VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY

UNIVERSITY OF INFORMATION TECHNOLOGY

ADVANCED PROGRAM IN INFORMATION SYSTEMS

TRAN HONG NGAN

DISCOVERING KNOWLEDGE IN THE UNIVERSITY ENTRANCE EXAMS

BACHELOR OF ENGINEERING IN INFORMATION SYSTEMS

HO CHI MINH CITY, 2014

VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY

UNIVERSITY OF INFORMATION TECHNOLOGY

ADVANCED PROGRAM IN INFORMATION SYSTEMS

TRAN HONG NGAN – 10520494

DISCOVERING KNOWLEDGE IN THE UNIVERSITY ENTRANCE EXAMS

BACHELOR OF ENGINEERING IN INFORMATION SYSTEMS

THESIS ADVISOR

ASSOC. PROF. DO PHUC

HO CHI MINH CITY, 2014

ASSESSMENT ADVISOR

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

…………………………………………………………………………………..

ASSESSMENT COMMITTEE

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

……………………………………………………………………………..

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Associate Professor Do Phuc, who suggests, guides and encourages me during the time I implement this thesis.

I would like to express my thankful to Vietnam National University – Ho Chi Minh City (VNU - HCM), who gave me the data, the collective results of the University Entrance Exams from year 2009 to year 2011, to do this thesis.

I also appreciate Tran Quang Hoa, an elder brother from class Advanced Education Program 2009, for his advices and assistance in writing this thesis.

I would like to express my thankful to the staffs in Faculty of Information Systems of University of Information Technology who emit the passion to me.

Finally, I would like to thank my family for providing me the best conditions to study in Advanced Education Program and finish this graduation thesis.

Winter 2014

Tran Hong Ngan - Student of CTTT2010.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

University Entrance Exam is a component of education environment which is one of the most important parts of our country. Exploring the relationships between components of education helps all countries to improve and develop their education system. Data mining analysis includes several techniques such as Association Rule, Classification, Clustering and so on to discover knowledge from databases. In this thesis, I would like to use association and classification techniques to discover knowledge from University Entrance Exams Results for three year periods in Viet Nam. My goal is finding the correlations between score, block and region, how these items affect the result of students. My thesis will provide a foundation for government and universities to make good decisions in developing education system of our country.

Chapter 1: Introduction

Chapter 1 is an overview about the topic of my thesis, the solution methodologies and the structure of my thesis.

1.1 Overview

Nowadays, data mining analysis is using more frequently because scientists are changing the research direction to discovering knowledge from databases. With data mining techniques, we can get valuable rules that support for making good decisions in future.

There are many theses in Educational Research. For example, some research works of predicting the result of students (both in Viet Nam and abroad) [5] [6]. All of them depend on the data mining techniques to get the rules from existing data and support for decision making process. However, one of the most important knowledge that we have not discovered in the past is the knowledge in the University Entrance Exams.

1.2 Problem Statement

Every year, we get a large number of scores of students from the University Entrance Exams. Each result reflects different meanings, we plan to use data mining to the University Entrance Exams to get the valuable rules. Some students can get high total score (addition score of three subjects) or others cannot get the average score (addition score of three subjects is less than 15 per 30 scale). Therefore, one of the most important problem is *"Whether existing any relationships among the regions where the students live, the blocks such as A, B, C, D1 and so on that the students choose and the exam results or not"*. In other word, we find the rules in the University Entrance Exams that help us to predict the result of students.

1.3 Solution Methodology

Association rules have many applications in our life such as credit card analysis, promotion analysis, power usage analysis and so on. Apriori algorithm belongs to association rules to find the association and correlation between several components. In addition, PART algorithm belongs to classification technique which is also a kind of data mining technique in discovering knowledge.

I use Apriori algorithm and PART algorithm to find the valuable relationships between regions, blocks and scores of students in University Entrance Exams from 2009 to 2011.

1.4 Structure of Thesis

The thesis is organized into five chapters as follows:

Chapter 1: Introduction

In this chapter, I give a short introduction of my problem and methodology to solve this problem.

Chapter 2: Fundamental Theories

In this chapter, I introduce about the basic theories that I use in this thesis, for examples: Association Rules basic concepts and then Apriori algorithm, Classification basic concepts and then PART algorithm.

Chapter 3: University Entrance Exams Database

In this chapter, I introduce about the collected database that I use for this thesis. Furthermore, I explain which properties are chosen for testing.

Chapter 4: System Implementation and Testing

In this chapter, I show my implementation and explain the results that I get from my demo application.

Chapter 5: Conclusions and Future Work

I summarize my work and suggest some branches for future work.

Chapter 2: Fundamental Theories

In chapter 2, the first content is an overview about data mining and then I will present about Apriori algorithm and PART algorithm because I will use them in my thesis.

2.1 Data Mining

2.1.1 Definition

The term data mining does not have an exactly definition, there are several definitions. Data mining, another name known as Knowledge Discovery in Databases (KDD), is the search or using a variety of techniques to find the relationships, patterns or useful information that are hidden in large databases.

Two primary goals of data mining in practice are prediction and description. Prediction is predicting the unknown values in the future from the past or current data. Description is finding the patterns of existing databases.

2.1.2 Data Mining Process

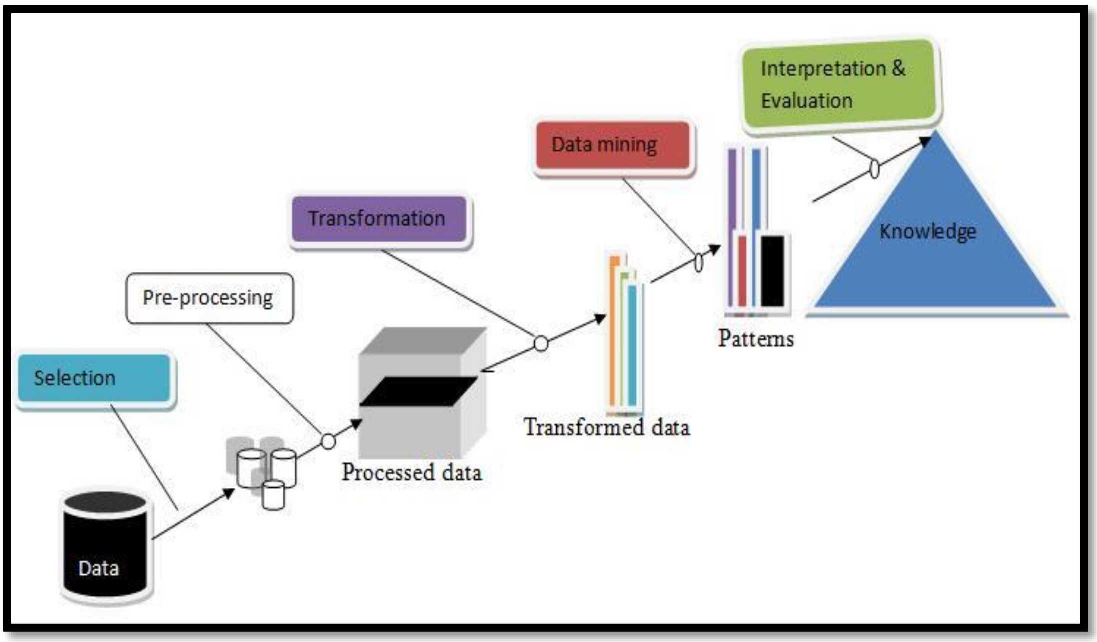


Figure 2.1. Stages/Process Identified in Data Mining

Data mining process is described in the Figure 2.1.

- *Selection*: selecting or choosing the attributes of data that we should use, this is the stage to create a subset of original database.
- *Preprocessing*: this is the stage to clean the unnecessary data, for example missing data, or format the data follow to the consistent format. We must have this stage because the original database can get from several sources with various formats or these unnecessary data can affect the data mining results.
- *Transformation*: the data is transferred to match the technique, the transformed data is usable for the main purpose.
- *Data mining*: this stage is finding the correlations, patterns which are hidden in the data. For example, correlations in supermarket transactions. When people buy A, they can also buy B, or a person has relationship with a lot of scientists so he/she can be a scientist.
- *Interpretation and evaluation*: the correlations will support human in making decisions such as predicting status, classifying objects or changing decisions.

2.1.3 Data Mining Methods

Some data mining tasks are listed as follows:

- *Classification* is learning a rule that helps to find predefined classes for a data item. Figure 2.2. shows an example of classification for the Loan Data Set.

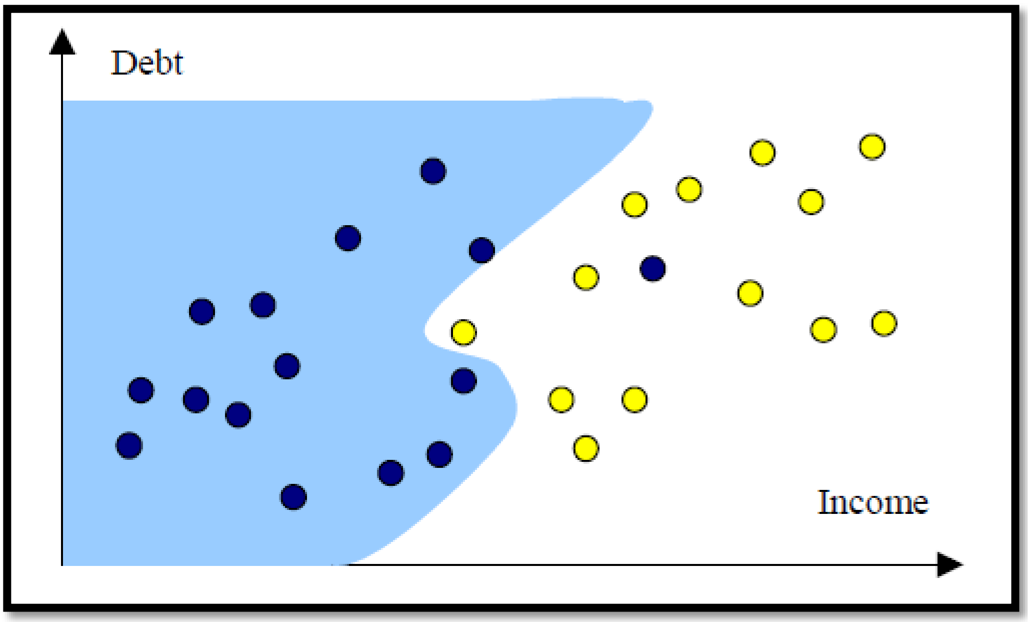


Figure 2.2. Classification Boundaries Learned by a Non-Linear Classifier for the Loan Data Set

- *Regression* is learning a function that a data item is mapped to a real-valued predictor variable. Figure 2.3. shows an example of regression for the Loan Data Set.

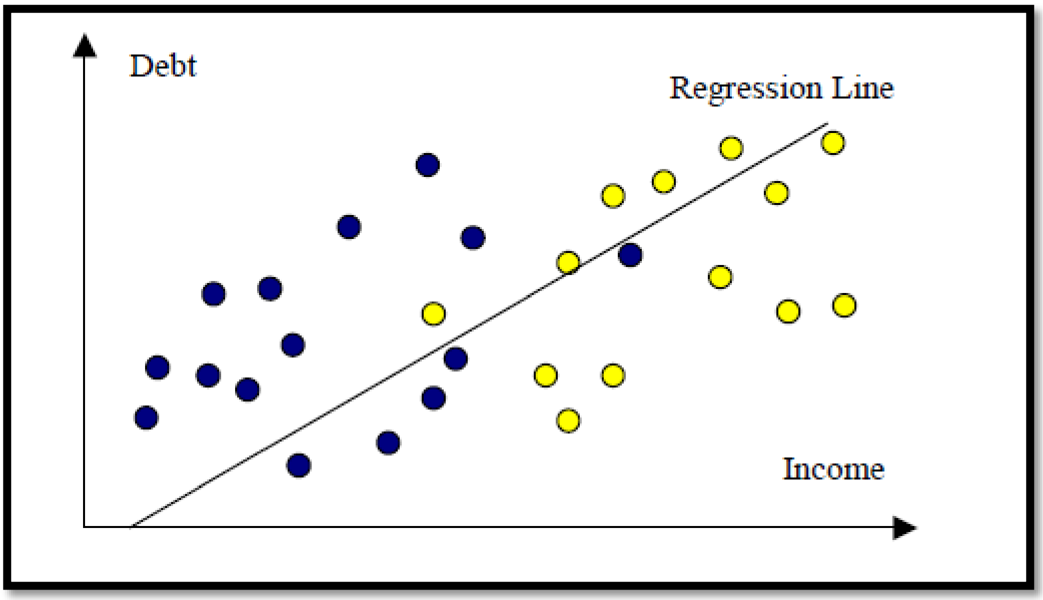


Figure 2.3. A Simple Linear Regression for the Loan Data Set

- *Clustering* is identifying the categories or clusters that exist in data.

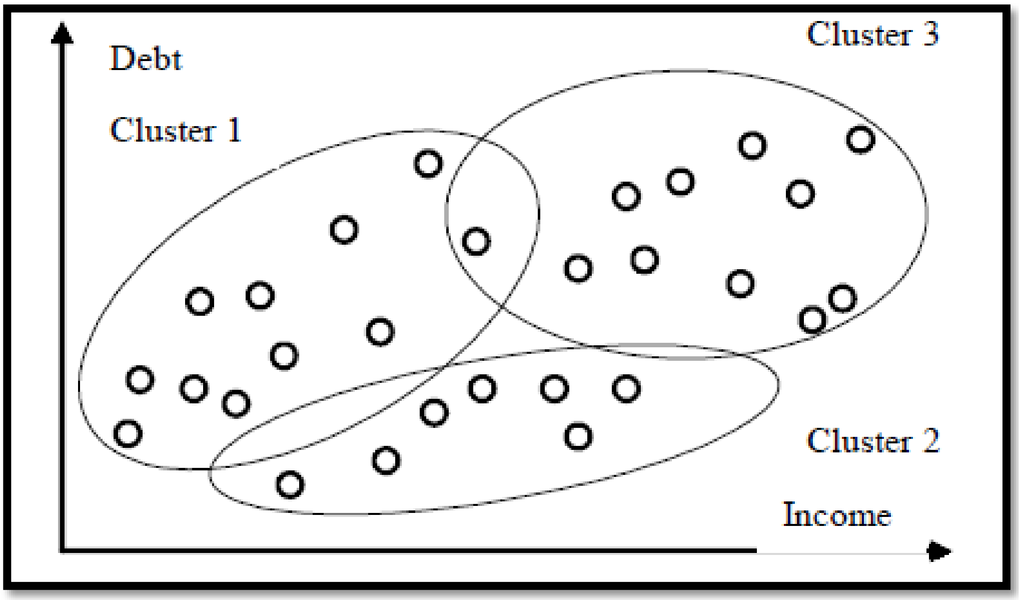


Figure 2.4. A Simple Clustering of the Loan Data Set into Three Clusters

- *Association Rules* express the correlations between attributes. Figure 2.5. shows an example of Association Rules, a market analyst finds all relationships between items that customers will buy together.

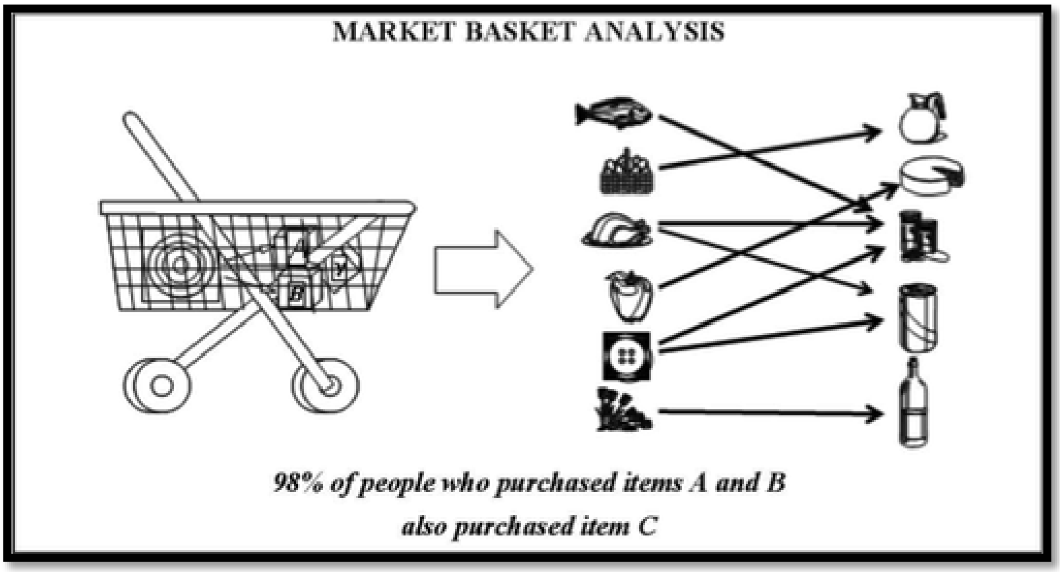


Figure 2.5. The famous example for Association Rules

2.2 Association Rules

In a large traditional or relational data sets, how can we find the correlations among items in them? This is the reason why association rules existing.

2.2.1 Basic Concepts

Let I = {I_1, I_2, ..., I_m} be a set of items. A set of items is also called an itemset and a set of k items is a k-itemset [4]. Let D be a set of database transactions. A transaction T, belong to D, is also a set of items such that $T \subseteq I$. A transaction T is said to contain A, is a set of items, if and only if $A \subseteq T$.

The form of an association rule is $A \Rightarrow B$, where $A \subset I$, $B \subset I$ and $A \cap B = \Phi$. The rule $A \Rightarrow B$ holds in the transaction set D with support s and has confidence c. Support s is the percentage of transactions in D that contains $A \cup B$ and the confidence c is the percentage of transactions in D containing A which also containing B in transaction set D. That is [3],

$$support\ (A\ \Rightarrow B) = P\ (A\ \cup B)$$

$$confidence\ (A\ \Rightarrow B) = P\ (B \mid A) = \frac{support\ (A\ \cup B)}{support\ (A)}$$

$$= \frac{support\_count\ (A\ \cup B)}{support\_count\ (A)}$$

The goal of association rules is finding all rules that satisfy a user-defined minimum support and a user-defined minimum confidence.

Two major steps of association rule generation [7] are:

- *Frequent Itemset Generation*: the purpose of this step is finding all itemsets, are called frequent itemsets, satisfy the minimum support.
- *Rule Generation*: the purpose of this step is finding all rules, from these frequent itemsets, satisfy the minimum confidence.

2.2.2 Apriori Algorithm

2.2.2.1 Frequent Itemset Generation

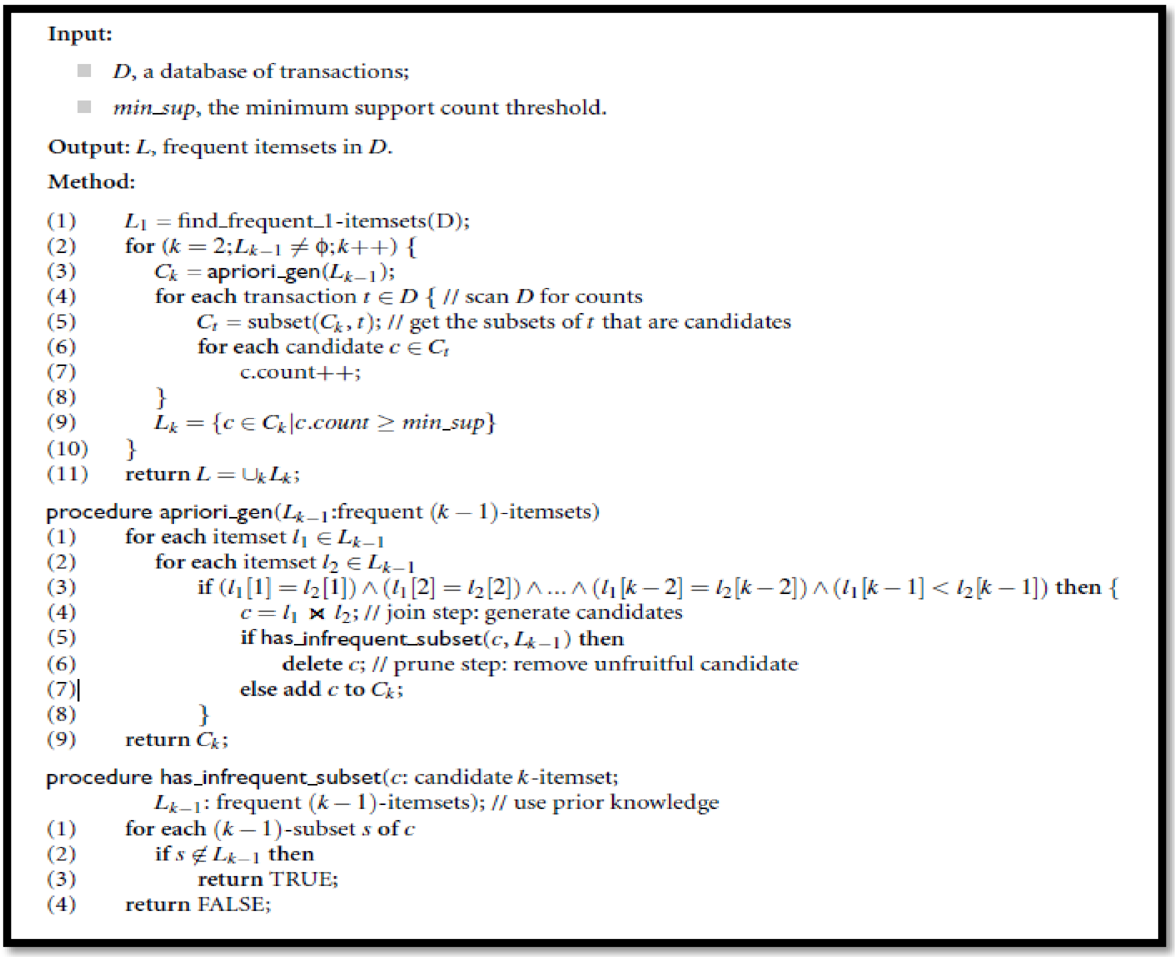
In 1994, R. Agrawal and R. Srikant proposed Apriori algorithm for mining frequent itemsets for Boolean association rule. Apriori algorithm uses candidate generation in finding frequent itemsets [3].

This algorithm starts with finding the set of frequent 1-itemsets, is called L1, by scanning all database and getting all items that satisfy the minimum support. Then, I use L1 to find the set of frequent 2-itemsets L2 by scanning all database again and continuing until cannot find Lk.

This algorithm is using Lk to find L(k+1) so it is known level-wise search, a form of iterative search. Furthermore, we use the Apriori property *"All non-empty subsets of a frequent itemset must also be frequent"* to make the algorithm to be more effective because we do not need to scan all database in each step.

