Data Guided Approach to Generate Multi-dimensional Schema for Targeted Knowledge Discovery

Muhammad Usman, Russel Pears, A.C.M. Fong
School of Computing and Mathematical Science
Auckland University of Technology
Auckland, New Zealand

muhammad.usman@aut.ac.nz, russel.pears@aut.ac.nz, alvis.fong@aut.ac.nz

Abstract

Data mining and data warehousing are two key technologies which have made significant contributions to the field of knowledge discovery in a variety of domains. More recently, the integrated use of traditional data mining techniques such as clustering and pattern recognition with data warehousing technique of Online Analytical Processing (OLAP) have motivated diverse research areas for leveraging knowledge discovery from complex real-world datasets. Recently, a number of such integrated methodologies have been proposed to extract knowledge from datasets but most of these methodologies lack automated and generic methods for schema generation and knowledge extraction. Mostly data analysts need to rely on domain specific knowledge and have to cope with technological constraints in order to discover knowledge from high dimensional datasets. In this paper we present a generic methodology which incorporates semi-automated knowledge extraction methods to provide data-driven assistance towards knowledge discovery. In particular, we provide a method for constructing a binary tree of hierarchical clusters and annotate each node in the tree with significant numeric variables. Additionally, we propose automated methods to rank nominal variables and to generate candidate multidimensional schema with highly significant dimensions. We have performed three case studies on three real-world datasets taken from the UCI machine learning repository in order to validate the generality and applicability of our proposed methodology.

Keywords: Data Mining, Data Warehousing, Schema Generation, Knowledge Discovery.

1 Introduction

Knowledge discovery from data is the result of an exploratory process involving the application of various algorithmic procedures for manipulating data (Bernstein et al., 2005). Data mining and warehousing are two key technologies for discovering knowledge from large datasets. Data mining permits targeted mining of large datasets in order to discover hidden trends, patterns and rules while data warehousing facilitates the interactive exploration and multidimensional analysis of summarized data. These two technologies are mature in their own right and have essentially the same set of objectives but little research has been carried out to seamlessly integrate the two. This is a challenging task as the techniques employed in each of the technologies are different.

In the past several years, a wide range of data mining techniques have made significant contributions to the field of knowledge discovery in a number of domains. In the banking sector, these techniques are used for loan payment prediction, customer credit policy analysis, classification of customers for targeted marketing, and the detection of money laundering schemes and other financial crimes. Similarly, in the retail industry, such techniques are used in the analysis of product sales and customer retention. In the telecommunication industry these techniques help in identifying and comparing data traffic, system workload, resource usage, profit and fraudulent pattern analysis (Han and Kamber, 2006).

Similarly, data warehousing has contributed extensively as a key technology for complex data analysis, decision support and automatic extraction of knowledge from huge data repositories (Nguyen et al., 2005). It provides analysts with a competitive advantage by providing relevant information to enhance strategic decision making. Moreover, warehousing has reduced costs by tracking trends, patterns, and exceptions over long periods in a consistent and reliable manner. Due to sophisticated analytical powers, these warehouse systems are being used broadly in many sectors such as financial services, consumer goods and retail, manufacturing, education, medical, media, and telecommunication.

The integrated use of data mining and warehousing techniques has been observed recently. A number of proposals (Fabris and Freitas, 2001, Goil and Choudhary, 2001, Messaoud et al., 2006, Missaoui et al., 2007, Pardillo et al., 2008, Usman and Asghar, 2011) emphasized the benefits of integrating these technologies and motivated diverse research directions in the area of knowledge discovery. However, it is a daunting task for data warehouse developers to integrate the outcomes of data mining techniques with data warehouse to perform analytical operations.

Data mining results need to be modelled in the form of a multidimensional schema to support interactive queries for the exploration of data. Multi-dimensional modelling is complex and requires extensive domain knowledge along with a familiarity with the data warehouse technologies.

Additionally, its techniques require multiple manual actions to discover measures and relevant dimensions from the dataset, creating a bottleneck in the knowledge discovery process. Even if the human data warehouse designer tries to resolve these problems, an incorrect design can still be generated if he/she doesn't understand the underlying relationships among the data items. In data warehouses, the choice of the dimensions and measures heavily influences the data warehouse effectiveness (Pighin and Ieronutti, 2008).

In prior research (Usman and Pears, 2010, Usman and Pears, 2011, Usman et al., 2011) methodologies have been developed in which the clustering and visualization techniques of data mining are integrated with the well-known multi-dimensional designing technique of data warehouse. This methodology constructs a renowned *STAR* schema through the use of hierarchical clustering and enhances the schema design by using the famous visualization technique known as multi-dimensional scaling. However, the design and implementation of generic methods is required for the process of knowledge discovery from this generated multidimensional schema. Moreover, the previously developed methodologies have two major limitations. Firstly, the methodologies need key extension for extracting more complex knowledge from the multidimensional schema. Secondly, it lacks automated support for data warehouse designers to identify more informative dimensions from high dimensional data to generate compact and informative schema for targeted knowledge discovery. This motivated us to formulate a generic methodology to provide data-driven assistance to a wide spectrum of users especially in those domains where very limited domain knowledge exists.

In this paper we make the following contributions in the field of integrating data mining with data warehousing. Firstly, we present a generic methodology for the generation of multidimensional schema by combining the benefits of hierarchical clustering and multi-dimensional scaling techniques of data mining discipline. Secondly, we provide an algorithm for constructing a binary tree from hierarchical clustering results (dendrogram). Thirdly, we identify the significant numeric variables that define the split of a cluster in the binary tree and rank them in order of significance. Fourthly, we rank nominal variables in each cluster by performing comparative analysis of similar variables distributed in neighbouring clusters (parent & sibling). Fifthly, we provide an algorithm to construct candidate schema with highly ranked dimensions (nominal variables) and measures (numeric variables). Finally, we highlight the significant interrelationships between the highly significant dimensions and measures present in the generated schema to be explored in an OLAP manner.

The rest of the paper is organized as follows. In the next section, we look at the previous work in the area of integrating data mining with warehousing. In section 3, we give an overview of the proposed methodology while section 4 presents the implementation details. The methodological steps are explained through an example in section 5. Our case study results are discussed in section 6. Section 7 gives a general comparison of proposed work with existing manual data analysis. Finally, we conclude the paper in section 8 with a summary of achievements and discussion of possible future research directions.

2 Related Work

In the last decade, many researchers have recognized the need for the integration of data mining with data warehousing (Kamber et al., 1997, Missaoui et al., 2007, Zubcoff et al., 2007, Goil and Choudhary, 2001). For instance, (Goil and Choudhary, 1997) identified that the decision making process requires complex operations on underlying data which can be very expensive in terms of computational time. They proposed an algorithm for the construction of data cubes on distributed memory parallel computers. In their approach they integrated OLAP and the Attribute Focusing (AF) technique, which relies on exploration and interpretation of attributes, of data mining. AF calculates associations between attributes using the notion of percentages and sub-populations. This integration allows interesting pattern mining on distributed data cubes.

A similar framework was proposed by the same authors that used the summary information present in the data cubes for association rule mining and decision tree classification (Goil and Choudhary, 1999). Data mining uses some pre-aggregating calculations to compute the probabilities needed for calculating support and confidence measures for association rules and the split point evaluation while building a classification tree (Goil and Choudhary, 2001). A limitation of these approaches is that they focused only on improving the OLAP query processing time by constructing the distributed data cubes. However, these distributed data cubes are unable to provide any automated assistance on which dimension and measures of the cubes should be explored in a high dimensional space for focused knowledge discovery.

Similarly, another system that implements a wide spectrum of data mining functions (association rule mining, clustering, etc.) called *DBMiner* has been proposed to support multiple mining functions (Han, 1998). The work focused on the efficiency and scalability of the data mining algorithms with the help of exploratory analysis provided by OLAP systems. The author emphasized that data mining should be performed interactively like an OLAP analysis and at different levels of data abstraction. This work has an obvious advantage of supporting multiple data mining functions. However, the author proposed mining to be applied on the pre-aggregated data cube, which fails to provide any assistance to the human data warehouse developers to build an appropriate schema in the first instance on which mining techniques can be applied later. Furthermore, the proposed system requires the analyst to possess knowledge of both data characteristics and the roles of data mining functions to select an appropriate mining algorithm. Analysts in various domains lack complete knowledge about the data and mining algorithms. It makes the application of such a proposal very difficult and identifies the need for an approach that can assist such analysts in the meaningful knowledge discovery. Similar to (Han, 1998), (Liu and Guo, 2001) put forward a new architecture for integrating data mining with OLAP. Yet again, the proposed architecture supported mining on the pre-aggregated data cubes built on top of manually constructed schema. However, the authors supported the idea of mining at different stages of the knowledge discovery process and at multiple levels of data abstraction. This approach is similar to our approach in terms of mining at multiple levels of data abstraction as we also produce hierarchical clusters at multiple levels of data abstraction for analysing information at multiple levels. However, the difference is that we not only apply mining at multiple levels of data abstraction but also provide a set of significant numeric and nominal variables and the strong relationships that exist between the two.

