Exploiting Data Mining Techniques in the Design of Multidimensional Schema for Enhanced Knowledge Discovery

A thesis submitted to

Auckland University of Technology (AUT)

in fulfillment of the requirements for the degree of

Doctor of Philosophy (PhD)

Muhammad Usman

2013

School of Computing and Mathematical Sciences

Primary Supervisor: Assoc. Prof. Russel Pears

Secondary Supervisor: Prof. Alvis Fong

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

“I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another author (expect where explicitly defined in the acknowledgements), nor material which to a substantial extension has been submitted to the award of any other degree or diploma of a university or other institute of higher learning.”

Signed: ____

Acknowledgements

The period I spent on my thesis was the most challenging and demanding, yet very rewarding period of my life. I dealt with a variety of research problems in the field of knowledge discovery and learnt valuable problem solving approaches.

First of all, I thankful to almighty ALLAH who taught me eloquence and intelligence. No doubt, these are few of the uncountable blessings of my praiseworthy Lord.

I would like to express my deepest gratitude to my primary supervisor Dr. Russel Pears for his continued guidance and support. His knowledge and experience in the field of Data Mining was of great help. His critiques enabled me to think deeper about the research problems in order to achieve a quality solution.

I would also like to thank my secondary supervisor Professor Alvis Fong for his motivation and encouragement throughout my PhD.

A special thanks to my friends Qi Guo, Affan Toor and Qaiser Maqbool for their endless support and optimistic attitude.

Finally, I want to thank my mother for always praying for my success. I wouldn’t have finished my PhD without her prayers and continuous encouragement.

Abstract

The work done in this thesis encapsulates an area of inquiry that has seen surprisingly little research in the broad and rapidly developing field of knowledge discovery. Both the Data Mining and Data Warehousing disciplines, which lie under the large umbrella of knowledge discovery, are well established in their own right but very little cross-fertilization has taken place between these two disciplines. The integration of Data Mining techniques with Data Warehousing is gaining popularity due to the fact that both disciplines complement each other in extracting knowledge from large datasets. However, the majority of approaches focus on applying data mining as a front end technology to mine data warehouses. Relatively little progress has been made in incorporating mining techniques in the design of data warehouses.

Recently though, there has been increasing interest in adapting clustering techniques that have been developed in the Data Mining discipline to the Data Warehousing environment. Such an adaptation is not easy from a technical viewpoint as a resource such as a Data Warehouse has generally a large community of users, each of which may potentially have different and conflicting data requirements which in turn translates to different clustering requirements for the same data resource. While methods such as data clustering applied on multidimensional data have been shown to enhance the knowledge discovery process, a number of fundamental issues remain unresolved with respect to the design of multidimensional schema which is an integral part of any Data Warehouse. These relate to automated support for the selection of informative dimension and fact variables in high dimensional and data intensive environments, an activity which may challenge the capabilities of human designers on account of the sheer scale of data volume and variables involved.

In this thesis, a novel methodology is proposed which facilitates knowledge workers to select a subset of dimension and fact variables from an initial large set of candidates for the discovery of interesting data cube regions. Unlike previous research in this area, the proposed approach does not rely on the availability of specialized domain knowledge and instead makes use of robust methods of data reduction such as Principal Component Analysis, and Multiple Correspondence Analysis to identify a small subset of numeric and nominal variables that are responsible for capturing the greatest degree of variation in the data and are thus used in generating data cubes of interest. Moreover, information theoretic measures such as Entropy and Information Gain have been exploited to filter out less informative dimensions to construct compact, useful and easily manageable schema. In terms of data analysis, we experiment with association rule mining to compare the rules generated with semi automatically generated schema with the rules gathered without the presence of such schema.

The three case studies that were conducted on real word datasets taken from UCI machine learning repository revealed that the methodology was able to capture regions

of interest in data cubes that were significant from both the application and statistical perspectives. Additionally, the knowledge discovered in the form of rules from the generated schema was more diverse, informative and have better prediction accuracy than the standard approach of mining the original data without the use of our methodology-driven multidimensional structure imposed on it.

Chapter 1

Introduction

This chapter introduces the general context, the aims and the rationale of the thesis with a brief description of each chapter's contents. The motivation for integrating data mining, data warehousing and machine learning in non-conventional application domains is given, followed by identifying the limitations of the existing frameworks to satisfy the requirements of intelligent and semi-automatic data analysis.

1.1 Knowledge discovery from large datasets

Knowledge discovery from large datasets is the result of an exploratory process involving the application of various algorithmic procedures for manipulating data (Bernstein, Provost et al. 2005). It aims to extract valid, novel, potentially useful, and ultimately understandable patterns from data (Fayyad, Piatetsky-Shapiro et al. 1996).
Data mining and data warehousing are two key technologies for discovering knowledge from large datasets. Data mining enables the discovery of hidden trends from large datasets, while data warehousing provides for interactive and exploratory analysis of data through the use of various data aggregation methods.

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).

Likewise, 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, Tjoa 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. More recently, there has been an increasing research interest in the knowledge engineering community towards integrating the two technologies (Goil and Choudhary 2001; Liu and Guo 2001; You, Dillon et al. 2001; Zhen and Minyi 2001; Ohmori, Naruse et al. 2007; Usman, Asghar et al. 2009; Usman and Pears 2010; Usman and Asghar 2011).

1.2 Integrated use of data mining and data warehousing

Both data mining and data warehousing technologies have essentially the same set of objectives and can potentially benefit from each other's methods to facilitate knowledge discovery. Each technology is mature in its own right, and despite the very clear synergy between these two technologies, they have developed largely independent of each other.

The integrated use of data mining and data warehousing techniques such as Online Analytical Processing (OLAP) has received considerable attention from researchers and practitioners alike, as they are key tools used in knowledge discovery from large data datasets (Han 1998; Sapia, Höfling et al. 1999; Goil and Choudhary 2001; You, Dillon et al. 2001; Zhen and Minyi 2001; Ohmori, Naruse et al. 2007; Zubcoff, Pardillo et al. 2007; Pardillo, Zubcoff et al. 2008). (Usman and Pears 2011) used a hierarchical clustering technique in conjunction with multidimensional scaling (Cox and Cox 2008) to design schema at different levels of data abstraction. They developed an iterative method that explores the similarities and differences in information contained across consecutive levels in the cluster hierarchy. The presentation of such information at different levels of abstraction provides decision makers with a better understanding of the patterns and trends present in the data. Although, a variety of integrated approaches have been proposed in the literature to mine large datasets for discovering knowledge. However, a number of issues remain unresolved in the previous work (Sarawagi, Agrawal et al. 1998; Sarawagi 2001; Kumar, Gangopadhyay et al. 2008; Ordonez and Zhibo 2009), especially on intelligent data analysis front.

1.3 Unresolved issues and motivation of the thesis

In this section, we discuss some of the important issues which remained unresolved in the previous approaches of integration. Firstly, the prior work assumed that data analysts could identify a set of candidate data cubes for exploratory analysis based on domain knowledge. Unfortunately, situations exist where such assumptions are not valid. These include high dimensional datasets where it may be very difficult or even impossible to predetermine which dimensions and which cubes are the most informative. In such environments it would be highly desirable to automate the process of finding the dimensions and cubes that hold the most interesting and informative content.

Secondly, reliance on domain knowledge tends to constrain the knowledge discovered to only encapsulate known knowledge, thus excluding the discovery of unexpected but nonetheless interesting knowledge (Koh, Pears et al. 2011). Another related issue is that it restricts the application of these methodologies to only those domains where such domain knowledge is available. However, a knowledge discovery system should be able to work in ill-defined domains (Nkambou, Fournier-Viger et al. 2011) and other domains where no background knowledge is available (Zhong, Dong et al. 2001).

Thirdly, there has been relatively less research in leveraging data mining techniques in the design of data warehouses or multidimensional schema (Sapia, Höfling et al. 1999; Zubcoff, Pardillo et al. 2007; Pardillo, Mazón et al. 2008; Pardillo and Mazón 2010; Usman, Asghar et al. 2010). It is a daunting task for data warehouse developers to integrate the outcomes of data mining techniques with data warehouse to perform analytical operations. The reason of this daunt is the requirement of a multidimensional model or schema for interactive data exploration and designing such schemas is a complex task as it requires extensive domain knowledge along with the expertise in data warehousing technologies. Additionally, modelling requires multiple manual actions to discover important facts and dimensions from the dataset, creating a bottleneck in the knowledge discovery process. Even if the human data warehouse designers try to resolve these problems, an incorrect design with the incorrect choice of dimensions and facts can still be generated if he/she doesn't understand the underlying relationships among the data items. Recent research has proved that in data warehouses the choice of the dimensions and measures heavily influences the data warehouse effectiveness (Pighin and Ieronutti 2008).

Fourthly, there remains a need for automated support in the design of data cubes, especially in domains containing high dimensional data. In such domains the sheer scale of the data, both in terms of data volume as well as in the number of dimensions, may make it difficult for human designers to decide which dimensions are the most informative and should thus be retained in the final version of the data cube. Furthermore, high dimensional and high volume datasets present significant challenges to domain experts in terms of identifying data cubes of interest. The presence of mixed data in the form of nominal and numeric variables present further complications as the interrelationship between nominal and numeric variables have also to be taken into account. A methodology that assists domain experts in identifying dimensions and facts of interest is highly desirable in these types of environments.

Finally, in high dimensional environments the design and data analysis processes need to be integrated with each other. With the use of appropriate information theoretic measures such as Entropy in the design process, less informative dimensions can be filtered out, thus leading to a more compact, useful and manageable schema. In terms of data analysis, the main tool used in multidimensional analysis in a data warehousing environment is the use of various data aggregation and exploratory techniques that form part of the On Line Analytical processing (OLAP) suite of methods. While traditional OLAP methods are excellent tools for exploratory data analysis they are limited as far as detecting hidden associations between items resident in a data warehouse. The discovery of such hidden relationships and associations often yields important insights into underlying trends and in general leads to an improved decision making capability.

The above mentioned issues motivated us to formulate a generic methodology for data cube identification and knowledge discovery that is applicable across any given application domain, including those environments where limited domain knowledge exists.

1.4 Research challenges considered to be out of Scope

In this section, we highlight some of the important research challenges which are relevant to our work but could not be addressed because of the limited timeframe of this PhD research.

- Firstly, the proposed research only targeted two data types namely, numeric and nominal data. However, there are a number of other data types such as multimedia data types (images, audio, video and graphical objects) which are not considered in this research. The analyses of these special data types require the formulation of specialized statistical methods and algorithms that would not be feasible within the limited timescale of a PhD.

- Secondly, we consider the design of only one type of multidimensional schema (STAR schema) in our research. However, there are two other schemas used in typical data warehouse environments known as *Snowflake* and *Fact Constellation* schema. Each schema has unique design and construction requirements and because we were focusing on automating the schema design process it was not feasible to consider multiple schema types.

- Finally, the dimensions that we design in our methodology-driven schema only support two level hierarchies, with the first level consisting of groups and the second consisting of individual values within each group. In practice, a typical data warehouse schema consists of dimensions defined on multiple hierarchical levels. However, the levels in the dimensional hierarchy are specified by human data warehouse designers and it poses a significant research challenge to automatically determine meaningful levels based purely on patterns within the data, without the use of human input. Although we were unable to automate the process of designing multi-level dimensions due to lack of time, we elaborated the procedure of extending our proposed method to accommodate multiple hierarchies in the future work section in Chapter 8 of this thesis.

1.5 Problems to be addressed

The main research problems addressed in this thesis are as follows:

- To provide automated support in multidimensional schema design and in identifying cubes of interest from multidimensional data viewed at different levels of data abstraction. This is a significant research issue in high volume, high dimensional data environments as specialized domain knowledge is likely to be of limited value in cube design in such environments.

- To facilitate analysts to effectively discover knowledge in the form of diverse association rules from large multidimensional cube structure.

1.6 Aims and Contributions

The title of this thesis reflects the overall goal of this work, which is to exploit data mining techniques in the design of multidimensional schema for enhancing knowledge discovery. Although data mining and data warehousing area have reached the state of maturity, new challenges arise when integrating the established technology especially to novel usage scenarios. Real-world case studies from the diverse domains of automobile, census and ecology are used to exemplify the challenges and motivate the proposed solution. The complementary ideas which determine the contributions of this thesis are as follows:

- To provide automated and data-driven support for the design and construction of multidimensional schema.
- To generate cubes of interest at different levels of data abstraction and study the effect of abstraction level on information content.
- To identify, at each level of data abstraction, the most significant interrelationships that exist between numeric and nominal variables, thus enabling the data analyst with pathways to explore the data.
- To discover diverse and meaningful association rules from multidimensional cube structure at various level of data abstraction.

The primary focus of the work is on extending the applicability of integrated approaches towards knowledge discovery by identifying the bottle-necks of the integrated approaches and searching for the ways to overcome the identified limitations. We have made the following main contributions in this thesis:

- Proposed a knowledge discovery methodology that utilizes a combination of machine learning and statistical methods to identify interesting regions of information and diverse association rules in large multidimensional data cubes.
- Provided an algorithm for constructing a binary tree from hierarchical clustering results (dendrogram).
- Proposed a measure based on Information Gain and Multiple Correspondence Analysis (MCA) to identify and rank the most informative dimensions among nominal variables that should be retained for schema design.

- Applied well-known dimension reduction techniques such as Principal Component Analysis (PCA) in order to identify and rank the most informative numeric facts present in high dimensional datasets.

- Generated informative cubes at different levels of data abstraction and studied the effect of abstraction level on information content. At each level of data abstraction we identify and rank the most significant interrelationships that exist between numeric and nominal variables, thus enabling the cubes of interest to be identified.

- Provided methods to construct candidate schema with highly ranked dimensions (nominal variables) and measures (numeric variables).

- Performed case studies on three real-world datasets to validate our methodology and showed that it enables analysts to find cubes of interest and the diverse association rules. Furthermore, we showed that rules generated from our semi-automatically generated multidimensional schema are in general more diverse and have better predictive accuracy than rules generated from the same data without the use of the multidimensional schema.

- Performed in-depth scalability study to validate that our methodology scales well with both large volume and high dimensional datasets.

1.7 Novelty and significance

The proposed methodology is novel from the viewpoint of its research objectives. To date, no systematic study has been proposed in the literature to investigate the following issues

- To study the dynamics of the relationships between nominal and numeric variables at different levels of data abstraction in a multidimensional data context.

- To provide automated support for the design of multidimensional schema and construction of informative data cubes in those environments where limited or no domain knowledge is available.

- To allow the discovery of diverse association rules from multidimensional cube structure.

The successful identification of regions of interest in data cubes and diverse association rules within high volume, high dimensional data represents a significant contribution to the research literature on multidimensional data analysis. To date, a few solutions have been proposed for this problem in the literature, but they tend to rely on expert users supplying information to guide the discovery process. Our proposed methodology does not assume that user input is available, but is flexible enough to accommodate such information should it be available. The proposed methodology would also be of interest to practitioners as commercial interest in high volume, high dimensional data analysis continues to grow. However, the proposed methodology definitely does not exhaust the challenges of comprehensive and completely automatic analysis for non-conventional domains, but we expect it to be useful for a wide range of application scenarios.

1.8 Thesis Outline

To address the specified research aims this thesis is outlined as follows:

Chapter 2 provides the review of the four main themes of related work in the context of our research. Section 2.1 discusses previous research in identifying data cubes that hold the greatest information content. Section 2.2 is dedicated to the review of previous research on identifying relationships between mixed data types. In Section 2.3, we present the work done so far in automating the design and construction of multidimensional schema. Section 2.4 is dedicated to the coverage of the application of association rule mining to enhance knowledge discovery from multidimensional schema. Finally, we review our prior work in section 2.5 which is closely related to the themes of the literature review presented in this chapter and which also forms the foundation of the work done in this thesis.

Chapter 3 presents an overview of the proposed methodology for multidimensional cube design that facilitates the discovery of interesting cube regions and diverse association rules. Moreover, we illustrate the methodological steps with a running example. We demonstrate that classical statistical methods for data analysis such as Principal Component Analysis (PCA) and Multiple Correspondence Analysis (MCA) can be successfully used in conjunction with hierarchical clustering to uncover useful information implicit in large multidimensional data cubes. Moreover, information gain measure can be effectively used to discover diverse association rules with high prediction accuracy. Section 3.1.7 shows that our methodology facilitates easy discovery of inter-relationships between numeric and nominal variables which are significant from both application and statistical perspectives. Furthermore, this useful and interesting knowledge can be discovered in the form of association rules without excessive reliance on specialized domain knowledge as explained in sections 3.1.8. Finally in section 3.1.9 we show that the multidimensional schema generated through our methodology gives diverse association rules with better predictive power as compared to the rules generated without the multidimensional structure imposed on it.

Chapter 4 presents the application of the proposed methodology on the first case study performed on real-world dataset, namely *Automobile* (Schlimmer 1985), taken from the well-known UCI machine learning repository (Asuncion and Newman 2010). This benchmark dataset describes the specification of an automobile in terms of various characteristics, its assigned insurance risk rating and its normalized (financial) losses in use as compared to other automobiles. The *Automobile* dataset has a small number of records, only 205, but has a rich mix of 11 nominal and 16 numeric variables that suits the objectives of our research.