The Apriori algorithm for discovering frequent itemsets is described in Figure 2.6. Especially, two smaller steps in finding frequent itemsets are join step and prune step:

- *The join step*: the purpose of join step is joining itemsets to find the next itemsets that means finding itemsets L_k by joining itemset L_{k-1} with itself. And we use C_k to denote the set of candidate k-itemsets. Let l_1 and l_2 be itemsets in L_{k-1}. The notation $l_i[j]$ refers to the jth item in l_i (e.g., $l_1[k - 2]$ refers to the second to the last item in l_1). Assumption in Apriori is itemsets are in lexicographic order, for example with the $(k - 1)$-itemset, l_i, this means that the items are in order $l_i[1] < l_i[2] < \ldots < l_i[k - 1]$. The join, $L_{k-1} \bowtie L_{k-1}$, is performed, where members of L_{k-1} are joinable if their first $(k - 2)$ items are in common. That is, members l_1 and l_2 of L_{k-1} are joined if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge \ldots \wedge (l_1[k - 2] = l_2[k - 2]) \wedge (l_1[k - 1] < l_2[k - 1])$. The condition $l_1[k - 1] < l_2[k - 1]$ simply ensures that no duplicates are generated. The resulting itemset formed by joining l_1 and l_2 is $l_1[1], l_1[2], \ldots , l_1[k - 2], l_1[k - 1], l_2[k - 1]$. [3]



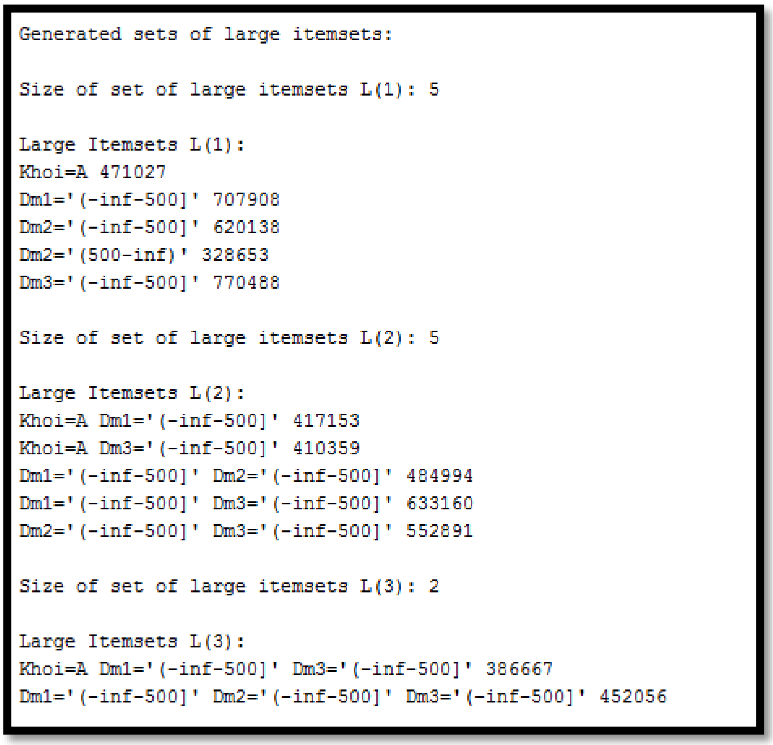
```
Input:
    ▪ $D$, a database of transactions;
    ▪ $min\_sup$, the minimum support count threshold.
Output: $L$, frequent itemsets in $D$.
Method:
    $L_1$ = find_frequent_1-itemsets(D);
    for ($k = 2; L_{k-1} \neq \phi; k{+}{+}$) {
        $C_k$ = apriori_gen($L_{k-1}$);
        for each transaction $t \in D$ { // scan $D$ for counts
            $C_t$ = subset($C_k, t$); // get the subsets of $t$ that are candidates
            for each candidate $c \in C_t$
                c.count++;
        }
        $L_k = \{c \in C_k | c.count \geq min\_sup\}$
    }
    return $L = \cup_k L_k$;
procedure apriori_gen($L_{k-1}$:frequent $(k-1)$-itemsets)
    for each itemset $l_1 \in L_{k-1}$
        for each itemset $l_2 \in L_{k-1}$
            if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge ... \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$ then {
                $c = l_1 \bowtie l_2$; // join step: generate candidates
                if has_infrequent_subset($c, L_{k-1}$) then
                    delete $c$; // prune step: remove unfruitful candidate
                else add $c$ to $C_k$;
            }
    return $C_k$;
procedure has_infrequent_subset($c$: candidate $k$-itemset;
            $L_{k-1}$: frequent $(k-1)$-itemsets); // use prior knowledge
    for each $(k-1)$-subset $s$ of $c$
        if $s \notin L_{k-1}$ then
            return TRUE;
    return FALSE;
```

Figure 2.6. The Apriori algorithm for discovering frequent itemsets [3]

- *The prune step*: the purpose of prune step is removing or deleting unsatisfied itemsets because not all candidates in C_k satisfy the requirement. Scanning database is necessary in this step to determine which candidate is accepted or not. We should use the Apriori property that was described above to reduce the complexity for this step. According to this property, if any (k - 1)-subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k [3].

An example of frequent itemsets that is created by Apriori algorithm is showed in Figure 2.7. Let minsup be equal to 0.3, from the University

Entrance Exams database in year 2010, we can get the large itemsets L(1), large itemsets L(2) and large itemsets L(3). For example, L(1) has five frequent itemsets “Khoi = A”, “Dm1 = ‘(-inf-500]’”, “Dm2 = ‘(-inf-500]’”, “Dm2 = ‘(500-inf)’”, “Dm3 = ‘(-inf-500]’”. The value is next to each itemset is the number of this occurrence in all the set.



```
Generated sets of large itemsets:

Size of set of large itemsets L(1): 5

Large Itemsets L(1):
Khoi=A 471027
Dm1='(-inf-500]' 707908
Dm2='(-inf-500]' 620138
Dm2='(500-inf)' 328653
Dm3='(-inf-500]' 770488

Size of set of large itemsets L(2): 5

Large Itemsets L(2):
Khoi=A Dm1='(-inf-500]' 417153
Khoi=A Dm3='(-inf-500]' 410359
Dm1='(-inf-500]' Dm2='(-inf-500]' 484994
Dm1='(-inf-500]' Dm3='(-inf-500]' 633160
Dm2='(-inf-500]' Dm3='(-inf-500]' 552891

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

Large Itemsets L(3):
Khoi=A Dm1='(-inf-500]' Dm3='(-inf-500]' 386667
Dm1='(-inf-500]' Dm2='(-inf-500]' Dm3='(-inf-500]' 452056
```

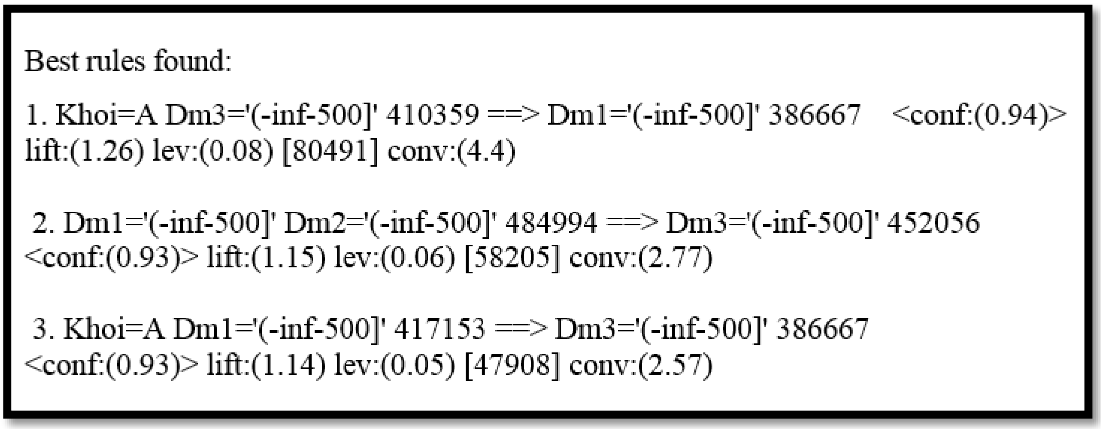
Figure 2.7. Frequent itemsets example (data in year 2010)

2.2.2.2 Rule Generation

Confidence equation was showed in item 2.2.1. Then, we use it to generate the association rules:

- Non-empty subsets will be generated for each frequent itemset *l*.
- For each non-empty subset *s* of *l*, get the rule "$s \Rightarrow (l - s)$" if $\frac{support\_count\ (l)}{support\_count\ (s)} \geq min\_conf$, where min_conf is the minimum confidence threshold.

After finding all frequent itemsets from the data in year 2010 (Figure 2.7), I get all association rules from this database that satisfy minimum confidence which is assigned by 0.9. There are three rules are found; the confidence number is next to these rules, all of them are greater than 0.9.



```
Best rules found:
1. Khoi=A Dm3='(-inf-500]' 410359 ==> Dm1='(-inf-500]' 386667    <conf:(0.94)>
lift:(1.26) lev:(0.08) [80491] conv:(4.4)

 2. Dm1='(-inf-500]' Dm2='(-inf-500]' 484994 ==> Dm3='(-inf-500]' 452056
<conf:(0.93)> lift:(1.15) lev:(0.06) [58205] conv:(2.77)

 3. Khoi=A Dm1='(-inf-500]' 417153 ==> Dm3='(-inf-500]' 386667
<conf:(0.93)> lift:(1.14) lev:(0.05) [47908] conv:(2.57)
```

Figure 2.8. Rule generation example (data in year 2009)

2.3 Classification

Classification is another way to find the hidden information from large database and help decision makers in predicting the future. Specifically, The Ministry of Education can know level of exams, the base scores and so on based on the Classifying of subjects, regions.

2.3.1 Basic Concepts

There are two steps in classification, the *learning step* and the *classification step*.

- *Learning*: Training data are analyzed by a classification algorithm.
- *Classification*: Classification rules can be used to classify new data if its accuracy is accepted. Accuracy of these rules which is found by test data is the percentage of correctly test set tuples by the classifier.

Depending on the status of the class label, we have *supervised learning* and *unsupervised learning*. If class label of each training tuple is provided, it is supervised learning. In vice versa, it is unsupervised learning.

2.3.2 PART Algorithm

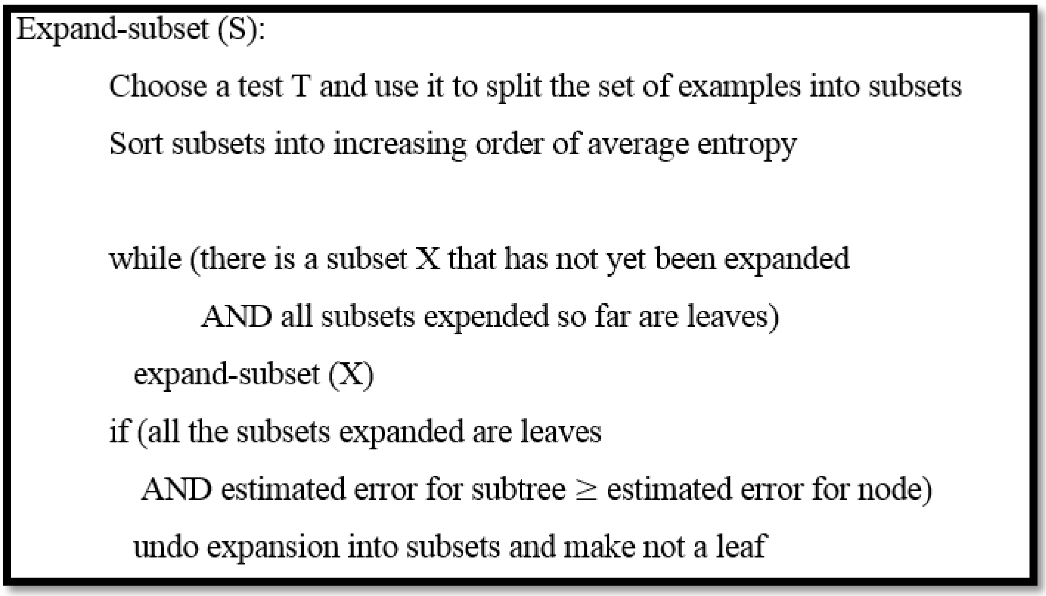
PART is a partial decision tree algorithm. PART is a developed version of C4.5 and RIPPER algorithms [2].

PART algorithm does not use global optimization, it still have to ensure the accuracy of rules so it combines two strategies, the divide-and-conquer and separate-and-conquer, to guarantee that. Separate-and-conquer is used in building rules recursively, starting with building a rule and remove this instance, then continuing with remaining instances. In principle, a pruned decision tree is built for the current set of instances when making a single rule; the leaf with the largest coverage is made into a rule then the tree is discarded [2].

A characteristic problem of the separate-and-conquer learner is overprune so we use a pruned tree to get rules to avoid this problem in this algorithm. The combination of decision trees and separate-and-conquer helps the algorithm to be more flexible and faster. It is so wasteful in building full decision tree just to get a rule but it can be more effective in another situation with its advantages.

Therefore, in PART algorithm, we will build a partial decision tree instead of building a full decision tree. A needed partial decision tree is a decision tree that includes branches to undefined subtrees. A stable subtree is created by combining of construction and pruning. When this subtree is found, tree building stops then a single rule is created.

The tree-building algorithm is summarized in Figure 2.9. A partial tree is created from a set of instances. Information-gain is the necessary property in building decision tree. Choosing a test and dividing the instances into subsets is the starting step. The next step is sorting subsets into increasing average entropy. This is explained by a subset with low average entropy is referred to be a small subtree then it can produce more general rule. Processing this step recursively until a subset is expanded into a leaf then that continues further by backtracking. After that, the subtree replacement operation of decision tree pruning is used to check the node which is replaced by a single leaf or not. A node is replaced by exploring siblings if the replacement is performed this algorithm backtracks. Nevertheless, the subtrees are left undefined if a node meets all other children which are not leaves [2]. The structure of this algorithm is recursive so this step finish automatically tree generation.



```
Expand-subset (S):
        Choose a test T and use it to split the set of examples into subsets
        Sort subsets into increasing order of average entropy

        while (there is a subset X that has not yet been expanded
                AND all subsets expended so far are leaves)
          expand-subset (X)
        if (all the subsets expanded are leaves
           AND estimated error for subtree $\geq$ estimated error for node)
          undo expansion into subsets and make not a leaf
```

Figure 2.9. Tree-building algorithm

The main advantage of this algorithm is simplicity because it achieves the same performance with other methods without global optimization. Furthermore, it is also effective method that applies decision tree.

An example of a partial decision tree is described in the Figure 2.10. In the top of this figure is Score3 and it has two subsets, interval (-inf-500] and interval (500-inf). Subset interval (-inf-500] finishes while another has Block subset. The process continues until finish all instances of database in year 2010. An example rule I get from this tree is “IF Score3 is greater than 5 and Block is B THEN Score2 is greater than 5”.

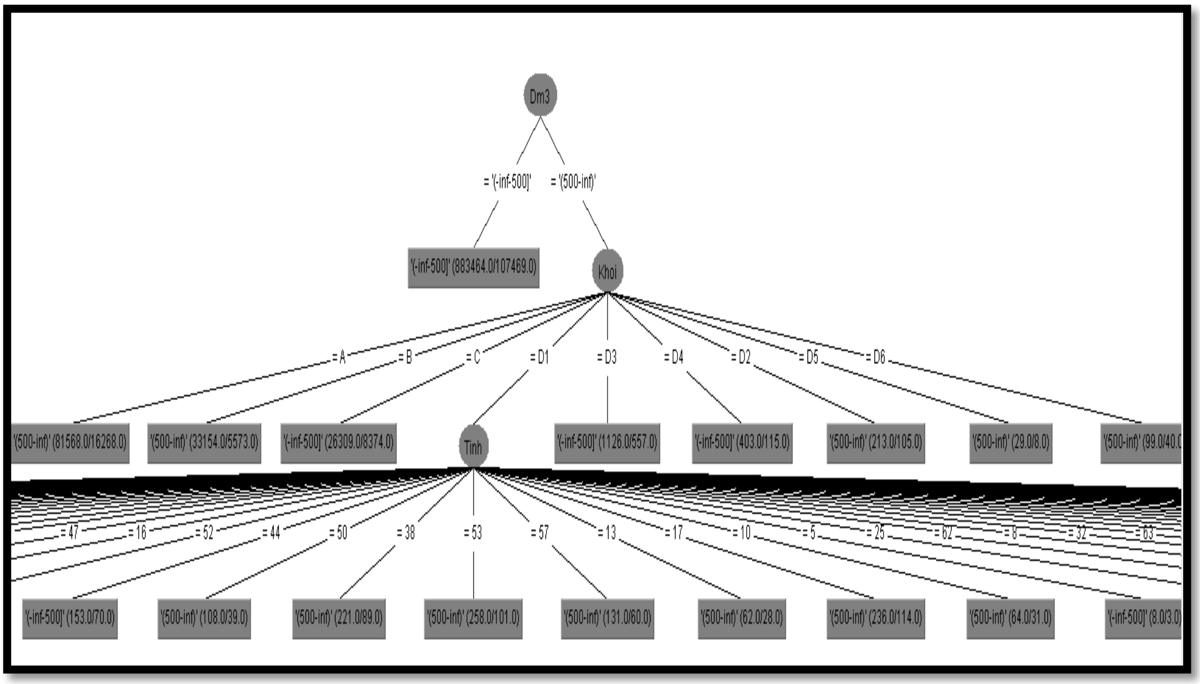


Figure 2.10. An example of decision tree for database in year 2010

In the Figure 2.11., some classification rules were discovered from the database in year 2009 by using PART algorithm. An example rule is that:

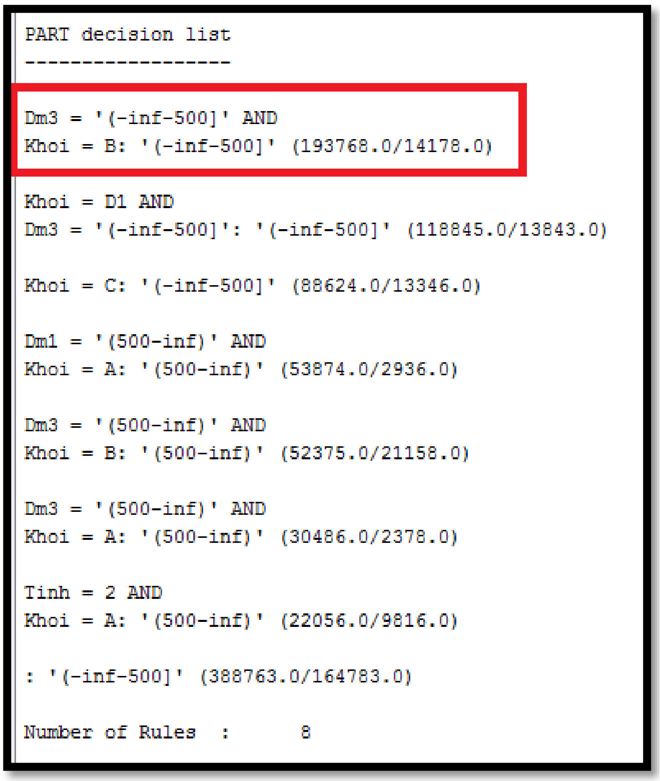
Dm3 = ‘(-inf-500]’ AND Khoi = B : ‘(-inf-500]’ (193768.0 / 14178.0)

- The meaning of this rule is:

  “IF Score3 is less than 5 and Block is B THEN Score2 is less than 5”

- The *(193768.0 / 14178.0)* property is:
  - The total number of condition is occurred (193768).
  - The total number of this rule is satisfied (14178.0).

Similarly, I can get another seven rules from database in year 2009. The last line in this figure is the total number of rules from this database.



```
PART decision list
------------------

Dm3 = '(-inf-500]' AND
Khoi = B: '(-inf-500]' (193768.0/14178.0)

Khoi = D1 AND
Dm3 = '(-inf-500]': '(-inf-500]' (118845.0/13843.0)

Khoi = C: '(-inf-500]' (88624.0/13346.0)

Dm1 = '(500-inf)' AND
Khoi = A: '(500-inf)' (53874.0/2936.0)

Dm3 = '(500-inf)' AND
Khoi = B: '(500-inf)' (52375.0/21158.0)

Dm3 = '(500-inf)' AND
Khoi = A: '(500-inf)' (30486.0/2378.0)

Tinh = 2 AND
Khoi = A: '(500-inf)' (22056.0/9816.0)

: '(-inf-500]' (388763.0/164783.0)

Number of Rules  :      8
```

Figure 2.11. Classification rules are extracted from database in year 2009

Chapter 3: University Entrance Exams Database

In this section, I will describe University Entrance Exams Database about the source, the attributes in database and so on and several reasons why I choose the attributes for this thesis. Later, I show my steps in pre-processing database before I apply it to my demo application.

3.1 Overview about University Entrance Exams Database

I got the collected results of the University Entrance Exams from Vietnam National University – Ho Chi Minh City (VNU - HCM) to do my research. This database is very large because it includes the result of three year periods (from year 2009 to year 2011). Database each year was stored in .dbf file format so I use the Microsoft Visual FoxPro 9.0 to process this database. In the original database, there are two types of exams which are University Entrance Exams and College Entrance Exams. I use the database of University Entrance Exams only because this thesis just focuses on University.

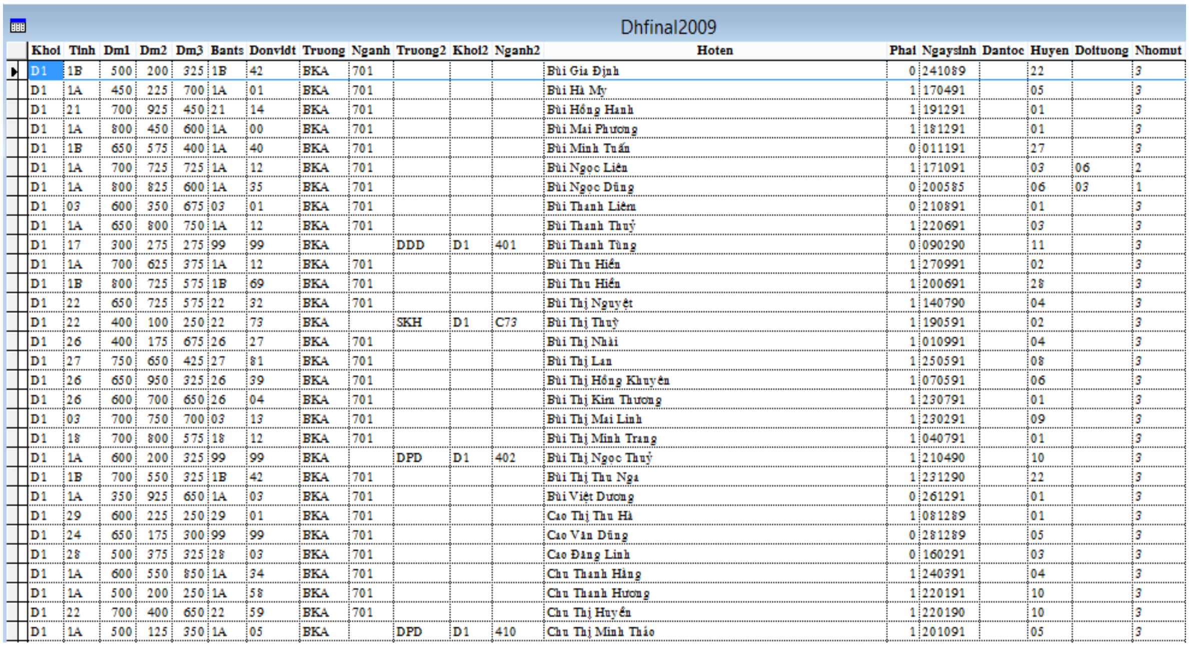
The collected result of the University Entrance Exams of each year includes many attributes, the explanation for abbreviated name are described in Table 3.1. The example of University Entrance Exam in year 2009 in .dbf file format of Visual FoxPro is shown in Figure 3.1. and Figure 3.2.

