 432 ==> doors=2 108 conf:(0.25)
337. doors=3 432 ==> maint=vhigh 108 conf:(0.25)
338. maint=vhigh 432 ==> doors=3 108 conf:(0.25)
339. doors=4 432 ==> maint=vhigh 108 conf:(0.25)
340. maint=vhigh 432 ==> doors=4 108 conf:(0.25)
341. doors=5more 432 ==> maint=vhigh 108 conf:(0.25)
342. maint=vhigh 432 ==> doors=5more 108 conf:(0.25)
343. doors=2 432 ==> maint=high 108 conf:(0.25)
344. maint=high 432 ==> doors=2 108 conf:(0.25)
345. doors=3 432 ==> maint=high 108 conf:(0.25)
346. maint=high 432 ==> doors=3 108 conf:(0.25)
347. doors=4 432 ==> maint=high 108 conf:(0.25)
348. maint=high 432 ==> doors=4 108 conf:(0.25)
349. doors=5more 432 ==> maint=high 108 conf:(0.25)
350. maint=high 432 ==> doors=5more 108 conf:(0.25)
351. doors=2 432 ==> maint=med 108 conf:(0.25)
352. maint=med 432 ==> doors=2 108 conf:(0.25)
353. doors=3 432 ==> maint=med 108 conf:(0.25)
354. maint=med 432 ==> doors=3 108 conf:(0.25)
355. doors=4 432 ==> maint=med 108 conf:(0.25)
356. maint=med 432 ==> doors=4 108 conf:(0.25)
357. doors=5more 432 ==> maint=med 108 conf:(0.25)
358. maint=med 432 ==> doors=5more 108 conf:(0.25)
359. doors=2 432 ==> maint=low 108 conf:(0.25)
360. maint=low 432 ==> doors=2 108 conf:(0.25)
361. doors=3 432 ==> maint=low 108 conf:(0.25)
362. maint=low 432 ==> doors=3 108 conf:(0.25)
363. doors=4 432 ==> maint=low 108 conf:(0.25)
364. maint=low 432 ==> doors=4 108 conf:(0.25)
365. doors=5more 432 ==> maint=low 108 conf:(0.25)
366. maint=low 432 ==> doors=5more 108 conf:(0.25)