More recently, (Usman et al., 2010) proposed an enhanced architecture for combining mining and warehousing to provide OLAP performance enhancement and visualization improvement. However, the analyst relies on a few levels of data abstraction and manual discovery of useful data chunks provided by the neural network technique with no way of interactively exploring the original data variables of a particular cluster in order to extract useful knowledge. Each cluster and the cluster hierarchy were assumed to be the navigational space for cube data exploration. That limited search space and limited navigation was not adequate for meaningful knowledge discovery form large data cubes. These limitations led to the work done by (Usman and Pears, 2010) to enhance the previous methodology by providing support for complex data containing a mix of numeric and nominal variables. Moreover, authors utilized visualization technique to identify the hidden relations present in the high cardinality nominal variables. Yet, the work was limited by its generic methods for constructing the binary tree from the clustering results and the method for automatic generation of schema for a given dataset. Additionally, the authors assumed that all the nominal variables in the dataset were candidates for dimensional variables. This proves to be very unrealistic as real-world datasets consist of a large number of nominal variables and all of these nominal variables cannot be taken as dimensions. There is a need to identify the highly informative dimensions and eliminating the least informative ones to form compact data cubes at multiple levels of data abstraction. Also, a large number of dimensions increase the cube construction time as many views need to be materialized. In this paper, we overcome these limitations and provide methods to identify and rank dimensions in order of significance to prune the search space and assist users in targeted discovery.

It is apparent that most of related work in this area is towards the application of data mining techniques on the top of a manually constructed schema or data cube. Little research has been carried out in exploiting data mining techniques to generate multidimensional schema. Multidimensional modelling is a complex task and we believe that mining techniques can not only assist in schema design process, but can also improve the knowledge discovery process. Moreover, the related work has a number of limitations and there is a strong need for tight coupling of the two disciplines by providing some generic methods to automate the knowledge discovery process, making it more data driven.

3 Overview of Proposed Methodology

In this section, we give an overview of our proposed methodology for the seamless integration of data mining and data warehousing. A critical question in the design of such an integrated methodology is how to integrate or couple the mining techniques within a data warehousing environment? According to (Han and Kamber, 2006) the possible integration schemes include no coupling, loose coupling, semi tight coupling and tight coupling. Tight coupling means that data mining system is smoothly (seamlessly) integrated into the data warehouse system. Such smooth integration is highly desirable because it facilitates efficient implementation of data mining algorithms, high system performance and an integrated knowledge discovery environment.

However, implementation of such a system is a hard task as the techniques employed by each discipline are substantially different from each other. Multidimensional modelling is a challenging task requiring domain knowledge, solid warehouse modelling expertise and deep understanding of data structure and variables (Han and Kamber, 2006). In a real world scenario, data warehouse designers possess the modelling expertise but lack the domain knowledge and thorough understanding of semantic relationships among data variables, which can lead to a poor warehouse design that can dramatically affect the knowledge discovery process. Data mining techniques such as clustering and pattern visualization can assist in understanding and visualizing the complex data structures. A methodology for such data driven multidimensional modelling is required to support both the human warehouse designers and the decision makers to extract useful knowledge from the data.

The aim of our proposed methodology is tight coupling of the two technologies. We employ a hierarchical clustering technique along with the well-known parallel coordinate technique (Rosario et al., 2004) to assist the analysts and designers in finding natural groupings in the data. In the absence of explicit domain expert input we rely on the strengths of hidden relationships and groupings among the variables in a given dataset. Figure 1 depicts the main steps of the proposed methodology. We provide an overview of each step in this section and the details of the tools and methods involved in each step are explained in the section 4.

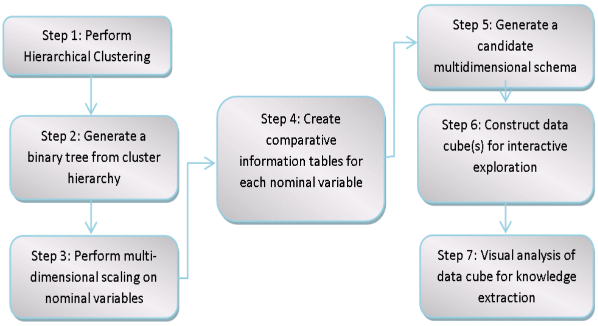

Step 1: Perform Hierarchical Clustering
Step 2: Generate a binary tree from cluster hierarchy
Step 3: Perform multi-dimensional scaling on nominal variables
Step 4: Create comparative information tables for each nominal variable
Step 5: Generate a candidate multidimensional schema
Step 6: Construct data cube(s) for interactive exploration
Step 7: Visual analysis of data cube for knowledge extraction

Figure 1: Proposed methodological steps for knowledge extraction

The first step is to apply agglomerative hierarchical clustering algorithm on the data to generate hierarchical clusters based on a similarity measure. For the purpose of generating clusters, we use only the numeric variables in order to get optimal clustering results, as clustering algorithms perform well on numeric data. In the second step, we generate the binary tree from the hierarchical dendrogram (tree) and efficiently determine the cut-off point in the tree. By efficiently determining the cut-off point we can limit the number of clusters produced by the mining algorithm for analysis and highlight the significant variables in each cluster which define the split of a particular cluster and name each cluster according to its data abstraction level in the binary tree.

For instance, we name the cluster at the first level as C1 and C2. Each cluster represents a parent child relationship in the decision tree, so C1 becomes the parent of C11 and C12 at second level in the hierarchy. In our methodology, we intend to compare the similarities and differences among parent, child and sibling (e.g. C1, C11 and C12) relationships. We take out nominal data from each cluster that did not play any part in the clustering process. In the third step, the effective visualization technique known as Distance-Quantification-Classing (DQC) proposed by (Rosario et al., 2004) for nominal variables is applied on extracted nominal data. This technique maps nominal values to numbers in a manner that conveys semantic relationships by assigning order and spacing among the values, helping in the visualization of the natural grouping among the data values in each nominal variable.

In the fourth step, we construct an information table for each nominal variable present in a cluster by comparing it with its values in neighbouring clusters (parent and sibling). For instance, C11 is compared with its parent (C1) and sibling (C12). We have developed a generic method that calculates the amount of change occurred in each nominal variable at various level of data abstraction and rank the variables based on the calculated value of change from higher to lower. Higher calculated change suggests more degree of variation in a given variable and thus it becomes the potential dimension for multidimensional schema. This highest degree of variation highlights the fact that a given nominal variable has highly significant characteristics as compared to its neighbouring clusters. Therefore, it should be taken as the most informative dimension to explore the numeric variables to discover hidden relationships among numeric and nominal variables.

Such variables with highly significant differences explain the hidden relations between the numeric and nominal variables. In the fifth step, we use the outcome of the visualization technique and comparative information tables to construct the multidimensional schema. The groupings present in ranked dimension in the previous helps in formulating the dimensional hierarchy. Finally, a set of generic *SQL* queries create dimension and fact tables and populate the tables of the multidimensional schema.

In the sixth step, we construct the data cube using the generated schema using automated queries. The final step of the methodology utilizes the data cube by employing the OLAP visualization technique. Users can visually explore the targeted cube structure to extract useful information and knowledge from the underlying data cube built upon multidimensional schema.

4 Integration Tools and Methods

In this section we present the details of the tools and methods involved in the methodology for the seamless integration.

4.1 Hierarchical Clustering of numeric variables

For the purpose of the hierarchical clustering of numeric data we have employed the Hierarchical Clustering Explorer (HCE) tool (Seo and Shneiderman, 2006) as it allows interactive analysis of multidimensional data and helps in identifying the natural number of clusters with interactive visual feedback and dynamic mining query controls. It supports various clustering parameters such as linkage methods (single, average or complete), clustering directions and similarity/distance measure. We use the most commonly used distance measure known as *Euclidean* distance and complete linkage method in our cluster analysis step.

4.2 Hierarchical Clustering of Numeric Variables

In this step, we generate a binary tree to a suitable data abstraction level. This is an important step because the clustering algorithm starts from the leaf nodes, which are the data points, and merges the nodes based on distance measure, creating a single cluster. In large data sets this approach can produce a huge number of clusters and it is not feasible to analyse the clusters which have very few records or no significant information available for extraction. It is a crucial stage of the methodology to determine the data abstraction level in the cluster hierarchy, which can give the analyst a meaningful picture of the overall data distribution. This leads to a sub step 2.1, determining the cut-off point.