Table 3.1. Abbreviated name of database attributes

No.	Abbreviated name	Description	Comment
1	Bants	The Admissions Committee	The organized university
2	Donvidt	Candidate units	Where students submit the profile
3	Truong	University	The second selected university
4	Khoi	Block	The second selected block
5	Nganh	Branch	The second selected branch

6	Truong2	University2	The second selected university
7	Khoi2	Block2	The second selected block
8	Nganh2	Branch2	The second selected branch
9	Hoten	Full Name	Full name of student
10	Phai	Gender	Gender of student
11	Ngaysinh	Date of Birth	Date of birth of student
12	Dantoc	Ethnic group	Ethic group
13	Tinh	Province / City	Province or city where student come from
14	Huyen	District	District of province or city
15	Doituong	Object group	Object group
16	Nhomut	Priority group	Priority group
17	Namtn	Graduation year	Graduation year
18	Lop12	12th class	12th representing
19	Lop11	11th class	11th representing
20	Lop10	10th class	10th representing
21	Khuvuc	Area group	Area group
22	Sobaodanh	Registration Number	Registration number
23	D1	Draft Score1	The first subject's score in in the first grading
24	D2	Draft Score2	The second subject's score in the first grading
25	D3	Draft Score3	The third subject's score in the first grading
26	Dm1	Score1	The official first subject's score
27	Dm2	Score2	The official second subject's

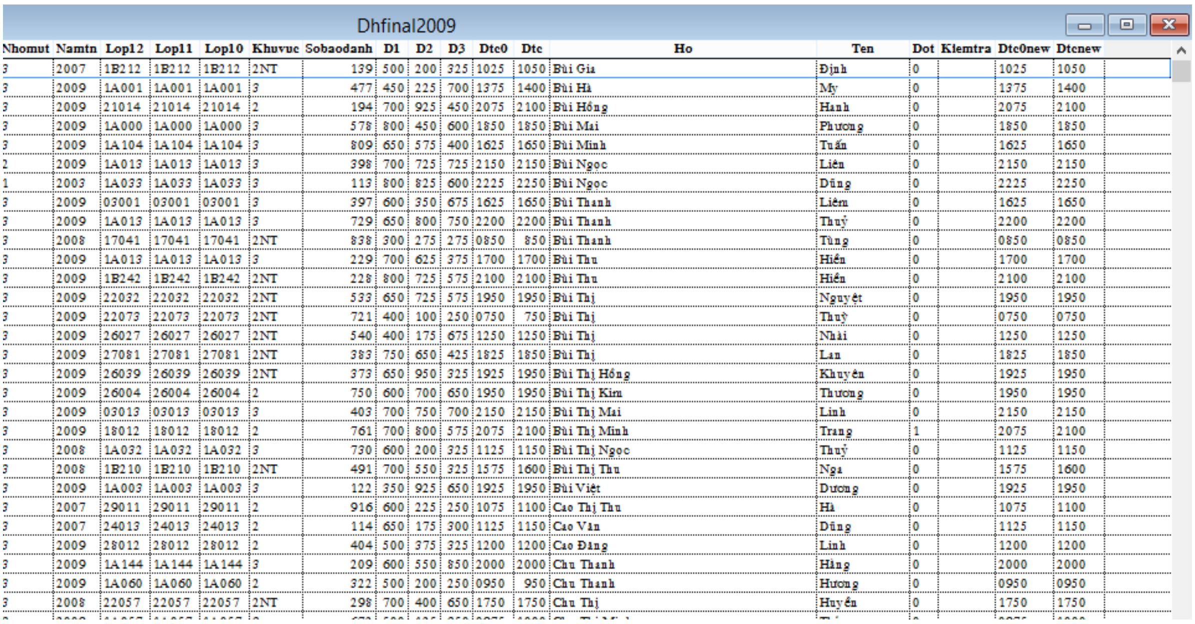
			score
28	Dm3	Score3	The official third subject's score
29	Dtc0	Draft Total Score	The total score from draft Score1, Score2, Score3
30	Dtc	Total Score	The total score after making even Dtc0
31	Ho	Lastname	The lastname
32	Ten	Firstname	The firstname
33	Dot	Phase	Phase in the exam
34	Kiemtra	Test	Testing in the exam
35	Dtc0new	Draft Total Score Final	The total score from official Score1, Score2, Score3
36	Dtcnew	Total Score Final	The official total score after making even Dtc0new



Dhfinal2009

Khoi	Tinh	Dm1	Dm2	Dm3	Bants	Donvidt	Truong	Nganh	Truong2	Khoi2	Nganh2	Hoten	Phai	Ngaysinh	Dantoc	Huyen	Doituong	Nhomut
D1	1B	500	200	325	1B	42	BKA	701				Bùi Gia Định	0	241089		22		3
D1	1A	450	225	700	1A	01	BKA	701				Bùi Hà My	1	170491		05		3
D1	21	700	925	450	21	14	BKA	701				Bùi Hồng Hạnh	1	191291		01		3
D1	1A	800	450	600	1A	00	BKA	701				Bùi Mai Phương	1	181291		01		3
D1	1B	650	575	400	1A	40	BKA	701				Bùi Minh Tuấn	0	011191		27		3
D1	1A	700	725	725	1A	12	BKA	701				Bùi Ngọc Liên	1	171091		03	06	2
D1	1A	800	825	600	1A	35	BKA	701				Bùi Ngọc Dũng	0	200585		06	03	1
D1	03	600	350	675	03	01	BKA	701				Bùi Thanh Liêm	0	210891		01		3
D1	1A	650	800	750	1A	12	BKA	701				Bùi Thanh Thuỷ	1	220691		03		3
D1	17	300	275	275	99	99	BKA		DDD	D1	401	Bùi Thanh Tùng	0	090290		11		3
D1	1A	700	625	375	1A	12	BKA	701				Bùi Thu Hiền	1	270991		02		3
D1	1B	800	725	575	1B	69	BKA	701				Bùi Thu Hiền	1	200691		28		3
D1	22	650	725	575	22	32	BKA	701				Bùi Thị Nguyệt	1	140790		04		3
D1	22	400	100	250	22	73	BKA		SKH	D1	C73	Bùi Thị Thuỳ	1	190591		02		3
D1	26	400	175	675	26	27	BKA	701				Bùi Thị Nhài	1	010991		04		3
D1	27	750	650	425	27	81	BKA	701				Bùi Thị Lan	1	250591		08		3
D1	26	650	950	325	26	39	BKA	701				Bùi Thị Hồng Khuyên	1	070591		06		3
D1	26	600	700	650	26	04	BKA	701				Bùi Thị Kim Thương	1	230791		01		3
D1	03	700	750	700	03	13	BKA	701				Bùi Thị Mai Linh	1	230291		09		3
D1	18	700	800	575	18	12	BKA	701				Bùi Thị Minh Trang	1	040791		01		3
D1	1A	600	200	325	99	99	BKA		DPD	D1	402	Bùi Thị Ngọc Thuỷ	1	210490		10		3
D1	1B	700	550	325	1B	42	BKA	701				Bùi Thị Thu Nga	1	231290		22		3
D1	1A	350	925	650	1A	03	BKA	701				Bùi Việt Dương	0	261291		01		3
D1	29	600	225	250	29	01	BKA	701				Cao Thị Thu Hà	1	081289		01		3
D1	24	650	175	300	99	99	BKA	701				Cao Văn Dũng	0	281289		05		3
D1	28	500	375	325	28	03	BKA	701				Cao Đăng Linh	0	160291		03		3
D1	1A	600	550	850	1A	34	BKA	701				Chu Thanh Hằng	1	240391		04		3
D1	1A	500	200	250	1A	58	BKA	701				Chu Thanh Hương	1	220191		10		3
D1	22	700	400	650	22	59	BKA	701				Chu Thị Huyền	1	220190		10		3
D1	1A	500	125	350	1A	05	BKA		DPD	D1	410	Chu Thị Minh Thảo	1	201091		05		3

Figure 3.1. Original database of University Entrance Exam in year 2009 (in .dbf file format of Visual FoxPro)



Dhfinal2009

Nhomut	Namtn	Lop12	Lop11	Lop10	Khuvuc	Sobaodanh	D1	D2	D3	Dtc0	Dtc	Ho	Ten	Dot	Kiemtra	Dtc0new	Dtcnew
3	2007	1B212	1B212	1B212	2NT	139	500	200	325	1025	1050	Bùi Gia	Định	0		1025	1050
3	2009	1A001	1A001	1A001	3	477	450	225	700	1375	1400	Bùi Hà	My	0		1375	1400
3	2009	21014	21014	21014	2	194	700	925	450	2075	2100	Bùi Hồng	Hạnh	0		2075	2100
3	2009	1A000	1A000	1A000	3	578	800	450	600	1850	1850	Bùi Mai	Phương	0		1850	1850
3	2009	1A104	1A104	1A104	3	809	650	575	400	1625	1650	Bùi Minh	Tuấn	0		1625	1650
2	2009	1A013	1A013	1A013	3	398	700	725	725	2150	2150	Bùi Ngọc	Liên	0		2150	2150
1	2003	1A033	1A033	1A033	3	113	800	825	600	2225	2250	Bùi Ngọc	Dũng	0		2225	2250
3	2009	03001	03001	03001	3	397	600	350	675	1625	1650	Bùi Thanh	Liêm	0		1625	1650
3	2009	1A013	1A013	1A013	3	729	650	800	750	2200	2200	Bùi Thanh	Thuỷ	0		2200	2200
3	2008	17041	17041	17041	2NT	838	300	275	275	0850	850	Bùi Thanh	Tùng	0		0850	0850
3	2009	1A013	1A013	1A013	3	229	700	625	375	1700	1700	Bùi Thu	Hiền	0		1700	1700
3	2009	1B242	1B242	1B242	2NT	228	800	725	575	2100	2100	Bùi Thu	Hiền	0		2100	2100
3	2009	22032	22032	22032	2NT	533	650	725	575	1950	1950	Bùi Thị	Nguyệt	0		1950	1950
3	2009	22073	22073	22073	2NT	721	400	100	250	0750	750	Bùi Thị	Thuỳ	0		0750	0750
3	2009	26027	26027	26027	2NT	540	400	175	675	1250	1250	Bùi Thị	Nhài	0		1250	1250
3	2009	27081	27081	27081	2NT	383	750	650	425	1825	1850	Bùi Thị	Lan	0		1825	1850
3	2009	26039	26039	26039	2NT	373	650	950	325	1925	1950	Bùi Thị Hồng	Khuyên	0		1925	1950
3	2009	26004	26004	26004	2	750	600	700	650	1950	1950	Bùi Thị Kim	Thương	0		1950	1950
3	2009	03013	03013	03013	3	403	700	750	700	2150	2150	Bùi Thị Mai	Linh	0		2150	2150
3	2009	18012	18012	18012	2	761	700	800	575	2075	2100	Bùi Thị Minh	Trang	1		2075	2100
3	2008	1A032	1A032	1A032	3	730	600	200	325	1125	1150	Bùi Thị Ngọc	Thuỷ	0		1125	1150
3	2008	1B210	1B210	1B210	2NT	491	700	550	325	1575	1600	Bùi Thị Thu	Nga	0		1575	1600
3	2009	1A003	1A003	1A003	3	122	350	925	650	1925	1950	Bùi Việt	Dương	0		1925	1950
3	2007	29011	29011	29011	2	916	600	225	250	1075	1100	Cao Thị Thu	Hà	0		1075	1100
3	2007	24013	24013	24013	2	114	650	175	300	1125	1150	Cao Văn	Dũng	0		1125	1150
3	2009	28012	28012	28012	2	404	500	375	325	1200	1200	Cao Đăng	Linh	0		1200	1200
3	2009	1A144	1A144	1A144	3	209	600	550	850	2000	2000	Chu Thanh	Hằng	0		2000	2000
3	2009	1A060	1A060	1A060	2	322	500	200	250	0950	950	Chu Thanh	Hương	0		0950	0950
3	2008	22057	22057	22057	2NT	298	700	400	650	1750	1750	Chu Thị	Huyền	0		1750	1750

Figure 3.2. Original database of University Entrance Exam in year 2009 (in .dbf file format of Visual FoxPro) (cont.)

3.2 Selected Attributes

Depending on the meaning of each attribute and the goal of my research, I choose five attributes province/city (region), block, score1, score2, score3 to find the relationships that are hidden in this database. I select score1, score2, score3 but not draft score1, draft score2, draft score3 because score1, score2, score3 are original score, it reflects the real result of students. In addition, the reason for my selection is I want to discover the correlation between subjects that students study, the strengths of each region. Furthermore, I can understand definitely about the quality of the teaching or quality of exams.

Attribute "block" has many values such as "A", "B", "C", "D", "T", "H" and so on. However, in this thesis, I just process in some main blocks "A", "B", "C", "D1", "D2", "D3", "D4", "D5", "D6" because number of students in these blocks are larger than other blocks; all subjects of these blocks are one multiple. I convert numeric, the type of score1, score2, score3, into nominal type that means the value of score data is classified into two intervals (from 0 to 5 and from 5 to 10). In addition, the subjects and its order are different from each block, all of them will be shown in Table 3.2. Table 3.3. shows the list of regions in year 2011. Through each year, some new regions will be established but the original regions are similar so this changing does not affect insignificantly the results.

Table 3.2. Subjects of each block in University Entrance Exams

Block	Subject 1	Subject 2	Subject 3
A	Mathematics	Physics	Chemistry
B	Biology	Mathematics	Chemistry
C	Literature	History	Geography
D[1]	Literature	Mathematics	Foreign Language

Table 3.3. List of regions of our country in year 2011

Region No.	Region Name	Region No.	Region Name
01[2]	Ha Noi City	**33**	Thua Thien - Hue Province
02	Ho Chi Minh City	**34**	Quang Nam Province
03	Hai Phong City	**35**	Quang Ngai Province
04	Da Nang City	**36**	Kon Tum Province
05	Ha Giang Province	**37**	Binh Dinh Province
06	Cao Bang Province	**38**	Gia Lai Province
07	Lai Chau Province	**39**	Phu Yen Province
08	Lao Cai Province	**40**	Dak Lak Province
09	Tuyen Quang Province	**41**	Khanh Hoa Province
10	Lang Son Province	**42**	Lam Dong Province
11	Bac Can Province	**43**	Binh Phuoc Province

[1] D1, D2, D3, D4, D5, D6 are different in final subject, Foreign Language, may be English, Japanese and so on.

[2] Ha Noi City includes two other sub-regions, these notations are 1A and 1B.

12	Thai Nguyen Province	44	Binh Duong Province
13	Yen Bai Province	45	Ninh Thuan Province
14	Son La Province	46	Tay Ninh Province
15	Phu Tho Province	47	Binh Thuan Province
16	Vinh Phuc Province	48	Dong Nai Province
17	Quang Ninh Province	49	Long An Province
18	Bac Giang Province	50	Dong Thap Province
19	Bac Ninh Province	51	An Giang Province
20[1]		52	Ba Ria – Vung Tau Province
21	Hai Duong Province	53	Tien Giang Province
22	Hung Yen Province	54	Kien Giang Province
23	Hoa Binh Province	55	Can Tho City
24	Ha Nam Province	56	Ben Tre Province
25	Nam Dinh Province	57	Vinh Long Province
26	Thai Binh Province	58	Tra Vinh Province
27	Ninh Binh Province	59	Soc Trang Province
28	Thanh Hoa Province	60	Bac Lieu Province
29	Nghe An Province	61	Ca Mau Province
30	Ha Tinh Province	62	Dien Bien Province
31	Quang Binh Province	63	Dak Nong Province
32	Quang Tri Province	64	Hau Giang Province

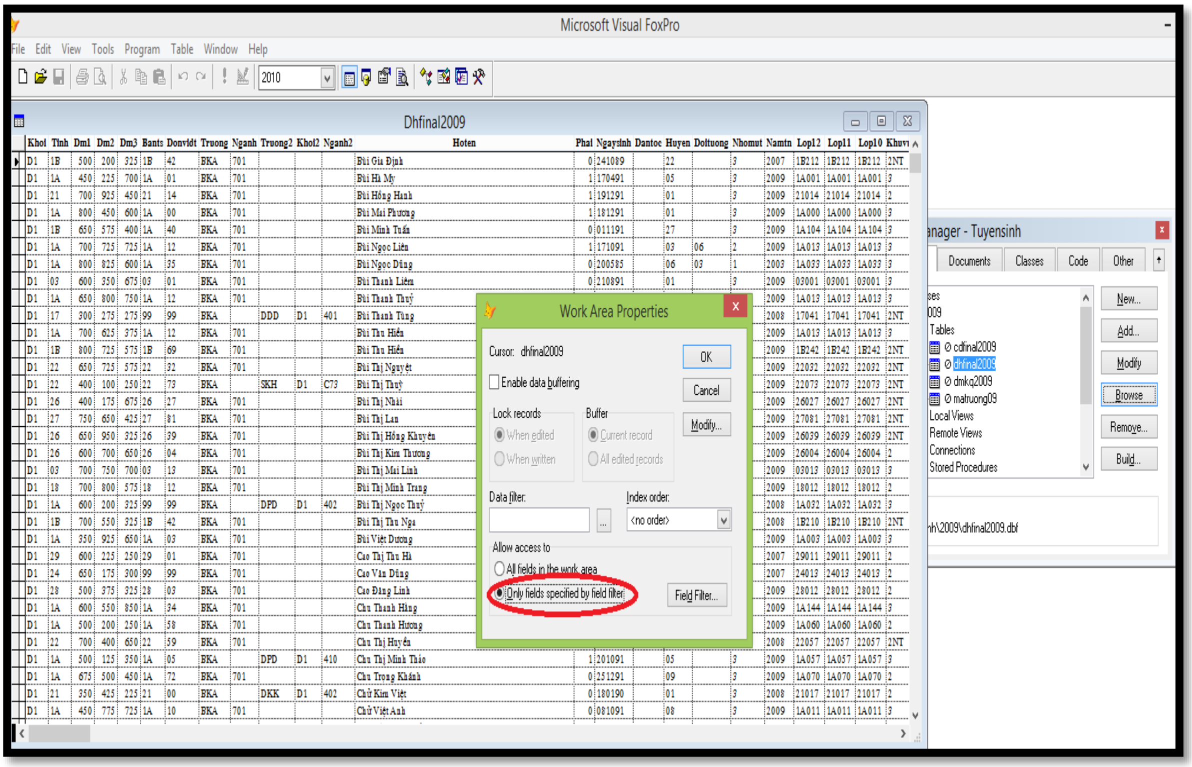
3.3 Data Processing

The original database has so many unnecessary fields. Moreover, the format of these file (.dbf) is hard to use. Before running my program, I must do an important

[1] This is not assigned to any regions.

step that is data processing. Especially, I use Microsoft Visual FoxPro 9.0, Excel 2013 and Notepad to convert .dbf file format into .csv file format.

First of all, I use Microsoft Visual FoxPro 9.0 to filter the necessary properties then converting .dbf file format into .txt file format. I choose "*Only fields specified by field filter*" in the property tab of this table (Figure 3.3.).



Dhfinal2009

Khoi	Tinh	Dm1	Dm2	Dm3	Bants	Donvidt	Truong	Nganh	Truong2	Khoi2	Nganh2	Hoten	Phai	Ngaysinh	Dantoc	Huyen	Doituong	Nhomut	Namtn	Lop12	Lop11	Lop10	Khuvi
D1	1B	500	200	325	1B	42	BKA	701				Bùi Gia Định	0	241089		22		3	2007	1B212	1B212	1B212	2NT
D1	1A	450	225	700	1A	01	BKA	701				Bùi Hà My	1	170491		05		3	2009	1A001	1A001	1A001	3
D1	21	700	925	450	21	14	BKA	701				Bùi Hồng Hạnh	1	191291		01		3	2009	21014	21014	21014	2
D1	1A	800	450	600	1A	00	BKA	701				Bùi Mai Phương	1	181291		01		3	2009	1A000	1A000	1A000	3
D1	1B	650	575	400	1A	40	BKA	701				Bùi Minh Tuấn	0	011191		27		3	2009	1A104	1A104	1A104	3
D1	1A	700	725	725	1A	12	BKA	701				Bùi Ngọc Liên	1	171091		03	06	2	2009	1A013	1A013	1A013	3
D1	1A	800	825	600	1A	35	BKA	701				Bùi Ngọc Dũng	0	200585		06	03	1	2003	1A033	1A033	1A033	3
D1	03	600	350	675	03	01	BKA	701				Bùi Thanh Liêm	0	210891		01		3	2009	03001	03001	03001	3
D1	1A	650	800	750	1A	12	BKA	701				Bùi Thanh Thuỷ							2009	1A013	1A013	1A013	3
D1	17	300	275	275	99	99	BKA		DDD	D1	401	Bùi Thanh Tùng							2008	17041	17041	17041	2NT
D1	1A	700	625	375	1A	12	BKA	701				Bùi Thu Hiền							2009	1A013	1A013	1A013	3
D1	1B	800	725	575	1B	69	BKA	701				Bùi Thu Hiền							2009	1B242	1B242	1B242	2NT
D1	22	650	725	575	22	32	BKA	701				Bùi Thị Nguyệt							2009	22032	22032	22032	2NT
D1	22	400	100	250	22	73	BKA		SKH	D1	C73	Bùi Thị Thuỳ							2009	22073	22073	22073	2NT
D1	26	400	175	675	26	27	BKA	701				Bùi Thị Nhài							2009	26027	26027	26027	2NT
D1	27	750	650	425	27	81	BKA	701				Bùi Thị Lan							2009	27081	27081	27081	2NT
D1	26	650	950	325	26	39	BKA	701				Bùi Thị Hồng Khuyên							2009	26039	26039	26039	2NT
D1	26	600	700	650	26	04	BKA	701				Bùi Thị Kim Thương							2009	26004	26004	26004	2
D1	03	700	750	700	03	13	BKA	701				Bùi Thị Mai Linh							2009	03013	03013	03013	3
D1	18	700	800	575	18	12	BKA	701				Bùi Thị Minh Trang							2009	18012	18012	18012	2
D1	1A	600	200	325	99	99	BKA		DPD	D1	402	Bùi Thị Ngọc Thuỷ							2008	1A032	1A032	1A032	3
D1	1B	700	550	325	1B	42	BKA	701				Bùi Thị Thu Nga							2008	1B210	1B210	1B210	2NT
D1	1A	350	925	650	1A	03	BKA	701				Bùi Việt Dương							2009	1A003	1A003	1A003	3
D1	29	600	225	250	29	01	BKA	701				Cao Thị Thu Hà							2007	29011	29011	29011	2
D1	24	650	175	300	99	99	BKA	701				Cao Văn Dũng							2007	24013	24013	24013	2
D1	28	500	375	325	28	03	BKA	701				Cao Đăng Linh							2009	28012	28012	28012	2
D1	1A	600	550	850	1A	34	BKA	701				Chu Thanh Hằng							2009	1A144	1A144	1A144	3
D1	1A	500	200	250	1A	58	BKA	701				Chu Thanh Hương							2009	1A060	1A060	1A060	2
D1	22	700	400	650	22	59	BKA	701				Chu Thị Huyền							2008	22057	22057	22057	2NT
D1	1A	500	125	350	1A	05	BKA		DPD	D1	410	Chu Thị Minh Thảo	1	201091		05		3	2009	1A057	1A057	1A057	3
D1	1A	675	500	450	1A	72	BKA	701				Chu Trọng Khánh	0	251291		09		3	2009	1A070	1A070	1A070	2
D1	21	350	425	225	21	00	BKA		DKK	D1	402	Chử Kim Việt	0	180190		01		3	2008	21017	21017	21017	2
D1	1A	450	775	725	1A	10	BKA	701				Chử Việt Anh	0	081091		08		3	2009	1A011	1A011	1A011	3

Work Area Properties

Cursor: dhfinal2009

□ Enable data buffering

Lock records: ● When edited ○ When written

Buffer: ● Current record ○ All edited records

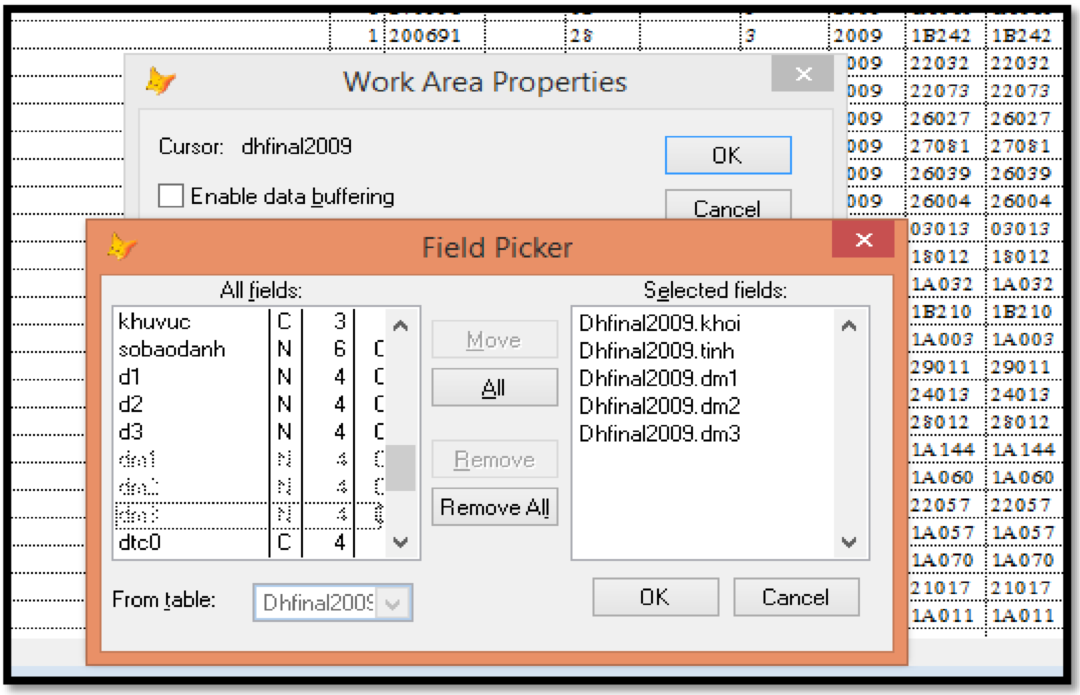
Data filter:

Index order: <no order>

Allow access to: ○ All fields in the work area ● Only fields specified by field filter

Figure 3.3. Table Property

In addition, I select necessary fields for my work in "Field Filter", they are "khoi", "tinh", "dm1", "dm2", "dm3" (Figure 3.4.). Then, I make condition for "khoi" in Expression Builder windows because I just choose some main blocks such as "A", "B", "C", "D". (Figure 3.5.).