Chapter 5 presents our second case study conducted using a larger dataset as compared to *Automobile* dataset. It is the *Adult* (Kohavi and Becker 1996) dataset which consists of 48,842 records with 8 nominal and 5 numeric variables. This benchmark dataset was extracted from the US Census bureau website using a data extraction system and is available for download from the UCI machine learning repository.

Chapter 6 presents our third case study conducted on a much larger dataset called *CoverType* (Blackard, Dean et al. 1998). This is currently one of the largest datasets in the UCI repository containing 58,1012 records with 54 variables (42 nominal and 12 numeric) and 7 target classes (Obradovic and Vucetic 2004). This benchmark dataset is used to predict forest cover types from cartographic variables. Forest cover type is basically defined as a descriptive classification of forest land based on occupancy of an area by the tree species present in it. The main motivation behind choosing this dataset is that it poses extreme challenges to the analysts in finding useful and interesting knowledge from the rich mix of nominal and numeric variables and sheer size of data. To give a glimpse of the difficulties involved in mining association rules from this complex dataset, we present some interesting facts for this dataset discovered by (Webb 2006) which motivated us to mine diverse association rules from large datasets. Webb 2006 reported the results of this particular dataset and observed that not a single non-redundant rule generated through *CoverType* dataset was found to be productive. The identified fact that all non-redundant associations for this dataset represented unproductive associations highlights the dangers of data mining without sound methodologies for discovering meaningful and diverse association rules.

Chapter 7 present experiments conducted on synthetic datasets to test the scalability of our proposed methodology. An important issue in our approach is to ensure that the proposed methods do not become a bottleneck in an environment where a large number of records or high dimensionality is present. To address this issue, the focus of this chapter is to show that the each step of the proposed methodology indeed scales with size and dimensionality of data. We have implemented a full-fledged prototype, i.e., for generating synthetic data with various parameters, and have conducted an extensive experimental evaluation to compare the processing time of each step of our proposed methodology. The key variables that we have identified for our scalability study are data size (in terms of number of records) and dimensionality (in terms of number of dimensions/variables).

Finally, Chapter 8 summarizes the contributions of this thesis, draws conclusions and identifies future research directions which we regard promising in the context of this thesis.

1.9 Publications from thesis

The following research papers have been written and published during the course of this research.

1. Usman, M., R. Pears, Fong. ACM (2013). "A data mining approach to knowledge discovery from multidimensional cube structures." Knowledge-Based Systems 40(0): 36-49.

2. Usman, M., R. Pears, Fong. ACM (2013). "Discovering Diverse Association Rules from Multidimensional Schema." Expert Systems with Applications 40(15): 5975-5996.

3. Usman, M., R. Pears, Fong. ACM (2012). "Data guided approach to generate multi-dimensional schema for targeted knowledge discovery." 10th Australasian Data Mining Conference (AusDM12), 229-240.

4. Usman, M. and R. Pears (2011). Multi Level Mining of Warehouse Schema. Networked Digital Technologies, Springer Berlin Heidelberg. 136: 395-408.

5. Usman, M. and R. Pears (2010). "Integration of Data Mining and Data Warehousing: A Practical Methodology." International Journal of Advancements in Computing Technology 2(3): 31 - 46.

6. Usman, M. & Pears, R. (2010) A methodology for integrating and exploiting data mining techniques in the design of data warehouses. 6th International Conference on Advanced Information Management and Service (IMS), Nov. 30 2010-Dec. 2 2010. 361-367.

Chapter 2

Literature Review

In this chapter, we review four main themes of literature related to the contributions made in this thesis. Firstly, we discuss previous research in identifying data cubes that hold the greatest information content. Secondly, we review previous research on identifying relationships between mixed data types. Thirdly, we present work in automating the design and construction of multidimensional schema. Fourthly, the application of association rule mining to enhance knowledge discovery from multidimensional schema is covered. Finally, we review our prior work which is closely related to the themes of the literature review presented in this chapter and which also serves to form the foundation of the work done in this thesis.

2.1 Identification of informative data cubes

A limited number of approaches have been proposed in the past in order to identify data cubes that hold the greatest information content. In this section we present the major contributions in this field and identify the most prominent techniques having similar objectives to the work done in this thesis.

Sarawagi et al. (1998) explored methods for guiding users towards the discovery of interesting cube regions. The authors focused on identifying regions within the data cube where cells contained values that were significantly different from an expected threshold value calculated via a regression model. This work was extended further by Sarawagi (Sarawagi 2001), whereby differences in cell values across regions were used to find surprising information in unexplored areas of a data cube based on the concept of maximum entropy.

According to Kumar et al. (Kumar, Gangopadhyay et al. 2008), the work done in (Sarawagi, Agrawal et al. 1998; Sarawagi 2001) defined surprises in a rigid manner, implying that users cannot view them differently according to their needs. Furthermore, the discovered surprises are not easy to understand and interpret by merely scanning high dimensional data presented in a large number of rows and columns. Kumar et al. (Kumar, Gangopadhyay et al. 2008) overcame these limitations by proposing an DIscovery of Sk-NAvigation Rules (*DISNAR)* algorithm for detecting surprises defined by users and establishing the concept of cube navigation using the detected surprises. The proposed *DINSAR* algorithm utilized a *Gaussian* distribution for detecting skewed nodes existing in cube lattices. It consisted of a four step recursive process. Firstly, it generates a set of candidate nodes for a given node. Secondly, it measures the skewness of candidate nodes. Thirdly, it applies a test of significance of skewness proposed by (D'Agostino and Stephens 1986) on candidate nodes and finally, transform nodes with significant skewness into cube navigation rules .The algorithm terminates either when it reaches the lowest level nodes in the cube lattice or when no more nodes of surprises are identified in the current iteration. The proposed rule based approach provides a method of guidance for cube navigation in order to enhance the cube exploration capabilities.

Our research exhibits some similarities with (Kumar, Gangopadhyay et al. 2008) and (Sarawagi 2001) in the sense that we also assist the user by providing candidate interesting cube regions for exploration at multiple levels of data abstraction. In addition to this, we provide greater level of guidance to users than (Kumar et al. 2008) by explicitly ranking paths on the basis of an information content measure based on entropy.

In the medical field, statistical methods were applied on cubes by (Ordonez and Zhibo 2009) to improve disease diagnostics. The authors proposed the integration of OLAP cube exploration with parametric statistical techniques to find significant differences in facts by identifying a small set of discriminating dimensions. However, this work is limited in terms of understanding the interrelationships between the significant facts and dimension variables. Additionally, the work can only be utilized after the construction of a data cube. There is no facility for the user to construct a constrained data cube having important dimensions and facts beforehand for further cube exploration, as proposed in this thesis. Furthermore, their application of statistical tests requires prior domain knowledge in order to pinpoint regions where the significant fact differences are suspected.

Moreover, as Koh et al. (2011) argue, heavy reliance on domain specific information only leads to the discovery of known patterns that fit a preconceived template and has the danger of inhibiting the discovery of unknown hidden patterns present in the data. Motivated by this, they proposed a generic solution for discovering informative rules by automatically assigning item weights based on the strength of interactions between them in a transactional database. In order to evaluate the informative rules generated by their proposed method, they utilized Principal Components Analysis (PCA) to capture the amount of variance by each rule term (the actionable component of the rule). The higher the variance captured for a rule term, the greater the significance of the rule is as a whole. Our work is similar as we also provide a solution without reliance on domain specific information. However, our proposal differs from their work as the emphasis is on the discovery of interesting cube regions. Additionally, our use of PCA is not to evaluate results but to rank the numeric variables in order of significance. Such a ranking provides guidance to the user to choose fact variables of his/her own choice.

More recently, a neural network based approach has been proposed by (Abdelbaki, Ben Messaoud et al. 2012) that predicts measures over high dimensional data cubes. The authors introduced a new two stage approach based on the novel concept of PCA-cubes. The first stage is data pre-processing in which PCA has been utilized to reduce data cube dimensionality. For the second stage, an OLAP oriented architecture embedded with a Multilayer Perceptron (MLP) was introduced for prediction purposes. The MLP learns from multiple training sets to perform prediction on each targeted measure. The authors termed their neural network based approach as: Neural Approach to Prediction over High Dimensional Cubes (NAP-HC). The experimental study showed that NAP-HC has largely met its goals in the case of data cubes exhibiting low levels of sparsity. However, its performance degrades when applied on highly sparse data cubes. Our work is similar in terms of the usage of PCA to reduce high dimensional data. However, we do not apply PCA after the construction of data cubes; instead we use PCA to filter out the less informative dimensions in order to construct a compact and more informative data cube.

It is apparent from the review in this section that a limited variety of approaches have been proposed in the literature to mine large data cubes for discovering knowledge. However, a number of issues remain unresolved in previous work (Sarawagi, Agrawal et al. 1998; Sarawagi 2001; Kumar, Gangopadhyay et al. 2008; Ordonez and Zhibo 2009), especially on the intelligent data analysis front.

Firstly, prior work has assumed that data analysts could identify a suitable set of candidate data cubes for exploratory analysis based on domain knowledge. Unfortunately, situations exist where such assumptions are not valid. These include high dimensional high volume (in terms of number of instances) datasets where it may be very difficult or even impossible to predetermine which dimensions and which cubes are the most informative. In such environments it would be highly desirable to automate the process of identifying dimensions and cubes that hold the most interesting and informative content.

Secondly, as stated earlier, excessive reliance on domain knowledge tends to constrain the knowledge discovered to only encapsulate known knowledge, thus excluding the discovery of unexpected but nonetheless interesting knowledge (Koh, Pears et al. 2011). Another related issue is that it restricts the application of these methodologies to only those domains where such domain knowledge is available. However, a knowledge discovery system should be able to work in ill-defined domains (Nkambou, Fournier-Viger et al. 2011) and other domains where no background knowledge is available (Zhong, Dong et al. 2001).

Finally, these approaches mostly target a specific data type. In the real world, datasets have a mix of numeric and nominal variables, often involving high cardinality nominal variables, thus challenging the analytic capability of the methods employed. In the following section, we review some of the machine learning approaches that tackled the problem of mixed data analysis.

2.2 Identification of relationships between mixed data types

Real world datasets consist of a mix of numeric and nominal data. Specifically, data sets with a large number of nominal variables, including some with large number of distinct values are becoming increasingly common (Rosario, Rundensteiner et al. 2004). For the purpose of efficient analysis of such mixed variable datasets, (Ahmad and Dey 2007) identified the problems associated with the traditional *k-means* algorithm as it is best suited for numeric data. In order to perform analysis on mixed data, the authors proposed a new algorithm which uses a cost function and distance measure based on co-occurrence of values. The proposed cost function alleviated the shortcoming of Huang's (Huang 1997) cost function. However, the limitation of the proposed work is that the analysis still relies on co-occurrences of data and discretization of numeric values which leads to a loss of information. In addition to this, the work does not support the identification of semantic relationships amongst values in nominal variables. For instance, nominal variables such as Product-category, Product-names, and Product-codes etc. in general contain a large number of distinct values and therefore require efficient methods for revealing inter-relationships amongst different values.

For the same purpose, a feature selection algorithm for mixed data containing both continuous and nominal features was introduced by (Tang and Mao 2007). The authors stressed that feature selection is a crucial step in pattern recognition and that most of the feature selection algorithms do not target mixed data containing both nominal and numeric features. Tang and Mao (2007) proposed a mixed forward selection (MFS) search algorithm for mixed feature space. MFS starts with an empty set and selects a one step-optimal feature at each step, but the selection is done through two stages. In the first stage, MFS searches the optimal nominal and continuous features separately. In the second stage it selects the step-optimal feature from the two candidate features identified in the first stage through the comparison of classification accuracy. In their experimental study, *Mahalanobis* distance and symmetrical uncertainty are employed as the evaluation criteria for continuous and nominal features respectively. The stopping criterion in the MFS algorithm is either a predefined feature subset size or the cross-validated error rate. The limitation of the proposed algorithm is that its performance on real word data was significantly inferior in comparison to that of synthetic data. In addition, the scalability of memory and runtime was not assessed with respect to the number of features.

In the same quest for mixed data analysis, three different distance measures for computing Mahalanobis-type distances were compared by McCane and Albert (McCane and Albert 2008). They identified the fact that there is a strong need to develop *Mahalanobis-type* distances for mixed data type variables. They observed that research done in mixed data analysis is either heuristic or is only based on the use of nominal data, with the exclusion of numeric variables. In their work, *Mahalanobis-type* distances were computed between random variables consisting of several categorical dimensions or mixed categorical and numeric dimensions. In each case, distances are computed via an interpretation of the categorical data in some real vector space. Authors tested the three methods on two application domains namely, classification and principal component analysis and found that overall only one method (regular simplex) was successful in both domains. The basic idea of the regular simplex method is to assume that any two distinct levels of a categorical variable are separated by the same distance. To achieve this, each level of an *n*-level variable is associated with a distinct vertex of a regular simplex in (*n*-1) dimensional space. The strength of their work is the comparison of measures for computing *Mahalanobis-type* distance measures between categorical and numeric dimensions. However, the authors have used very small data sets having only a few records to perform the validation; hence, it is unclear how their method will scale for large datasets.

Hsu et. al (Hsu, Chen et al. 2007) focused on hierarchical clustering of mixed data based on a distance hierarchy. The proposed work differs from the work of (McCane and Albert 2008) as their work only considered nominal variables in the computation of distance measure. For clustering mixed data, it was reported that most of the clustering algorithms operate on numeric data and only a few can support the analysis of mixed numeric and nominal data (Milenova and Campos 2002). The authors extended the existing Orthogonal partitioning clustering (O-Cluster) algorithm (Milenova and Campos 2002) to work in domains containing both nominal and mixed variable data types. The O-Cluster algorithm combines a novel active sampling technique with an axis-parallel partitioning strategy to identify continuous areas of high density in the input space. It computes the histograms of active partitions to find the best splitting points. It operates on a limited memory buffer and requires at most a single scan through the data. Furthermore, it proposes the use of a statistical test, namely *chi-squared* test for the identification of good splitting points along data projections and makes possible automated selection of high quality separators. The algorithm relies on an active sampling method to accomplish scalability with large volumes of data. Similar to the work of (Hsu, Chen et al. 2007), the proposed extended (O-Cluster) algorithm uses axis-parallel partitioning to build a hierarchy and identifies hyper-rectangular regions in the input feature space.

Doring et al. (Doring, Borgelt et al. 2004) proposed a fuzzy clustering approach based on a probabilistic distance feature. Authors stressed on the fact that clustering mixed feature-type data is a major data analysis task. This algorithm is based on a probabilistic model and thus circumvents the problems of weighting dissimilar components that can result from separately computing distances based on different data types. The clusters formed from this approach contain the weighted means and covariance matrices of numeric attributes and weighted frequencies of the nominal attributes categories. The weakness of the proposed work is that the clustering process is driven purely by the of nominal variables without explicit contribution from the numeric variables. Moreover, the experiments were performed only on synthetic data sets without the use of real-world data.

Luo et. al (Luo, Kong et al. 2006) targeted the same area of clustering mixed data. Luo et al presented an Evidence based Spectral Clustering Algorithm (EBSC) that works well for data containing a mix of both nominal and numeric features. Authors proposed a method for building the co-association matrix (Fred and Jain 2005) for datasets with mixed variable data types. The co-association matrix is a widely used data structure for combining information from multiple clustering runs (Tsaipei 2011). The idea of evidence accumulation is based on the co-association matrix as it views each clustering result as an independent evidence of data organization and combines the results into a single data partition. The proposed EBSC algorithm first obtains N clustering results by running k-means N times with random initializations on a pure numeric subset and M clustering results for M nominal attributes. Secondly, co-occurrences of pairs in the same clusters are taken as votes for association. Thirdly, data partitions are mapped into a similarity matrix of patterns and finally a spectral clustering method is applied to obtain the final clustering result.

The performance of the EBSC algorithm was evaluated on real world data sets. It was claimed that the measure based on evidence accumulation works well with mixed data types. The major weakness in their work is that they created either numeric features or nominal features but neglected the use of mixed features in their experimentation on synthetic data sets. Furthermore, the real world data set selected for experimentation had very few high cardinality nominal variables and the proposed method does not specify how such variables will be dealt with by the algorithm. In addition to this, the size of the data set used was very small and thus a reliable accuracy comparison could not be made with previous existing algorithms on mixed data analysis.

Li and Biswas (Li and Biswas 2002) demonstrated that the similarity measure proposed by (Goodall 1966) works well with data with mixed nominal and numeric features. Authors proposed a Similarity-based agglomerative clustering (SBAC) algorithm that utilizes Goodall similarity measure and hierarchical agglomerative approach for clustering. The Goodall similarity measure is determined by the uncommonness of attribute-value matches and defines a structure to deal with mixed variables with similarity among objects. Specifically, a pair of objects (i, j) is more similar than another pair of objects (x, y), if, and only if, the objects i and j have a larger match in attribute values which occur relatively less often in the overall dataset (Hsu and Huang 2008). (Hsu and Chen 2007) identified that their proposed approach assumes that the variables are independent and a major limitation of their work is the use of simple matching approach of computing the discrepancy for handling categorical variables. This simple matching technique ignores the semantic information embedded between categorical values of each variable as it uses a traditional value subtraction method for comparing distances between nominal variables.