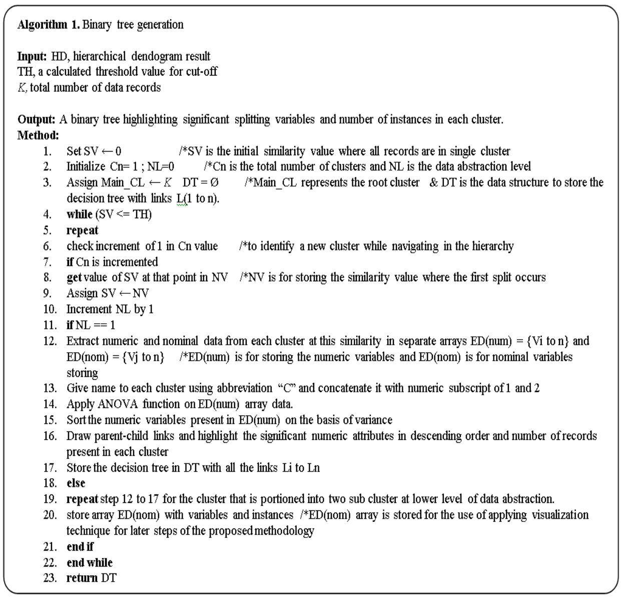
4.2.1 Determining cut-off point in cluster hierarchy

Various ways to determine a suitable cut-off point of the hierarchical tree (dendrogram) have been proposed in the literature. (Basak and Krishnapuram, 2005) have used a lower bound on the minimum number of data points in a cluster as the stopping criterion. However, they suggested the depth of the decision can also be used in combination with the minimum number of data points. In the Biofuel industry (Riviére and Marlair, 2010) relied on cutting the tree at various levels and finally decided (based on domain expertise) the cut-off point where the clustering was more understandable. A more appropriate and flexible approach is adopted by (Seo and Gordish-Dressman, 2007) using the HCE tool built in controls of the minimum similarity bar and detailed cut-off bar. The minimum similarity bar helps users find the right number of clusters and the detail cut-off bar helps users control the level of detail by rendering the sub clusters below the bar with their averages.

In the proposed methodology we retain the flexibility of choosing the specific number of clusters supported by the HCE tool; however, users often do not have a clear idea how many clusters will be adequate for the analysis. To tackle this problem we use the linkage inconsistency coefficient threshold, which is defined as the length of a link in a cluster hierarchy minus the average length of all links divided by the standard deviation of all links (Cordes et al., 2002), to find the natural cluster division in the dataset. If the length of the link in dendrogram has approximately the same length as neighbouring links then the objects have similar features (high consistency) and vice versa. We calculate this threshold by using the *MATLAB* inconsistent function that returns data about the links and returns an inconsistency coefficient value. This calculated threshold value helps in determining the suitable cut-off point for a given dataset.

4.2.2 Cluster Naming, Significant Splitting Variable Identification and Data Extraction

After determining the suitable threshold value for the cut-off point the next task is to name the cluster in a logical and meaningful way. Cluster naming helps identify the data group that needs further exploration. As all the numeric variables are used in the clustering process, it is vital to identify the significant variables in a given cluster say C1. These significant numeric variables define the splitting of a cluster into two sub clusters, for instance, C11 and C12, and assist in picking the suitable measures or facts for the multidimensional model to be built. For this purpose, we perform a statistical function called the analysis of variance (ANOVA) and sort the variables based on their variance value. We construct the binary tree showing the cluster names and significant variables in each cluster. However, the high dimensionality of the data sets still hinders users from finding interesting patterns and outliers. One reason for this is the presence of nominal variables in each cluster which are not involved in the clustering process. In order to find hidden relationships among nominal and numeric data, we extract the nominal data from each cluster and store it in temporary files to be used later. Algorithm 1 elaborates the steps involved in generating a binary tree.



Algorithm 1. Binary tree generation

Input: HD, hierarchical dendogram result
TH, a calculated threshold value for cut-off
K, total number of data records

Output: A binary tree highlighting significant splitting variables and number of instances in each cluster.
Method:

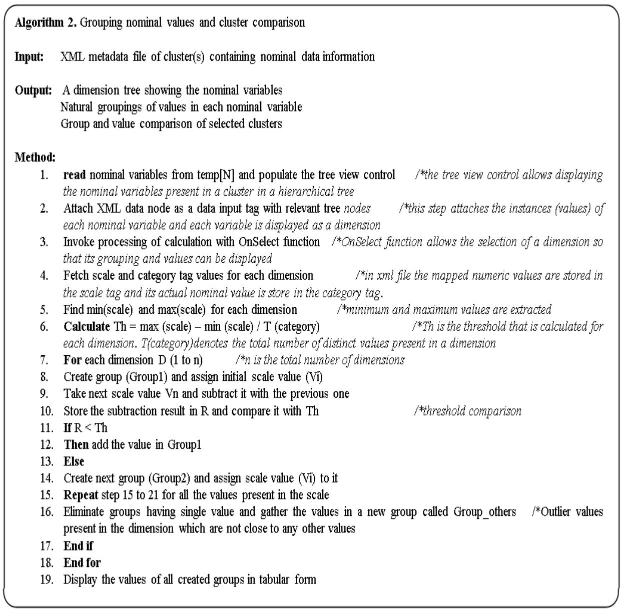
```
1.  Set SV ← 0                 /*SV is the initial similarity value where all records are in single cluster
2.  Initialize Cn= 1 ; NL=0        /*Cn is the total number of clusters and NL is the data abstraction level
3.  Assign Main_CL ← K   DT = Ø      /*Main_CL represents the root cluster  & DT is the data structure to store the
    decision tree with links L(1 to n).
4.  while (SV <= TH)
5.  repeat
6.  check increment of 1 in Cn value     /*to identify a new cluster while navigating in the hierarchy
7.  if Cn is incremented
8.  get value of SV at that point in NV   /*NV is for storing the similarity value where the first split occurs
9.  Assign SV ← NV
10. Increment NL by 1
11. if NL == 1
12. Extract numeric and nominal data from each cluster at this similarity in separate arrays ED(num) = {Vi to n} and
    ED(nom) = {Vj to n}    /*ED(num) is for storing the numeric variables and ED(nom) is for nominal variables
    storing
13. Give name to each cluster using abbreviation "C" and concatenate it with numeric subscript of 1 and 2
14. Apply ANOVA function on ED(num) array data.
15. Sort the numeric variables present in ED(num) on the basis of variance
16. Draw parent-child links and highlight the significant numeric attributes in descending order and number of records
    present in each cluster
17. Store the decision tree in DT with all the links Li to Ln
18. else
19. repeat step 12 to 17 for the cluster that is portioned into two sub cluster at lower level of data abstraction.
20. store array ED(nom) with variables and instances /*ED(nom) array is stored for the use of applying visualization
    technique for later steps of the proposed methodology
21. end if
22. end while
23. return DT
```

Figure 2: Algorithmic steps of binary tree generation

4.3 Perform Multidimensional Scaling on Nominal Variables

After retrieving the nominal data, we apply the DQC technique to efficiently map the nominal values into numbers. This technique is implemented in a java based program developed by (Rosario et al., 2004). The program maps the nominal values efficiently to numbers so that the semantic relationships among the values can be easily visualized. Two output files are created by the program, one holding the mapped values and the other an *xml* file for meta-data. The purpose of this implementation is the visualization of nominal values using a well-known visualization technique called parallel coordinate. The major drawback of this parallel coordinate technique is that it is only suitable when there is small number of nominal variables. Visualization of more than 10 nominal variables becomes very unclear using parallel coordinate display.

Moreover, this technique fails when there are a large number of instances in a single nominal variable. For instances, *Country* is a nominal variable which usually have more than 40 distinct values (country names). Such variables are difficult to visualize using parallel coordinates as most of the values overlap each other. This motivated us to find a solution for extracting the groupings produced by this visualization technique. In our investigation, we identified that the output xml file produced by the Java based tool holds the structure of all the nominal variables and their instances. This led to the development of our own prototype that reads this output xml file and groups the values based on automatically calculated thresholds for each nominal variable. In addition to this, the developed prototype also allows comparative analysis of a given cluster with neighbouring clusters. However, our prototype is flexible enough to allow comparison of a given cluster with any other cluster present in the hierarchical tree. Algorithm 2 illustrates the steps involved in our developed prototype for finding natural groupings in nominal variables.