Cursor: dhfinal2009

□ Enable data buffering

All fields:

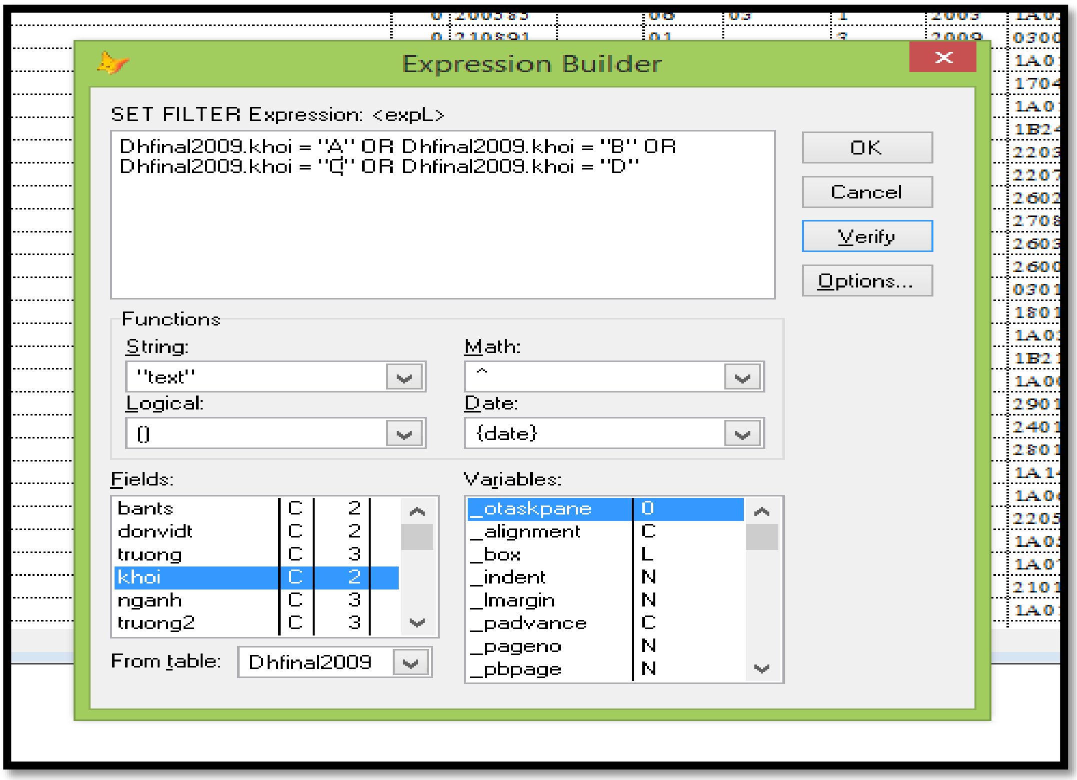
```
khuvuc     C  3
sobaodanh  N  6
d1         N  4
d2         N  4
d3         N  4
dm1        N  4
dm2        N  4
dm3        N  4
dtc0       C  4
```

Selected fields:

```
Dhfinal2009.khoi
Dhfinal2009.tinh
Dhfinal2009.dm1
Dhfinal2009.dm2
Dhfinal2009.dm3
```

From table: Dhfinal2009

Figure 3.4. Choosing needed fields



SET FILTER Expression: <expL>

```
Dhfinal2009.khoi = "A" OR Dhfinal2009.khoi = "B" OR
Dhfinal2009.khoi = "C" OR Dhfinal2009.khoi = "D"
```

Fields:

```
bants     C  2
donvidt   C  2
truong    C  3
khoi      C  2
nganh     C  3
truong2   C  3
```

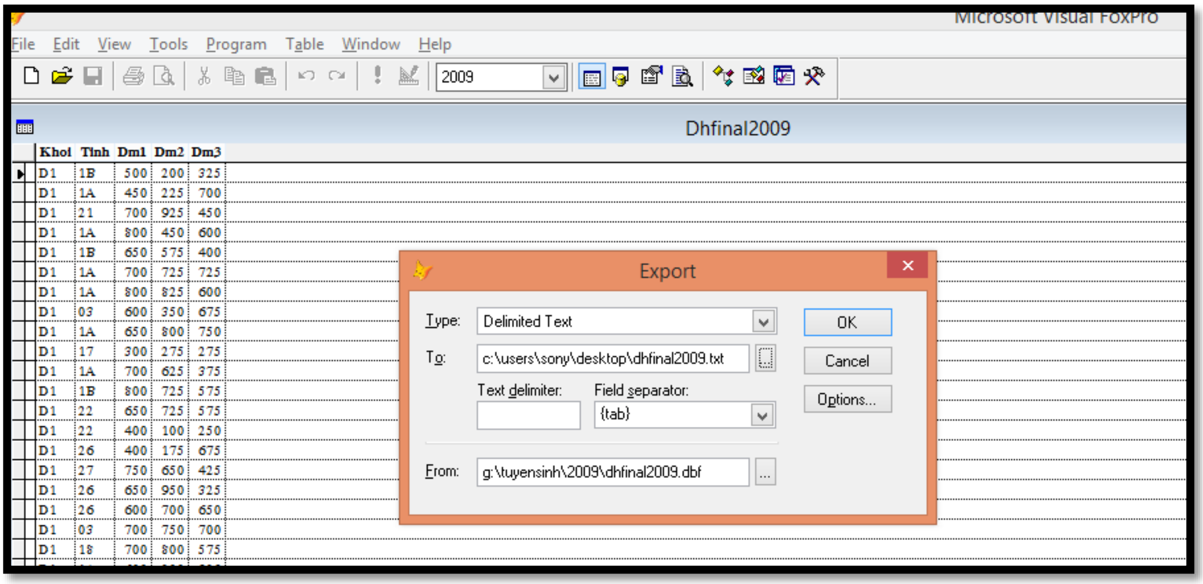
From table: Dhfinal2009

Variables:

```
_otaskpane  O
_alignment  C
_box        L
_indent     N
_lmargin    N
_padvance   C
_pageno     N
_pbpage     N
```

Figure 3.5. Condition for block field

I choose tab “Export” to export .dbf file format to .txt file format. File .txt is delimited text and separate each field by {tab}. In this case, I cannot convert directly into .csv file format because it does not support for this type (Figure 3.6.).

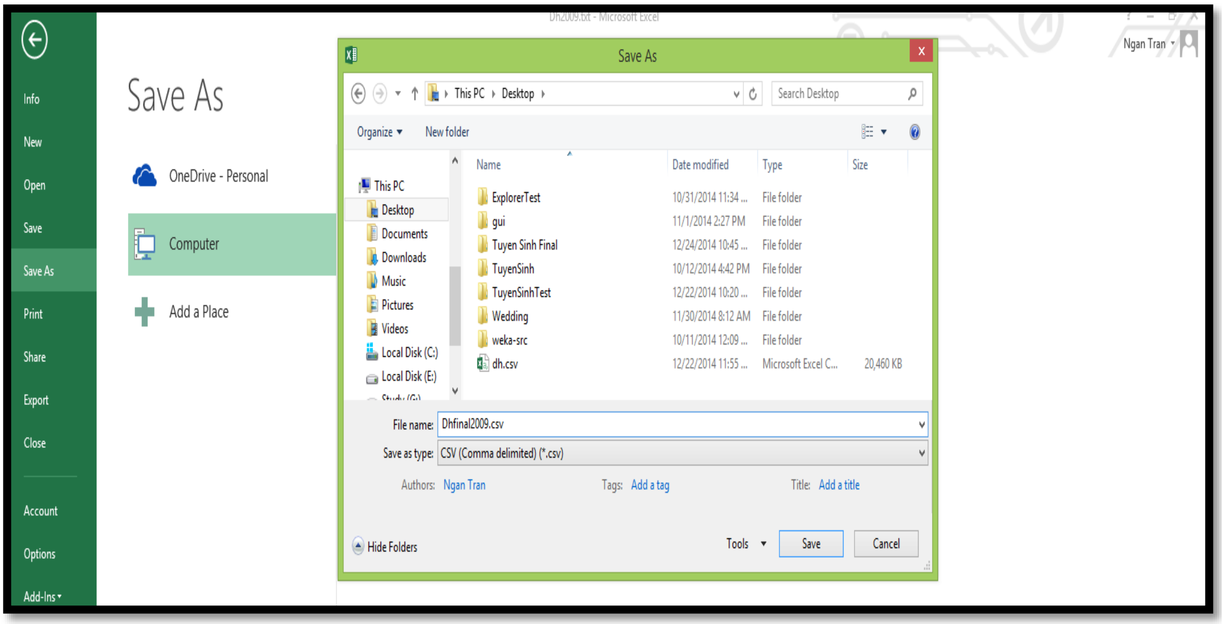


Khoi	Tinh	Dm1	Dm2	Dm3
D1	1B	500	200	325
D1	1A	450	225	700
D1	21	700	925	450
D1	1A	800	450	600
D1	1B	650	575	400
D1	1A	700	725	725
D1	1A	800	825	600
D1	03	600	350	675
D1	1A	650	800	750
D1	17	300	275	275
D1	1A	700	625	375
D1	1B	800	725	575
D1	22	650	725	575
D1	22	400	100	250
D1	26	400	175	675
D1	27	750	650	425
D1	26	650	950	325
D1	26	600	700	650
D1	03	700	750	700
D1	18	700	800	575

Type: Delimited Text
To: c:\users\sony\desktop\dhfinal2009.txt
Text delimiter:
Field separator: {tab}
From: g:\tuyensinh\2009\dhfinal2009.dbf

Figure 3.6. Exporting .dbf file format to .txt file format

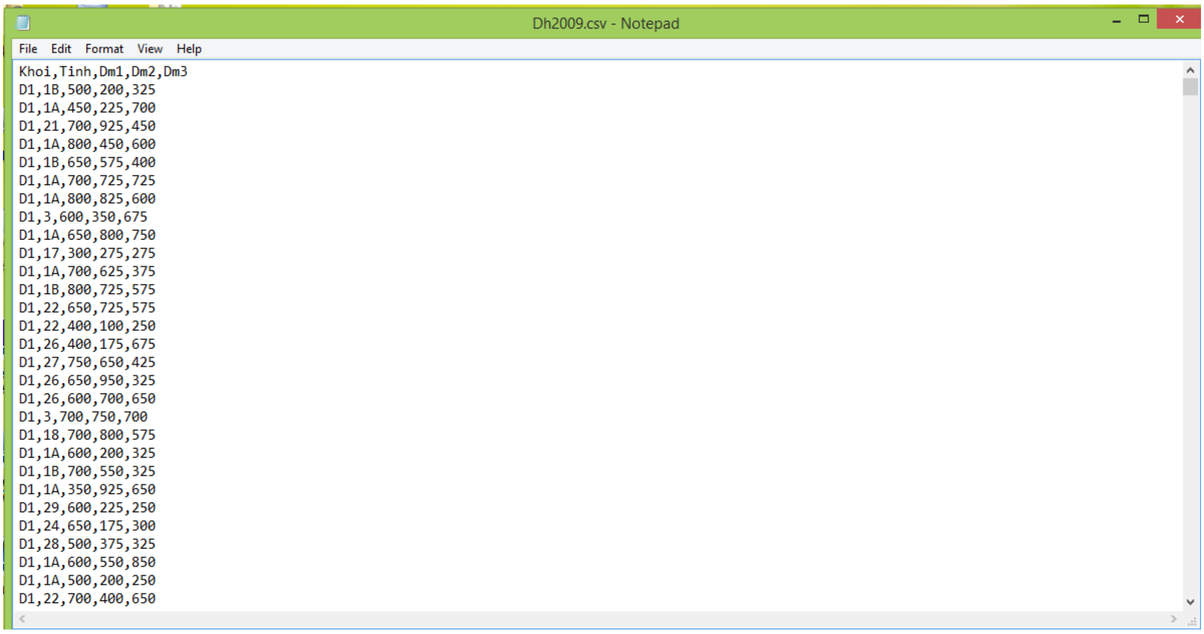
Secondly, I convert file format from .txt into .csv by using Excel 2013 (Figure 3.7.). Finally, I open .csv file by Notepad to confirm that all data is correct (Figure 3.8.).



Name	Date modified	Type	Size
ExplorerTest	10/31/2014 11:34 ...	File folder	
gui	11/1/2014 2:27 PM	File folder	
Tuyen Sinh Final	12/24/2014 10:45 ...	File folder	
TuyenSinh	10/12/2014 4:42 PM	File folder	
TuyenSinhTest	12/22/2014 10:20 ...	File folder	
Wedding	11/30/2014 8:12 AM	File folder	
weka-src	10/11/2014 12:09 ...	File folder	
dh.csv	12/22/2014 11:55 ...	Microsoft Excel C...	20,460 KB

File name: Dhfinal2009.csv
Save as type: CSV (Comma delimited) (*.csv)
Authors: Ngan Tran
Tags: Add a tag
Title: Add a title

Figure 3.7.Converting .txt file format to .csv file format by Excel 2013



```
Khoi,Tinh,Dm1,Dm2,Dm3
D1,1B,500,200,325
D1,1A,450,225,700
D1,21,700,925,450
D1,1A,800,450,600
D1,1B,650,575,400
D1,1A,700,725,725
D1,1A,800,825,600
D1,3,600,350,675
D1,1A,650,800,750
D1,17,300,275,275
D1,1A,700,625,375
D1,1B,800,725,575
D1,22,650,725,575
D1,22,400,100,250
D1,26,400,175,675
D1,27,750,650,425
D1,26,650,950,325
D1,26,600,700,650
D1,3,700,750,700
D1,18,700,800,575
D1,1A,600,200,325
D1,1B,700,550,325
D1,1A,350,925,650
D1,29,600,225,250
D1,24,650,175,300
D1,28,500,375,325
D1,1A,600,550,850
D1,1A,500,200,250
D1,22,700,400,650
```

Figure 3.8. Database after processing in .csv file format

The size of the original database and processed data are described in Table 3.4. The size of processed database is smaller than the original database because I filter by several conditions. Furthermore, the size of data is opened in Excel in limit so the sizes of database in year 2010 and database in year 2010 are approximately equal.

Table 3.4. The size of database before and after processing[1]

Year	Size of Original Database	Size of Database after processing
2009	996,443	948,791
2010	1,072,439	1,048,575
2011	1,328,267	1,048,575

[1] The size of database in year 2010, 2011 after processing are approximately equal because the limited of Excel in opening large data.

Chapter 4: System Implementation and Testing

This chapter includes two parts, the first is describing my demo application and the second is explaining the results of my application.

4.1 System Implementation

4.1.1 Programming Framework

- Minimum Java is version java 1.6.

  This Java version support for NetBeans to run correctly.

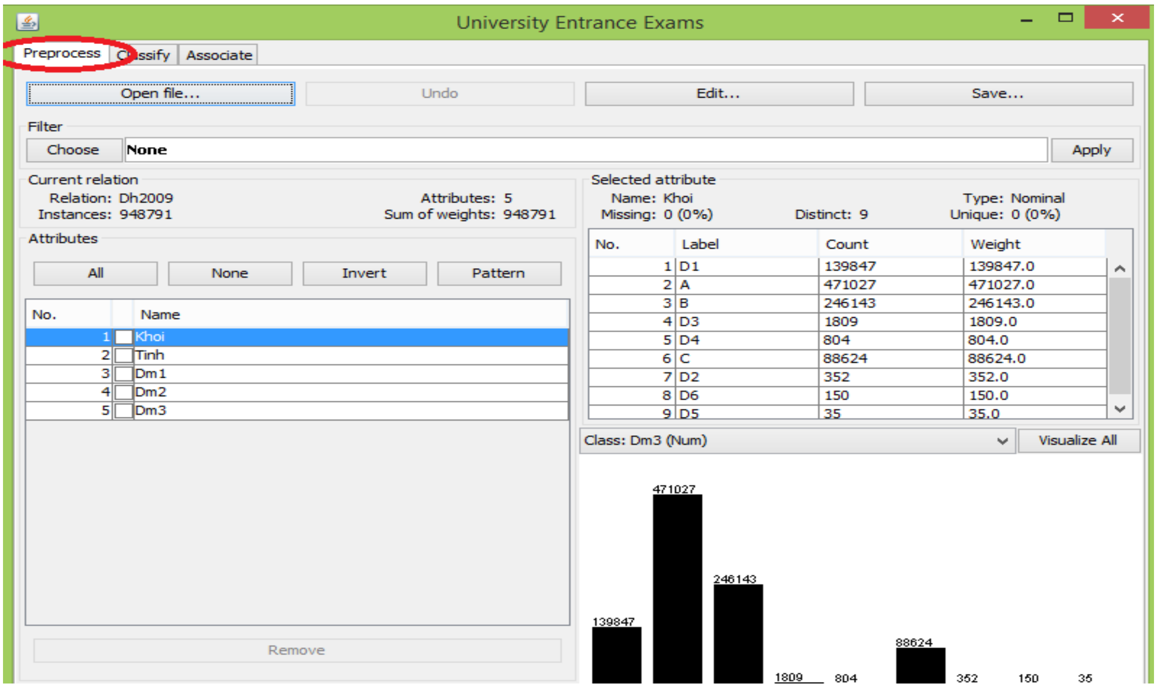
- Programming language is Java and program for developing is NetBeans IDE 8.0.1

  NetBeans is an open-source software development project and a supporter for Java Programming Language, an object-oriented programing. NetBeans has friendly graphic user interface so it is easy to code.

4.1.2 Graphic User Interface Implementation

My demo application has three main interfaces such as Preprocessing interface (Figure 4.1.), Classifying interface (Figure 4.4.), Associating interface (Figure 4.5.).

- Preprocessing interface has five components that are described in the Table 4.1. When clicking on the "*Open file*" button, the Open Dialog will appear for choosing input data file (Figure 4.2.). After processing the data, we can save this data file by clicking on button "*Save*" then a Save Dialog will appear for saving data file after processing (Figure 4.3.).



Filter: None

Current relation
Relation: Dh2009
Instances: 948791
Attributes: 5
Sum of weights: 948791

Attributes

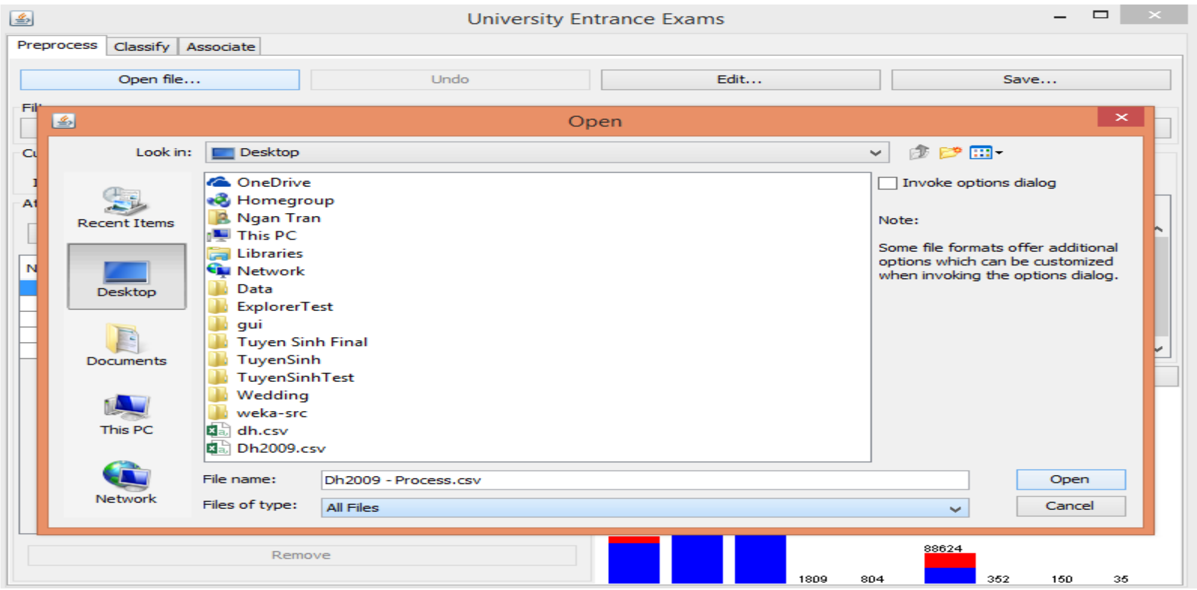
No.		Name
1	□	Khoi
2	□	Tinh
3	□	Dm1
4	□	Dm2
5	□	Dm3

Selected attribute
Name: Khoi
Missing: 0 (0%)
Distinct: 9
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	D1	139847	139847.0
2	A	471027	471027.0
3	B	246143	246143.0
4	D3	1809	1809.0
5	D4	804	804.0
6	C	88624	88624.0
7	D2	352	352.0
8	D6	150	150.0
9	D5	35	35.0

Class: Dm3 (Num)

Figure 4.1. Preprocessing Interface (first tab)



Look in: Desktop

OneDrive
Homegroup
Ngan Tran
This PC
Libraries
Network
Data
ExplorerTest
gui
Tuyen Sinh Final
TuyenSinh
TuyenSinhTest
Wedding
weka-src
dh.csv
Dh2009.csv

□ Invoke options dialog

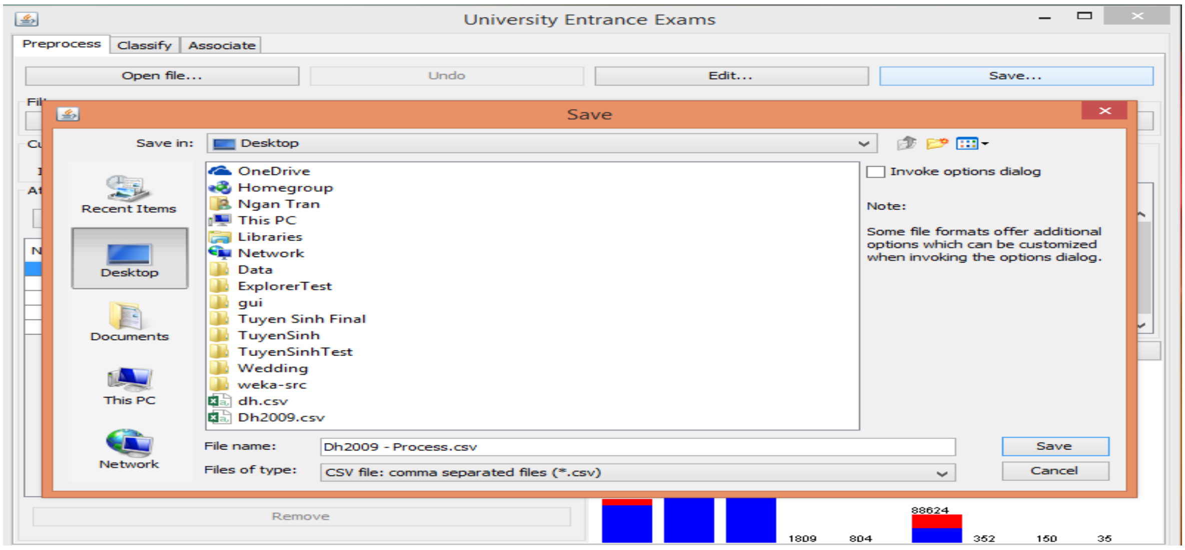
Note:

Some file formats offer additional options which can be customized when invoking the options dialog.