For the purpose of identifying semantic information between nominal variables and the effective visualization of high cardinality nominal variables a new technique was proposed by (Rosario, Rundensteiner et al. 2004), called Distance-Quantification-Classing (DQC) approach. The (DQC) approach pre-process the nominal variables, calculates the distance between the variables, assigns order and spacing among the nominal values in each variable and finally determines which values are similar to each other and thus can be grouped together. The authors investigated an assignment of order and spacing among nominal data with a large number of distinct values to highlight the relationships among the data points. Categorical data has been displayed using a well-established visualization technique called parallel coordinates (Inselberg and Dimsdale 1991). By this the space on each coordinate is used more efficiently because the spaces become meaningful as similar values are positioned close to each other (Kosara, Bendix et al. 2006). In this thesis, we have utilized the DQC technique for efficient mapping of nominal values to numbers in order to discover the semantic relationship among values in nominal variables.

It is evident from the literature discussed in this section that to tackle the long standing problem of mixed data analysis a variety of clustering algorithms have been proposed (Li and Biswas 2002; Doring, Borgelt et al. 2004; Luo, Kong et al. 2006; Ahmad and Dey 2007; Becue-Bertaut and Pages 2008; Hsu and Huang 2008; Chatzis 2011; Ji, Han et al. 2012) ranging from hierarchical clustering, k-means clustering, fuzzy clustering and incremental clustering algorithms. However, none of these approaches have been integrated with statistical methods to provide assistance towards the discovery of interesting information, as proposed in this thesis. The main consequence of not using statistical methods in the past is that it led users to the discovery of previously known patterns because the data exploration process relied heavily on user’s subjective knowledge. In the real world, it is extremely hard for novice users, and even for experts, to have a clear idea of the underlying data in a large multi-dimensional space. Statistical methods such as Principal Component Analysis (PCA) and Multiple Correspondence Analysis (MCA) help in constraining a large multi-dimensional space by filtering out less informative dimensions and retaining the important ones. This allows users to have meaningful statistical information that they can use together with any specialized domain knowledge that may be relevant in identifying relationships between mixed data types.

We close this section by presenting the overall limitations of previous research in relation to the problem that we are examining. It is evident that limited research has been conducted in the area of finding interrelationships between numeric and nominal variables that are increasingly becoming common in real-world datasets. Moreover, the predominant statistical techniques for analysis of such variables lack integration with machine learning methods for finding interrelationships between variables.

In the following section, we present the third theme of our literature review which covers the work done in in automating the design and construction of multidimensional schema.

2.3 Automating the design of multidimensional schema

A number of schema design approaches have been proposed in the literature to provide automated support for the design of multidimensional schema. Pardillo et al. (2010) identified that most of the research in schema design focuses on the automatic derivation of database schemata from conceptual models but does not address the problem of design of multidimensional schema. Pardillo et al. emphasized that the main issue in data warehouse construction is multidimensional modelling. To resolve this issue, they proposed a Model Driven Architecture Framework (MDA) approach for multidimensional modelling. In their approach, they built different MDA models using an extension of the Unified Modelling Language (UML) and Common Warehouse Meta-model (CWM) (Poole and Mellor 2001) to formally establish transformations between MDA models using a query language. UML integration with CWM allows end-user tools to query the multidimensional schema accurately and reduce design/development time.

Likewise, Dori et al. (2008) suggested an Object-process-based Data Warehouse Construction Method (ODWC) for designing multidimensional schema. Dori et al. suggested that the suitability of current multidimensional modelling methods for large-scale systems is questionable, as they require multiple manual actions to discover measures and relevant dimensional entities and they tend to disregard the system's dynamic aspects. They proposed the ODWC method which utilizes the conceptual model of operational systems to construct a corresponding multidimensional schema. The method operates by first selecting business processes and models them in the form of snowflake schemas. Secondly, it selects the schema that is most appropriate for the organization’s data mining needs. The main limitation of the ODWC method is the strong assumption that business processes are well defined by the organization. There could be cases in which the organization’s business processes are not well established or not aligned with the organization’s data mining needs. In such cases, the proposed method will require additional manual actions to configure business processes for the generation of meaningful snowflake schemas.

A similar semi-automatic technique has been proposed by Palopoli et al. (2002) for generating multidimensional schema from operational databases. The proposed approach first collects subsets of operational schema in the form of Entity-Relationship (ER) models and a dictionary of lexical synonymy properties into homogeneous clusters and then integrates those schemas on a cluster-by-cluster basis. Each integrated schema thus obtained is then abstracted, to construct a global schema representing the cluster. The aforementioned process is iterated over the set of cluster schemas, until only one schema is left. Based on the final schema a single unified data warehouse schema is generated.

However, in spite of its advantages, the proposed approach has a fundamental limitation in those application environments where complex heterogeneous operational systems exist. In such environments, the single global scheme obtained by integrating schemes of operational databases is likely to consist of a large number of data objects, and becomes enormously complex to be effectively used. As a result, the derivation of decision support information becomes quite a difficult task. Apart from this limitation, this approach has only been tested on a single case study and requires further cases to be studied before it can be considered as a comprehensive modelling technique with wider applicability.

To accomplish ease in schema design, Tryfona et al. (1999) built a conceptual model (StarER) for multidimensional modelling on the basis of user modelling requirements. The StarER model combines the star structure, which is dominant in data warehouses, with the semantically rich constructs of the ER model. Examples from a mortgage data warehouse environment, in which StarER has been tested, revealed the ease of understanding the model, as well as its efficiency in representing complex information at the semantic level.

In the quest to propose a generic modelling technique, Hahn et al. (2000) proposed the generation of a tool specific OLAP schemata from conceptual graphical models. A new approach named Bablefish, has been suggested to generate multidimensional schema. It allows graphical representation of a conceptual schema for interactive modelling purposes. Furthermore, the proposed approach discusses the issue of translating graphical representations to configurations for real-world OLAP tools and introduces a View concept that allows designers to model interconnections of static schema with other aspects of warehouse design such as transformation modelling, data source modelling, security modelling etc. The main benefit of the approach is its applicability to Greenfield situations when no system is already in place. In real world development, there exist cases in non-business domains (Mansmann 2009) where the data warehouse developer needs to design schema where an operational system does not exist. Furthermore, such domains may be ill-defined domains (Nkambou, Fournier-Viger et al. 2011), thus compounding the difficulty of the design problem. Therefore, their research is similar to the work undertaken in this paper as we also target those cases where limited domain knowledge exists and no operational system is in place. However, the scope of our work is wider as we not only generate schema but also equip analysts to discover knowledge from the generated schema.

Peralta et al. (2003) stated that design automation usually focuses on data models, data structures and criteria for defining table partitions and indexes. A rule-based mechanism was proposed to automate the design of multidimensional schema. A set of design rules embedding design strategies decide the application of suitable transformations in order to generate logical multidimensional schema. The proposed system has been prototyped by applying design rules using an algorithm that takes into account frequent occurring design problems suggested in existing methodologies. Likewise, an automatic tool for generating a star schema from an Entity-Relationship Diagram (ERD) was introduced by Song et al. (2008). A prototype named SAMSTAR was presented, which was used for the automatic generation of star schema from an ERD. With this automatic generation of star schema, the system helps designers to reduce their effort and time in building data warehouse schemas.

More recently (Usman, Pears et al. 2013) proposed a methodology for the design of multidimensional schema and discovery of interesting cube regions in multidimensional datasets. The distinctive feature of their approach is the use of robust data reduction methods such as PCA and Multiple Correspondence Analysis (MCA) to identify variables that capture the greatest degree of variation in high dimensional data. Such variables were utilized to design schema and construct informative data cubes which contain interesting regions of information. These informative data cubes were generated at different levels of data abstraction and the effect of abstraction level on information content was studied through OLAP analysis.

However, OLAP analysis is limited to exploratory analysis and was not designed to discover interesting associations among data variables (Ben Messaoud, Loudcher Rabaseda et al. 2007). This limitation of OLAP motivated us in our current research to augment OLAP analysis with association rule mining methods to discover interesting relationships and associations among data variables at multiple levels of data abstraction. In the following section, we present the work done to augment OLAP analysis with association rule mining for enhanced knowledge discovery.

2.4 Enhanced knowledge discovery from multidimensional schema

As cited by a number of authors (Kaya and Alhajj 2003; Nestorov and Jukic 2003; Ben Messaoud, Loudcher Rabaseda et al. 2007), Kamber et al. (1997) were the first to target the issue of discovering associations rules in a multidimensional environment. In their proposed approach, a user specifies hypotheses in the form of meta-rules or pattern templates. A mining system then attempts to confirm the provided hypotheses by searching for patterns that match given meta-rules. The use of pattern templates ensures that rules found are of interest to the user. This method also has the advantage of making the rule discovery process efficient as the search for rules is conducted in a space constrained by the templates specified. However, the main drawback is that interesting rules that fall outside the template scope will not be discovered and this will happen when the user is unaware of unexpected and interesting patterns due to limited knowledge of the underlying data.

Zhu and Han in Zhu (1998) proposed an approach towards mining three types of multidimensional association rules, namely intra-dimensional, inter-dimensional and hybrid association rules. The proposed method leveraged OLAP technology to perform multi-level association rule mining on different levels of the dimensional hierarchy. However, interestingness evaluation of the generated rules was confined to correlation analysis of the left-hand side with the right-hand side of the rules and it is not clear how the rules would perform on other objective rule interest measures such as, for example, the rule diversity measure (Geng and Hamilton 2006; Zbidi, Faiz et al. 2006).

In order to generalize the methods of multidimensional association rule mining, Psaila et al. (2000) proposed a new approach to exploit the concept hierarchies present in the multidimensional model. With this approach, data miners reduce complexity in multidimensional data by exploiting concept hierarchies to guide the mining process towards potentially interesting mining queries. To estimate the degree of rule interestingness, the authors employed a metric that was specifically designed for analyzing sales data. The utility of rules generated against generic interest measures such as confidence, lift, etc was not explored in this research.

Ng et al. (2002) focused on applying association rule mining to the most commonly used warehouse schema type, the STAR schema. They proposed an efficient algorithm which makes use of the properties of the STAR schema and experimental results showed that the proposed algorithm significantly outperforms the conventional approach of joining the dimension tables first and then mining rules from the logically joined tables. This proposed joining process of merging the dimension tables present in the schema before mining rules, however, requires the data to be loaded from the data warehouse before applying association rule mining which is computationally expensive when the volume of data stored is large.

In order to overcome this limitation, Chung and Mangamuri (2005) introduced another improved algorithm named Star-miner that can be implemented directly on the relational database system without the need for relational joins. However, their experimentation was confined to synthetic data and hence there was no indication of how well the approach would work on real world data. A similar approach for mining the STAR schema was proposed in Nestorov and Jukic (2003). The authors proposed a framework that enabled ad-hoc data mining queries to be run directly on the multidimensional warehouse structure. The proposed framework expanded the domain of application of association rule mining from the transactional level to the aggregate data level. The use of dimensional information resulted in more detailed and actionable rules. Experimentation also showed that the rules could be generated much faster than with the conventional approach of generating rules from the transactional level. This research revealed new insights into the usefulness of extracting knowledge from multidimensional data vis-a-vis transactional data.

In order to mine association rules from data warehouses, Tjioe and Taniar (2005) proposed a pruning approach to filter non informative data in a data warehouse with the objective of discovering interesting rules. The authors proposed four algorithms *VAvg, HAvg, WMAvg* and *ModusFilter* which focus on pruning all rows present in the fact table that have less than the average quantity sold, average price, and so forth. After pruning, the resultant data is stored in tables called *initialized tables* as they provide efficient data initialization for mining rules. The algorithms then efficiently use these tables to mine association rules by focusing on summarized data present in the data warehouse. Authors have conducted a performance evaluation to show the effectiveness of the proposed row pruning methods. However, no evaluation has been conducted to assess the interestingness of the rules discovered through these proposed methods.

Messaoud et. al in (Messaoud, Rabaséda et al. 2006) highlighted another limitation in Tjioe's work, referring to the fact that it is limited only to the COUNT measure for mining rules from aggregated data. They proposed another method of mining rules based on aggregation measures such as *sum, avg, min* and *max*. They used criteria for the evaluation of rule importance such as *Lift* and *Loevinger* measures. This work has been extended by proposing an online environment for mining association rules called OLEMAR (Ben Messaoud, Loudcher Rabaseda et al. 2007). In the extended work visual representation has been added in order to visualize the importance of the discovered rules using Graphic Semiology principles. A real world case study conducted on a breast cancer dataset illustrated the efficiency and effectiveness of the proposed work. However, there is still a need to evaluate the performance of the discovered rules on objective measures of interest.

As with research in the area of multidimensional schema design, gaps exist in previous research in the area of knowledge discovery from multidimensional data. It is apparent from the review that all the previous approaches support constraining search space for rule discovery, either in the form of pattern templates or in the form of aggregated data. While aggregated data has been shown to be a better foundation for generating rules, the evaluation for the most part, with the exception of Messaoud et al. (2006), used support and confidence criteria to measure the importance of rules. These measures are best suited to transactional data but do not adequately measure the effectiveness of rules generated from data represented at different levels of data granularity. The complexity of multidimensional structures requires the use of more sophisticated measures to quantify the interestingness of rules discovered at different levels of data abstraction.

2.5 A multi-level approach to design multidimensional schema

In this section, we summarize the prior work of (Usman and Pears 2011) which forms the foundation for our current research. The authors suggested that the expertise of a human data warehouse designer, with his/her limited knowledge of the domain may not be effective in high data volume and high dimensional environments. Furthermore, they pointed out that nominal variables, while being candidates for dimension variables, may not always be suitable candidates for use. This was for two reasons: firstly nominal variables which have low information content do not add value to the knowledge discovery process and could thus be excluded. Secondly, even when nominal variables have high information content it may not be appropriate to use them in raw form to define dimensions. This is typically the case in high cardinality nominal variables where the grouping of nominal values will lead to the discovery of more meaningful and useful patterns. They reasoned that the use of data mining techniques to aid in the discovery of meaningful dimensions could augment domain knowledge and thus enrich the design process.

They also pointed out that relationships between nominal and numeric variables may be subject to change, depending on the level of data granularity. To test this premise they applied a hierarchical clustering algorithm to the numeric variables to generate a dendrogram with nodes representing individual clusters containing a mix of numeric and nominal variables that are candidates for multidimensional schema. A multidimensional scaling method (Cox and Cox 2008) was then applied on the nominal values within a cluster in order to transform them into numerical form. The motivation was to obtain a grouping of the nominal variables based on their pattern of co-occurrence within a given cluster. However, no concrete algorithm was proposed for deriving groups and it was left to the human designer's subjective judgement to decide boundaries between groups.

Despite the afore-mentioned contributions, their approach suffers from a number of limitations. Firstly, the methodology does not provide a clear indication of how many levels in the cluster hierarchy are required to optimize the knowledge discovery process. Human judgement is required to decide the cluster cut-off point and if this cut-off point is underestimated then valuable knowledge may be lost. On the other hand, overestimation leads to the situation of an unnecessarily large dendrogram with attendant space and computational inefficiencies. We address this problem in this paper by utilizing the linkage inconsistency threshold proposed by Cordes et al. (2002) to determine the cut-off point in a dendrogram. The use of a rigorous method of cut-off determination via the inconsistency threshold removes the need for the manual error-prone method of determination.

Secondly, a manual method was used for the extraction of clustered data and the labelling of clusters at the various levels of the hierarchy to generate a binary tree. This represents a laborious task for the analyst to extract data from each cluster and label each cluster, one by one. Besides the manual work of naming and extracting cluster data, users have to manually construct a binary tree structure in order to visualize the cluster hierarchy in the form of a hierarchical tree. We believe that cluster extraction, labelling and binary tree generation should be automated in order to ensure that knowledge discovery is efficient and robust. In this paper, we propose an algorithm that generates a binary tree of clusters based on an automatically identified cut-off point and labels clusters with automatically determined labels that are based on their position in the data hierarchy.

Thirdly, as mentioned previously, full automation for grouping of nominal variable is not provided. Instead, users are required to visualize the results of the multidimensional scaling technique in the form of a parallel coordinate display and to group similar values present in each dimension. Such grouping by visual inspection may not be feasible in cases where similar values lie very close to each other in a dimensional coordinate. Nominal variables with high cardinality such as Country, Product codes etc., having more than 40 distinct names, are difficult to visualize and group, and thus there is a need for a generic method that can create groups of similar values within each dimension automatically. In this work, we provide an algorithm which creates groups of similar values based on an automatically calculated threshold for each dimension.

Fourthly, no ranking mechanism for filtering non informative dimensions was provided. Users are required to decide the dimensions of their choice with no indication of the underlying information content. Thus it would be useful to provide guidance to users by ranking dimensions based on objective information theoretic measures such as entropy and information gain, thus enabling users to factor in information content in addition to their specialized domain knowledge in the decision making process.