Algorithm 2. Grouping nominal values and cluster comparison

Input: XML metadata file of cluster(s) containing nominal data information

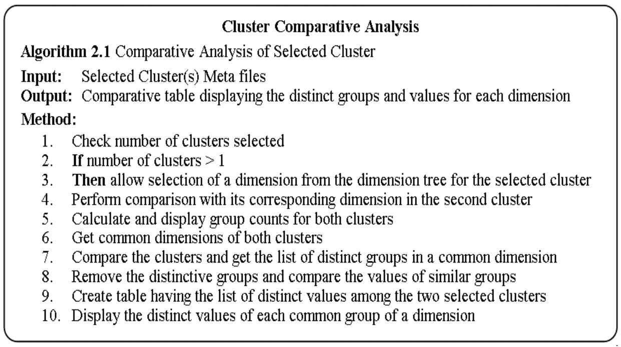
Output: A dimension tree showing the nominal variables
Natural groupings of values in each nominal variable
Group and value comparison of selected clusters

Method:

```
1.  read nominal variables from temp[N] and populate the tree view control    /*the tree view control allows displaying
    the nominal variables present in a cluster in a hierarchical tree
2.  Attach XML data node as a data input tag with relevant tree nodes     /*this step attaches the instances (values) of
    each nominal variable and each variable is displayed as a dimension
3.  Invoke processing of calculation with OnSelect function  /*OnSelect function allows the selection of a dimension so
    that its grouping and values can be displayed
4.  Fetch scale and category tag values for each dimension        /*in xml file the mapped numeric values are stored in
    the scale tag and its actual nominal value is store in the category tag.
5.  Find min(scale) and max(scale) for each dimension        /*minimum and maximum values are extracted
6.  Calculate Th = max (scale) – min (scale) / T (category)                /*Th is the threshold that is calculated for
    each dimension. T(category)denotes the total number of distinct values present in a dimension
7.  For each dimension D (1 to n)       /*n is the total number of dimensions
8.  Create group (Group1) and assign initial scale value (Vi)
9.  Take next scale value Vn and subtract it with the previous one
10. Store the subtraction result in R and compare it with Th                 /*threshold comparison
11. If R < Th
12. Then add the value in Group1
13. Else
14. Create next group (Group2) and assign scale value (Vi) to it
15. Repeat step 15 to 21 for all the values present in the scale
16. Eliminate groups having single value and gather the values in a new group called Group_others   /*Outlier values
    present in the dimension which are not close to any other values
17. End if
18. End for
19. Display the values of all created groups in tabular form
```

Figure 3: Algorithmic steps of grouping similar nominal values

Algorithm 2.1 describes the steps for comparative analysis (Parent-Sibling) of selected clusters. Moreover, Algorithm 2.1 highlights the similarities and differences among the dimensions in different clusters. The outcomes of cluster comparison steps are fed into our next step for the generation of rich information tables for each cluster in the tree.

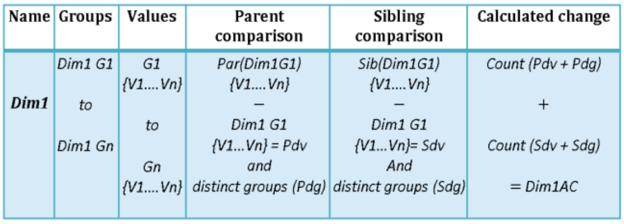


```
Cluster Comparative Analysis
Algorithm 2.1 Comparative Analysis of Selected Cluster
Input:   Selected Cluster(s) Meta files
Output:  Comparative table displaying the distinct groups and values for each dimension
Method:
  1.  Check number of clusters selected
  2.  If number of clusters > 1
  3.  Then allow selection of a dimension from the dimension tree for the selected cluster
  4.  Perform comparison with its corresponding dimension in the second cluster
  5.  Calculate and display group counts for both clusters
  6.  Get common dimensions of both clusters
  7.  Compare the clusters and get the list of distinct groups in a common dimension
  8.  Remove the distinctive groups and compare the values of similar groups
  9.  Create table having the list of distinct values among the two selected clusters
  10. Display the distinct values of each common group of a dimension
```

Figure 4: Algorithmic steps of comparative analysis of clusters

4.4 Create Comparative Information Tables for Each Nominal Variable

This step consists of two main tasks. First, providing a method for calculating the variable change at multiple levels of data abstraction and second, ranking the variables to highlight the potential dimensions for the multidimensional schema. Table 1 shows the generic structure of the information that is provided for each dimension in a given cluster, for instance cluster C1.



Name	Groups	Values	Parent comparison	Sibling comparison	Calculated change
Dim1	*Dim1 G1* *to* *Dim1 Gn*	*G1* *{V1....Vn}* *to* *Gn* *{V1....Vn}*	*Par(Dim1G1)* *{V1....Vn}* – *Dim1 G1* *{V1...Vn} = Pdv* *and* *distinct groups (Pdg)*	*Sib(Dim1G1)* *{V1....Vn}* – *Dim1 G1* *{V1...Vn}= Sdv* *And* *distinct groups (Sdg)*	*Count (Pdv + Pdg)* + *Count (Sdv + Sdg)* *= Dim1AC*

Table 1: Structure of information table for a single dimension in a cluster

We explain the notations used in each column of Table 1. In the first column the name of a dimension is displayed. Each cluster consists of a set of dimension say *Dim {1,2,…,n}* where n is the total number of dimensions.

The second column shows the natural groups present in the first dimension *(Dim1)*. The groups in a dimension can vary from 1 to n number of groups. We name the groups in ascending order starting from Group1 *(G1)*, Group2 *(G2)* and so on. The third column shows the values which are grouped together using the procedure shown in Algorithm 2.

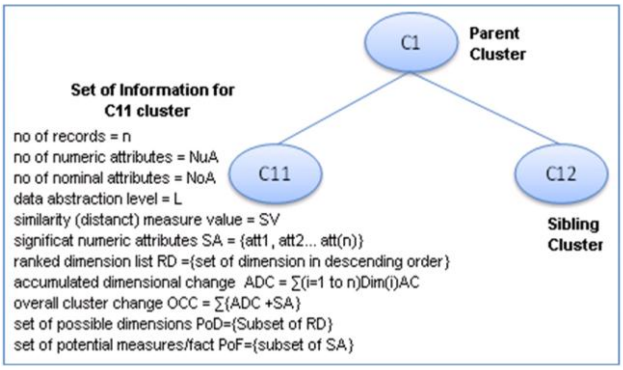
Similarly, every group has *n* number of values *{V1,V2,…,Vn}* in it. The forth column is the calculation column that calculates the change of a given dimension in a cluster with its corresponding dimension in the parent, with two sets of comparisons performed in this column. First we perform a value comparison. Each group of selected cluster dimension is compared with its corresponding group in the parent cluster. For instance, Group1 of C11 is compared with Group1 of C1 for a particular dimension Dim1. The resultant is a set of distinct values present in the parent dimension but that are not present in *Dim1*. This is achieved by applying the minus operator and the result is stored in the *Pdv* variable. Secondly, we compare the groups of the Dim1 with the groups of its parent *(Par(Dim1))*. We calculate the group change in another variable called Pdg. In the fifth column the same set of comparisons are performed with the sibling dimension, *(Sib(Dim1))*. In the last column, we count the parent change *(Pdv + Pdg)* and sibling change *(Sdv + Sdg)* and store the result of accumulated change in the *Dim1AC* variable. This accumulated change is calculated for each dimension *Dim1* to *Dim(n)* and the dimensions are ranked on the basis of this calculated value. The dimension that possesses higher accumulated changed is ranked higher and vice versa. The generic method of calculation is applied to all the dimensions present in a cluster to get the accumulated dimensional change represented by equation 1.

$$ADC = \sum_{i=1}^{n} \{Dim(i)\,AC\} \qquad (1)$$

This helps in presenting the overall change present in a cluster as compared to its neighbouring clusters. The formula for calculating overall cluster change is represented in equation 2.

$$OCC = \sum_{i=1}^{n} \{ADC + SA\} \qquad (2)$$

here *SA* represents the count of significant variables present in a cluster as a result of performing the ANOVA function described in step 2.2 of the proposed methodology. Figure 5 depicts an example of information provided to the user after the comparison with its neighbouring clusters.



Set of Information for C11 cluster
no of records = n
no of numeric attributes = NuA
no of nominal attributes = NoA
data abstraction level = L
similarity (distanct) measure value = SV
significat numeric attributes SA = {att1 , att2... att(n)}
ranked dimension list RD ={set of dimension in descending order}
accumulated dimensional change ADC = ∑(i=1 to n)Dim(i)AC
overall cluster change OCC = ∑{ADC +SA}
set of possible dimensions PoD={Subset of RD}
set of potential measures/fact PoF={subset of SA}

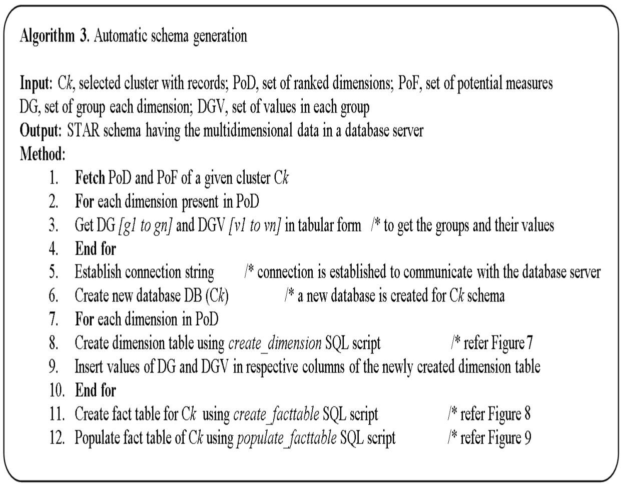
Figure 5: Set of information for cluster C11 after parent (C1) and sibling (C12) comparison

After performing comparisons a set of highly ranked dimensions and measures are fed into the automatic schema generator that constructs the well-known *STAR* schema for multidimensional analysis.