File name: Dh2009 - Process.csv

Files of type: All Files

Figure 4.2. Open dialog for choosing input data file



Save in: Desktop

OneDrive
Homegroup
Ngan Tran
This PC
Libraries
Network
Data
ExplorerTest
gui
Tuyen Sinh Final
TuyenSinh
TuyenSinhTest
Wedding
weka-src
dh.csv
Dh2009.csv

□ Invoke options dialog

Note:

Some file formats offer additional options which can be customized when invoking the options dialog.

File name: Dh2009 - Process.csv

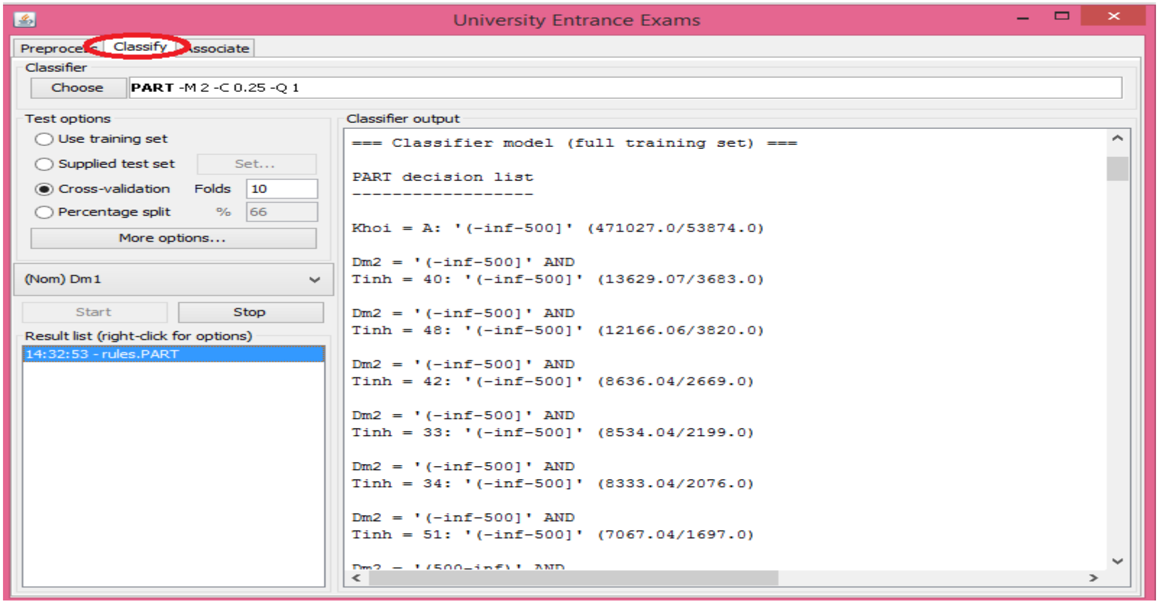
Files of type: CSV file: comma separated files (*.csv)

Figure 4.3. Save dialog for saving data file after process

Table 4.1. Components in the Preprocessing interface

Components	Element in component	Describe
Button	Open file	Choose a file from computer to load to this demo application.
	Undo	Go back to the current activity.
	Edit	Modify the database.
	Save	Save changes.
Filter	Choose button	Choose the method we want to do.
	Property	Display the properties of chosen method.
	Apply button	Apply the method we choose.
Current relation	Relation	The name of the database we choose.
	Attributes	The number of attributes in this database.
	Instances	Number of instances in this database.
	Sum of weights	The total number of data in this database.

Attributes	All button	Choose all attributes.
	None button	Do not choose any attributes.
	Invert button	Convert from chosen attributes to not chosen attributes and vice versa.
	Attributes table	Show all attributes in chosen database.
	Remove button	Remove the chosen attributes.
Selected attribute	Name	The name of chosen attribute.
	Type	The type of chosen attribute.
	Missing	The number of missed values.
	Distinct	The number of distinct values.
	Unique	The number of unique values.
	Selected attributes table	Show all values of chosen attribute.



PART -M 2 -C 0.25 -Q 1

○ Use training set
○ Supplied test set   Set...
● Cross-validation   Folds 10
○ Percentage split   % 66

(Nom) Dm1

14:32:53 - rules.PART

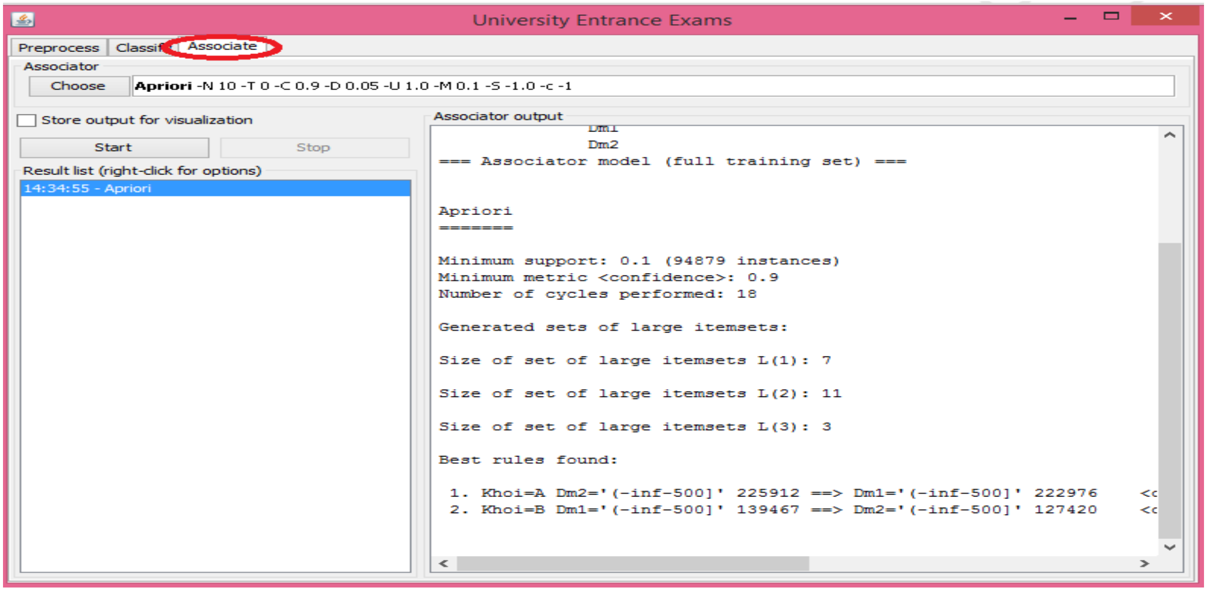
```
=== Classifier model (full training set) ===

PART decision list
------------------

Khoi = A: '(-inf-500]' (471027.0/53874.0)

Dm2 = '(-inf-500]' AND
Tinh = 40: '(-inf-500]' (13629.07/3683.0)

Dm2 = '(-inf-500]' AND
Tinh = 48: '(-inf-500]' (12166.06/3820.0)

Dm2 = '(-inf-500]' AND
Tinh = 42: '(-inf-500]' (8636.04/2669.0)

Dm2 = '(-inf-500]' AND
Tinh = 33: '(-inf-500]' (8534.04/2199.0)

Dm2 = '(-inf-500]' AND
Tinh = 34: '(-inf-500]' (8333.04/2076.0)

Dm2 = '(-inf-500]' AND
Tinh = 51: '(-inf-500]' (7067.04/1697.0)

Dm2 = '(500-inf)' AND
```

Figure 4.4. Classifying Interface (second tab)

- Classifying interface has five components that are described in Table 4.2.

Table 4.2. Components in the Classifying interface

Components	Element in component	Describe
Classifier	Choose button	Choose a classifier algorithm.
	Property	Display the property of chosen algorithm.
Test options	Radio button options	Some options for testing.
	More Options button	Other options that we can change.
Buttons	Attributes list	Choose attribute that we want to classify.
	Start button	Start chosen algorithm.
	Stop button	Stop running algorithm.
Result list	Result list window	Show list of algorithms that we has already run.
Classifier output	Output window	The output for the chosen algorithm after running successful.


University Entrance Exams
Preprocess
Classif
Associate
Associator
Choose
□ Store output for visualization
Start
Stop
Result list (right-click for options)
Associator output

Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

14:34:55 - Apriori

```
                  Dm1
                  Dm2
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.1 (94879 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 7

Size of set of large itemsets L(2): 11

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

Best rules found:

 1. Khoi=A Dm2='(-inf-500]' 225912 ==> Dm1='(-inf-500]' 222976    <c
 2. Khoi=B Dm1='(-inf-500]' 139467 ==> Dm2='(-inf-500]' 127420    <c
```

Figure 4.5. Associating Interface (third tab)

- Associating interface has four components that are described in Table 4.3.

Table 4.3. Components in the Associating interface

Components	Element in component	Describe
Associator	Choose button	Choose a associsation algorithm.
	Property	Display the property of chosen algorithm.
Buttons	Start button	Start chosen algorithm.
	Stop button	Stop running algorithm.
Result list	Result list window	Show list of algorithms that we has already run.
Associator ouput	Output window	The output for the chosen algorithm after running successful.

4.2 System Testing

In this section, I explain my demo application in four aspects: running time, preprocessing step, results and result explanation.

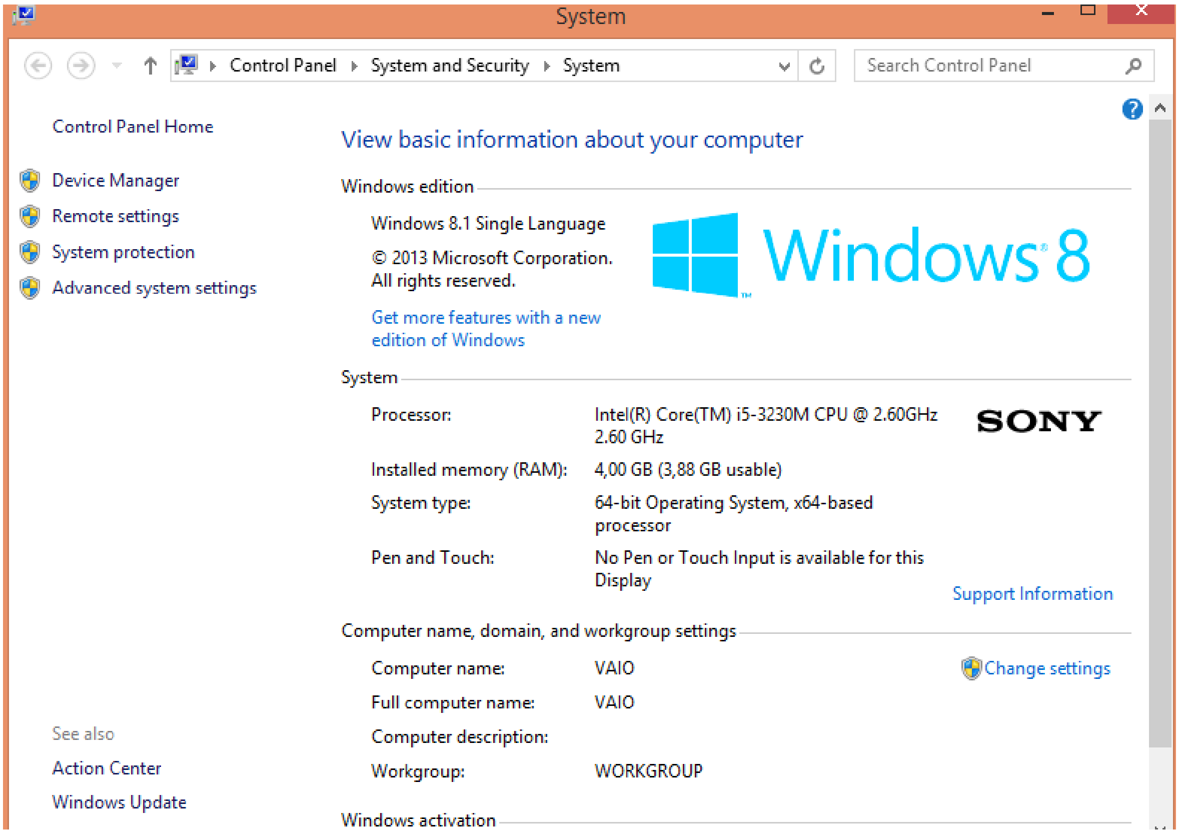
4.2.1 Time Testing

The running time of Preprocessing step, PART algorithm and Apriori algorithm is described in the Table 4.4. I use my laptop for testing my application. The information about my laptop is shown in Figure 4.6.

Table 4.4. Time testing of demo application

Step or Algorithm	Properties	Time
Preprocessing	2009	4s60
	2010	9s
	2011	8s
PART algorithm	Block	Build model: 4s54
		Total time:1m29s84

	Region	Build model: 5s64
		Total time: 1m57s43
	Score1	Build model: 5s30
		Total time:1m17s47
	Score2	Build model: 5s43
		Total time: 1m55s97
	Score3	Build model: 3s98
		Total time: 1m18s57
Apriori algorithm	2009	8s08
	2010	9s27
	2011	7s03



View basic information about your computer

Windows edition

Windows 8.1 Single Language

© 2013 Microsoft Corporation. All rights reserved.

Get more features with a new edition of Windows

System

Processor: Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz 2.60 GHz

Installed memory (RAM): 4,00 GB (3,88 GB usable)

System type: 64-bit Operating System, x64-based processor

Pen and Touch: No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name: VAIO

Full computer name: VAIO

Computer description:

Workgroup: WORKGROUP

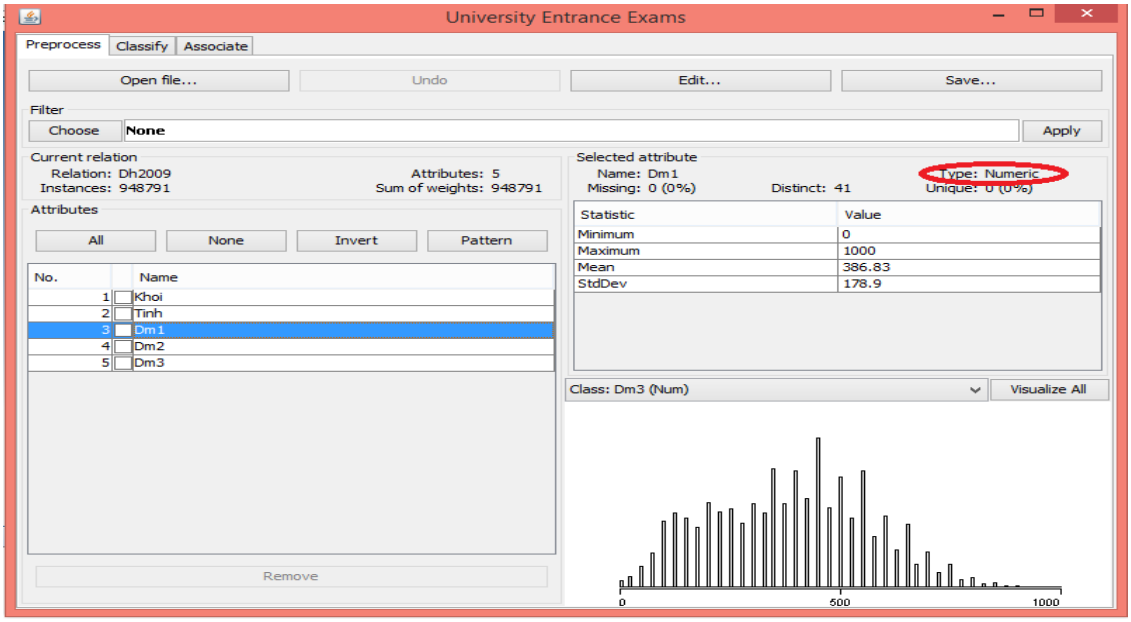
Windows activation

Figure 4.6. Information of tested laptop

With large databases, the preprocessing time is less than ten seconds, it depends on the size of databases. PART algorithm runs in less than six seconds to build a model and completes its running that includes building model and testing model in less than two minutes. Moreover, Apriori running time is not greater than 10 seconds for a database. These time results are so good with large databases which include larger than nine hundred thousand instances and with a normal laptop. Therefore, if data will be run by a supercomputer, the running time will be perfect.

4.2.2 Preprocessing Step

The fields “Dm1”, “Dm2”, “Dm3” from database is in numeric type. However, the type which is used in PART algorithm and Apriori algorithm must be nominal so I have another step to convert numeric into nominal.



Filter: None

Current relation
Relation: Dh2009
Instances: 948791
Attributes: 5
Sum of weights: 948791

Attributes

No.		Name
1	□	Khoi
2	□	Tinh
3	□	Dm1
4	□	Dm2
5	□	Dm3

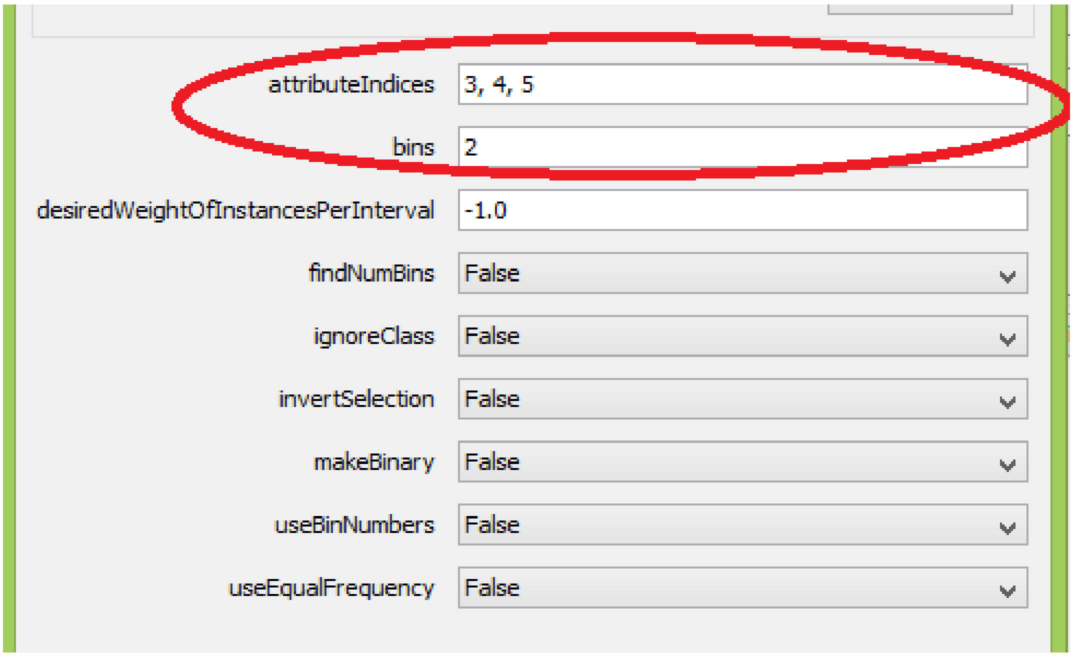
Selected attribute
Name: Dm1
Missing: 0 (0%)
Distinct: 41
Type: Numeric
Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1000
Mean	386.83
StdDev	178.9

Class: Dm3 (Num)

Figure 4.7. The original type of three fields “Dm1”, “Dm2”, “Dm3”

In the tab Preprocess of the program, in “Choose” of “Filter”, I choose “Discretize” method in Unsupevised/Attribute. In the property windows, I set “3, 4, 5” to “*attributeIndices*” because the fields I want to convert types is in 3, 4, 5 index. Next, “bins” is set to 2 because I want to divide into two intervals of score which are *(-inf-500]* that means interval of score from 0 to 5 and *(500-inf)* that means interval of score from 5 to 10 (Figure 4.8.). Finally, I click "*Apply*" to apply these changes. The result is shown in Figure 4.9.



attributeIndices: 3, 4, 5

bins: 2

desiredWeightOfInstancesPerInterval: -1.0

findNumBins: False

ignoreClass: False

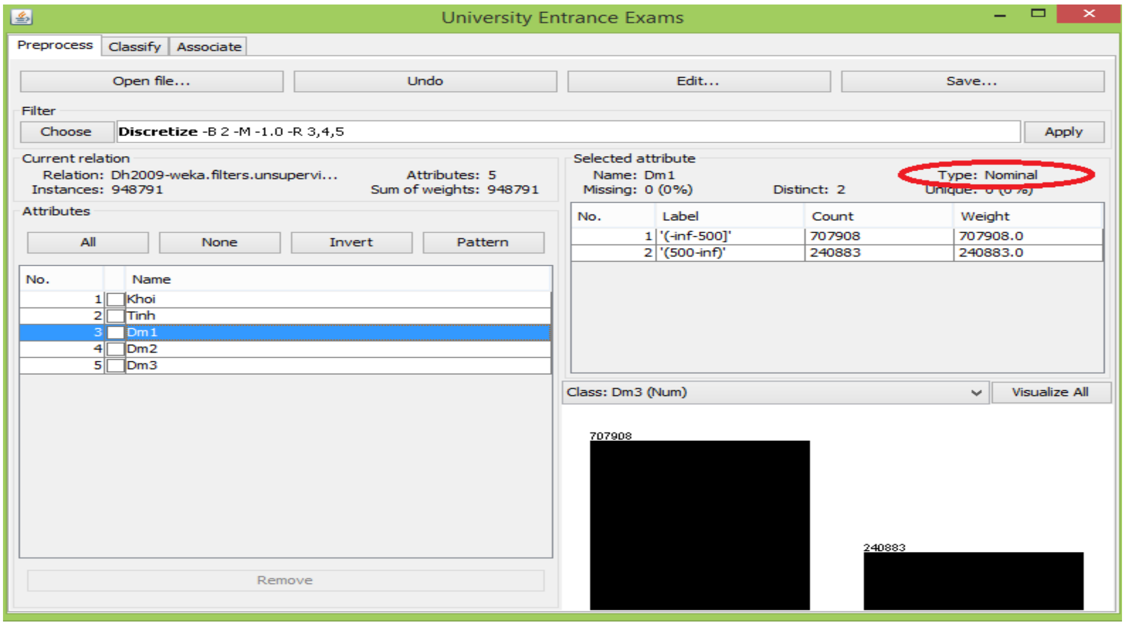
invertSelection: False

makeBinary: False

useBinNumbers: False

useEqualFrequency: False

Figure 4.8. Converting attribute value from numeric to nominal



University Entrance Exams

Filter: **Discretize** -B 2 -M -1.0 -R 3,4,5

Current relation

Relation: Dh2009-weka.filters.unsupervi...   Attributes: 5

Instances: 948791   Sum of weights: 948791

Attributes

No.		Name
1	□	Khoi
2	□	Tinh
3	□	Dm1
4	□	Dm2
5	□	Dm3

Selected attribute

Name: Dm1   Type: Nominal

Missing: 0 (0%)   Distinct: 2   Unique: 0 (0%)

No.	Label	Count	Weight
1	'(-inf-500]'	707908	707908.0
2	'(500-inf)'	240883	240883.0

Class: Dm3 (Num)

Figure 4.9. Result after apply Discretize method.