Finally, no explicit support was provided for the discovery of hidden relationships and associations which often yield important insights into underlying trends. To overcome the limitation of OLAP’s incapability of finding hidden associations, we applied association rule mining on our generated multidimensional schema. This enables the mining of hidden trends and patterns in the form of association rules from logically constrained schema. Moreover, we evaluate the interestingness of rules with respect to multiple objective rule interest measures proposed in Geng and Hamilton (2006) under the diversity criterion. We believe that rules containing diverse information convey more knowledge and hence, such rules are of more interest to the user.

Summary

In this chapter we presented the four main themes of literature review which are closely related to the contributions in this thesis. It is apparent from the review that a variety of approaches have been proposed in the literature to mine large data cubes for discovering knowledge. However, a number of issues remain unresolved in that previous work especially on the intelligent data analysis front. Firstly, the prior work assumed that data analysts could identify a set of candidate data cubes for exploratory analysis based on domain knowledge. Unfortunately, situations exist where such assumptions are not valid. These include high dimensional datasets where it may be very difficult or even impossible to predetermine which dimensions and which cubes are the most informative. In such environments it would be highly desirable to automate the process of finding the dimensions and cubes that hold the most interesting and informative content. Moreover, there remains a need for automated support in the design of multidimensional schema, especially in domains containing high dimensional data. In such domains the sheer scale of the data, both in terms of data volume as well as in the number of dimensions, may make it difficult for human designers to decide which dimensions are the most informative and should thus be retained in the final version of the cube design.

Secondly, reliance on domain knowledge tends to constrain the knowledge discovered to only encapsulate known knowledge, thus excluding the discovery of unexpected but nonetheless interesting knowledge (Koh, Pears et al. 2011). Another related issue is that it restricts the application of these methodologies to only those domains where such domain knowledge is available. However, a knowledge discovery system should be able to work in ill-defined domains (Nkambou, Fournier-Viger et al. 2011) and other domains where no background knowledge is available (Zhong, Dong et al. 2001). To the best of our knowledge, none of the work done in the past focused on cases where limited or no domain knowledge exists. Most of the work done in the past targeted the business domain and hence it would be of interest to investigate the effectiveness of rule discovery across non-business domains. Additionally, there is a strong requirement to assist data warehouse designers to construct informative schema that can overcome design pitfalls and provide analysts a base to counterpart knowledge discovery challenges from large multidimensional space.

Thirdly, these approaches mostly target a specific data type. In the real world, datasets have a mix of numeric and nominal variables, often involving high cardinality nominal variables, thus challenging the analytical capability of the methods employed. It is evident that limited research has been conducted in the area of finding interrelationships between numeric and nominal variables that are increasingly becoming common in real-world datasets. Moreover, the predominant statistical techniques for analysis of such variables lack integration with machine learning methods for finding interrelationships between variables.

As with research in the area of multidimensional schema design, gaps exist in previous research in the area of knowledge discovery from multidimensional data. It is apparent from the review that all the previous approaches support constraining search space for rule discovery, either in the form of pattern templates or in the form of aggregated data. While aggregated data has been shown to be a better foundation for generating rules, the evaluation for the most part, with the exception of Messaoud et al. (2006), used support and confidence criteria to measure the importance of rules. These measures are best suited to transactional data but do not adequately measure the effectiveness of rules generated from data represented at different levels of data granularity. The complexity of multidimensional structures requires the use of more sophisticated measures to quantify the interestingness of rules discovered at different levels of data abstraction.

In terms of data analysis, the main tool used in multidimensional analysis in a data warehousing environment is the use of various data aggregation and exploratory techniques that form part of the On Line Analytical processing (OLAP) suite of methods. While traditional OLAP methods are excellent tools for exploratory data analysis they are limited as far as detecting hidden associations between items resident in a data warehouse. The discovery of such hidden relationships and associations often yields important insights into underlying trends and in general leads to an improved decision making capability.

The above mentioned issues motivated us to formulate a generic methodology for data cube identification and knowledge discovery that is applicable across any given application domain, including those environments where limited domain knowledge exists. High dimensional and high volume datasets present significant challenges to domain experts in terms of identifying data cubes of interest. The presence of mixed data in the form of nominal and numeric variables present further complications as the interrelationships between nominal and numeric variables have also to be taken into account. A methodology that assists domain experts in identifying dimensions and facts of interest is highly desirable in these types of environments. Moreover the proposed methodology should provide automated assistance to constrain a multidimensional schema, supports advanced evaluation of discovered rules interestingness, and offer easy implementation methods for non-business domains.

Chapter 3

Discovery of interesting cube regions and diverse association rules

In this chapter, we present an overview of the proposed methodology for multidimensional cube design that facilitates the discovery of interesting cube regions and diverse association rules. As mentioned earlier, the two main objectives of our research are: firstly, to equip knowledge workers with essential information to intelligently analyze high dimensional datasets containing mixed data types; and, secondly, to assist in the automated cube design process by complementing automated techniques for cube design with specialized knowledge that domain specialists may possess with their expert knowledge of the application domain.

3.1 Methodological Framework

We first present an overview of the framework we propose for data cube design and analysis before discussing the details of each step involved in implementing the framework. Figure 3.1 depicts the major phases involved. Steps 1 to 6 cover the design aspects while the 3 remaining steps deal with analysis of the informative cubes generated.

We use a hypothetical example to illustrate each of the phases in the proposed methodology. Consider a mixed variable dataset ***D*** having 3 numeric (*Profit, Quantity, Weight)* and 3 nominal variables *(quality, color, size)* with ***X*** number of records. This dataset, although being small in terms of dimensionality of variable is of mixed data type which is a key facet of multidimensional data used in the construction of a data warehouse. We now describe each of the phases involved in implementing the framework.

3.1.1 Generate Hierarchical Clusters

In the first phase, we apply Agglomerative Hierarchical Clustering (AHC) on numeric variables of the given dataset to generate a dendrogram. Each level in the dendrogram contains a set of child clusters that were split off a single parent. A key issue with any form of clustering is to determine the number of clusters; and with respect to hierarchical clustering this reduces to determining at what point to terminate generation of the dendrogram. We use the linkage inconsistency threshold (Cordes, Haughton et al. 2002) to determine the cut-off point. The threshold is defined by equation 1.

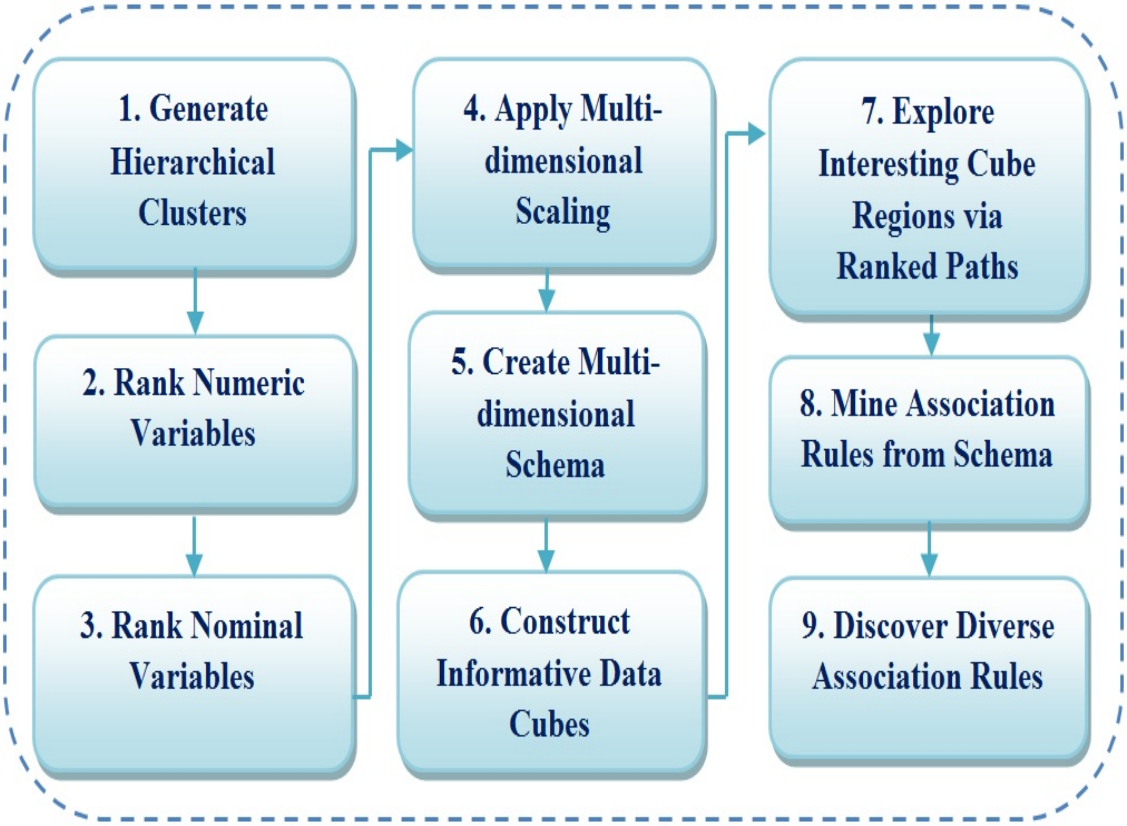
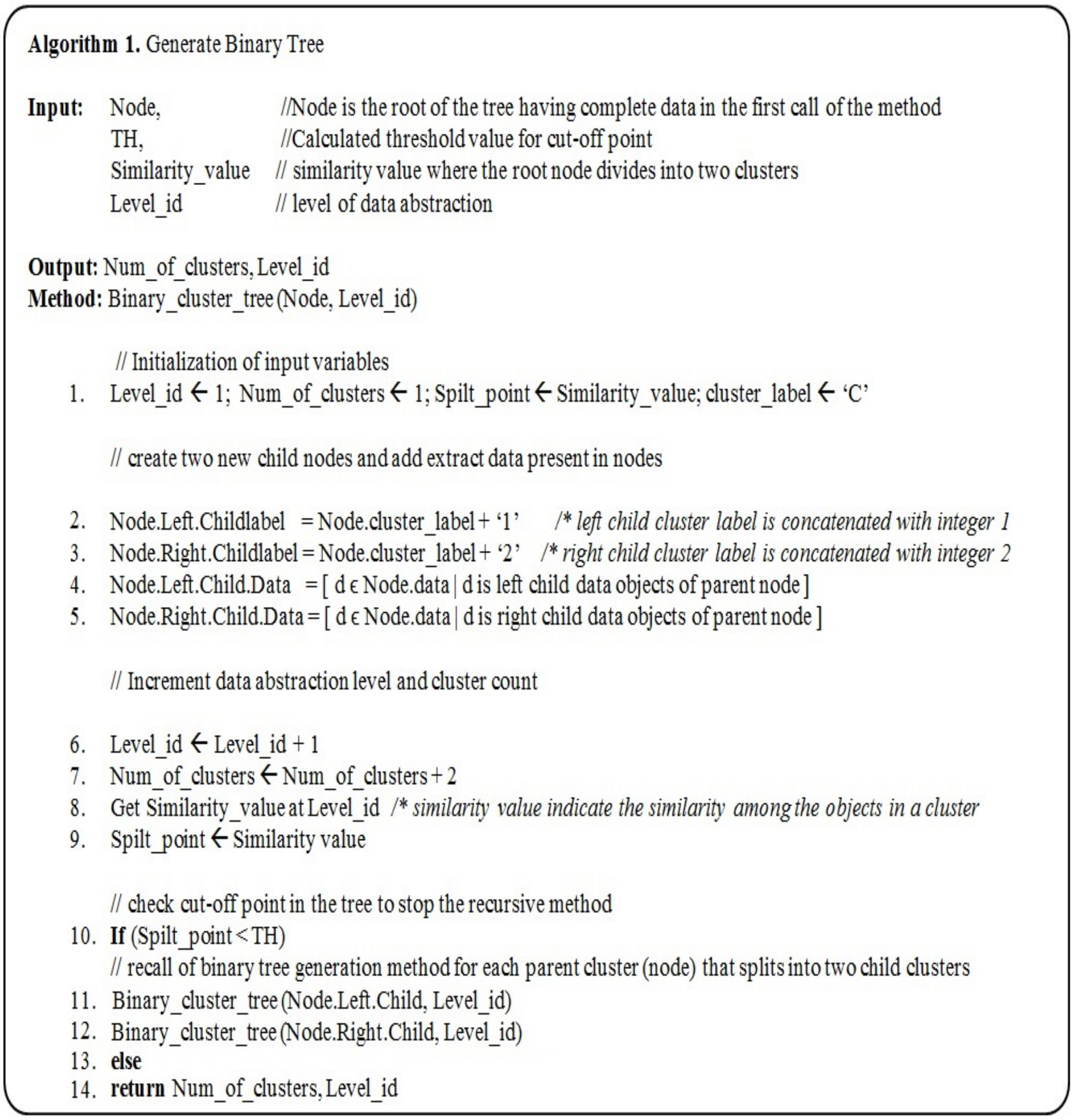

1. Generate Hierarchical Clusters
2. Rank Numeric Variables
3. Rank Nominal Variables
4. Apply Multi-dimensional Scaling
5. Create Multi-dimensional Schema
6. Construct Informative Data Cubes
7. Explore Interesting Cube Regions via Ranked Paths
8. Mine Association Rules from Schema
9. Discover Diverse Association Rules

Figure 3.1: Methodological Framework for discovery of interesting cube regions and diverse association rules

$$ITh\left(link1\right)=\frac{length\left(link1\right)-\mu\left(links\right)}{\sigma\left(links\right)} \qquad (1)$$

In equation 1, the distance between two clusters is represented as the length of the link, *link1*. The term μ represents the mean of all the links present in the dendrogram and σ is the calculated standard deviation across all links. The higher the values of the threshold *ITh*, the less similar are the clusters connected by the link. This threshold thus provides an objective method of determining the number of clusters without heavy reliance on domain specific information. The inconsistency coefficient of the links in the cluster tree structure identifies cluster divisions where similarities between data objects change abruptly. A link whose height differs significantly from the height of the links below it indicates that the clusters at this level in the dendrogram are much farther apart than that of their corresponding child clusters. Such a link is said to be inconsistent with the links below it.

As we move from the top level (root node) towards the lower levels (leaf nodes) the heights of the links at a particular level will become approximately the same height as the links below it, thus indicating that there is no distinct division between clusters objects at the particular level in the hierarchy. We take the inconsistency threshold value at such a level in the hierarchy as the value that determines the cut-off point. After determining the cut-off point, we give each cluster a unique label and extract the clustered data from each level using the procedure shown in lines 2 to 5 of Algorithm 1. We then increment the data abstraction level and cluster count in lines 6 and 7.



Algorithm 1. Generate Binary Tree

```
Input:   Node,              //Node is the root of the tree having complete data in the first call of the method
         TH,                //Calculated threshold value for cut-off point
         Similarity_value   // similarity value where the root node divides into two clusters
         Level_id           // level of data abstraction

Output: Num_of_clusters, Level_id
Method: Binary_cluster_tree (Node, Level_id)

         // Initialization of input variables
    1.   Level_id ← 1;  Num_of_clusters ← 1; Spilt_point ← Similarity_value; cluster_label ← 'C'

         // create two new child nodes and add extract data present in nodes

    2.   Node.Left.Childlabel   = Node.cluster_label + '1'     /* left child cluster label is concatenated with integer 1
    3.   Node.Right.Childlabel = Node.cluster_label + '2'     /* right child cluster label is concatenated with integer 2
    4.   Node.Left.Child.Data   = [ d ϵ Node.data | d is left child data objects of parent node ]
    5.   Node.Right.Child.Data = [ d ϵ Node.data | d is right child data objects of parent node ]

         // Increment data abstraction level and cluster count

    6.   Level_id ← Level_id + 1
    7.   Num_of_clusters ← Num_of_clusters + 2
    8.   Get Similarity_value at Level_id  /* similarity value indicate the similarity among the objects in a cluster
    9.   Spilt_point ← Similarity value

         // check cut-off point in the tree to stop the recursive method
    10.  If (Spilt_point < TH)
         // recall of binary tree generation method for each parent cluster (node) that splits into two child clusters
    11.  Binary_cluster_tree (Node.Left.Child, Level_id)
    12.  Binary_cluster_tree (Node.Right.Child, Level_id)
    13.  else
    14.  return Num_of_clusters, Level_id
```

After, incrementing the abstraction level, we obtain a similarity value and store it as a spilt point, as shown in lines 8 and 9. This similarity value represents the *Euclidean* distance between the data points on which a cluster splits into two child clusters.

Line 10 checks the threshold (cut-off point) in the tree. If the split point is less than the threshold value then we recursively call the *Binary_cluster_tree* method for left and right child clusters as shown in line 11 and 12. This method recursively assigns unique labels to the two left and right child clusters and extracts clustered data. The recursion is terminated when the spilt point equals the calculated cut-off point. Finally, the number of clusters and total number of levels in the cluster hierarchy are output.