4.5 Generate a Candidate Multidimensional Schema

For the generation of schema, we have developed a generic method that constructs the *STAR* schema in

Microsoft SQL Server 2005 database. The steps are presented in Algorithm 3. Figure 7 depicts the script that is used by the schema generator to create a dimension table in the database server. Each dimension consists of a group level and an individual value level and defines the concept hierarchy of the generated dimensions for the dimensional data navigation. Figure 8 represents the structure of the fact table creation script. First, it creates the fact table ID followed by selected dimension IDs. In the end, it applies primary key and foreign key constraints on the dimensions.

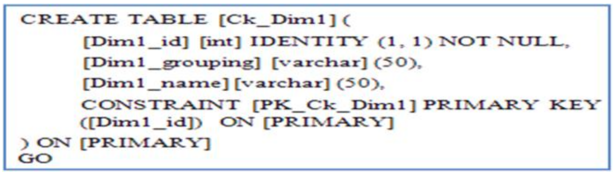


```
Algorithm 3. Automatic schema generation

Input: Ck, selected cluster with records; PoD, set of ranked dimensions; PoF, set of potential measures
DG, set of group each dimension; DGV, set of values in each group
Output: STAR schema having the multidimensional data in a database server
Method:
    1.  Fetch PoD and PoF of a given cluster Ck
    2.  For each dimension present in PoD
    3.  Get DG [g1 to gn] and DGV [v1 to vn] in tabular form   /* to get the groups and their values
    4.  End for
    5.  Establish connection string       /* connection is established to communicate with the database server
    6.  Create new database DB (Ck)           /* a new database is created for Ck schema
    7.  For each dimension in PoD
    8.  Create dimension table using create_dimension SQL script              /* refer Figure 7
    9.  Insert values of DG and DGV in respective columns of the newly created dimension table
    10. End for
    11. Create fact table for Ck  using create_facttable SQL script            /* refer Figure 8
    12. Populate fact table of Ck using populate_facttable SQL script          /* refer Figure 9
```

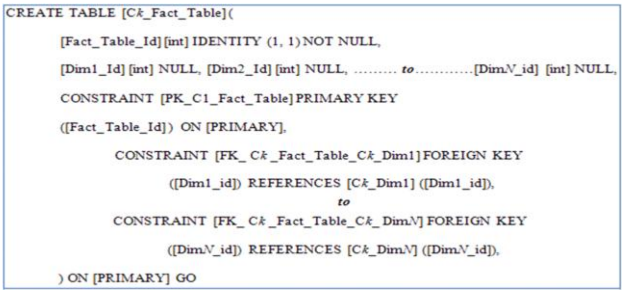
Figure 6: Algorithmic steps of schema generation

After creating the fact table, the script in Figure 9 is used to insert (populate) data in it. It selects the dimension IDs from the dimension table and the measures or facts in the given cluster Ck where the names in source data cluster Ck are equivalent to the distinct names in the dimensional table. These scripts create a complete multidimensional model in the database server for the use of cube construction.



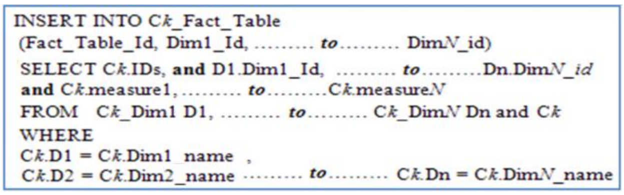
```
CREATE TABLE [Ck_Dim1] (
        [Dim1_id] [int] IDENTITY (1, 1) NOT NULL,
        [Dim1_grouping] [varchar] (50),
        [Dim1_name] [varchar] (50),
        CONSTRAINT [PK_Ck_Dim1] PRIMARY KEY
        ([Dim1_id]) ON [PRIMARY]
) ON [PRIMARY]
GO
```

Figure 7: Structure of SQL script for creating dimension table



```
CREATE TABLE [Ck_Fact_Table](
        [Fact_Table_Id] [int] IDENTITY (1, 1) NOT NULL,
        [Dim1_Id] [int] NULL, [Dim2_Id] [int] NULL, ......... to ............[DimN_id] [int] NULL,
        CONSTRAINT [PK_C1_Fact_Table] PRIMARY KEY
        ([Fact_Table_Id]) ON [PRIMARY],
                CONSTRAINT [FK_ Ck _Fact_Table_Ck_Dim1] FOREIGN KEY
                        ([Dim1_id]) REFERENCES [Ck_Dim1] ([Dim1_id]),
                                        to
                CONSTRAINT [FK_ Ck _Fact_Table_Ck_ DimN] FOREIGN KEY
                        ([DimN_id]) REFERENCES [Ck_DimN] ([DimN_id]),
        ) ON [PRIMARY] GO
```

Figure 8: Structure of SQL script for creating fact table and setting dimension and fact tables' relationships



```
INSERT INTO Ck_Fact_Table
 (Fact_Table_Id, Dim1_Id, ......... to ......... DimN_id)
 SELECT Ck.IDs, and D1.Dim1_Id,  ......... to .........Dn.DimN_id
 and Ck.measure1, ......... to .........Ck.measureN
 FROM   Ck_Dim1 D1, ......... to ......... Ck_DimN Dn and Ck
 WHERE
 Ck.D1 = Ck.Dim1  name  ,
 Ck.D2 = Ck.Dim2_name ......... to ......... Ck.Dn = Ck.DimN_name
```

Figure 9: Structure of SQL script for populating data in fact table

4.6 Construction of Data Cubes for Visual Exploration

In this step, we use Microsoft Analysis services 2005 software to construct the cube from the automatically generated multidimensional schema. We use MOLAP because this storage type maps a multidimensional view directly to the data cube array structure. This gives the advantage of fast indexing of pre-computed summarized data. The proposed methodology allows construction of data cubes containing highly informative dimensions and measures present in each cluster and eliminate the lowly ranked (less informative) dimensions. It efficiently narrows down the cube search space and improves cube construction time as fewer views remain for cube materialization.

4.7 Visual Analysis of Cube for Knowledge Extraction

In the final step of the methodology, we have developed an application for the front-end OLAP analysis by using the OLAP Services Control developed by Dundas Data Visualization, Incorporation. The application connects with the Cube server and shows the data cubes for performing OLAP operations such as drill-down, roll-up, slice and dice. Analysts can interactively and visually explore the data cube which has the most significant dimensions and measures of a given cluster. These most significant dimensions assist the users to quickly identify the hidden relationships between the highly informative dimensions and thus lead the analyst to extract knowledge from a constrained yet informative search space.

5 An Example of application – Automobile dataset

In this section, we discuss an example to illustrate the steps of our proposed methodology. We use an *Automobile* (Schlimmer, 1985) dataset which has 205 records and 26 variables. This data set consists of three types of entities: (a) the specification of an auto in terms of various characteristics, (b) its assigned insurance risk rating, (c) its *normalized losses* in use as compared to other cars. The second rating corresponds to the degree to which the auto is more risky than its price indicates. Cars are initially assigned a risk factor symbol associated with its price. Then, if it is more risky (or less), this symbol is adjusted by moving it up (or down) the scale. This process is called "symbolling". A value of +3 indicates that the auto is risky, -3 that it is probably pretty safe. A full description of all the variables of this dataset can be found at the UCI machine learning website (Asuncion and Newman, 2010). For the first step, we use the 16 numeric variables present in the dataset to perform hierarchical clustering.

From this dendrogram we determine the suitable cut-off point and generate the binary tree using Algorithm 1, which helps the analyst to pick the cluster of his choice for further analysis. At this stage, we provide the significant numeric variables present in each cluster of the tree. Figure 10 depicts the binary tree generated for the automobile dataset. Each cluster is given a unique name and the significant variables are shown in the surrounding of the cluster. The next step of the methodology is the application of Distance-Quantification-Classing (DQC) technique in order to identify semantic relationships among high cardinal nominal variables.