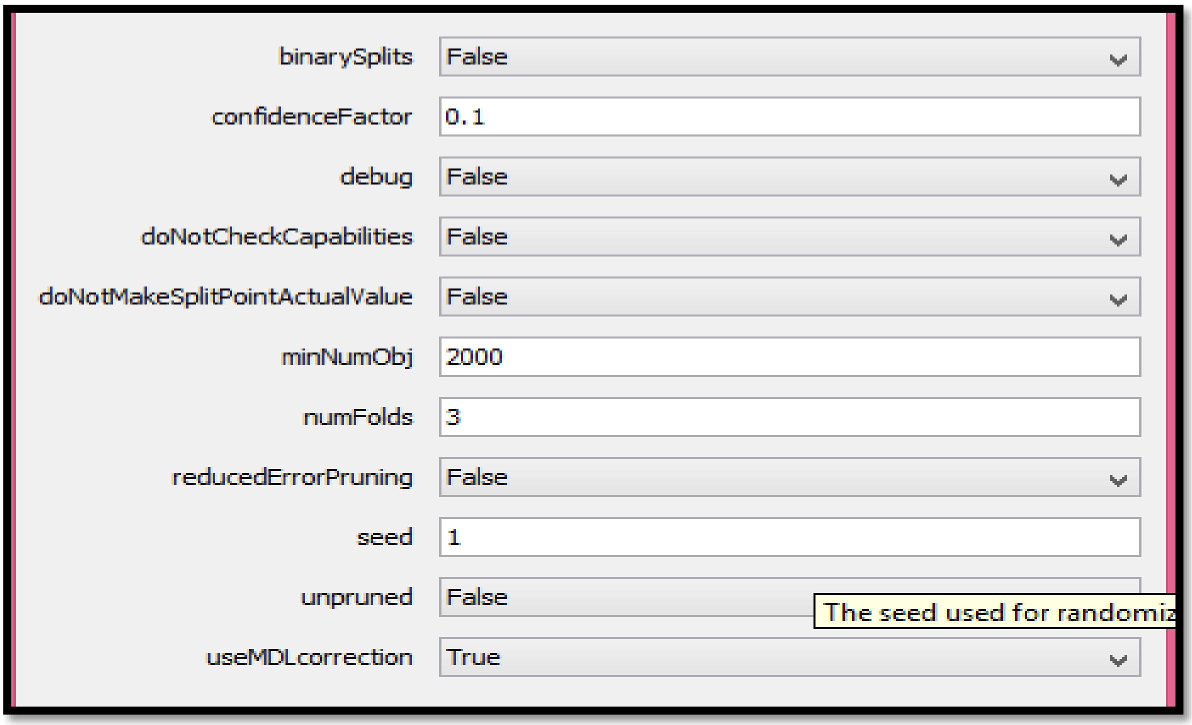
4.2.3 Results

When all data is prepared, I run my demo application "University Entrance Exams" to find the relationships between five attributes (block, region, score1, score2, score3). I run algorithm for each year because the capacity of my laptop does not allow me to run more than one year (the size of database is too large).

First, I run PART algorithm to get all classification rules which are hidden in this database. I will run five times with separate attributes "block", "region", "score1", "score2", "score3".

The options for this algorithm is shown in Figure 4.10. From the top to the bottom, first option is *binarySplits*, true or false value, which is used on nominal attributes when building the partial trees. Next option is *confidenceFactor* which is used to prune the partial trees. Specially, if it is smaller, the partial trees would be pruned more. *Debug* is true-false option; if it is set to true, the information of this classifier would be showed more. The other true-false option is *doNotCheckCapabilities*, the capacities of this classifier is checked or not before it is built. The option *doNotMakeSplitPointActualValue* is also a true-false option, it is useful for numeric attributes because of speed process. The minimum number of instances per rule is *minNumObj*. The option is used to reduced-error running is *numFolds*, number value, one is used for pruning and the others are used for develop rules. If *reducedErrorPruning* is false, the default C4.5 pruning is used. When *reducedErrorPruning* is true, the *seed* is used for randomizing data. The option *unpruned* is used to set pruning or not. The final option is *useMDLcorrection* that has correlation with finding spits on numeric attributes.

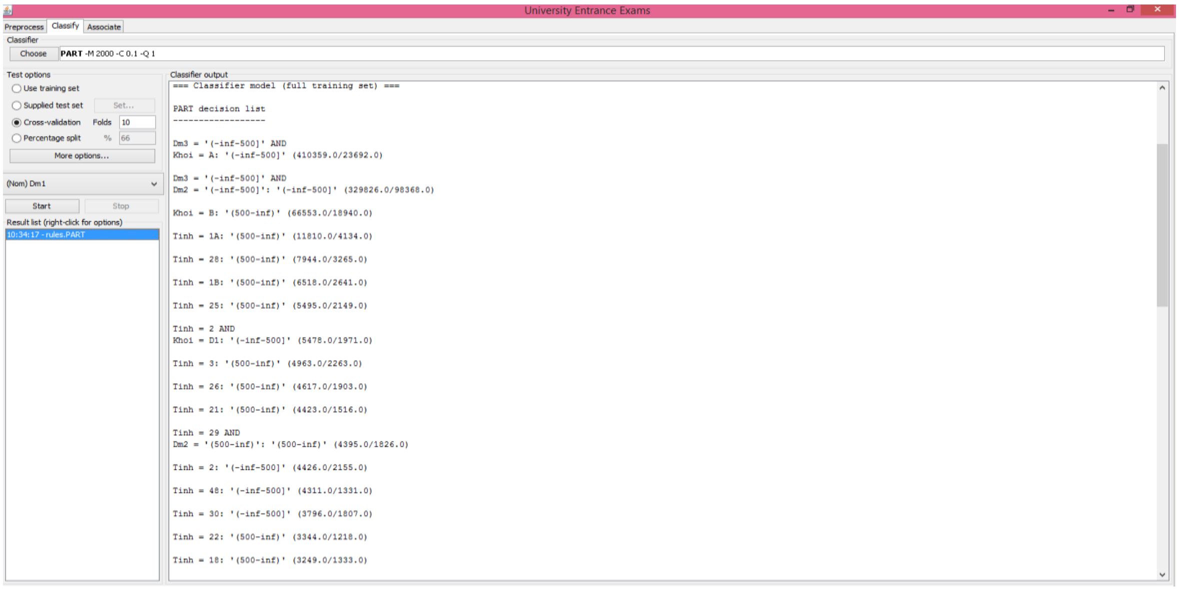
Figure 4.11. to Figure 4.15. are shown the results from my application for 2009 data with attributes: Block, Region, Score1, Score2 and Score3.



binarySplits	False
confidenceFactor	0.1
debug	False
doNotCheckCapabilities	False
doNotMakeSplitPointActualValue	False
minNumObj	2000
numFolds	3
reducedErrorPruning	False
seed	1
unpruned	False
useMDLcorrection	True

The seed used for randomiz

Figure 4.10. The properties of PART algorithm



PART -M 2000 -C 0.1 -Q 1

● Cross-validation Folds 10

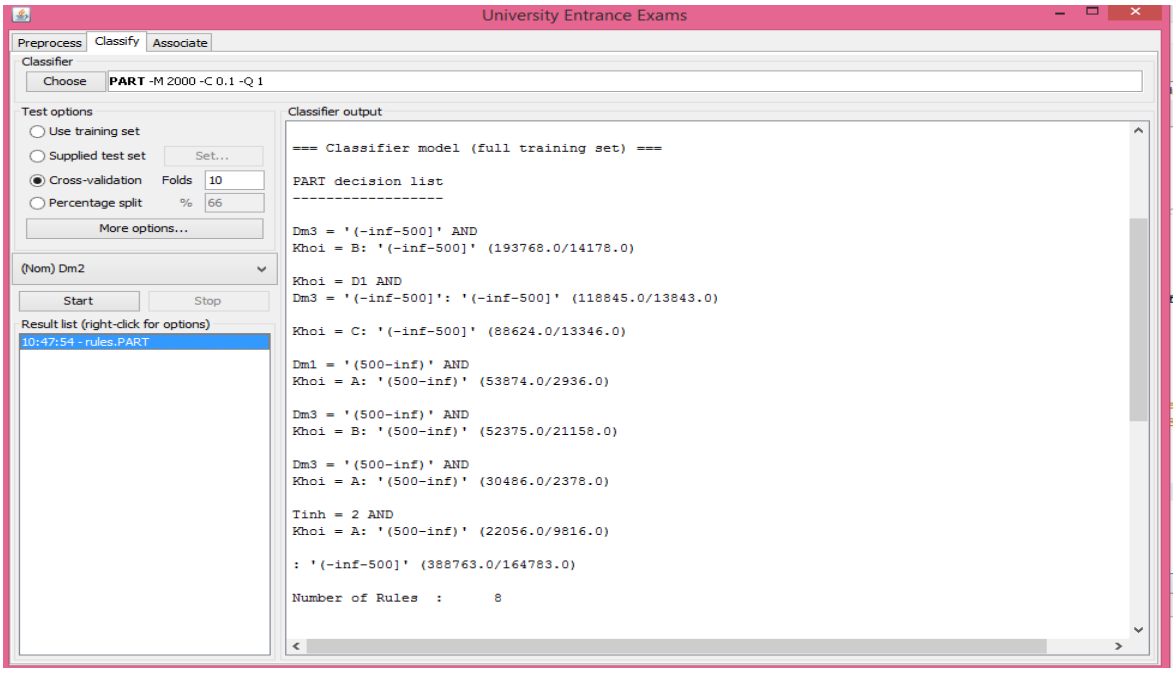
○ Percentage split % 66

(Nom) Dm1

10:34:17 - rules.PART

```
=== Classifier model (full training set) ===

PART decision list
------------------

Dm3 = '(-inf-500]' AND
Khoi = A: '(-inf-500]' (410359.0/23692.0)

Dm3 = '(-inf-500]' AND
Dm2 = '(-inf-500]': '(-inf-500]' (329826.0/98368.0)

Khoi = B: '(500-inf)' (66553.0/18940.0)

Tinh = 1A: '(500-inf)' (11810.0/4134.0)

Tinh = 28: '(500-inf)' (7944.0/3265.0)

Tinh = 1B: '(500-inf)' (6518.0/2641.0)

Tinh = 25: '(500-inf)' (5495.0/2149.0)

Tinh = 2 AND
Khoi = D1: '(-inf-500]' (5478.0/1971.0)

Tinh = 3: '(500-inf)' (4963.0/2263.0)

Tinh = 26: '(500-inf)' (4617.0/1903.0)

Tinh = 21: '(500-inf)' (4423.0/1516.0)

Tinh = 29 AND
Dm2 = '(500-inf)': '(500-inf)' (4395.0/1826.0)

Tinh = 2: '(-inf-500]' (4426.0/2155.0)

Tinh = 48: '(-inf-500]' (4311.0/1331.0)

Tinh = 30: '(-inf-500]' (3796.0/1807.0)

Tinh = 22: '(500-inf)' (3344.0/1218.0)

Tinh = 18: '(500-inf)' (3249.0/1333.0)
```

Figure 4.11. Classification results from data in year 2009 with Score1 attribute



PART -M 2000 -C 0.1 -Q 1

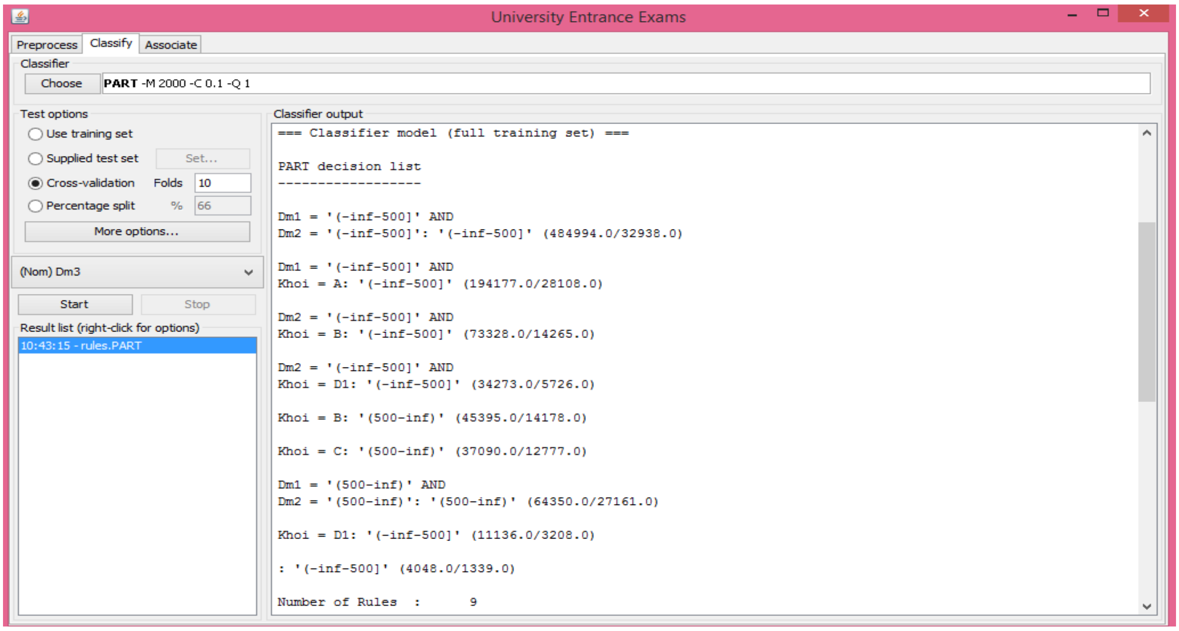
○ Use training set
○ Supplied test set   Set...
● Cross-validation   Folds 10
○ Percentage split   % 66
More options...

(Nom) Dm2

10:47:54 - rules.PART

```
=== Classifier model (full training set) ===

PART decision list
------------------

Dm3 = '(-inf-500]' AND
Khoi = B: '(-inf-500]' (193768.0/14178.0)

Khoi = D1 AND
Dm3 = '(-inf-500]': '(-inf-500]' (118845.0/13843.0)

Khoi = C: '(-inf-500]' (88624.0/13346.0)

Dm1 = '(500-inf)' AND
Khoi = A: '(500-inf)' (53874.0/2936.0)

Dm3 = '(500-inf)' AND
Khoi = B: '(500-inf)' (52375.0/21158.0)

Dm3 = '(500-inf)' AND
Khoi = A: '(500-inf)' (30486.0/2378.0)

Tinh = 2 AND
Khoi = A: '(500-inf)' (22056.0/9816.0)

: '(-inf-500]' (388763.0/164783.0)

Number of Rules  :      8
```

Figure 4.12. Classification results from data in year 2009 with Score2 attribute



PART -M 2000 -C 0.1 -Q 1

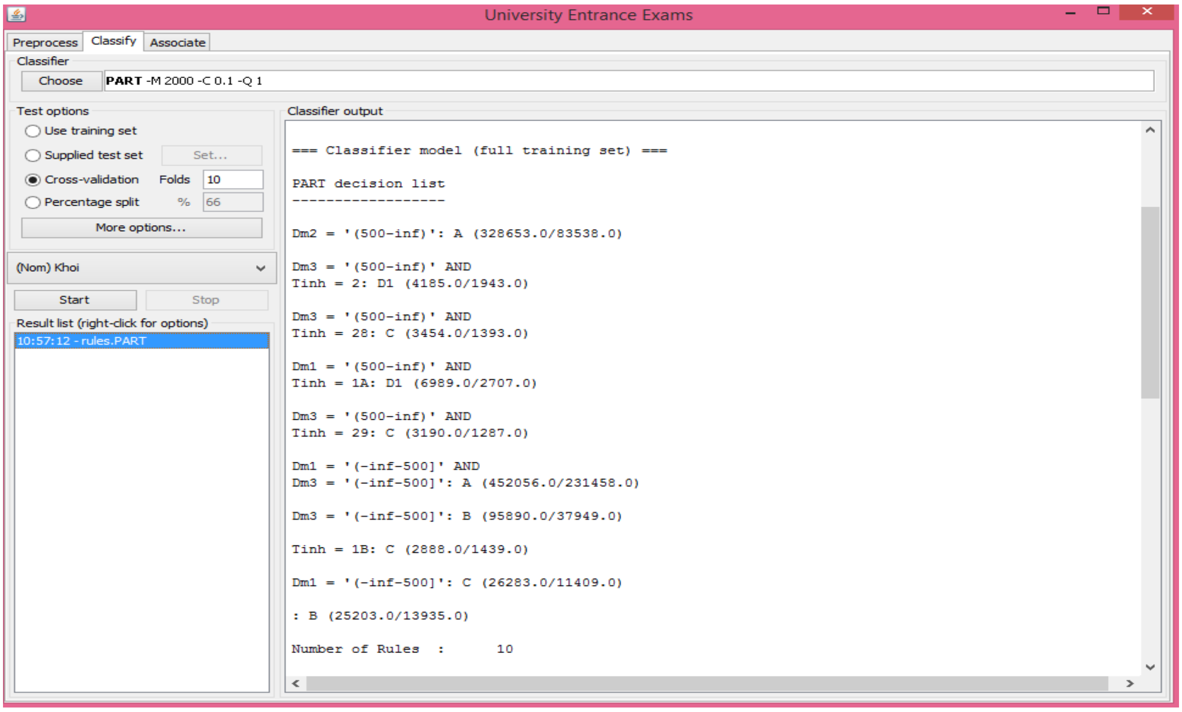
○ Use training set
○ Supplied test set   Set...
● Cross-validation   Folds 10
○ Percentage split   % 66
More options...

(Nom) Dm3

10:43:15 - rules.PART

```
=== Classifier model (full training set) ===

PART decision list
------------------

Dm1 = '(-inf-500]' AND
Dm2 = '(-inf-500]': '(-inf-500]' (484994.0/32938.0)

Dm1 = '(-inf-500]' AND
Khoi = A: '(-inf-500]' (194177.0/28108.0)

Dm2 = '(-inf-500]' AND
Khoi = B: '(-inf-500]' (73328.0/14265.0)

Dm2 = '(-inf-500]' AND
Khoi = D1: '(-inf-500]' (34273.0/5726.0)

Khoi = B: '(500-inf)' (45395.0/14178.0)

Khoi = C: '(500-inf)' (37090.0/12777.0)

Dm1 = '(500-inf)' AND
Dm2 = '(500-inf)': '(500-inf)' (64350.0/27161.0)

Khoi = D1: '(-inf-500]' (11136.0/3208.0)

: '(-inf-500]' (4048.0/1339.0)

Number of Rules  :      9
```

Figure 4.13. Classification results from data in year 2009 with Score3 attribute



Classifier
Choose PART -M 2000 -C 0.1 -Q 1

Test options
○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

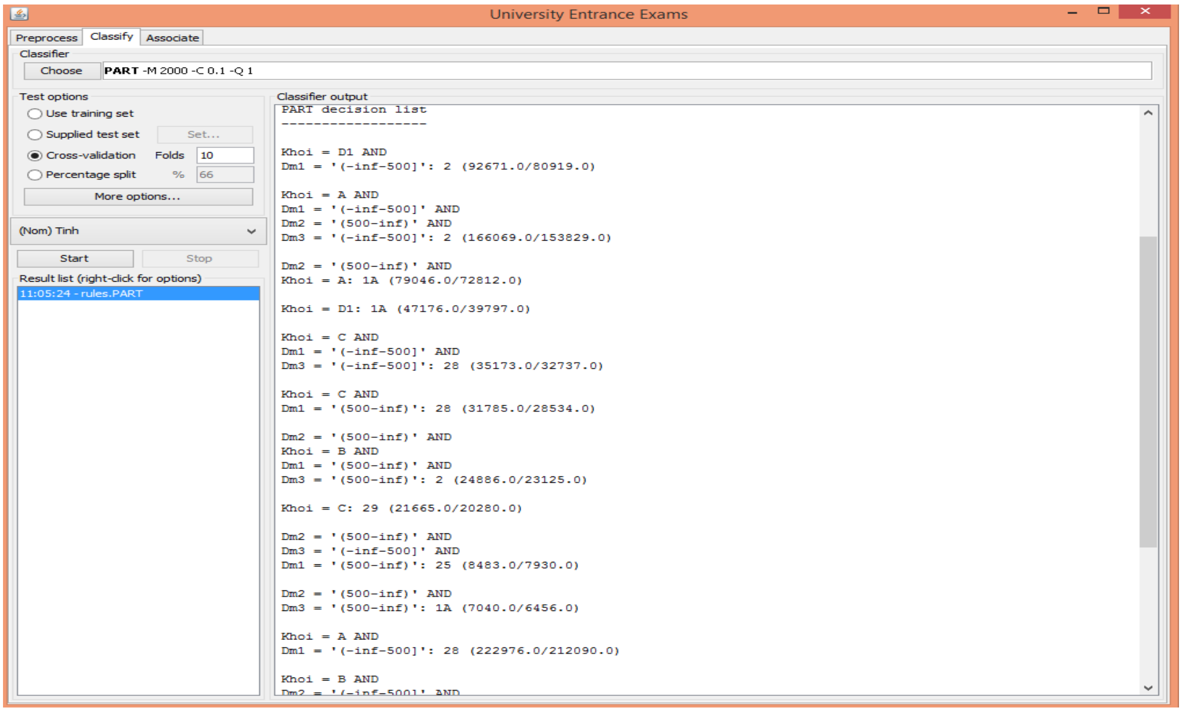
(Nom) Khoi

Result list (right-click for options)
10:57:12 - rules.PART

Classifier output

```
=== Classifier model (full training set) ===

PART decision list
------------------

Dm2 = '(500-inf)': A (328653.0/83538.0)

Dm3 = '(500-inf)' AND
Tinh = 2: D1 (4185.0/1943.0)

Dm3 = '(500-inf)' AND
Tinh = 28: C (3454.0/1393.0)

Dm1 = '(500-inf)' AND
Tinh = 1A: D1 (6989.0/2707.0)

Dm3 = '(500-inf)' AND
Tinh = 29: C (3190.0/1287.0)

Dm1 = '(-inf-500]' AND
Dm3 = '(-inf-500]': A (452056.0/231458.0)

Dm3 = '(-inf-500]': B (95890.0/37949.0)

Tinh = 1B: C (2888.0/1439.0)

Dm1 = '(-inf-500]': C (26283.0/11409.0)

: B (25203.0/13935.0)

Number of Rules  :      10
```

Figure 4.14. Classification results from data in year 2009 with Block attribute



Classifier
Choose PART -M 2000 -C 0.1 -Q 1

Test options
○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) Tinh

Result list (right-click for options)
11:05:24 - rules.PART

Classifier output

```
PART decision list
------------------

Khoi = D1 AND
Dm1 = '(-inf-500]': 2 (92671.0/80919.0)

Khoi = A AND
Dm1 = '(-inf-500]' AND
Dm2 = '(500-inf)' AND
Dm3 = '(-inf-500]': 2 (166069.0/153829.0)

Dm2 = '(500-inf)' AND
Khoi = A: 1A (79046.0/72812.0)

Khoi = D1: 1A (47176.0/39797.0)

Khoi = C AND
Dm1 = '(-inf-500]' AND
Dm3 = '(-inf-500]': 28 (35173.0/32737.0)

Khoi = C AND
Dm1 = '(500-inf)': 28 (31785.0/28534.0)

Dm2 = '(500-inf)' AND
Khoi = B AND
Dm1 = '(500-inf)' AND
Dm3 = '(500-inf)': 2 (24886.0/23125.0)

Khoi = C: 29 (21665.0/20280.0)

Dm2 = '(500-inf)' AND
Dm3 = '(-inf-500]' AND
Dm1 = '(500-inf)': 25 (8483.0/7930.0)

Dm2 = '(500-inf)' AND
Dm3 = '(500-inf)': 1A (7040.0/6456.0)

Khoi = A AND
Dm1 = '(-inf-500]': 28 (222976.0/212090.0)

Khoi = B AND
Dm2 = '(-inf-500]' AND
```

Figure 4.15. Classification results from data in year 2009 with Region attribute

After getting all classification rules from three years[1], I analyze and summarize to get the general classification rules in the Table 4.5. Each year will have different rules. In addition, not all of rules were appeared together so I must compare these rules to get the main rules for three years.