The knowledge contained within a cluster is captured by the relationships that exist between the numeric variables and nominal variables. In general, relationships between nominal and numeric variables are subject to change depending on the range that the numeric variables are constrained on. As the range tightens at the lower levels of the dendrogram, significant differences in the relationships emerge, as shown in the results of the three case studies that we undertake in Chapters 4, 5 and 6.

Our preference for Agglomerative Hierarchical Clustering (AHC) is based on the fact that it tends to capture a natural hierarchy more faithfully than other clustering approaches (Seo, Bakay et al. 2003; Seo, Bakay et al. 2004; Usman, Asghar et al. 2010; Usman and Pears 2010; Usman and Pears 2011). In hierarchical agglomerative clustering, only numeric variables play an active part in cluster formation, in common with many other clustering approaches. With AHC, nominal variables are normally required to be transformed into numeric form in order to be involved in the clustering process. However, our methodology does not require any such mapping and we believe that nominal variables should retain their original data format as ad-hoc and unnecessary mappings could result in loss of information or may lead to erroneous results. In place of ad-hoc transformations we rely on the use of formal methods such as Multiple Correspondence Analysis (MCA) and entropy/information gain measures to extract natural groupings of nominal variables within a cluster.

To illustrate the first phase of the methodology using the hypothetical dataset, we applied the AHC algorithm on the 3 numeric variables from dataset *D* to generate hierarchical clusters at different data abstraction levels. It produced the hypothetical dendogram depicted in Figure 3.2.

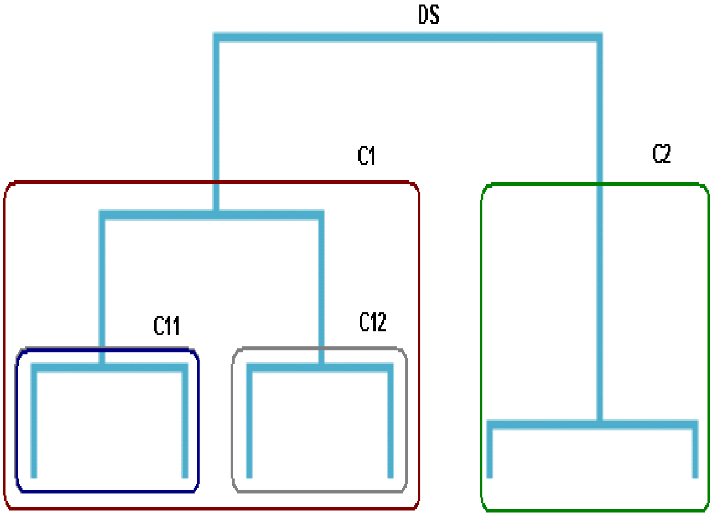


Figure 3.2: Dendogram structure of hierarchical clusters

Using Algorithm 1, we identify and label the hierarchical clusters by giving simple abbreviations such as C1, C2 etc. at different levels of data abstraction in the form of binary tree as represented in Figure 3.3.

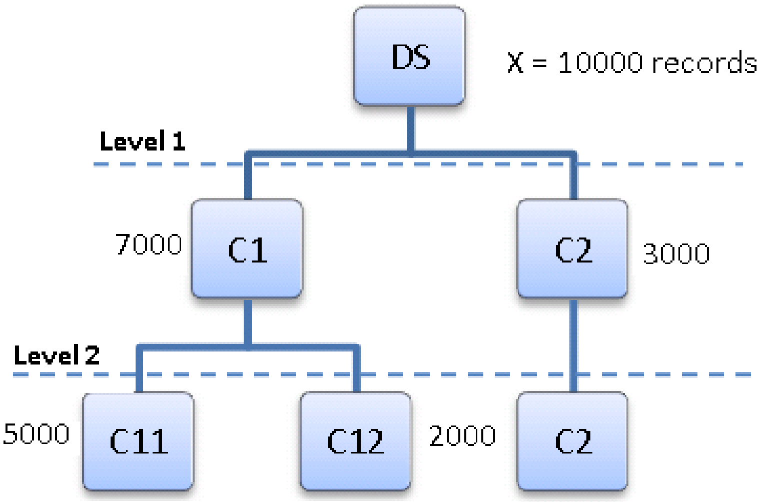


Figure 3.3: Tree structure of clusters

3.1.2 Rank Numeric Variables

After the dendrogram is generated, each of the numeric variables within a cluster is ranked by Principal Component Analysis (PCA) in terms of the degree of variance it captures across the data present in the cluster. PCA (Jolliffe 2002) is a popularly used statistical technique that has been applied in a wide variety of applications for finding patterns in high dimensional data (Uguz 2011). The main advantage of PCA is its ability to transform a given set of variables into a new (smaller) set of variables that capture the most variation. In the following paragraphs, we provide an overview of PCA as a method of data reduction.

Suppose that the dataset to be reduced has *n* numeric variables. PCA projects the original dataset onto a smaller dataset that captures most of the variation present in the original dataset. It accomplishes this by finding a set of Eigen vectors E_1, $E_{2,}$…, *En*. Given a dataset *D*, we first project the dataset onto its numeric variables and obtained another dataset *D'*. Now from *D'*, the covariance of every pair of items can be expressed in terms of its covariance matrix *M*. The matrix *P* of all possible Eigen vectors can then be derived from:

$$P^{-1}MP = Q \qquad (2)$$

where *Q* is the diagonal matrix of Eigen values of *M*. Our use of PCA is to obtain a set of factor loadings from the set of Eigen vectors obtained from equation 2 above. In practice, only a subset of Eigen vectors that capture t% of the total variance across dataset *D'* is used. Each Eigen vector E_i is associated with an Eigen value e_i that represents the proportion of variance that is explained by that Eigen vector. The Eigen vectors can then be arranged in ranked order of their Eigen values and the first *m* such vectors that collectively capture at least t% (generally set to 0.90) are chosen for extraction of the factor loadings. The factor loading $\mathrm{F_i}$ for an original numeric variable $\mathrm{V_i}$ is then given by its commonality (Tryfos 1998).

Thus, $$F_i = \sum_{j=1}^{m} \left(E_{ij}\right)^2 \quad \forall\, i = 1,\ldots,n. \tag{3}$$

The factor loadings F_i obtained are then used to rank the numeric variables. In order to obtain the ranked list of numeric variables in a parent cluster, say C1, we apply PCA on the numeric variables of the two child clusters, namely C11 and C12 and obtain the factor loadings (Eigen values) for each numeric variable present in these child clusters. We then compare differences between the loadings for each numeric variable across clusters C11 and C12. Each variable is assigned a ranking at a (parent) cluster that is equal to the difference in factor loadings for that variable across the child clusters. The higher the difference in loadings for a given variable, the higher is the rank for that variable.

The rationale behind this approach is that the two mutually exclusive child clusters have the necessary information to identify the numeric variables that defined the split. Thus, if Profit, Quantity and Weight defined the split of cluster C1 (parent) into clusters C11 and C12 (children), then the variable that discriminates most between the two clusters would tend to capture a high degree of variation in one of the clusters while expressing itself to a much lesser extent in the other cluster. Thus for example, Profit expresses itself much more strongly in cluster C11 when compared to cluster C12. The variable Profit has the highest difference in factor loadings amongst the 3 variables, thus acquiring the highest rank, followed by Weight and Quantity, as shown in Table 3.1.

Table 3.1: Ranking of numeric variables in cluster C1

Numeric Variables	C11 Factor Loadings	C12 Factor Loadings	Comparison Results	Ranking of variables
Profit	0.627	0.283	0.343	Rank # 1
Quantity	0.742	0.540	0.201	Rank # 3
Weight	0.896	0.619	0.276	Rank # 2

3.1.3 Rank Nominal Variables

In order to rank the nominal variables, we apply two separate data analysis techniques, namely Multiple Correspondence Analysis (MCA) and Information Gain. MCA is a counterpart to PCA and is used to detect and represent underlying structure information for nominal or categorical data, while Information Gain is a variable selection measure.

The rationale behind the usage of these techniques for ranking is that it would be useful to provide guidance to analysts by ranking dimensions based on objective information theoretic measures, thus enabling analysts to factor in information content in addition to their own specialized domain knowledge in the decision making process.

Moreover, these two ranking methods provide assistance in achieving the two main objectives of this research; i) identification of interesting cube regions and ii) discovery of diverse association rules. Firstly, MCA based ranking helps in identifying interesting regions in data cubes via highly ranked paths. These paths are determined through outlier analysis of nominal values present in each variable. Details of this method are explained through an example in Section 3.1.3.1. Secondly, information gain based ranking assists in capturing informative dimensions for discovering diverse association rules from multidimensional schema.

The selection of the ranking method depends on the knowledge discovery task at hand. If users are interested in exploring data cubes to find interesting regions then MCA based ranking has to be utilized as the information gain ranking method does not have the added advantage of highlighting the ranked paths in cube space for finding interesting knowledge. Otherwise, if the purpose of discovery task is to uncover diverse association rules then information gain based ranking provides a better understanding of the underlying information content in each dimension. Thus the two methods are alternative to each other and in any given situation a choice is normally made between the two, depending on circumstances, as just explained.

However, there could be cases where users have an interest in both types of knowledge discovery tasks. For example, a user may first want to investigate data cube regions and after finding interesting regions, would like to discover the diverse rules with high prediction accuracy from such dense informative regions. In such cases, we suggest using both ranking methods in parallel to enhance knowledge discovery.

For instance, MCA based ranking assists the user in selecting informative dimensions and regions in data cube and information gain based ranking provide extra information about the information content in each dimension selected. In this way, user gets added information for the efficient discovery of diverse rules with greater prediction accuracy. In the case studies on real world datasets presented in the following Chapters, we have utilized both ranking methods and explained how these methods work in parallel in enhancing the knowledge discovery process.

3.1.3.1 Ranking via Multiple Correspondence Analysis (MCA)

In this step, we rank the nominal variables present in each data cluster. To achieve this objective, we adopt MCA (Greenacre 1991; Abdi and Valentin 2007; Le Roux and Rouanet 2009), which is conceptually similar to PCA but is specially designed for the analysis of nominal variables. MCA is an extension of the simple correspondence analysis technique to account for more than two variables. It is applicable to large sets of nominal variables, each of which may have high cardinality (large number of categorical values). It can also be seen as a generalization of the PCA technique when variables to be analyzed are qualitative instead of quantitative. After the application of MCA, a factor loading is computed for each nominal variable. We compute these factor loadings and rank the nominal variables in descending order.

Large factor loadings correspond to a large spread among the categories of nominal variables and consequently indicate a higher degree of discrimination between the categories of a nominal variable.

It should be noted that we do not automatically reduce the dimensions in this phase because our purpose is to provide a ranked list of all dimensions and the user has the flexibility to choose the dimensions that are more meaningful for his/her analysis.

In addition to ranking nominal variables, MCA also provides a calculated value for each category in a nominal variable. The category values can be used to segment nominal variables on the basis of their values. Accordingly, some of these categories may be at a much greater distance apart when compared to the average distance, taken across all categories. Such categories have distinct characteristics and play a vital role in determining the interesting regions in a data cube.

Similar to PCA, we obtain the factor loadings for each nominal variable present in our example dataset *D* and rank them according to their individual factor loading values, from largest to smallest. Unlike our numeric ranking approach, we rank on the basis of individual clusters instead of comparing child clusters. The basic reason for this approach is that the nominal variables do not play a direct role in clustering the dataset. Table 3.2 shows the ranking obtained for cluster C1.

Another advantage of using MCA for nominal variable analysis is that we can easily identify the significant values of the highly ranked nominal variables by means of outlier analysis. For instance, if 50 distinct Products are present and each has a unique color and shape then a plot of each of these variables with the first two Principal Components as axes will reveal any products having color and/or shapes that have very different values with respect to one of the numeric variables, say Profit.

Figure 3.4 clearly shows that two products, (*Products A & D*), have large deviations from the average. The same holds true for the Color (for *green & blue* values) and Shape (with values *star & diamond*) variables.

Table 3.2: Ranking of nominal variables in cluster C1

Nominal Variables	Factor Loadings	Ranking of variables
Product Name	0.627	Rank # 2
Color	0.701	Rank # 1
Shape	0.525	Rank # 3

The identification of such outliers plays a central role in our methodology for knowledge discovery via exploration of ranked paths in data cubes. However, the visual exploration of these outlying values in plot diagrams is a laborious and time-consuming process, especially when a large number of values are present in the plot or the plotted values lie very close to each other. For example, if there are 100 *Products* plotted with

50 unique colours and 40 types of shapes, then it becomes difficult to analyse which colours and shapes are outliers in the plot. Additionally, users may not be interested in the most outlying colours and shapes, as such distinctive colours and shapes may already be known to them. Instead, users may be more interested in a set of top *k* ranked values for the colour and shape variables. Here *k* can be any number of values less than the total values projected by MCA.

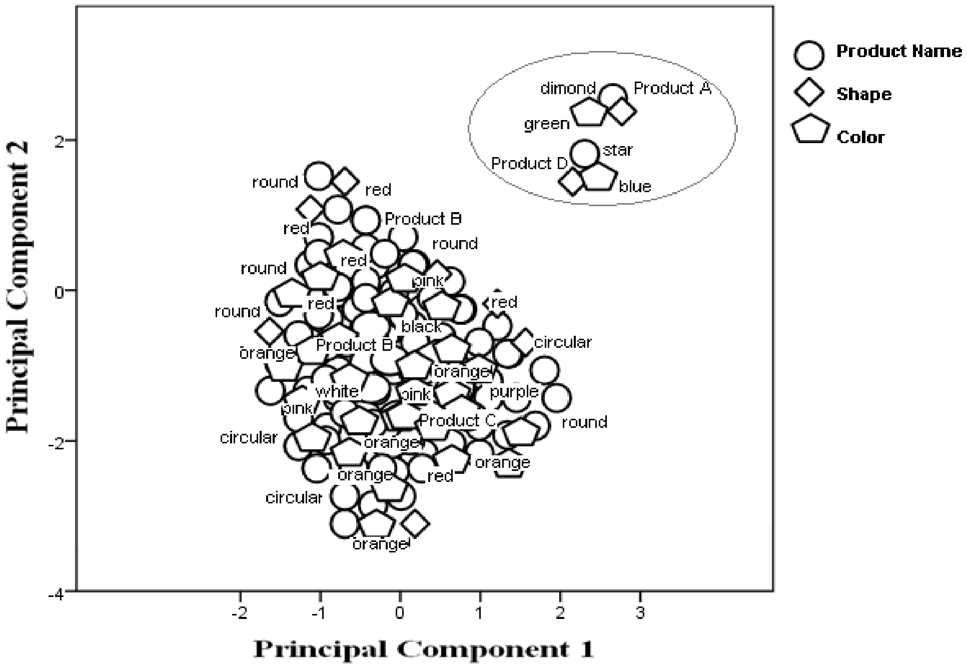
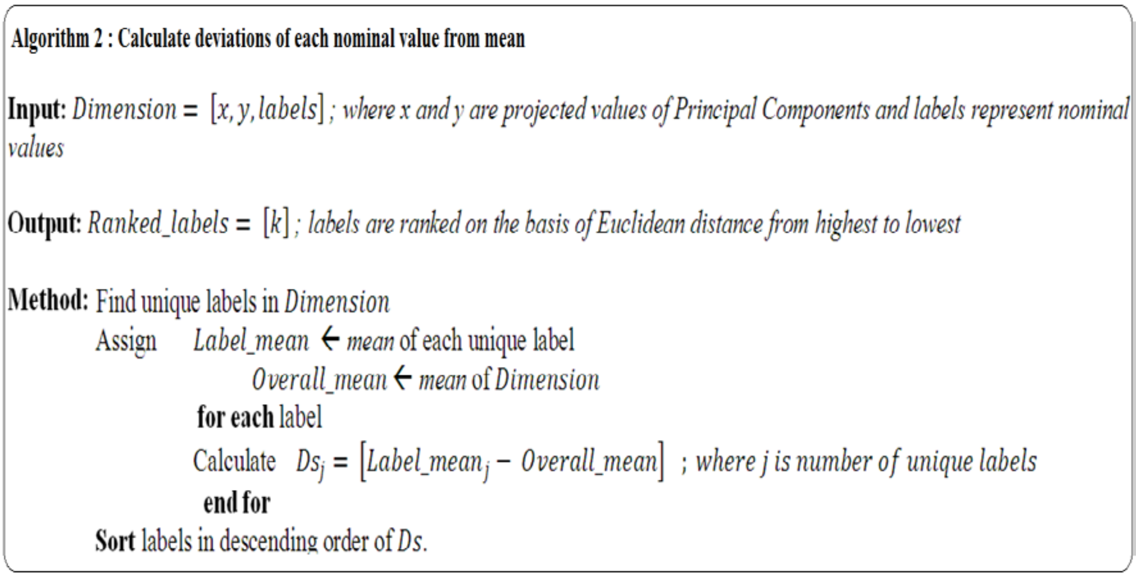


Figure 3.4: Project values of ranked nominal variables in cluster C1

In order to give users greater flexibility at this stage, we automate the process of outlier detection by taking the *Euclidean distances* of each distinct nominal value from the overall mean and sorting the calculated distances in descending order. The procedure for calculation is shown in Algorithm 2.



```
Algorithm 2 : Calculate deviations of each nominal value from mean

Input: $Dimension = [x, y, labels]$ ; where x and y are projected values of Principal Components and labels represent nominal values

Output: $Ranked\_labels = [k]$ ; labels are ranked on the basis of Euclidean distance from highest to lowest

Method: Find unique labels in Dimension
        Assign  $Label\_mean \leftarrow$ mean of each unique label
                $Overall\_mean \leftarrow$ mean of Dimension
                for each label
                    Calculate  $Ds_j = [Label\_mean_j - Overall\_mean]$ ; where j is number of unique labels
                end for
        Sort labels in descending order of Ds.
```

3.1.3.2 Ranking via Entropy and Information Gain Measures

We adopt entropy and information gain measures to devise an alternative nominal ranking method. Entropy is a measure that indicates the degree of impurity in a variable. It can be measured in bits for a variable, say, *v* through equation 4.

$$Entropy\ (v) = -\ (p * \log\ (p) + (1-p) * \log\ (1-p)) \qquad (4)$$

Generally, entropy is greater if the distinct values in a variable are evenly distributed and vice versa. Information gain, on the other hand, is a measure of purity in a variable.