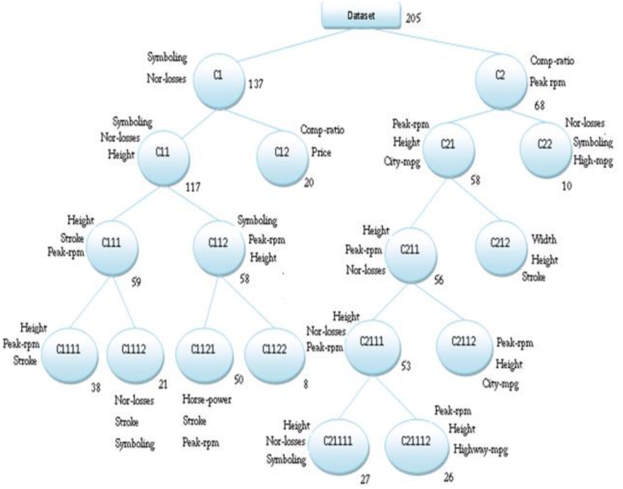
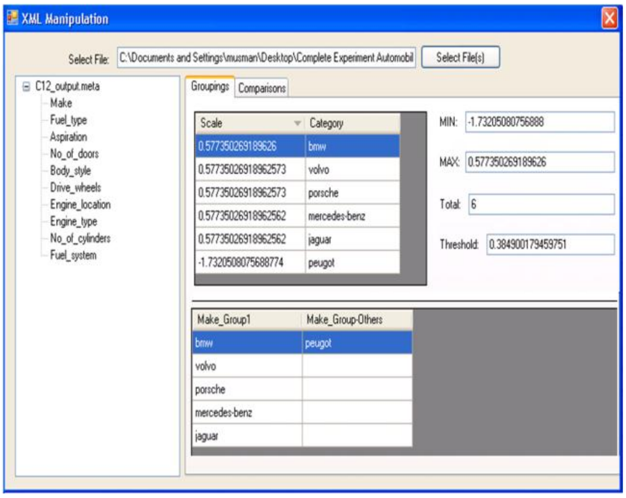


Figure 10: Binary tree for generated from the dendrogram showing significant numeric variables

Our developed prototype takes clusters nominal data as input to show the groupings for each nominal variable. Figure 11 shows the groupings of the *Make* variable in Cluster C12 using our developed prototype.



Select File: C:\Documents and Settings\musman\Desktop\Complete Experiment Automobil

C12_output.meta
- Make
- Fuel_type
- Aspiration
- No_of_doors
- Body_style
- Drive_wheels
- Engine_location
- Engine_type
- No_of_cylinders
- Fuel_system

Scale	Category
0.577350269189626	bmw
0.57735026918962573	volvo
0.57735026918962573	porsche
0.57735026918962562	mercedes-benz
0.57735026918962562	jaguar
-1.7320508075688774	peugot

MIN: -1.73205080756888

MAX: 0.577350269189626

Total: 6

Threshold: 0.384900179459751

Make_Group1	Make_Group-Others
bmw	peugot
volvo	
porsche	
mercedes-benz	
jaguar	

Figure 11: View of the groupings achieved for Cluster C12 through the developed prototype

It can be seen that Cluster C12 has two groups and the automobiles that have a strong association in the underlying cluster records are grouped together. For instance, *BMW*, *Volvo*, *Porsche*, *Mercedes-Benz* and *Jaguar* are closer to each other. On the other hand, *Peugeot* is a local outlier because no other automobile has similar variables. At this stage, the analyst can identify the strong relationships as well as the anomalies present in each nominal variable in a given cluster. Using Algorithm 3, we generate the multidimensional schema and construct data cubes, allowing various analytical operations and giving the analyst a targeted multidimensional space for meaningful exploration for discovering knowledge using OLAP operations.

6 Real-world Case Studies (Results and Discussion)

In this section, we discuss the results of the case studies which have been performed on real-world datasets namely *Automobile* (Schlimmer, 1985), *Statlog German Credit data* (Hofmann, 1994) and *Adult* (Kohavi and Becker, 1996) datasets taken from from the UCI machine learning repository. In Table 2, we give the summary of the variables present in these three datasets. Further detailed descriptions of each dataset can be obtained from UCI's machine learning website.

Data Set	Variable types	No of attributes	Nominal attributes	Numeric attributes	No of records
Automobile	numeric, nominal	26	10	16	205
German Credit data	numeric, nominal	20	13	7	1000
Adult	numeric, nominal	14	8	6	48842

Table 2: Summary of datasets used for case studies

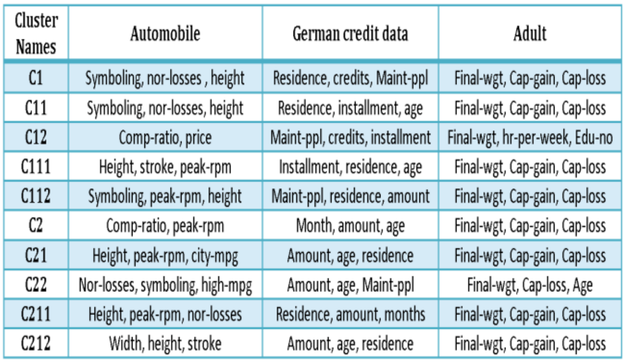
For our three case studies we first performed hierarchical clustering on numeric variables to get a hierarchy of clusters in the form of a dendrogram. For the adult dataset we removed the records having missing values and used 30162 (61%) records for hierarchical clustering. As clustering results tend to be better with numeric variables, all numeric variables were involved in the clustering process. Other clustering parameters used in this step are explained above in section 4. However, agglomerative hierarchical clustering algorithm only works well for small datasets. It took 329 seconds for the HCE tool to produce a dendrogram for the Adult dataset on a machine with 2 GHz processor and 4 GB of RAM.

A suitable cut-off point in the dendrogram was calculated for each of the datasets using the inconsistency function. This calculation helped in setting up the detailed cut-off bar and suggested the number of levels of data abstraction that should be adequate for extracting knowledge from a given dataset. In Table 3, we highlight the calculated similarity value (cut-off point), the number of levels and number of clusters at the bottom level.

Data Set	Calculated Similarity value for cut-off point (0 to 1)	Total levels of data abstraction for analysis	No of clusters at lowest data abstraction level	Total Clusters sufficient for overall analysis
Automobile	0.676	5	10	18
German Credit data	0.557	7	14	26
Adult	0.623	9	18	34

Table 3: Calculated cut-off points and cluster information at lowest data abstraction

It can be seen from Table 3 that regardless of the size of dataset the hierarchical clustering tree can be efficiently cut at approximately 0.60 similarity measures. However, the levels of data abstraction and the number of clusters are directly proportional to the size of the datasets. After calculating the cut-off point and determining the number of clusters, we extract data from the clusters and apply ANOVA function and find the significant variables in each cluster that define a split. These significant numeric variables are ranked at this stage of the methodology for each cluster and become the possible measures in a ranked order for a given cluster in the hierarchy. The significant variables found for the datasets are shown in Table 4. Due to limited space, we only show the top 3 significant variables of 10 distinct clusters from each dataset. It can be seen from the results in Table 4 that the Automobile dataset each cluster has its own significant variables. However, in a particular section of the tree (or subset of the tree) *{C1, C11, C12, C111 & C112}* the important measures are symbolling, normalized losses (nor-losses) and height.



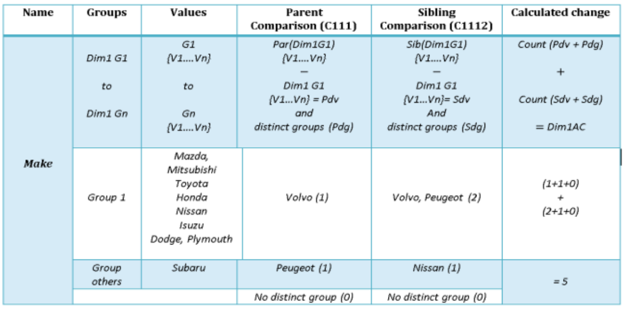
Cluster Names	Automobile	German credit data	Adult
C1	Symboling, nor-losses , height	Residence, credits, Maint-ppl	Final-wgt, Cap-gain, Cap-loss
C11	Symboling, nor-losses, height	Residence, installment, age	Final-wgt, Cap-gain, Cap-loss
C12	Comp-ratio, price	Maint-ppl, credits, installment	Final-wgt, hr-per-week, Edu-no
C111	Height, stroke, peak-rpm	Installment, residence, age	Final-wgt, Cap-gain, Cap-loss
C112	Symboling, peak-rpm, height	Maint-ppl, residence, amount	Final-wgt, Cap-gain, Cap-loss
C2	Comp-ratio, peak-rpm	Month, amount, age	Final-wgt, Cap-gain, Cap-loss
C21	Height, peak-rpm, city-mpg	Amount, age, residence	Final-wgt, Cap-gain, Cap-loss
C22	Nor-losses, symboling, high-mpg	Amount, age, Maint-ppl	Final-wgt, Cap-loss, Age
C211	Height, peak-rpm, nor-losses	Residence, amount, months	Final-wgt, Cap-gain, Cap-loss
C212	Width, height, stroke	Amount, age, residence	Final-wgt, Cap-gain, Cap-loss

Table 4: Significant variables present in each cluster of the three datasets

The only exception is in the C12 cluster in which the compression ratio (comp-ratio) and price variables are significant. This information can play a vital role in the knowledge extraction process. For instance, an analyst interested in looking at the variation of price of the automobiles will focus mainly on cluster C12. On the other hand, symbolling value, which represents the safety measure of the car, can be looked at a number of clusters to see its variation. Similarly, in German credit data, the analyst can easily extract the information from a subset of the tree *{C2, C21, C22, C211 & C212}* that *Amount* and *Age* are the two dominant variables. This means that by concentrating on this sub-tree the analyst can find interesting correlation between the two variables namely *Amount* and *Age*.