Table 4.5. The general classification rules in three years (2009 to 2011)

No.	Classification Rules

1	Dm2 = '(500-inf)' AND Khoi = B AND
	Dm3 = '(500-inf)' AND Dm1 = '(500-inf)': Tinh = 2
	➢ ***IF Score1, Score2, Score3 are greater than 5 and Block is B THEN the region is 2 (Ho Chi Minh City).***

2	Khoi = C AND Dm1 = '(500-inf)': Tinh = 28 ➢ ***IF Block is C and Score1 is greater than 5 THEN the region is 28 (Thanh Hoa Province).***

3	Khoi = D1 AND Dm3 = '(500-inf)' AND
	Dm1 = '(500-inf)' AND Dm2 = '(500-inf)': Tinh = 1A
	➢ ***IF Block is D1 and Score1, Score2, Score3 are greater than 5 THEN the region is 1A (Ha Noi City).***

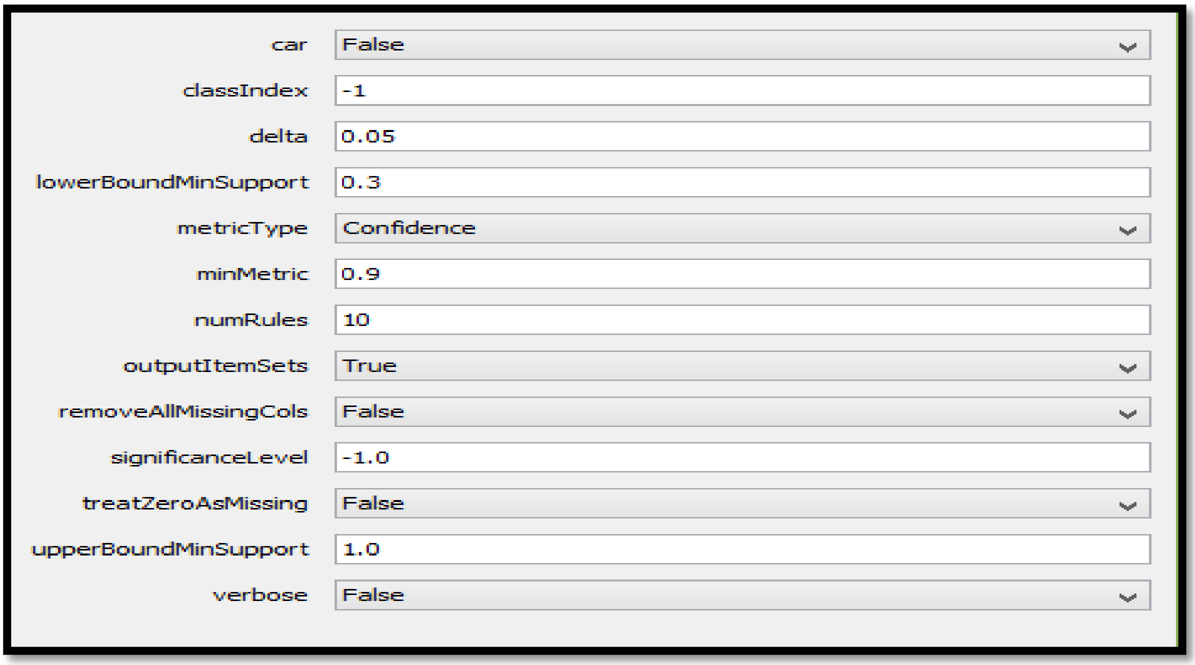
4	Dm2 = '(500-inf)' AND Khoi = A AND
	Dm1 = '(500-inf)' AND Dm3 = '(500-inf)': Tinh = 1A
	➢ ***IF Block is A and Score1, Score2, Score3 are greater than 5 THEN the region is 1A (Ha Noi City).***

5	Dm1 = '(500-inf)' AND Dm2 = '(500-inf)': Dm3 = '(500-inf)' ➢ ***IF Score1 and Score2 are greater than 5 THEN Score3 is greater than 5.***
6	Dm1 = '(500-inf)' AND Dm3 = '(500-inf)': Dm2 = '(500-inf)' ➢ ***IF Score1 and Score3 are greater than 5 THEN Score2 is greater than 5.***

[1] The full results of PART algorithm can see in APPENDICES of this thesis.

7	Dm3 = '(500-inf)' AND Dm2 = '(500-inf)': Dm1 = '(500-inf)' ➢ ***IF Score2 and Score3 are greater than 5 THEN Score1 is greater than 5.***
8	Dm1 = '(-inf-500]' AND Dm2 = '(-inf-500]': Dm3 = '(-inf-500]' ➢ ***IF Score1 and Score2 are less than or equal to 5 THEN Score3 is less than or equal to 5.***
9	Dm3 = '(-inf-500]' AND Dm1 = '(-inf-500]': Dm2 = '(-inf-500]' ➢ ***IF Score1 and Score3 are less than or equal to 5 THEN Score2 is less than or equal to 5.***
10	Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]': Dm1 = '(-inf-500]' ➢ ***IF Score2 and Score3 are less than or equal to 5 THEN Score1 is less than or equal to 5.***
11	Khoi = C: Dm2 = '(-inf-500]' ➢ ***IF Block is C THEN Score2 is less than or equal to 5.***

Second, I apply Apriori algorithm to gel all association rules which are hidden in this database. The properties for Apriori algorithm are showed in Figure 4.16. From the top to bottom, there are many options for Apriori algorithm. First option is *car*, a true-false option, if it is true, class association rules are mined. The next option is *classIndex*, the default is -1 that means the class attribute is the last attribute. The *delta* option is used in reducing support iteratively. The lower bound for minimum support is defined in *lowerBoundMinSupport*. The type of metric is set by *metricType*, for example confidence is one metricType. Minimum metric score is defined in *minMetric*. Setting the number of rules is found by *numRules*. Another true-false option is *outputItemSets*, itemsets are shown or not. Setting true or false for *removeAllMissingCols* will recognize remove or not all missing values. The next option is *significanceLevel* which is used in Significance test (confidence metric only). The option *treatZeroAsMissing* is used to set missing value by zero. The upper bound for minimum support is defined in *upperBoundMinSupport*. The final option is verbose; if it is true, algorithm will be run in verbose mode.



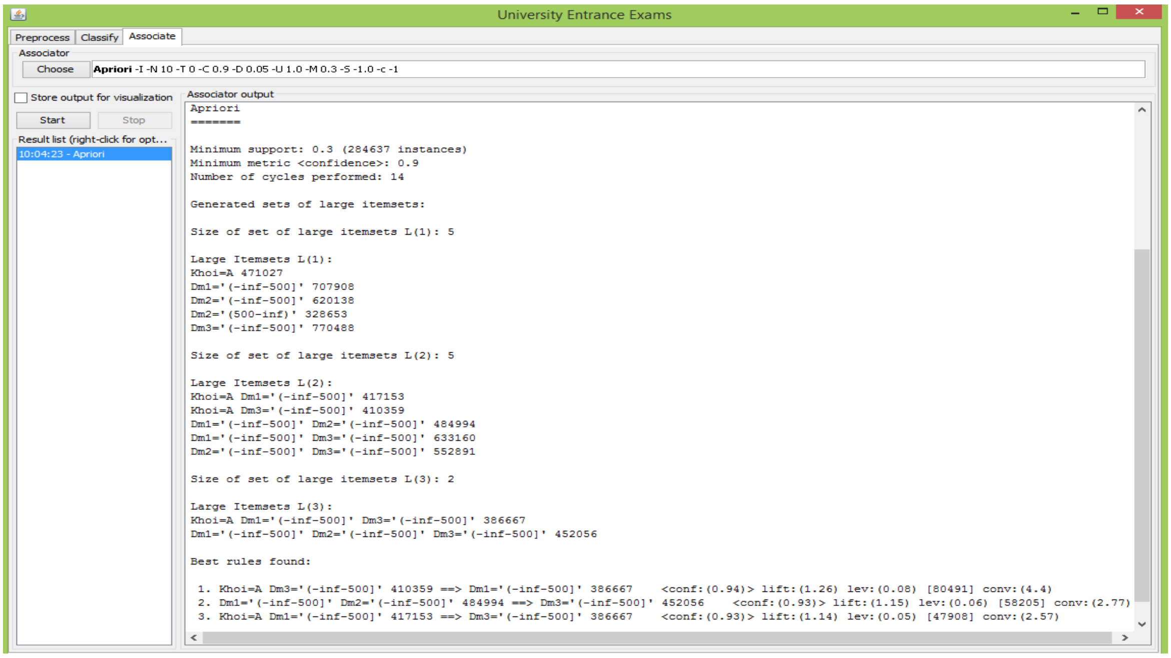
car	False
classIndex	-1
delta	0.05
lowerBoundMinSupport	0.3
metricType	Confidence
minMetric	0.9
numRules	10
outputItemSets	True
removeAllMissingCols	False
significanceLevel	-1.0
treatZeroAsMissing	False
upperBoundMinSupport	1.0
verbose	False

Figure 4.16. The properties of Apriori algorithm

Each year has several association rules so I must compare and analyze the results to find the main rules for three years. The general association rules is described in Table 4.6. The Figure 4.17 is the association rules from data in year 2009.

Table 4.6. The general association rules in three years (2009 to 2011)

No.	Association Rules
1	Dm1='(-inf-500]' Dm2='(-inf-500]' ==> Dm3='(-inf-500]' ➢ ***If Score1 and Score2 are less than or equal to 5 together, Score3 is also less than or equal to 5.***
2	Khoi=A Dm1='(-inf-500]' ==> Dm3='(-inf-500]' ➢ ***If Block is equal to A and Score1 is less than or equal to 5 together, Score3 is also less than or equal to 5.***


University Entrance Exams
Preprocess
Classify
Associate
Associator
Choose
Store output for visualization
Start
Stop
Result list (right-click for opt...
10:04:23 - Apriori
Associator output

```
Apriori -I -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.3 -S -1.0 -c -1
```

```
Apriori
=======

Minimum support: 0.3 (284637 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Generated sets of large itemsets:

Size of set of large itemsets L(1): 5

Large Itemsets L(1):
Khoi=A 471027
Dm1='(-inf-500]' 707908
Dm2='(-inf-500]' 620138
Dm2='(500-inf)' 328653
Dm3='(-inf-500]' 770488

Size of set of large itemsets L(2): 5

Large Itemsets L(2):
Khoi=A Dm1='(-inf-500]' 417153
Khoi=A Dm3='(-inf-500]' 410359
Dm1='(-inf-500]' Dm2='(-inf-500]' 484994
Dm1='(-inf-500]' Dm3='(-inf-500]' 633160
Dm2='(-inf-500]' Dm3='(-inf-500]' 552891

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

Large Itemsets L(3):
Khoi=A Dm1='(-inf-500]' Dm3='(-inf-500]' 386667
Dm1='(-inf-500]' Dm2='(-inf-500]' Dm3='(-inf-500]' 452056

Best rules found:

 1. Khoi=A Dm3='(-inf-500]' 410359 ==> Dm1='(-inf-500]' 386667    <conf:(0.94)> lift:(1.26) lev:(0.08) [80491] conv:(4.4)
 2. Dm1='(-inf-500]' Dm2='(-inf-500]' 484994 ==> Dm3='(-inf-500]' 452056    <conf:(0.93)> lift:(1.15) lev:(0.06) [58205] conv:(2.77)
 3. Khoi=A Dm1='(-inf-500]' 417153 ==> Dm3='(-inf-500]' 386667    <conf:(0.93)> lift:(1.14) lev:(0.05) [47908] conv:(2.57)
```

Figure 4.17. Association rules from data in year 2009

The meaning of each format in each result is different. With PART algorithm, the format when I classify by "Score3" is:

"*Dm1='(-inf-500]' AND Dm2='(-inf-500]' : '(-inf-500]' (484994.0/32938.0)*"

That means "***If score1 is in interval [0, 5] and score2 is in interval [0, 5] then score3 is in interval [0, 5], with 32938 instances satisfy this rule in 484994 total number of this occurred condition.***".

This format is divided into two main parts "*IF A THEN B*" with a/b property:

- In the left of colon: The condition of a rule, the IF clause (A).
- In the right of colon: The conclusion of this rule, the THEN clause (B).
- The number final (a/b): the total number of IF condition is occurred (a) / the total number of this rule is occurred (b).

In another case, the format of Apriori algorithm is:

"*Khoi=A Dm2='(-inf-500]' 504589 ==> Dm3='(-inf-500]' 485623*
<conf:(0.96)>"

That means "***A student chooses block A and has score2 in interval [0, 5] together so he/she also has score3 in interval [0, 5], with 485623 instances satisfy this rule in 504589 instances of left side and confidence is 0.96 > 0.9***".

This format is divided into two main parts "$A \Rightarrow B$" with some properties of this rule:

- A: is antecedent of this rule (in the left of the right double arrow).
- B: is consequent of this rule (in the right of the right double arrow).
- Some properties: the number of instances satisfy this rule, confidence of this rule and so on.

4.2.4 Explanation for the Results

From these rules, I specify several conclusions about University Entrance Exams.

- Rule ***"IF Score1, Score2, Score3 are greater than 5 and Block is B THEN the region is 2 (Ho Chi Minh City)."***

  Rule ***"IF Block is D1 and Score1, Score2, Score3 are greater than 5 THEN the region is 1A (Ha Noi City)."***

  Rule ***"IF Block is A and Score1, Score2, Score3 are greater than 5 THEN the region is 1A (Ha Noi City)."***

  Ho Chi Minh City and Ha Noi City are two biggest cities of our country so they have several reasons to attract a lot of students to attend the University Entrance Exams. The first reason is that these cities have many large and famous universities in our country. The other reason is a lot of opportunities for students that means a lot of competitions, good students from other cities, provinces attend the University Entrance Exams so the scores of the main blocks such as "A", "B", "D1" are higher than other places. This is the explanation for rule 1, 3 and 4 of classification rules above and the explanation for higer benchmark at these cities.

On the other hand, I can see that the quality of education system between regions is not equal so it lets a place is too much students while another places do not have any students. Therefore, these rules support for the decisions in improving and balancing the quality between regions.

- Rule “***IF Block is C and Score1 is greater than 5 THEN the region is 28 (Thanh Hoa Province).***”

  In recent years, we are nervous about block C because students just chose blocks A, B or D for development. From the second classification rule, I can see that students from Thanh Hoa province choose block C a lot and get good score in Literature (greater than five). Therefore, this rule is a point for government to concentrate in Thanh Hoa to develop social studies. In addition, other schools should learn the ways that Thanh Hoa schools teach Literature.

- Rule “***IF Score1 and Score2 are greater than 5 THEN Score3 is greater than 5.***”

  Rule “***IF Score1 and Score3 are greater than 5 THEN Score2 is greater than 5.***”

  Rule “***IF Score2 and Score3 are greater than 5 THEN Score1 is greater than 5.***”

  Rule “***IF Score1 and Score2 are less than 5 THEN Score3 is less than 5.***”

  Rule “***IF Score1 and Score3 are less than 5 THEN Score2 is less than 5.***”

  Rule “***IF Score2 and Score3 are less than 5 THEN Score1 is less than 5.***”

  The meaning of six classification rules above is that if two of three subjects are good then the result of remaining subject is good, in contrast, if two of three subjects are bad then the result of remaining subject is bad, too. This may be because of people’ s abilities, for example, if he/she is good in natural subjects then Math, Physics and Chemistry are good or if he/she is bad in social subjects then History, Literature are bad. This is just the almost situations because some people are also good in all natural and social subjects.

- Rule ***"IF Block is C THEN Score2 is less than or equal to 5."***

The last classification rule shows a real and serious situation that is the History's score is not good. In all three years, the History's score is usually less than average score (five score). We should consider why students do not get high score in this subject. The reasons may be the exam questions are too long so they cannot finish or the History theory is too much so they cannot learn all of them or they are boring in study history and so on. We must consider seriously and find the ways to improve this situation such as changing the style of exam questions or changing the teaching methods.

- Rule ***"If Score1 and Score2 are less than or equal to 5 together, Score3 is also less than or equal to 5."***

The first association rule tells us the relationship between three subjects in each block. As explaining in six classification rules above, if two first subjects are good then the last subject is also good but if two first subjects are bad then the last subject is also bad. According to this rule, we can predict the result of a student is good or bad.

- Rule ***"If Block is equal to A and Score1 is less than or equal to 5 together, Score3 is also less than or equal to 5."***

The second association rule tells us the relationship between Mathematics and Chemistry. That means if a student is not good in Math so he/she is also not good in Chemistry. This rule may be explained by the similar properties of two subjects, Mathematics and Chemistry. Chemistry requires students must be flexible in calculate and this is the requirement of a good Mathematics person.

In summary, some rules from my application are easy to see, the relationships between the results of subjects, but some of them are also strange, the bad score in History or students at Thanh Hoa Province take care of social studies a lot. These rules present for data mining that with a large database, we can get familiar rules or strange rules with scientific techniques. Moreover, these rules can help the government or universities to get the advantages or disadvantages of our education systems, then they propose policies to improve it better.

Chapter 5: Conclusions and Future Work

5.1 Conclusions

In summary, Association Rule Mining and Classification help me to get some important rules, relationships between regions, blocks and exam results. These rules will help The Ministry of Education or universities to make valuable decisions in University Entrance Exams, for example thinking about the way to make good exam questions. Besides, they can also use these results to improve the quality of education systems. For example, they will change the teaching methods for some social subjects or find a good student depend on his/her results. Furthermore, they will propose just one policy but it will have good influence on many aspects of education.

5.2 Future Work

In future, I can apply my research for another branch of education, it is College, to get the rules and relationship between different components of College Entrance Exams. Then, I can extract the general rules for all exams, University Entrance Exam and College Entrance Exams.

Furthermore, I will improve my application for multiple file format and solve the limit problem of database size (because Excel just can open a limited size). I will divide my database into more specific intervals to find effective rules more exactly.

REFERENCES

[1] D. Phuc, Khai thác dữ liệu, Đại Học Quốc Gia - Thành Phố Hồ Chí Minh, 2006.

[2] I. H. Witten, E. Frank and M. A. Hall, Data Mining - Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2011.

[3] J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2006.

[4] J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," in *Proc. of 1995 Int'l Conf. on Very Large Data Bases (VLDB'95),* Zürich, Switzerland: ABC, September 1995.

[5] N. T. T. Thuy, "Ứng dụng khai phá dữ liệu xây dựng công cụ dự đoán kết quả học tập của sinh viên," in *Hội Nghị Sinh Viên Nghiên Cứu Khoa Học lần thứ 8*, Đà Nẵng, 2012.

[6] O. Oladipupo and O. Oyelade, "Knowledge Discovery from Students' Result Repository: Association Rule Mining Approach," *International Journal of Computer Science & Security (IJCSS),* vol. 4, no. 2, pp. 199-207, 2014.

[7] P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, Addison-Wesley, 2005.

APPENDICES

- The full results from PART algorithm in year 2009 are showed in five tables below:

Results from PART algorithm with Block attribute in year 2009

	Block
Results	<u>*10 rules*</u> - Dm2 = '(500-inf)': A (328653.0/83538.0) - Dm3 = '(500-inf)' AND Tinh = 2: D1 (4185.0/1943.0) - Dm3 = '(500-inf)' AND Tinh = 28: C (3454.0/1393.0) - Dm1 = '(500-inf)' AND Tinh = 1A: D1 (6989.0/2707.0) - Dm3 = '(500-inf)' AND Tinh = 29: C (3190.0/1287.0) - Dm1 = '(-inf-500]' AND Dm3 = '(-inf-500]': A (452056.0/231458.0) - Dm3 = '(-inf-500]': B (95890.0/37949.0) - Tinh = 1B: C (2888.0/1439.0) - Dm1 = '(-inf-500]': C (26283.0/11409.0) - : B (25203.0/13935.0)

Results from PART algorithm with Region attribute in year 2009

	Region
Results	<u>*15 rules*</u> - Khoi = D1 AND Dm1 = '(-inf-500]': 2 (92671.0/80919.0) - Khoi = A AND Dm1 = '(-inf-500]' AND Dm2 = '(500-inf)' AND - Dm3 = '(-inf-500]': 2 (166069.0/153829.0) - Dm2 = '(500-inf)' AND Khoi = A: 1A (79046.0/72812.0) - Khoi = D1: 1A (47176.0/39797.0) - Khoi = C AND Dm1 = '(-inf-500]' AND Dm3 = '(-inf-500]': 28 (35173.0/32737.0) - Khoi = C AND Dm1 = '(500-inf)': 28 (31785.0/28534.0) - Dm2 = '(500-inf)' AND Khoi = B AND Dm1 = '(500-inf)' AND Dm3 = '(500-inf)': 2 (24886.0/23125.0) - Khoi = C: 29 (21665.0/20280.0) - Dm2 = '(500-inf)' AND Dm3 = '(-inf-500]' AND Dm1 = '(500-inf)': 25 (8483.0/7930.0) - Dm2 = '(500-inf)' AND Dm3 = '(500-inf)': 1A (7040.0/6456.0) - Khoi = A AND Dm1 = '(-inf-500]': 28 (222976.0/212090.0) - Khoi = B AND Dm2 = '(-inf-500]' AND Dm3 = '(500-inf)' AND

	Dm1 = '(500-inf)': 2 (14265.0/13285.0) - Khoi = B AND Dm2 = '(-inf-500]' AND Dm3 = '(-inf-500]': 29 (179589.0/169510.0) - Khoi = B: 28 (12609.0/11694.0) - : 1A (5356.0/4785.0)

Results from PART algorithm with Score1 attribute in year 2009

	Score1
Results	*34 rules* - Dm3 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (410359.0/23692.0) - Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]': '(-inf-500]' (329826.0/98368.0) - Khoi = B: '(500-inf)' (66553.0/18940.0) - Tinh = 1A: '(500-inf)' (11810.0/4134.0) - Tinh = 28: '(500-inf)' (7944.0/3265.0) - Tinh = 1B: '(500-inf)' (6518.0/2641.0) - Tinh = 25: '(500-inf)' (5495.0/2149.0) - Tinh = 2 AND Khoi = D1: '(-inf-500]' (5478.0/1971.0) - Tinh = 3: '(500-inf)' (4963.0/2263.0) - Tinh = 26: '(500-inf)' (4617.0/1903.0) - Tinh = 21: '(500-inf)' (4423.0/1516.0) - Tinh = 29 AND Dm2 = '(500-inf)': '(500-inf)' (4395.0/1826.0) - Tinh = 2: '(-inf-500]' (4426.0/2155.0) - Tinh = 48: '(-inf-500]' (4311.0/1331.0) - Tinh = 30: '(-inf-500]' (3796.0/1807.0) - Tinh = 22: '(500-inf)' (3344.0/1218.0) - Tinh = 18: '(500-inf)' (3249.0/1333.0) - Tinh = 19: '(500-inf)' (3155.0/1246.0) - Tinh = 33: '(-inf-500]' (2965.0/1047.0) - Tinh = 34: '(-inf-500]' (2797.0/1003.0) - Tinh = 40: '(-inf-500]' (2705.0/849.0) - Tinh = 15: '(500-inf)' (2606.0/1033.0) - Tinh = 4: '(-inf-500]' (2542.0/897.0) - Tinh = 16: '(500-inf)' (2348.0/935.0) - Tinh = 29: '(-inf-500]' (2290.0/976.0) - Tinh = 24: '(500-inf)' (2177.0/761.0) - Tinh = 37: '(-inf-500]' (2142.0/888.0) - Tinh = 27: '(500-inf)' (2120.0/845.0) - Tinh = 35: '(-inf-500]' (2089.0/865.0) - Tinh = 42: '(-inf-500]' (2081.0/851.0) - Dm3 = '(500-inf)' AND Dm2 = '(-inf-500]': '(-inf-500]'

	(14343.0/5158.0) - Dm3 = '(500-inf)' AND Khoi = A: '(-inf-500]' (11620.0/5125.0) - Dm3 = '(500-inf)': '(500-inf)' (5351.0/2433.0) - : '(-inf-500]' (3953.0/1273.0)

Results from PART algorithm with Score2 attribute in year 2009

	Score2
Results	*8 rules* - Dm3 = '(-inf-500]' AND Khoi = B: '(-inf-500]' (193768.0/14178.0) - Khoi = D1 AND Dm3 = '(-inf-500]': '(-inf-500]' (118845.0/13843.0) - Khoi = C: '(-inf-500]' (88624.0/13346.0) - Dm1 = '(500-inf)' AND Khoi = A: '(500-inf)' (53874.0/2936.0) - Dm3 = '(500-inf)' AND Khoi = B: '(500-inf)' (52375.0/21158.0) - Dm3 = '(500-inf)' AND Khoi = A: '(500-inf)' (30486.0/2378.0) - Tinh = 2 AND Khoi = A: '(500-inf)' (22056.0/9816.0) - : '(-inf-500]' (388763.0/164783.0)

Results from PART algorithm with Score3 attribute in year 2009

	Score3
Results	*9 rules* - Dm1 = '(-inf-500]' AND Dm2 = '(-inf-500]': '(-inf-500]' (484994.0/32938.0) - Dm1 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (194177.0/28108.0) - Dm2 = '(-inf-500]' AND Khoi = B: '(-inf-500]' (73328.0/14265.0) - Dm2 = '(-inf-500]' AND Khoi = D1: '(-inf-500]' (34273.0/5726.0) - Khoi = B: '(500-inf)' (45395.0/14178.0) - Khoi = C: '(500-inf)' (37090.0/12777.0) - Dm1 = '(500-inf)' AND Dm2 = '(500-inf)': '(500-inf)' (64350.0/27161.0) - Khoi = D1: '(-inf-500]' (11136.0/3208.0) - : '(-inf-500]' (4048.0/1339.0)

- The full results from PART algorithm in year 2010 are showed in five tables below:

Results from PART algorithm with Block attribute in year 2010

	Block
Results	*87 rules* - Dm1 = '(-inf-500]' AND Tinh = 25 AND Dm2 = '(-inf-500]': A (17379.0/5341.0) - Dm1 = '(-inf-500]' AND Tinh = 26 AND Dm2 = '(-inf-500]': A (17741.0/5788.0) - Dm1 = '(-inf-500]' AND Tinh = 21 AND Dm2 = '(-inf-500]': A (14219.0/5129.0) - Tinh = 29 AND Dm3 = '(-inf-500]' AND Dm1 = '(-inf-500]' AND Dm2 = '(-inf-500]': A (31483.0/10798.0) - Tinh = 35: A (22746.0/9082.0) - Tinh = 3 AND Dm1 = '(-inf-500]': A (18857.0/6908.0) - Tinh = 28 AND Dm1 = '(-inf-500]' AND Dm2 = '(-inf-500]': A (33271.0/11396.0) - Tinh = 1A AND Dm1 = '(-inf-500]': A (25779.0/9916.0) - Dm1 = '(-inf-500]' AND Tinh = 18 AND Dm2 = '(-inf-500]': A (13742.0/5165.0) - Dm1 = '(-inf-500]' AND Tinh = 1B AND Dm2 = '(-inf-500]': A (21206.0/7553.0) - Dm1 = '(-inf-500]' AND Tinh = 12: A (12952.0/5581.0) - Tinh = 2 AND Dm3 = '(-inf-500]': A (53219.0/22942.0) - Dm1 = '(-inf-500]' AND Tinh = 17: A (10044.0/3450.0) - Dm1 = '(-inf-500]' AND Tinh = 52: A (10025.0/4165.0) - Dm1 = '(-inf-500]' AND Tinh = 24: A (9293.0/3386.0) - Dm1 = '(-inf-500]' AND Tinh = 27: A (8821.0/3361.0) - Dm1 = '(-inf-500]' AND Tinh = 22 AND Dm2 = '(-inf-500]': A (8554.0/3369.0) - Dm1 = '(-inf-500]' AND Tinh = 19 AND Dm2 = '(-inf-500]': A (9203.0/3555.0) - Dm1 = '(-inf-500]' AND Tinh = 37 AND Dm2 = '(-inf-500]': A (27221.0/11360.0) - Dm1 = '(-inf-500]' AND Tinh = 48: A (23565.0/9939.0) - Dm1 = '(-inf-500]' AND Tinh = 42: A (16757.0/7413.0) - Dm1 = '(-inf-500]' AND Tinh = 30 AND Dm2 = '(-inf-500]': A (15271.0/6672.0) - Dm1 = '(-inf-500]' AND Tinh = 40: A (25639.0/12335.0) - Dm1 = '(-inf-500]' AND Tinh = 34: A (22043.0/9995.0) - Dm1 = '(-inf-500]' AND Tinh = 47: A (14641.0/6746.0)

	- Dm1 = '(-inf-500]' AND Tinh = 41: A (14569.0/6409.0) - Dm1 = '(-inf-500]' AND Tinh = 51: A (13369.0/6725.0) - Dm1 = '(-inf-500]' AND Tinh = 49: A (13025.0/5729.0) - Dm1 = '(-inf-500]' AND Tinh = 38: A (12721.0/5857.0) - Dm1 = '(-inf-500]' AND Tinh = 53: A (12647.0/5394.0) - Dm1 = '(-inf-500]' AND Tinh = 50: A (11692.0/5492.0) - Dm1 = '(-inf-500]' AND Tinh = 56: A (10534.0/4738.0) - Dm1 = '(-inf-500]' AND Tinh = 15: A (9859.0/4265.0) - Dm1 = '(-inf-500]' AND Tinh = 4: A (9329.0/4484.0) - Dm1 = '(-inf-500]' AND Tinh = 55: A (9255.0/4246.0) - Dm1 = '(-inf-500]' AND Tinh = 43: A (8990.0/3711.0) - Dm1 = '(-inf-500]' AND Tinh = 16: A (8811.0/3566.0) - Dm1 = '(-inf-500]' AND Tinh = 54: A (8551.0/3847.0) - Dm1 = '(-inf-500]' AND Tinh = 57: A (8319.0/3433.0) - Dm1 = '(-inf-500]' AND Tinh = 46: A (7278.0/3307.0) - Dm1 = '(-inf-500]' AND Tinh = 10: A (6855.0/3350.0) - Dm1 = '(-inf-500]' AND Tinh = 44: A (6544.0/2922.0) - Dm1 = '(-inf-500]' AND Tinh = 61: A (6304.0/2882.0) - Dm1 = '(-inf-500]' AND Tinh = 63: A (5971.0/2896.0) - Dm1 = '(-inf-500]' AND Tinh = 58: A (5863.0/2682.0) - Dm1 = '(-inf-500]' AND Tinh = 45: A (5710.0/2837.0) - Dm1 = '(-inf-500]' AND Tinh = 14: A (5476.0/2813.0) - Dm1 = '(-inf-500]' AND Tinh = 64: A (4998.0/2416.0) - Dm1 = '(-inf-500]' AND Tinh = 9: A (4898.0/2337.0) - Dm2 = '(500-inf)' AND Dm1 = '(500-inf)': A (118726.0/44774.0) - Dm1 = '(-inf-500]' AND Tinh = 6: A (4775.0/2510.0) - Dm1 = '(-inf-500]' AND Tinh = 28: B (4746.0/2362.0) - Dm1 = '(-inf-500]' AND Tinh = 13: A (4278.0/1928.0) - Dm1 = '(-inf-500]' AND Tinh = 25: B (4208.0/1887.0) - Dm1 = '(-inf-500]' AND Tinh = 1B: B (4132.0/2207.0) - Dm1 = '(-inf-500]' AND Tinh = 26: B (3834.0/1616.0) - Dm1 = '(-inf-500]' AND Tinh = 29 AND Dm3 = '(-inf-500]': B (3487.0/1749.0) - Dm1 = '(-inf-500]' AND Tinh = 33 AND Dm2 = '(-inf-500]': A (16670.0/8694.0) - Dm1 = '(-inf-500]' AND Tinh = 31: A (13955.0/7097.0) - Dm1 = '(-inf-500]' AND Tinh = 32: A (10958.0/5865.0) - Dm1 = '(-inf-500]' AND Tinh = 39: A (9998.0/4743.0) - Dm1 = '(-inf-500]' AND Tinh = 59: A (6247.0/3053.0) - Dm1 = '(-inf-500]' AND Tinh = 23: A (4552.0/2387.0) - Dm1 = '(-inf-500]' AND Tinh = 2: D1 (3964.0/2074.0) - Dm1 = '(-inf-500]' AND Tinh = 60: A (3864.0/1924.0) - Dm1 = '(-inf-500]' AND Tinh = 8: A (3422.0/1533.0)

	- Dm1 = '(-inf-500]' AND Tinh = 29: C (3402.0/2266.0) - Dm1 = '(-inf-500]' AND Dm2 = '(500-inf)': B (17103.0/8675.0) - Tinh = 1A: D1 (7095.0/3495.0) - Tinh = 28: A (6753.0/4265.0) - Tinh = 1B: D1 (4705.0/3008.0) - Tinh = 29: A (4387.0/2308.0) - Tinh = 25: A (4384.0/1984.0) - Tinh = 26: A (3698.0/1918.0) - Tinh = 3: D1 (3318.0/1705.0) - Tinh = 36: A (3269.0/1474.0) - Tinh = 5: A (3208.0/1566.0) - Tinh = 21: A (2848.0/1598.0) - Tinh = 11: A (2783.0/1588.0) - Tinh = 18: A (2743.0/1777.0) - Tinh = 62: A (2615.0/1318.0) - Tinh = 48: D1 (2603.0/1352.0) - Tinh = 37: A (2594.0/1615.0) - Tinh = 19: A (2130.0/1230.0) - Tinh = 24: A (2042.0/1260.0) - Dm3 = '(-inf-500]': A (37950.0/23502.0) - : C (11890.0/7752.0)

Results from PART algorithm with Region attribute in year 2010

	Region
Results	<u>*18 rules*</u> - Khoi = D1 AND Dm1 = '(500-inf)' AND Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]': 2 (30308.0/27651.0) - Khoi = D1 AND Dm1 = '(-inf-500]': 2 (95635.0/84566.0) - Dm2 = '(500-inf)' AND Khoi = A AND Dm1 = '(500-inf)' AND Dm3 = '(500-inf)': 1A (52055.0/48227.0) - Khoi = A AND Dm1 = '(500-inf)': 2 (66652.0/61117.0) - Khoi = A AND Dm2 = '(500-inf)' AND Dm3 = '(-inf-500]': 2 (27060.0/25549.0) - Khoi = A AND Dm2 = '(-inf-500]': 2 (422346.0/397972.0) - Khoi = D1: 1A (23446.0/19039.0) - Dm2 = '(500-inf)' AND Khoi = B AND Dm3 = '(500-inf)' AND Dm1 = '(500-inf)': 2 (13825.0/12820.0) - Dm2 = '(500-inf)' AND Khoi = B AND Dm3 = '(-inf-500]' AND Dm1 = '(-inf-500]': 29 (27483.0/25745.0) - Khoi = C AND Dm1 = '(-inf-500]' AND Dm3 = '(-inf-500]': 28 (44012.0/41969.0)

	- Khoi = C AND Dm1 = '(500-inf)': 28 (22939.0/20578.0) - Khoi = B AND Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]' AND Dm1 = '(-inf-500]': 2 (156323.0/149404.0) - Khoi = B AND Dm1 = '(-inf-500]': 1B (17354.0/16251.0) - Khoi = A: 48 (13245.0/12301.0) - Khoi = C: 29 (13184.0/12048.0) - Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]': 28 (10477.0/9951.0) - Dm3 = '(-inf-500]': 2 (8386.0/7816.0) - : 1A (3845.0/3408.0)

Results from PART algorithm with Score1 attribute in year 2010

	Score1
Results	26 rules - Dm3 = '(-inf-500]' AND Dm2 = '(-inf-500]': '(-inf-500]' (775995.0/83483.0) - Dm3 = '(500-inf)' AND Dm2 = '(500-inf)': '(500-inf)' (114673.0/33300.0) - Khoi = B: '(-inf-500]' (41404.0/10323.0) - Tinh = 1A: '(500-inf)' (7377.0/2851.0) - Tinh = 29: '(-inf-500]' (5792.0/2604.0) - Tinh = 28: '(500-inf)' (5710.0/2499.0) - Tinh = 1B: '(500-inf)' (4718.0/2151.0) - Tinh = 2 AND Khoi = A: '(500-inf)' (4567.0/1988.0) - Tinh = 2: '(-inf-500]' (4438.0/1923.0) - Tinh = 25: '(500-inf)' (4219.0/1742.0) - Tinh = 3: '(500-inf)' (3735.0/1795.0) - Tinh = 26: '(500-inf)' (3630.0/1554.0) - Tinh = 48: '(-inf-500]' (3528.0/1306.0) - Tinh = 21: '(500-inf)' (3104.0/1339.0) - Tinh = 34: '(-inf-500]' (2854.0/1051.0) - Tinh = 30: '(-inf-500]' (2649.0/1121.0) - Tinh = 37: '(-inf-500]' (2565.0/1027.0) - Tinh = 40: '(-inf-500]' (2542.0/925.0) - Tinh = 33: '(-inf-500]' (2418.0/845.0) - Tinh = 18: '(500-inf)' (2416.0/1152.0) - Tinh = 19: '(500-inf)' (2295.0/949.0) - Tinh = 22: '(500-inf)' (2059.0/887.0) - Khoi = A: '(-inf-500]' (25695.0/11281.0) - Khoi = C: '(-inf-500]' (10474.0/3962.0) - Dm2 = '(500-inf)': '(-inf-500]' (6171.0/2658.0)

	- : '(500-inf)' (3547.0/1504.0)

Results from PART algorithm with Score2 attribute in year 2010

	Score2
Results	*7 rules* - Dm3 = '(-inf-500]': '(-inf-500]' (883464.0/107469.0) - Khoi = A: '(500-inf)' (81568.0/16268.0) - Khoi = B: '(500-inf)' (33154.0/5573.0) - Khoi = C: '(-inf-500]' (26309.0/8374.0) - Tinh = 2: '(500-inf)' (4475.0/1924.0) - Dm1 = '(500-inf)': '(500-inf)' (14180.0/5670.0) - : '(-inf-500]' (5425.0/2357.0)

Results from PART algorithm with Score3 attribute in year 2010

	Score3
Results	*13 rules* - Dm2 = '(-inf-500]' AND Dm1 = '(-inf-500]': '(-inf-500]' (719680.0/27168.0) - Dm2 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (41416.0/7070.0) - Dm2 = '(-inf-500]' AND Khoi = D1: '(-inf-500]' (36307.0/5999.0) - Dm1 = '(-inf-500]' AND Khoi = B: '(-inf-500]' (41239.0/13756.0) - Dm1 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (40305.0/13245.0) - Dm2 = '(-inf-500]' AND Khoi = C: '(-inf-500]' (16466.0/7666.0) - Dm2 = '(500-inf)' AND Dm1 = '(500-inf)' AND Khoi = A: '(500-inf)' (77291.0/25236.0) - Dm2 = '(500-inf)' AND Dm1 = '(500-inf)' AND Khoi = B: '(500-inf)' (22173.0/8348.0) - Khoi = D1 AND Dm1 = '(-inf-500]': '(-inf-500]' (11866.0/3216.0) - Dm2 = '(-inf-500]': '(-inf-500]' (12564.0/2535.0) - Khoi = C: '(500-inf)' (10923.0/2549.0) - Tinh = 1A: '(500-inf)' (3640.0/1259.0) - : '(500-inf)' (14705.0/6884.0)

- The full results from PART algorithm in year 2011 are showed in five tables below:

Results from PART algorithm with Block attribute in year 2011

	Block
Results	*11 rules* - Dm1 = '(-inf-500]' AND Tinh = 2 AND Dm3 = '(-inf-500]': A (52011.0/17263.0) - Dm1 = '(-inf-500]': A (756040.0/264345.0) - Dm2 = '(500-inf)': A (102703.0/45821.0) - Tinh = 2: D1 (9311.0/4761.0) - Tinh = 1A: D1 (7199.0/3234.0) - Dm3 = '(-inf-500]' AND Tinh = 28: B (5457.0/3783.0) - Dm3 = '(-inf-500]' AND Tinh = 37: B (4303.0/1918.0) - Dm3 = '(-inf-500]' AND Tinh = 1B: D1 (4006.0/2588.0) - Dm3 = '(-inf-500]' AND Tinh = 25: D1 (3443.0/2177.0) - Dm3 = '(-inf-500]': B (81066.0/41089.0) - : C (23036.0/12615.0)

Results from PART algorithm with Region attribute in year 2011

	Region
Results	*17 rules* - Khoi = D1 AND Dm3 = '(500-inf)' AND Dm1 = '(500-inf)' AND Dm2 = '(500-inf)': 1A (10180.0/8029.0) - Khoi = D1 AND Dm3 = '(-inf-500]' AND Dm1 = '(500-inf)' AND Dm2 = '(-inf-500]': 2 (32538.0/29383.0) - Khoi = D1 AND Dm3 = '(-inf-500]' AND Dm1 = '(-inf-500]': 2 (114999.0/101584.0) - Khoi = D1 AND Dm1 = '(-inf-500]': 2 (9420.0/6378.0) - Khoi = D1 AND Dm2 = '(-inf-500]': 2 (7307.0/5912.0) - Dm2 = '(500-inf)' AND Khoi = A: 2 (102092.0/93565.0) - Khoi = A AND Dm1 = '(500-inf)' AND Dm3 = '(-inf-500]': 1A (17487.0/15924.0) - Khoi = A: 2 (487102.0/454085.0) - Dm2 = '(500-inf)' AND Khoi = B AND Dm1 = '(500-inf)': 2 (25284.0/23774.0) - Dm2 = '(500-inf)' AND Khoi = B: 29 (12250.0/11394.0) - Khoi = B AND Dm1 = '(-inf-500]': 28 (106950.0/100717.0) - Khoi = B AND Dm3 = '(-inf-500]': 2 (49371.0/46970.0)

	- Khoi = C AND Dm1 = '(-inf-500]' AND Dm3 = '(-inf-500]': 34 (30090.0/28476.0) - Khoi = C AND Dm1 = '(500-inf)': 28 (24535.0/22303.0) - Khoi = D1: 1A (6953.0/6029.0) - Khoi = C: 33 (6693.0/6260.0) - : 2 (5324.0/4794.0)

Results from PART algorithm with Score1 attribute in year 2011

	Score1
Results	<u>*8 rules*</u> - Dm3 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (533019.0/33499.0) - Dm2 = '(-inf-500]' AND Dm3 = '(-inf-500]': '(-inf-500]' (335419.0/93167.0) - Dm2 = '(500-inf)' AND Dm3 = '(500-inf)': '(500-inf)' (84545.0/20494.0) - Khoi = B: '(500-inf)' (29380.0/11272.0) - Khoi = A: '(-inf-500]' (18966.0/5869.0) - Tinh = 2: '(-inf-500]' (5908.0/2254.0) - Dm2 = '(-inf-500]': '(500-inf)' (27315.0/10341.0) - : '(-inf-500]' (14023.0/6602.0)

Results from PART algorithm with Score2 attribute in year 2011

	Score2
Results	<u>*9 rules*</u> - Dm3 = '(-inf-500]' AND Dm1 = '(-inf-500]': '(-inf-500]' (760456.0/50068.0) - Dm3 = '(-inf-500]' AND Khoi = B: '(-inf-500]' (64694.0/15323.0) - Khoi = D1 AND Dm3 = '(-inf-500]': '(-inf-500]' (39491.0/6953.0) - Khoi = A AND Dm3 = '(500-inf)': '(500-inf)' (73662.0/18966.0) - Khoi = C: '(-inf-500]' (31228.0/3105.0) - Khoi = B: '(500-inf)' (16620.0/4233.0) - Dm1 = '(500-inf)' AND Dm3 = '(500-inf)': '(500-inf)' (18635.0/7788.0) - Khoi = D1: '(-inf-500]' (9420.0/3670.0) - : '(-inf-500]' (34369.0/16216.0)

Results from PART algorithm with Score3 attribute in year 2011

	Score3
Results	*10 rules* - Dm2 = '(-inf-500]' AND Dm1 = '(-inf-500]': '(-inf-500]' (737489.0/27101.0) - Dm2 = '(-inf-500]' AND Khoi = B: '(-inf-500]' (52156.0/2785.0) - Dm2 = '(-inf-500]' AND Khoi = D1: '(-inf-500]' (39845.0/7307.0) - Dm1 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (45210.0/13826.0) - Dm1 = '(-inf-500]' AND Khoi = D1: '(-inf-500]' (12317.0/3670.0) - Dm1 = '(500-inf)' AND Dm2 = '(-inf-500]' AND Khoi = A: '(-inf-500]' (23356.0/5869.0) - Dm1 = '(500-inf)' AND Khoi = A: '(500-inf)' (56882.0/16012.0) - Dm1 = '(500-inf)' AND Khoi = B: '(-inf-500]' (25284.0/9961.0) - Dm1 = '(500-inf)': '(500-inf)' (43001.0/18575.0) - : '(-inf-500]' (13035.0/2998.0)

➢ The full results from Apriori Algorithm from year 2009 to year 2011 are showed below:

Apriori algorithm results in three years (from year 2009 to year 2011)

	Result
Dh2009	1. Khoi=A Dm3='(-inf-500]' 410359 ==> Dm1='(-inf-500]' 386667 <conf:(0.94)> lift:(1.26) lev:(0.08) [80491] conv:(4.4) 2. Dm1='(-inf-500]' Dm2='(-inf-500]' 484994 ==> Dm3='(-inf-500]' 452056 <conf:(0.93)> lift:(1.15) lev:(0.06) [58205] conv:(2.77) 3. Khoi=A Dm1='(-inf-500]' 417153 ==> Dm3='(-inf-500]' 386667 <conf:(0.93)> lift:(1.14) lev:(0.05) [47908] conv:(2.57)
Dh2010	1. Khoi=A Dm1='(-inf-500]' Dm2='(-inf-500]' 422346 ==> Dm3='(-inf-500]' 413148 <conf:(0.98)> lift:(1.16) lev:(0.05) [57305] conv:(7.23) 2. Khoi=A Dm2='(-inf-500]' 463762 ==> Dm3='(-inf-500]' 447494 <conf:(0.96)> lift:(1.15) lev:(0.05) [56757] conv:(4.49) 3. Dm1='(-inf-500]' Dm2='(-inf-500]' 719680 ==> Dm3='(-inf-500]' 692512 <conf:(0.96)> lift:(1.14) lev:(0.08) [86154] conv:(4.17) 4. Khoi=A Dm1='(-inf-500]' 462651 ==> Dm3='(-inf-500]' 440208 <conf:(0.95)> lift:(1.13) lev:(0.05) [50407] conv:(3.25) 5. Dm2='(-inf-500]' 826433 ==> Dm3='(-inf-500]' 775995 <conf:(0.94)> lift:(1.11) lev:(0.08) [79694] conv:(2.58) 6. Khoi=A Dm1='(-inf-500]' Dm3='(-inf-500]' 440208 ==> Dm2='(-inf-500]' 413148 <conf:(0.94)> lift:(1.19) lev:(0.06) [66198]

	conv:(3.45) 7. Dm1='(-inf-500]' 817727 ==> Dm3='(-inf-500]' 757259 <conf:(0.93)> lift:(1.1) lev:(0.07) [68293] conv:(2.13) 8. Khoi=A Dm2='(-inf-500]' Dm3='(-inf-500]' 447494 ==> Dm1='(-inf-500]' 413148 <conf:(0.92)> lift:(1.18) lev:(0.06) [64171] conv:(2.87) 9. Dm1='(-inf-500]' Dm3='(-inf-500]' 757259 ==> Dm2='(-inf-500]' 692512 <conf:(0.91)> lift:(1.16) lev:(0.09) [95679] conv:(2.48) 10. Khoi=A Dm1='(-inf-500]' 462651 ==> Dm2='(-inf-500]' 422346 <conf:(0.91)> lift:(1.16) lev:(0.06) [57708] conv:(2.43)
Dh2011	1. Dm1='(-inf-500]' Dm2='(-inf-500]' 737489 ==> Dm3='(-inf-500]' 710388 <conf:(0.96)> lift:(1.11) lev:(0.07) [70529] conv:(3.6) 2. Khoi=A Dm2='(-inf-500]' 504589 ==> Dm3='(-inf-500]' 485623 <conf:(0.96)> lift:(1.11) lev:(0.05) [47832] conv:(3.52) 3. Khoi=A Dm2='(-inf-500]' 504589 ==> Dm1='(-inf-500]' 481233 <conf:(0.95)> lift:(1.24) lev:(0.09) [92387] conv:(4.96) 4. Khoi=A Dm1='(-inf-500]' 526443 ==> Dm3='(-inf-500]' 499520 <conf:(0.95)> lift:(1.09) lev:(0.04) [42768] conv:(2.59) 5. Dm1='(-inf-500]' 808051 ==> Dm3='(-inf-500]' 760456 <conf:(0.94)> lift:(1.08) lev:(0.06) [59376] conv:(2.25) 6. Dm2='(-inf-500]' 875310 ==> Dm3='(-inf-500]' 821042 <conf:(0.94)> lift:(1.08) lev:(0.06) [61607] conv:(2.14) 7. Khoi=A Dm3='(-inf-500]' 533019 ==> Dm1='(-inf-500]' 499520 <conf:(0.94)> lift:(1.22) lev:(0.08) [88765] conv:(3.65) 8. Dm1='(-inf-500]' Dm3='(-inf-500]' 760456 ==> Dm2='(-inf-500]' 710388 <conf:(0.93)> lift:(1.12) lev:(0.07) [75588] conv:(2.51) 9. Khoi=A Dm1='(-inf-500]' 526443 ==> Dm2='(-inf-500]' 481233 <conf:(0.91)> lift:(1.1) lev:(0.04) [41778] conv:(1.92) 10. Dm1='(-inf-500]' 808051 ==> Dm2='(-inf-500]' 737489 <conf:(0.91)> lift:(1.09) lev:(0.06) [62959] conv:(1.89)