The information gain for a given variable v is given by equation 5 below.

$$Information\ Gain\ (v) = Entropy\ (v)\ before\ spilt – Entropy\ (v)\ after\ split \qquad (5)$$

We calculate the information gain for each nominal variable present in a cluster in order to rank the variable in terms of significance. The variable with the highest information gain acquires the highest rank as it minimizes the information required (i.e. has least randomness) to cluster records from the parent cluster, say C1, into child clusters, C11 and C12. However, we need to take into account the entropy on left child cluster (C11) and right child cluster (C12) in order to calculate the entropy after a parent cluster (C1) splits. Equation 6 defines entropy of a variable after the spilt.

$$Entropy\ (v)\ after\ split = Wfl * (Entropy\ (v)\ left\ child\ cluster) + Wfr * (Entropy\ (v)\ right\ child\ cluster) \qquad (6)$$

In equation 6, *Wfl and Wfr* represents the weight factors which are the ratios of the number of records on the left child (C11) and right child (C12) clusters respectively to the total number of records in the parent cluster (C1). Therefore, by comparing the entropy before and after the split, we obtain a measure of information gain, or in simple terms, we assess the information that was gained by performing a split with a given variable *v*.

We illustrate this step using our running example having three nominal variables, namely *quality, color and size.* Our objective is to rank the nominal variables present in the parent cluster C1. We start by first calculating the entropy of each variable present in parent cluster C1 and the two child clusters C11 and C12 using equation 5. The results obtained are shown in Table 3.3. Thereafter, we calculate the entropy of each variable after the split by substituting the values from Table 3.3 in equation 6. For example, the entropy of variable *Quality* can be calculated as follows:

Entropy after split = *Wfl* * (Entropy (Quality) left child) + *Wfr* * (Entropy (Quality) right child)

$$= 5000/7000 * (2.042) + 2000/7000 * (1.304)$$

$$= (0.714 * 2.042) + (0.285 * 1.304)$$

= 1.457 + 0.371

= **1.828**

We calculate the weight factors for the left and right child clusters by utilizing the distribution of records across clusters given in Figure 3.2. We then calculate the information gained for the Quality variable using equation 5.

Information Gain (Quality) = Entropy before spilt – Entropy after split

= 2.028 – 1.828

= 0.2

Similarly, we calculate the information gain for the other two variables and rank them accordingly. The results of the ranking for this example are shown in Table 3.4.

Table 3.3: Calculated entropy of nominal variables

Variables	Cluster C1 entropy (parent)	Cluster C11 entropy (left-child)	Cluster C12 entropy (right-child)
Quality	2.028	2.042	1.304
Color	1.845	1.936	1.687
Size	1.596	1.600	1.404

Table 3.4: Ranked list based on information gain

Cluster C1				
Variables	**Entropy before**	**Entropy after**	**Information Gain**	**Rank**
Quality	2.028	1.828	0.2	Rank # 1
Color	1.845	1.864	-0.01	Rank # 3
Size	1.596	1.544	0.05	Rank # 2

3.1.4 Apply Multidimensional Scaling

After ranking of nominal variables, the next phase of our methodology involves identification of natural groupings of the nominal variables. To achieve this, we apply multidimensional scaling (Borg and Groenen 2005) to identify the semantic relationships among values in each nominal variable. With multidimensional scaling semantic relationships between multiple nominal variables can easily be visualized through a parallel coordinate display. In a parallel coordinate display each nominal variable is represented on a vertical scale. The values are displayed on the scale and the spacing between the values signifies the similarities and differences between the values.

Figure 3.5 depicts the use of this technique for visualizing each of the nominal variables for our running example.

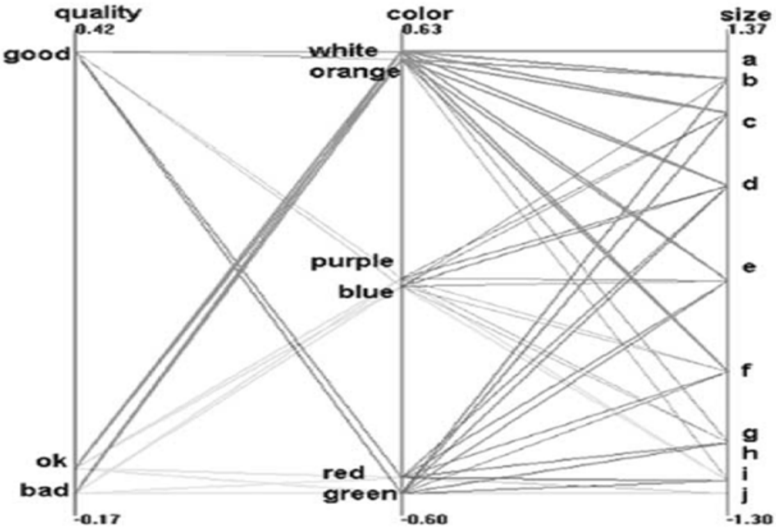


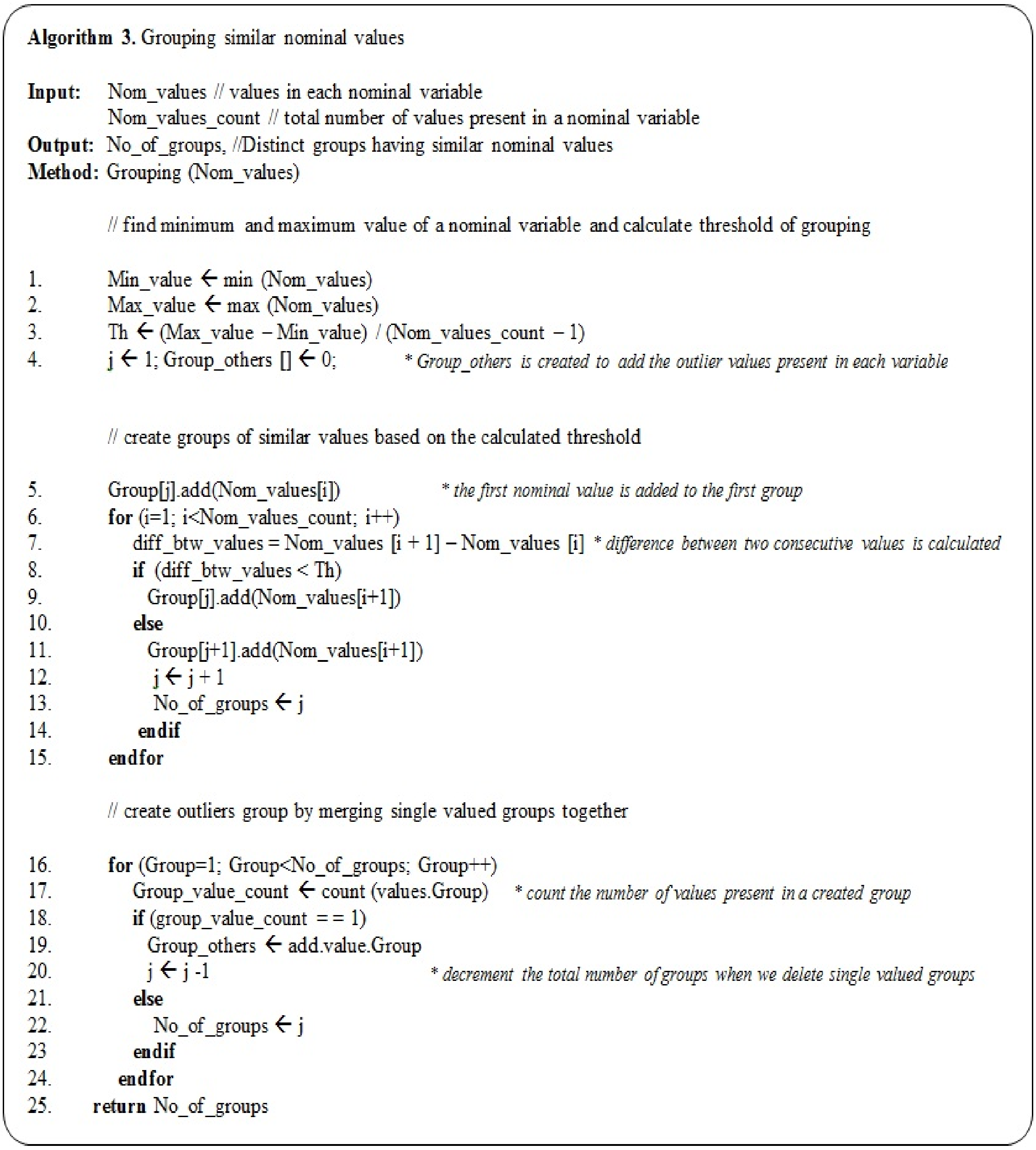
Figure 3.5: Parallel coordinates display showing the similarities among nominal values

Figure 3.5 reveals that objects with quality *ok* or *bad* have a similar underlying distribution for *Color* and *Size* in contrast to the objects with quality as *good* which have different *color* and size characteristics. Each of these values represents a numeric value on the scale. As can be seen from Figure 3.5 the parallel coordinate display enables the easy grouping of *Color* values. Three natural and distinct groups, each having two different sets of colors such as (white, orange), (purple, blue) and (red, green) are clearly defined on the display. However, the *Size* variable is difficult to group through a simple visual inspection of the parallel coordinate display. There are 10 different sizes, ranging from *a* to *j* and the distribution of these values on the scale does not permit a precise grouping based on visual inspection. For instance, sizes *a* and *b* are closer to each other but it is difficult to visually determine whether size *c* should be grouped with sizes *a* and *b* or whether it should be contained within another group with size *d*.

In a real world scenario variables with large cardinality such as *Country* or *Product codes* are common and it is next to impossible to group the values by visualization alone. An automated method for grouping is then clearly required. Algorithm 3 generates groups for a given nominal variable given its coordinates produced by the multidimensional scaling method.

A generic grouping strategy is used to assign values to groups, where each group corresponds to a collection of semantically related values. We first take the minimum and maximum value of each coordinate and calculate a threshold for assigning values. The threshold is computed as the average range, taken across all nominal values. Nominal values are then assigned to groups on the basis of proximity to each other (lines 5 to 15). If two consecutive values are not further from each other than the threshold distance then they are assigned to the same group, otherwise they fall into two neighbouring groups. After generating the groups, we check for singleton groups (lines

16 to 24). If such singleton groups exist they are merged into a single group called the “outlier” group. After ranking the numeric and nominal variables and obtaining the natural groupings in each of the nominal variables, we move to the next phase of creating a multidimensional schema.



```
Algorithm 3. Grouping similar nominal values

Input:   Nom_values // values in each nominal variable
         Nom_values_count // total number of values present in a nominal variable
Output:  No_of_groups, //Distinct groups having similar nominal values
Method:  Grouping (Nom_values)

         // find minimum and maximum value of a nominal variable and calculate threshold of grouping

1.       Min_value ← min (Nom_values)
2.       Max_value ← max (Nom_values)
3.       Th ← (Max_value − Min_value) / (Nom_values_count − 1)
4.       j ← 1; Group_others [] ← 0;       * Group_others is created to add the outlier values present in each variable


         // create groups of similar values based on the calculated threshold

5.       Group[j].add(Nom_values[i])            * the first nominal value is added to the first group
6.       for (i=1; i<Nom_values_count; i++)
7.         diff_btw_values = Nom_values [i + 1] – Nom_values [i]  * difference between two consecutive values is calculated
8.         if (diff_btw_values < Th)
9.           Group[j].add(Nom_values[i+1])
10.        else
11.          Group[j+1].add(Nom_values[i+1])
12.          j ← j + 1
13.          No_of_groups ← j
14.        endif
15.      endfor

         // create outliers group by merging single valued groups together

16.      for (Group=1; Group<No_of_groups; Group++)
17.        Group_value_count ← count (values.Group)   * count the number of values present in a created group
18.        if (group_value_count = = 1)
19.          Group_others ← add.value.Group
20.          j ← j -1                            * decrement the total number of groups when we delete single valued groups
21.        else
22.          No_of_groups ← j
23         endif
24.      endfor
25.    return No_of_groups
```

3.1.5 Create Multidimensional Schema

After receiving ranked lists of numeric and nominal variables, a multidimensional *STAR* schema is created by treating nominal variables as dimensions and numeric variables as facts. We produce a schema with all dimensions and facts present in a data cluster. The groupings information assists in defining the dimensional hierarchy or dimensional levels. Each dimension in a cluster has a group level and value level. For example, if Color is a dimension then it has Color (All) level → Color_groups (Group) level → Color_names (Value) level. A physical structure is created with the use of generic *SQL* queries. These queries create the necessary tables (fact and dimension) and define table relationships that are needed to implement the multidimensional schema. These generic queries are automatically structured to support the quick generation of multidimensional schema for any given cluster in the hierarchical tree.

For our running example, we construct the multidimensional schema by taking the nominal variables as dimensions and numeric variables as facts. Figure 3.6 depicts the multidimensional schema for cluster C1.

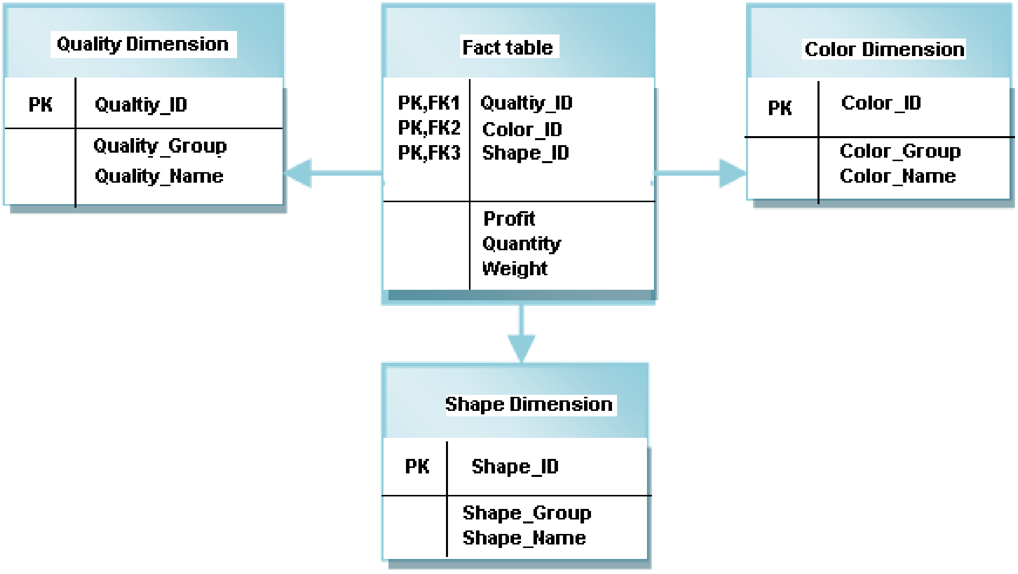


Figure 3.6: Multidimensional schema of cluster C1

In this step, the schema contains all the dimensions and facts present in the data cluster C1. The multidimensional schema is used to construct informative data cubes in the next step.

3.1.6 Construct Informative Data Cubes

In this phase, a data cube is constructed by using the highly ranked dimensions and facts present in the generated multi-dimensional schema. At this stage the user has the option of specifying values for the top *k* and top *m* thresholds, where *k* is the number of highest ranked dimensions and *m* is the number of highest ranked facts to be selected for data cube construction. Users can input the top *k* and *m* threshold to constrain the cube search space. They can either select either the top ranked dimensions/facts or the dimensions/facts of his/her own choice from the generated schema. We believe that each user has specific data analysis requirements and there may be certain cases when the user would like to see highly ranked dimensions with low ranked facts or vice versa.

The construction of informative data cubes allows the user to apply basic OLAP operations such as *Drill-down*, *Roll-up*, *Slice* and *Dice* in order to interactively explore the data to find interesting patterns. Figure 3.7 shows the data cube constructed with ranked dimensions and facts for cluster C1.