The third dataset yields the most interesting set of variables from all the clusters. Final Weight (*Final-wgt*) is the weight assigned to people taking into account the (*age, sex and race*) variables. This assigned weight is leading in all clusters. However, there are a few obvious distinctions in cluster C22 and C12. These clusters can be of focus if (*hours per week* or *Age*) variable related data needs to be explored. Up to this point, our methodology targeted only numeric variables but in real world datasets there are a number of highly cardinal nominal variables exist. Mix of numeric and nominal variables require efficient analysis in order to discover knowledge from these complex real-world datasets with mixed variables.

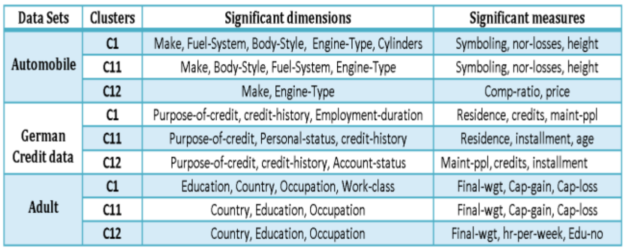
The prevalent mining technique used for the efficient analysis of mixed data is Clustering. A number of authors (Li and Biswas, 2002, Ahmad and Dey, 2007) have adopted this technique and a variety of algorithms are proposed to tackle the mixed data analysis problem. Hierarchical Clustering, in particular, has shown good results in this area. We also have adopted hierarchical clustering technique for efficient analysis of numeric data. However, we use the visualization technique along with our own information processing method (details given in step 3; section 4). We extract the nominal data from each of the clusters in the tree and apply the visualization technique (DQC) on the nominal variables. After, identifying the groupings in each nominal variable we proceed to construct the rich information tables for the dimension in a cluster. Table 5, shows the information table data produced for the top dimension of *Automobile* dataset. Similarly, the dimensional change value is calculated for the given cluster with respect to its neighbouring clusters to provide detailed information about the nominal variables and the distribution among a specific region in the hierarchical tree. Up to this step, the proposed methodology has identified and ranked the possible measures (significant numeric variables) and the potential dimension (ranked nominal variables). This set of information is given to the binary tree to depict more detailed information about the underlying data in a cluster as shown in Figure 5. It leads to the next step which concentrates on the automatic generation of multidimensional schema.



Name	Groups	Values	Parent Comparison (C111)	Sibling Comparison (C1112)	Calculated change
Make	*Dim1 G1* *to* *Dim1 Gn*	*G1* *{V1....Vn}* *to* *Gn* *{V1....Vn}*	*Par(Dim1G1)* *{V1....Vn}* *–* *Dim1 G1* *{V1...Vn} = Pdv* *and* *distinct groups (Pdg)*	*Sib(Dim1G1)* *{V1....Vn}* *–* *Dim1 G1* *{V1...Vn}= Sdv* *And* *distinct groups (Sdg)*	*Count (Pdv + Pdg)* *+* *Count (Sdv + Sdg)* *= Dim1AC*
	Group 1	*Mazda,* *Mitsubishi* *Toyota* *Honda* *Nissan* *Isuzu* *Dodge, Plymouth*	*Volvo (1)*	*Volvo, Peugeot (2)*	*(1+1+0)* *+* *(2+1+0)*
	Group others	*Subaru*	*Peugeot (1)*	*Nissan (1)*	*= 5*
			No distinct group (0)	*No distinct group (0)*	

Table 5: Information summary of values present in top dimension (Make) of Cluster C111

After the application of Algorithm 3, we generated the schema for each of three clusters in each dataset used in our case studies. Table 6 shows the potential dimensions and significant measures used for generating the STAR schema for three clusters from each dataset representing the parent-child relationship.



Data Sets	Clusters	Significant dimensions	Significant measures
Automobile	C1	Make, Fuel-System, Body-Style, Engine-Type, Cylinders	Symboling, nor-losses, height
	C11	Make, Body-Style, Fuel-System, Engine-Type	Symboling, nor-losses, height
	C12	Make, Engine-Type	Comp-ratio, price
German Credit data	C1	Purpose-of-credit, credit-history, Employment-duration	Residence, credits, maint-ppl
	C11	Purpose-of-credit, Personal-status, credit-history	Residence, installment, age
	C12	Purpose-of-credit, credit-history, Account-status	Maint-ppl, credits, installment
Adult	C1	Education, Country, Occupation, Work-class	Final-wgt, Cap-gain, Cap-loss
	C11	Country, Education, Occupation	Final-wgt, Cap-gain, Cap-loss
	C12	Country, Education, Occupation	Final-wgt, hr-per-week, Edu-no

Table 6: Information of dimensions and measures for cluster C1, C11 and C12 for each dataset

The automatic schema generation method gives two advantages for targeted knowledge discovery by reducing the number of dimension present in a data cluster and providing the significant measures present in a cluster. Moreover, it gives an abstract level relationship between the most influential numeric and nominal variables. For instance, the C12 cluster of the *Automobile* dataset highlights a relationship between *Make (model name)* and *Engine Type* variables with the *compression ratio (comp-ratio)* and *price* of the automobile. Similarly, the *Purpose of credit* and *credit history* variables can be used to explore the credit amount and residential years from the credit cards data.

The *Adult* dataset suggests the prominent dimensions of *country, education and occupation*, with cluster C12 predicting a strong association between the *hours per week (hrs-per-week)* variable with prominent dimensions in this cluster. Data warehouse designers and knowledge workers can use this information as a starting point to explore further and extract hidden patterns from the data using OLAP functions. To perform the OLAP functions, we constructed data cubes for each cluster based on the identified dimensions and measures. With minimum number of dimensions and measures, our methodology limits the large number of views to be materialized and also provides narrow search space for the rapid discovery of knowledge from the underlying data. In each step of the proposed methodology, we provide step-by-step guidance to the data warehouse designers and knowledge workers to find useful knowledge. The set of tools, techniques and integration methods assist in recognizing complex and large data sets. In this work, we intend to offset the individual weaknesses of existing knowledge discovery methods and techniques and integrate the strengths tightly in a logical way. The proposed work can be applied in many areas where knowledge workers deal with complex datasets and have very limited knowledge of the domain. Data warehouse designers rely heavily on the user requirements and domain experts for modelling the warehouse schema. Because user requirements are unpredictable and constantly change with time, a design based solely on such requirements is unstable and poor choice of dimension and measures can intensively affect the knowledge discovery process and quality of decisions. Furthermore, even domain experts ignore the semantic relationships among data variables as they cannot be identified without the assistance of automated techniques.

It is apparent from the case studies that the proposed methodology helps in recognizing the predominant variables and finding their hidden relationships at every step. The identified patterns and relationship information has the potential to lead to the discovery of knowledge.

7 Comparison of Proposed Automated Analysis with Manual Analysis

In this section we give a general discussion on the proposed system and compare it with the traditional method of manual data analysis. It is important to clarify that the proposed integrated system doesn’t solve a classical data mining or data analysis problem so there is no point to show that the system is able to complete a mining or analysis task on some standard dataset. Moreover, we emphasize that our proposed methodology is suitable in cases where very limited domain knowledge exist and analysts depend on the systems to guide him/her in discovering knowledge. In our proposed work, we start the automated analysis by performing hierarchical clustering. It is assumed that data miners have a good idea of the important variables on basis of which clustering should be performed and later they interpret the clustering results. Also, based on their domain knowledge they extract the number of clusters which seem to be adequate for analysis. In high dimensional data, it is difficult for the human analyst to come up with important dimensions on the basis of which clustering is performed. Moreover, it is not feasible to rely heavily on limited domain knowledge to decide the number of clusters adequate for discovering knowledge.

In our work we automate this process by taking all the numeric variables into account for clustering and later perform statistical analysis to highlight the important variables in each cluster. Furthermore, we help in identifying the number of clusters to be taken for analysis by automatically calculating the cut-off point in the hierarchy. Likewise, we automate the manual way of identifying and by visualizing nominal variables in each cluster via multi-dimensional scaling technique. Additionally, we provide a generic way to assign nominal values in various groups which are near to impossible for the human analyst to visualize and create manually. These data guided numeric and nominal variables and their groupings information assist the human data warehouse designers to use this information to design a multi-dimensional schema for analytical analysis.