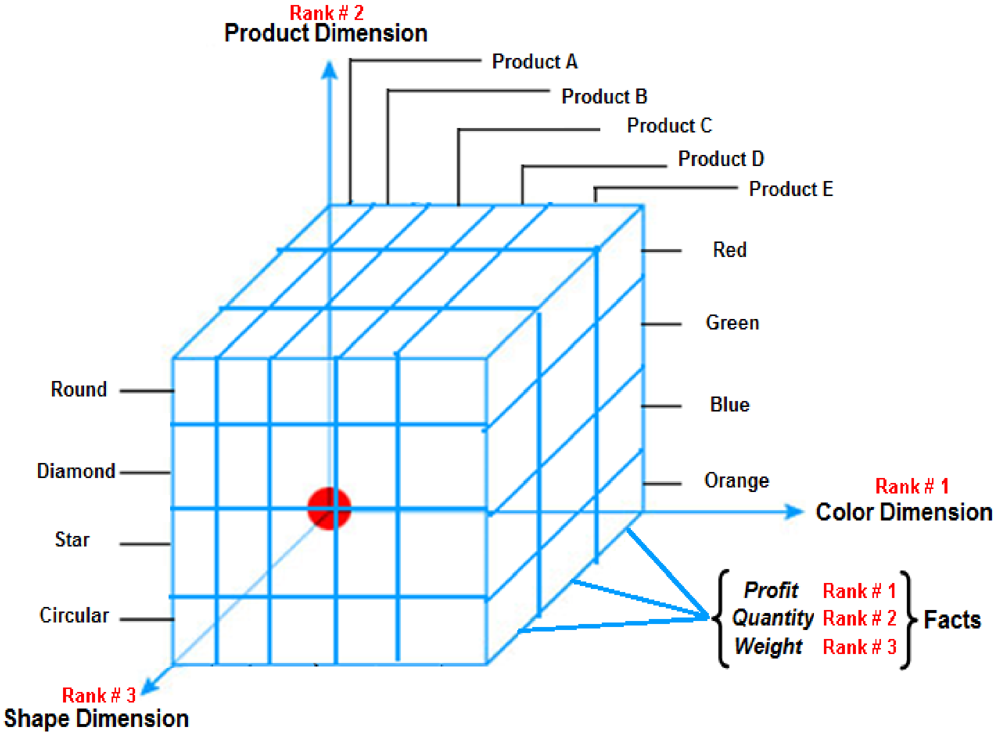


Figure 3.7: Cube structure of cluster C1

It is important to point out that we have represented all 3 dimensions and facts in Figure 3.7 to show the 3 dimensional structure of the data cube. However, as explained earlier, the user can construct data cubes by providing any number of dimensions and facts for the construction of informative cubes.

3.1.7 Explore Interesting Cube Regions via Ranked Paths

The final phase of the proposed methodology is to visually explore the interesting cube regions with the help of distinct categories (values) in each nominal variable determined through MCA. As explained in Section 3.3 some of these categories lies further apart from the rest of the categories, therefore, they reveal distinct information when viewed with respect to certain facts in a data cube. Even in a constrained data cube with only highly ranked dimensions and facts it may not be easy to discover interesting patterns or regions where the facts are highly distinctive as compared to all other regions in the cube. This is due to the sheer number of regions to be explored. Our methodology alleviates this problem by rankings paths according to the amount of information that they contain.

By utilizing unique categories in each dimension present in a data cube, we define highly ranked paths for the visual exploration of interesting cube regions. A path is represented by a set of ordered pairs and is given by:

P = [(V, v) | where V denotes a nominal variable and v is the value taken by V]

Our methodology assists users in identifying those regions of data cube that possess highly significant information when compared to all other regions. For instance, if cluster C1's data cube is to be explored, then Profit should be explored with respect to Color dimension as a first choice. The combination of the most highly ranked dimension (Color) and fact (Profit) suggests that certain regions in the C1 data cube are significantly different from the rest of the regions. Therefore, instead of exploring all regions of this cube, our methodology assists the analyst in picking up Profit as a fact and Color as a dimension to reveal the significant differences.

Additionally, using Algorithm 1, the values present in the Color dimension are also ranked in order of deviation from highest to lowest. Users can further pick those colors which show the most deviation from the average. For instance in our running example, the green color shows extreme deviation followed by the blue color. Similarly, Products A and D show the most deviation from the average. A combination of unique color and product corresponds to a particular cell in the data cube that has significant differences from cells with this combination of variables.

For example, the profit earned by Product A, having green color, is the highest when compared to the other products. Furthermore, the profit earned on Product D, with the color blue, is the second most significant cell in the data cube. We believe that these distinct differences in certain cells of a data cube disclose interesting information present in the underlying data in the form of navigational paths. We explain this further with the help of the results presented in Table 3.5.

Table 3.5: Comparison results of high ranked and low ranked dimensions and facts

Ranked Paths	Dimensions	Fact #1 Average (Profit)	Fact # 2 Average (Weight)
	All	6800	**50**
P 1	Color (*green*) & Product name (*Product A*)	10200	**45**
	Mean Deviation	**3400**	**5**
	All other paths	7000	**49**
	Mean Deviation	200	**1**
P 2	Color (*blue*) & Product name (*Product D*)	8000	**47**
	Mean Deviation	**1200**	**3**
	All other paths	6900	**49.5**
	Mean Deviation	100	**0.5**
--------	---	--------	**-------**
P n	Shape (*circular*)	6850	**50.2**
	Absolute Difference from average	50	**0.2**

Table 3.5 shows that the overall average profit is 6800 dollars when measured over all possible paths.

When the cube is explored through ranked path 1, the average profit rises to 10200. At the same time the (mean) deviation of *Profit* with respect to highly ranked path 1 is 3400 dollars. This compares to a mean profit value of 7000, taken across all possible paths. At the same time, the mean deviation in profit is only 200, thus illustrating the utility of path 1. Similarly, (ranked path 2) also shows correspondingly significant differences in both facts (Profit and Weight). On the other hand, the lowly ranked dimension *Shape*, with the low ranked dimensional value of (Circular), shows the least deviation from the average on both facts. This clearly shows that the amount of increase in profit is much higher with the highly ranked path when compared to the lowly ranked one.

These results also reveal the importance of MCA as an analytical tool in our proposed methodology. This technique not only assists in ranking dimensions, but also assists in identifying the dimensional values that have significant differences in the underlying data. This enabled us to pick the green and blue colors, as identified by Algorithm 1. Correspondingly, we picked a circular shape instead of any other shape for the low ranked path because the circular shape type has the lowest value for the *Shape* dimension. The hypothetical example presented in this section illustrates the suitability of our proposed methodology for discovering interesting information using highly ranked navigation paths (based on correspondence analysis) in data cubes.

The construction of informative data cubes allows the user to apply basic OLAP operations such as Drill-down, Roll-up, Slice and Dice in order to interactively explore the data to find meaningful patterns. However, pattern discovery through the use of OLAP is ultimately limited by the analyst's insights. Such insights may not extend to patterns that are hidden due to the sheer data volume and dimensionality of the data. Furthermore, data granularity introduces another complicating factor: patterns themselves change depending on the level of granularity, as we proceed down the dendogram the dynamics of relationship between variables are likely to change. For these reasons we apply association rule mining to enhance the knowledge discovery process.

3.1.8 Mine Association Rules from Schema

In this step, we apply the well-known *Apriori* algorithm (Agrawal, Imieliński et al. 1993) in order to generate rules from the multidimensional schema. In association rule mining, a rule is defined as an implication A → B where A, B are frequent items in the data. Strong rules meet user-specified thresholds on minimum support and minimum confidence. Support reflects the percentage of records that contain both A and B, while Confidence refers to the percentage of records containing B that also contain A. Both these measures are used to specify the significance of a rule.

We utilize the ranked dimensions to discover significant associations between them. For instance, in a 4 dimensional data cube, the two dimensions with the highest degree of impurity (largest entropy) can be targeted for rule discovery. Dimensions with high impurity are not easily predictable as the underlying data distribution tends to be uniform. Furthermore, the larger the entropy of a variable, the lesser information we know about the underlying variable as the underlying data distribution tends to be random in nature. Entropy is a measure of unpredictability or information content. Consider an example of a poll on some controversial sports issue. Often such polls happen because the outcome is not already obvious. In other words, the outcome is unpredictable, and learning the results after performing the poll highlight previously unknown information. This is basically an alternate way of saying that entropy of poll results is large. Another example is of a coin toss with a coin that has two heads and no tails has zero entropy and the outcome can be predicted perfectly as the coin will always come up heads. On the other hand, when the coin is fair, that it has the same probability of heads as well as tails, then entropy of the coin toss is largest as there is no way to predict the outcome of the toss ahead of time.

Low entropy means that the distribution is less random; the variable may have many low values and a few extreme values. Hence, such variables tend to be more predictable. Hence association rules that contain high entropy variables on the rule right hand side (consequent) will in general provide more insights than their low entropy counterparts, provided they meet standard rule evaluation thresholds such as rule Confidence and/or Importance. However, association rule mining has the potential to generate a large number of trivial rules (Tuzhilin and Adomavicius 2002). Although the rule base can be pruned by setting the rule *support* and *confidence* thresholds appropriately, there is no guarantee that the rules that survive would capture interesting patterns in the multidimensional data. One reason is that the support and confidence measures were designed to evaluate rules derived from transactional data and are not necessarily effective for multidimensional data (Nestorov and Jukic 2003). This limitation motivated us to introduce the next step of the methodology that is designed to evaluate the interestingness of rules generated from multidimensional schema.

3.1.9 Discover Diverse Association Rules

In this step, we evaluate the interestingness of the generated rules with the help of alternative evaluation measures. One such measure is known as *Importance* which captures the usefulness of a rule.

Rule importance is defined in equation 7.

$$\textit{Importance}\ (A \rightarrow B) = \log\ (\textit{probability}\ (B|A)\ /\ \textit{probability}\ (B|\ \textit{not}\ A)) \qquad (7)$$

The importance measure assesses the degree of correlation between dimensions. For instance, if importance of a rule is greater than 0 then it means the dimensions are positively correlated and vice versa. A positive importance means that the probability of observing the right hand side of the rule increases when the left hand side is true. Table

3.6 shows a list of hypothetical rules generated from cluster C1. We contrast the rules generated from cluster C1 with those generated from the same cluster that has the multidimensional schema structure imposed on it.

Such rules help in understanding the underlying association among different dimensions. For instance, Rule 1 without the schema predicts that the Color of a product will be ‘white” if Quality = good and Size = a. Also, the positive importance value indicates that there is a strong relationship between the Quality and Size dimensions. On the other hand, Rule 1 with the use of the schema predicts the same color by giving an association among a group of values present in each dimension, namely G1 and G2. Each group contains a set of diverse values which are semantically related.

For instance, Rule 1 with schema predicts the white color product if its *Quality* belongs to the values of 1st group G1 = [ok, bad] and size belongs to the values of 2nd group G2 = [*a*,*b*,*c*]. It is apparent that the importance value of rules generated with and without schema is the same but the rules generated from schema are more diverse in comparison to the rules without schema. For instance, rule 1 with schema predicts the same *Color* but is predicated on diverse *Quality* and *Size* values present in semantically related groups. Intuitively, for two rules R and S with the same importance value, rule R which has more triggering conditions in the rule antecedent for any given rule consequent is more valuable than a rule S with fewer trigger conditions as rule R is fired in a greater diversity of situations than rule S.

Table 3.6: Hypothetical rules from cluster C1

Rule #	Without multidimensional schema		With multidimensional schema	
	Rules	Imp	Rules	Imp
1	If Quality = [Good] and Size [10] → Color = white	1.80	If Quality [G1]and Size [G2] → Color = white	1.80
2	If Quality [ok] and Size [b] → Color = orange	1.29	If Quality [G2]and Size [G4] → Color = orange	1.29
3	If Quality [ok] and Size [e] → Color = purple	1.10	If Quality [G2]and Size [G3] → Color = purple	1.10
4	If Quality [bad] and Size [f] → Color = green	1.05	If Quality [G2]and Size [G3] → Color = green	1.05

However, we cannot conclude merely on the basis of multiple values present in the dimensional groups that the rules generated from the schema are more diverse. We need further concrete evidence to support our claim.

In order to do that, we conducted further evaluation using objective measures of diversity criteria, namely Rae, CON and Hill proposed by Zbidi et. al in (Zbidi, Faiz et al. 2006) . These measures provide concrete statistical evidence that the diversity of the set of rules produced from our semi-automatically generated multidimensional schema is higher when compared to the rules generated without schema.

We employ the above mentioned diversity measures for the evaluation of summary tables generated for the purpose of rule evaluation. These summary tables are basically deduced from the main dataset according to a given rule. For instance, given a rule R1 = Quality [Good] and Size [*b*] → Color = [*white*] generated from our example dataset *D*, the summary table S1 for this rule is a table with the set of records containing Quality = (good) with size = (b). Using these summary tables we evaluate the interestingness of rules. The three measures Rae, CON and Hill are defined in equations 8, 9 and 10 respectively.

$$Rae = \sum_{i=0}^{m} \frac{n_i\,(n_i - 1)}{N\,(N-1)} \qquad (8) \qquad\qquad CON = \sqrt{\frac{\left(\sum_{i=1}^{m} Pi^2\right) - \bar{q}}{1-\bar{q}}} \qquad (9)$$

$$Hill = 1 - \frac{1}{\sqrt{\sum_{i=1}^{m} Pi^3}} \qquad (10)$$

Here, m denotes the total number of rows in a summary table; n_i is the value of derived count attribute of each row in the summary table; N is the total count $N = \sum_{i=1}^{m} n_i$; $P_i = \frac{n_i}{N}$ is the actual probability of row r_i ; $\bar{q} = \frac{1}{\mathrm{m}}$ is the uniform probability of row r_i. With the help of these measures we rank the generated rules in terms of the diversity of information contained in each rule.

Table 3.7: Rule evaluation using advanced diversity measures for cluster C1

	Without multidimensional schema			With multidimensional schema		
Rule set	**Rae**	**CON**	**Hill**	**Rae**	**CON**	**Hill**
R1-R2	0.53	0.56	-2.7	**0.84**	**0.87**	**-0.3**
R3-R4	0.25	0.36	-3.7	**0.53**	**0.58**	**-1.7**

Table 3.7 depicts the values obtained for the diversity evaluation measures on the rules that were generated. It is apparent from Table 3.7 that all three diversity measures show a significant improvement for the rules generated with the multidimensional schema.

Summary

In this chapter, we presented a methodology for enhanced knowledge discovery and explained each step of our methodology with the help of a running example. The proposed methodology integrates mining techniques such as hierarchical clustering with multidimensional scaling in order to design multidimensional warehouse schema. We demonstrated that classical statistical methods for data analysis such as Principal Component Analysis and Multiple Correspondence Analysis can be successfully used in conjunction with hierarchical clustering to uncover useful information implicit in large multidimensional data cubes. Moreover, information gain measure can be effectively used to discover diverse association rules with high prediction accuracy. The methodology facilitated easy discovery of inter-relationships between numeric and nominal variables which are significant from both application and statistical perspectives. Furthermore, this useful and interesting knowledge can be discovered without excessive reliance on specialized domain knowledge.

Domain specialists, however, knowledgeable, cannot be expected to predict with high precision the dynamics of such relationships at different levels of data abstraction. The methodology allows users to efficiently design data cubes at multiple data abstraction levels, to find interesting regions in cubes, and to discover diverse association rules from multidimensional schema. In order to validate our claims, we now turn our attention to the application of the methodology on three real-world datasets in Chapter 4, 5 and 6.

Chapter 4

Case Study 1: Automobile Dataset

In this chapter, we present our first case study conducted on a real-world dataset taken from the University of California Irvine (UCI) machine learning repository (Asuncion and Newman 2010) , namely Automobile (Schlimmer 1985). The Automobile dataset has a small number of records only 205 has a rich mix of 11 nominal and 16 numeric variables that suits the objectives of our research. This benchmark dataset describes the specification of an automobile in terms of various characteristics, its assigned insurance risk rating and its normalized (financial) losses in use as compared to other automobiles. More detailed description of this dataset can be found at University of California – machine learning website - http://archive.ics.uci.edu/ml

4.1 Application of Agglomerative Hierarchical Clustering

As per the first step of the proposed methodology, we applied Agglomerative Hierarchical Clustering using the Hierarchical Clustering Explorer (HCE) tool developed by Jinwook, et. al (Jinwook and Shneiderman 2002) to generate a dendrogram. From the dendrogram generated we determined the suitable cut-off point and generated the binary tree using the procedure explained in Algorithm 1. The calculated threshold for cut-off point was 0.676, and we cut the dendrogram at this value and considered it to be the last level of our data abstraction hierarchy. There were a total of 5 levels of data abstraction till the cut-off point and the last level had 10 clusters in it.

4.2 Ranking of Numeric Variables via PCA

For implementing the second step of our proposed methodology, we used IBM's SPSS package to apply PCA analysis on each cluster. Firstly, we plotted the 16 numeric variables present in each cluster and identified that most of the clusters were discriminating well on only 1 component or factor as shown in Figure 4.1.