Without the presence of this automated system, data warehouse designers have to rely heavily on domain knowledge to manually model dimensions and dimensional hierarchies. Moreover, the manually modelled schema is unable to highlight the natural grouping of values which are interesting and worth exploring using OLAP tool. For instance, *Country* could be taken as a dimension by human data warehouse designer and one meaningful way of grouping countries is to assign countries based on their regions such as *Asia*, *Europe*, and *Africa*. However, the automated way proposed in this paper, *Country* dimension can form natural groups and these groups show the semantic relationships. For example, in an automated way of grouping *Australia*, *Mexico* and *Spain* could be together to highlight strong relations (i.e. almost same GDP) in underlying data. This interesting information is not obvious in manually modelled schema and will take unnecessary time to discover it manual inspection of OLAP cubes.

We believe that the proposed system facilitates a broad range of users (data warehouse designers, data miners, analysts) as different users have diverse analytical needs. For instance, a data miner may be interested in finding natural grouping (clusters) of data whereas the warehouse designer is more interested in finding important dimensions and measures in order to design a multi-dimensional scheme and may not be interested in knowing the natural clusters that exist in the data. It shows that certain information which appears to be knowledge for one type of user may not appear the same for the other. It is near to impossible to discover knowledge without the automated aid provided by integrating the strengths of data mining and warehousing technologies. However, we do not challenge the existing manual usage of integrated techniques and emphasize that our proposal should be taken as a complementary method to assist knowledge discovery.

8 Conclusion and Future Work

In this paper, we have presented a generic methodology for the seamless integration of data mining and data warehousing with flexibility and efficiency objectives in consideration. In particular, we employed hierarchical clustering and multidimensional scaling technique for better understanding and efficient analysis of datasets. Moreover, the generic methods are provided to use the clustering results for the automatic generation of binary tree, which provides rich information of data variables at different levels of data abstraction. This rich information not only helps users to identify clusters of interest, but also highlights the semantic relationships and associations between the numeric and nominal variables within a cluster. These clusters and their data association information guide the human data warehouse developers to select the best possible set of dimensions and measures for the multidimensional model construction. We also propose an automated method for the generation of multidimensional schema to construct compact and informative data cubes. The case studies performed on real world datasets validated our proposal and elaborated its significance at various stages of knowledge discovery process. Results show that a hybrid solution complemented by automated methods seamlessly enhances the knowledge discovery process. In the end, we discussed the applicability of the proposed methodology in those domains where the data warehouse designers need automated assistance to design the schema and knowledge workers need data drive assistance and flexibility to explore complex dataset to find knowledge.

The future work is mainly focused on overcoming the existing limitations of the methodology. For instance, we intend to use more statistical functions such as entropy and information gain to rank dimensions and measures in the data cube. In addition to this, we are in a process of applying association rule mining on our generated schema to produce association rules and compare them with rules produced without schema (flat-file).

9 References

AHMAD, A. & DEY, L. 2007. A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering,* 63**,** 503-527.

ASUNCION, A. & NEWMAN, D. J. 2010. UCI machine learning repository http://archive.ics.uci.edu/ml Irvine, CA: University of California, School of Information and Computer Science.

BASAK, J. & KRISHNAPURAM, R. 2005. Interpretable hierarchical clustering by constructing an unsupervised decision tree. *IEEE Transactions on Knowledge and Data Engineering***,** 121-132.

BERNSTEIN, A., PROVOST, F. & HILL, S. 2005. Toward intelligent assistance for a data mining process: An ontology-based approach for cost-sensitive classification. *Knowledge and Data Engineering, IEEE Transactions on,* 17**,** 503-518.

CORDES, D., HAUGHTON, V., CAREW, J. D., ARFANAKIS, K. & MARAVILLA, K. 2002. Hierarchical clustering to measure connectivity in fMRI resting-state data. *Magnetic resonance imaging,* 20**,** 305-317.

FABRIS, C. C. & FREITAS, A. A. Incorporating deviation-detection functionality into the OLAP paradigm. the 16th Brazilian Symposium on Databases (SBBD 2001), 2001 Rio de Janeiro,Brazil. 274-285.

GOIL, S. & CHOUDHARY, A. 1997. High performance OLAP and data mining on parallel computers. *Data Mining and Knowledge Discovery,* 1**,** 391-417.

GOIL, S. & CHOUDHARY, A. A parallel scalable infrastructure for OLAP and data mining. 1999. Published by the IEEE Computer Society, 178.

GOIL, S. & CHOUDHARY, A. 2001. PARSIMONY: An infrastructure for parallel multidimensional analysis and data mining. *Journal of parallel and distributed computing,* 61**,** 285-321.

HAN, J. 1998. Towards on-line analytical mining in large databases. *ACM Sigmod Record,* 27**,** 97-107.

HAN, J. & KAMBER, M. 2006. *Data mining: concepts and techniques*, Morgan Kaufmann.

HOFMANN, H. 1994. Statlog (GermanCreditData) [Online]http://archive.ics.uci.edu/ml/datasets/Statlog

KAMBER, M., HAN, J. & CHIANG, J. Y. Metarule-guided mining of multi-dimensional association rules using data cubes. KDD'97, 1997. 207-210.

KOHAVI, R. & BECKER, B. 1996. *Adult dataset* [Online]Available:http://archive.ics.uci.edu/ml/datasets/Adult.

LI, C. & BISWAS, G. 2002. Unsupervised learning with mixed numeric and nominal data. *IEEE Transactions on Knowledge and Data Engineering***,** 673-690.

LIU, Z. & GUO, M. 2001. A proposal of integrating data mining and on-line analytical processing in data warehouse. *Info-tech and Info-net, 2001. Proceedings. ICII 2001 - Beijing. 2001 International Conferences on* Beijing , China.

MESSAOUD, R. B., RABASÉDA, S. L., BOUSSAID, O. & MISSAOUI, R. Enhanced mining of association rules from data cubes. DOLAP '06 Proceedings of the 9th ACM international workshop on Data warehousing and OLAP 2006 New York. ACM, 11-18.

MISSAOUI, R., JATTEAU, G., BOUJENOUI, A. & NAOUALI, S. 2007. Toward Integrating Data Warehousing with Data Mining Techniques. *Data warehouses and OLAP: concepts, architectures, and solutions***,** 253.

NGUYEN, T. M., TJOA, A. M. & TRUJILLO, J. 2005. Data warehousing and knowledge discovery: A chronological view of research challenges. *Data warehousing and knowledge discovery*, 530-535.

PARDILLO, J., ZUBCOFF, J., MAZÓN, J. N. & TRUJILLO, J. 2008. Applying MDA to integrate mining techniques into data warehouses: a time series case study. *Mining Multiple Information Sources MMIS 08.* Las Vegas.

PIGHIN, M. & IERONUTTI, L. 2008. A Methodology Supporting the Design and Evaluating the Final Quality of Data Warehouses. *International Journal of Data Warehousing and Mining (IJDWM),* 4, 15-34.

RIVIÉRE, C. & MARLAIR, G. 2010. The use of multiple correspondence analysis and hierarchical clustering to identify incident typologies pertaining to the biofuel industry. *Biofuels, Bioproducts and Biorefining,* 4, 53-65.

ROSARIO, G. E., RUNDENSTEINER, E. A., BROWN, D. C., WARD, M. O. & HUANG, S. 2004. Mapping nominal values to numbers for effective visualization. *Information Visualization,* 3, 80-95.

SCHLIMMER, J. C. 1985. *Automobile dataset* [Online]. Available:http://archive.ics.uci.edu/ml/datasets/ Automobile

SEO, J. & GORDISH-DRESSMAN, H. 2007. Exploratory data analysis with categorical variables: An improved rank-by-feature framework and a case study. *International Journal of Human-Computer Interaction,* 23, 287-314.

SEO, J. & SHNEIDERMAN, B. 2006. Knowledge discovery in high-dimensional data: Case studies and a user survey for the rank-by-feature framework. *IEEE Transactions on Visualization and Computer Graphics*, 311-322.

USMAN, M. & ASGHAR, S. 2011. An Architecture for Integrated Online Analytical Mining. *Journal of Emerging Technologies in Web Intelligence,* 3, 74-99.

USMAN, M., ASGHAR, S. & FONG, S. 2010. Integrated Performance and Visualization Enhancement of OLAP Using Growing Self Organizing Neural Networks. *Journal of Advances in Information Technology,* 1, 26-37.

USMAN, M. & PEARS, R. 2010. Integration of Data Mining and Data Warehousing: A Practical Methodology. *International Journal of Advancements in Computing Technology,* 2, 31 - 46.

USMAN, M. & PEARS, R. 2011. Multi Level Mining of Warehouse Schema. Networked Digital Technologies, *Communications in Computer and Information Science. Springer Berlin Heidelberg.*136:395-408

ZUBCOFF, J., PARDILLO, J. & TRUJILLO, J. 2007. Integrating clustering data mining into the multidimensional modeling of data warehouses with UML profiles. *Data Warehousing and Knowledge Discovery*, 199-208.