The number of components to be extracted is based on Eigen value analysis of all 16 numeric variables present in each cluster. The cut-off point for component extraction can be set by the user to a certain percentage of variance captured by a variable, for example 80%, 85%, 90% or 95%. As explained in the example presented in Chapter 3, we compared the factor loadings of two child clusters to rank the numeric variables in the parent cluster. Figure 4.2 shows the ranking of top and bottom 3 numeric variables in the cluster hierarchy after performing PCA and comparative analysis as explained in section 3.2 of Chapter 3.

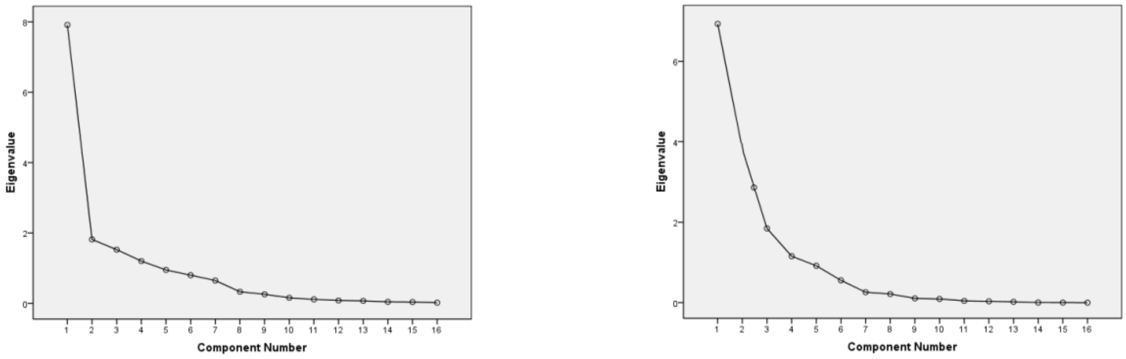
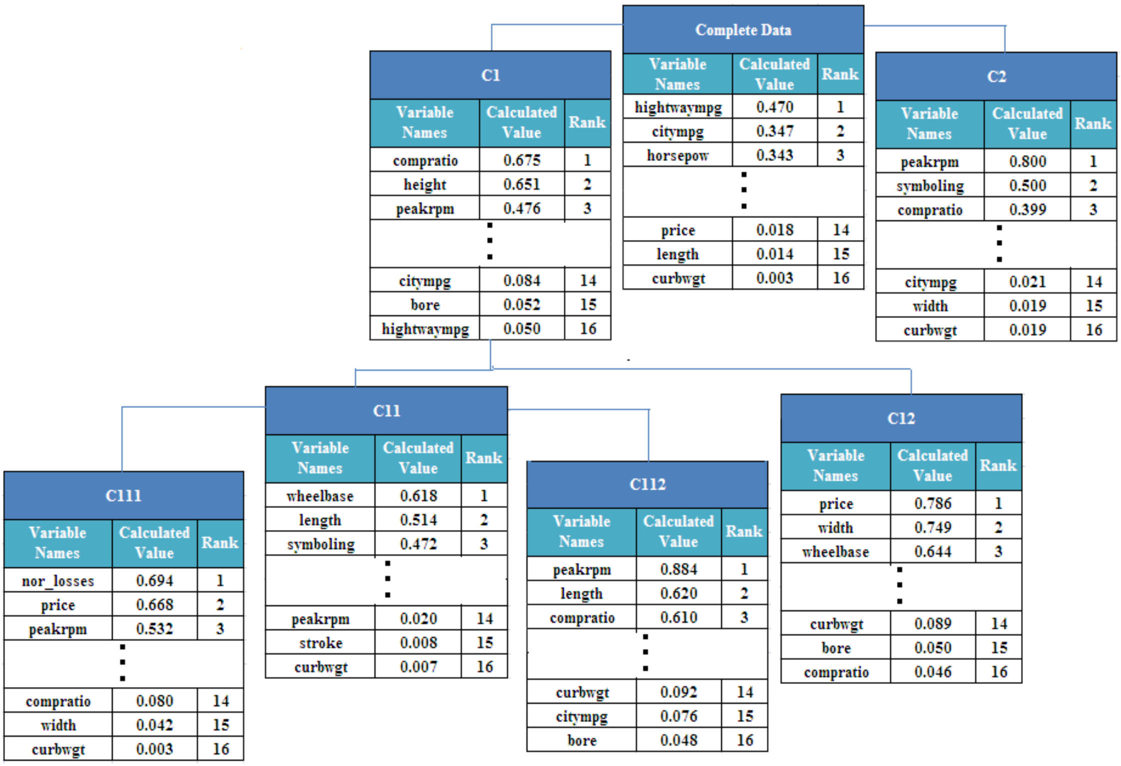


Figure 4.1: Scree plot of showing Eigen values of cluster C11 (left) and C12 (right)



Complete Data

Variable Names	Calculated Value	Rank
hightwaympg	0.470	1
citympg	0.347	2
horsepow	0.343	3
⋮		
price	0.018	14
length	0.014	15
curbwgt	0.003	16

C1

Variable Names	Calculated Value	Rank
compratio	0.675	1
height	0.651	2
peakrpm	0.476	3
⋮		
citympg	0.084	14
bore	0.052	15
hightwaympg	0.050	16

C2

Variable Names	Calculated Value	Rank
peakrpm	0.800	1
symboling	0.500	2
compratio	0.399	3
⋮		
citympg	0.021	14
width	0.019	15
curbwgt	0.019	16

C11

Variable Names	Calculated Value	Rank
wheelbase	0.618	1
length	0.514	2
symboling	0.472	3
⋮		
peakrpm	0.020	14
stroke	0.008	15
curbwgt	0.007	16

C12

Variable Names	Calculated Value	Rank
price	0.786	1
width	0.749	2
wheelbase	0.644	3
⋮		
curbwgt	0.089	14
bore	0.050	15
compratio	0.046	16

C111

Variable Names	Calculated Value	Rank
nor_losses	0.694	1
price	0.668	2
peakrpm	0.532	3
⋮		
compratio	0.080	14
width	0.042	15
curbwgt	0.003	16

C112

Variable Names	Calculated Value	Rank
peakrpm	0.884	1
length	0.620	2
compratio	0.610	3
⋮		
curbwgt	0.092	14
citympg	0.076	15
bore	0.048	16

Figure 4.2: Ranking of numeric variables present in Automobile dataset

It is clear from Figure 4.2 that the ranked lists of numeric variables have sharp differences across the data hierarchy. In other words, each data cluster has its own unique set of significant numeric variables. For instance, *highwaympg* (Highway miles per gallon), *citympg* (City miles per gallon) and *horsepow* (Horse power), which happen to be the most significant variables in the complete dataset, do not appear to be significant at lower levels of data abstraction.

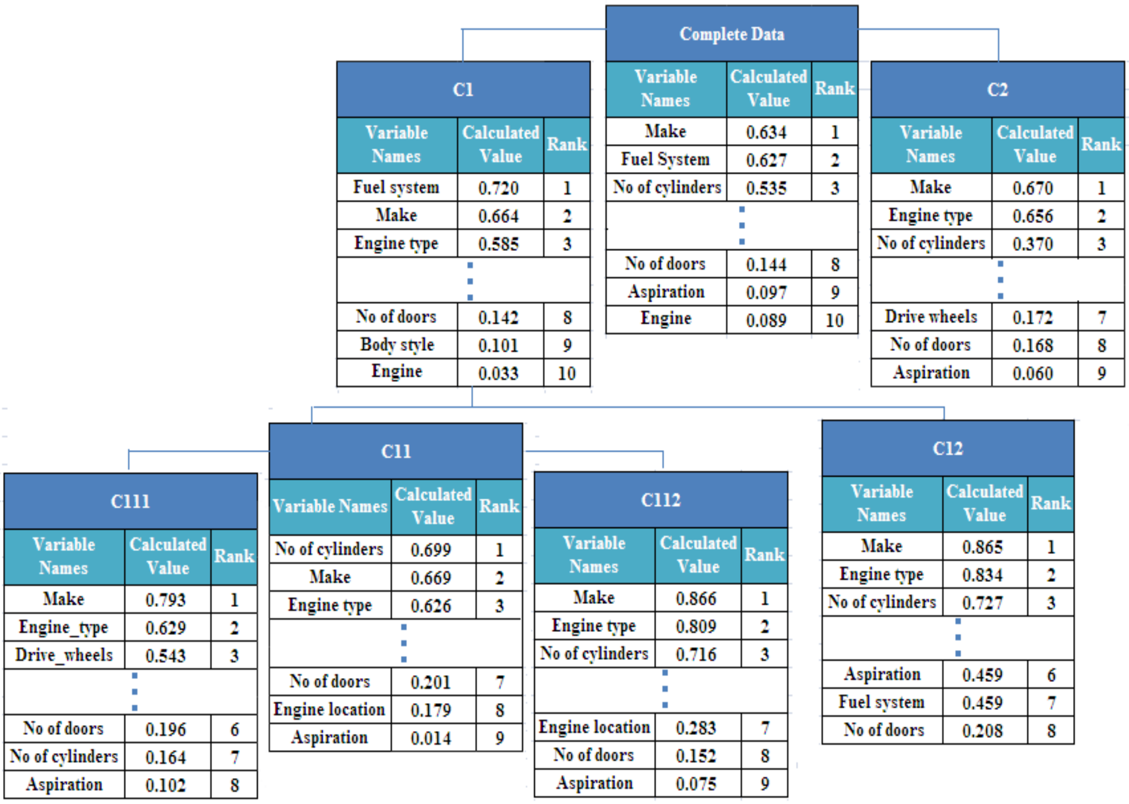
Interestingly, the most significant variable *highwaympg* in the complete unclustered dataset appears to be the least significant variable in cluster C1. Similarly, *citympg*, the second most significant variable in the dataset, took only 14th place in the ranking for cluster C2. This implies that the automobiles in the complete dataset are split into two separate clusters primarily on the basis of *higwaympg* and *citympg* variables and marginally on the basis of other variables such as *length* and *curbwgt* (Curb Weight) variables.

4.3 Ranking of Nominal Variables

In order to rank the nominal variables, we applied the two techniques discussed in Chapter 3, namely MCA and Information Gain to provide assistance to analysts on the basis of objective measures of information content in order to complement existing domain knowledge.

4.3.1 Application of Multiple Correspondence Analysis (MCA)

After establishing the significant numeric variables in each cluster, we rank the nominal variables in the cluster as part of the next step in our proposed methodology. We apply MCA using the SPSS package obtain factor loading (Eigen values) for each nominal variable. We rank the nominal variables based on their corresponding Eigen values from highest to lowest, and the results are depicted in Figure 4.3.



Complete Data

Variable Names	Calculated Value	Rank
Make	0.634	1
Fuel System	0.627	2
No of cylinders	0.535	3
⋮		
No of doors	0.144	8
Aspiration	0.097	9
Engine	0.089	10

C1

Variable Names	Calculated Value	Rank
Fuel system	0.720	1
Make	0.664	2
Engine type	0.585	3
⋮		
No of doors	0.142	8
Body style	0.101	9
Engine	0.033	10

C2

Variable Names	Calculated Value	Rank
Make	0.670	1
Engine type	0.656	2
No of cylinders	0.370	3
⋮		
Drive wheels	0.172	7
No of doors	0.168	8
Aspiration	0.060	9

C111

Variable Names	Calculated Value	Rank
Make	0.793	1
Engine_type	0.629	2
Drive_wheels	0.543	3
⋮		
No of doors	0.196	6
No of cylinders	0.164	7
Aspiration	0.102	8

C11

Variable Names	Calculated Value	Rank
No of cylinders	0.699	1
Make	0.669	2
Engine type	0.626	3
⋮		
No of doors	0.201	7
Engine location	0.179	8
Aspiration	0.014	9

C112

Variable Names	Calculated Value	Rank
Make	0.866	1
Engine type	0.809	2
No of cylinders	0.716	3
⋮		
Engine location	0.283	7
No of doors	0.152	8
Aspiration	0.075	9

C12

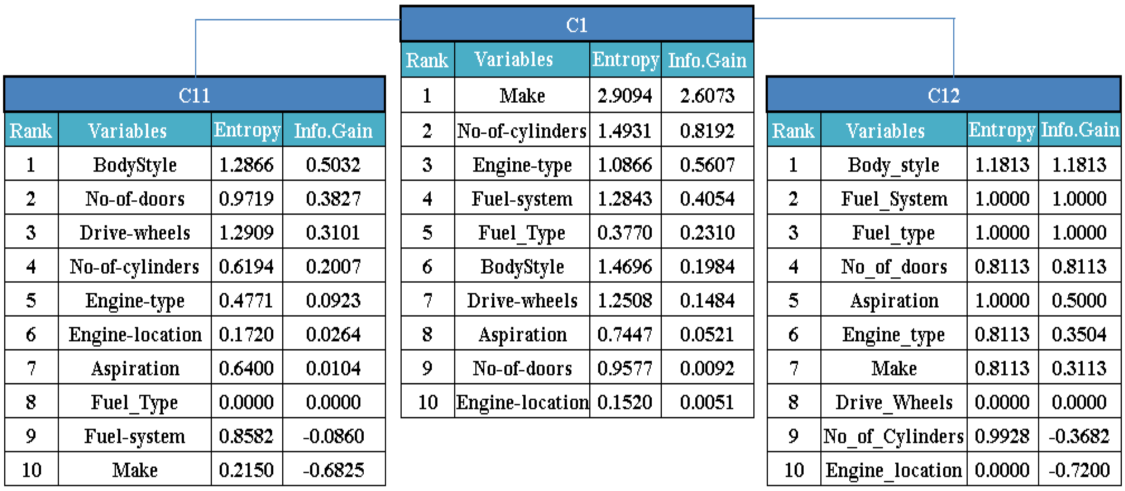
Variable Names	Calculated Value	Rank
Make	0.865	1
Engine type	0.834	2
No of cylinders	0.727	3
⋮		
Aspiration	0.459	6
Fuel system	0.459	7
No of doors	0.208	8

Figure 4.3: Ranking of nominal variables present in Automobile dataset

Similar to the ranking of numeric variables, the nominal variable ranking varies from cluster to cluster, depending on its position in the hierarchy. Each cluster has its own unique set of significant nominal variables.

4.3.2 *Analysis of nominal variables via Information Gain*

As an alternative, we rank the same nominal variables in each cluster based on information gain measure. Figure 4.4 shows the information gain based ranking for three clusters, at consecutive level in the hierarchy, namely C1, C11 and C12.



C11			
Rank	Variables	Entropy	Info.Gain
1	BodyStyle	1.2866	0.5032
2	No-of-doors	0.9719	0.3827
3	Drive-wheels	1.2909	0.3101
4	No-of-cylinders	0.6194	0.2007
5	Engine-type	0.4771	0.0923
6	Engine-location	0.1720	0.0264
7	Aspiration	0.6400	0.0104
8	Fuel_Type	0.0000	0.0000
9	Fuel-system	0.8582	-0.0860
10	Make	0.2150	-0.6825

C1			
Rank	Variables	Entropy	Info.Gain
1	Make	2.9094	2.6073
2	No-of-cylinders	1.4931	0.8192
3	Engine-type	1.0866	0.5607
4	Fuel-system	1.2843	0.4054
5	Fuel_Type	0.3770	0.2310
6	BodyStyle	1.4696	0.1984
7	Drive-wheels	1.2508	0.1484
8	Aspiration	0.7447	0.0521
9	No-of-doors	0.9577	0.0092
10	Engine-location	0.1520	0.0051

C12			
Rank	Variables	Entropy	Info.Gain
1	Body_style	1.1813	1.1813
2	Fuel_System	1.0000	1.0000
3	Fuel_type	1.0000	1.0000
4	No_of_doors	0.8113	0.8113
5	Aspiration	1.0000	0.5000
6	Engine_type	0.8113	0.3504
7	Make	0.8113	0.3113
8	Drive_Wheels	0.0000	0.0000
9	No_of_Cylinders	0.9928	-0.3682
10	Engine_location	0.0000	-0.7200

Figure 4.4: Ranking of nominal variables based on Information Gain measure

Similar to numeric variables, the list of ranked nominal variables also show sharp differences as we move from a higher level of data abstraction to a lower level. It can be seen from Figure 4.4 that Make which is a top ranked in cluster 1 is lowly ranked in cluster C11. Similarly, No-of-Cylinders which is the second most significant variable took 9th place in cluster C12. In other words, Make and No-of-Cylinders had the least randomness in parent cluster C1 but at the immediate lower level in the hierarchy these variables appear to be have the most randomness or impurity. We suggest that these highly ranked variables should be explored as the first choice in a given cluster C1 as they possess more versatile information that defines the split as compared to the lowly ranked variables which play a minimum role in the cluster split.

4.4 Grouping of Nominal Values via Multidimensional Scaling

After ranking nominal variables, we apply multidimensional scaling and group the values present in each nominal variable using Algorithm 2 presented in Chapter 3. Figure 4.5 shows the groupings obtained for the top ranked variable (Body-Style) in cluster C11 and C12.

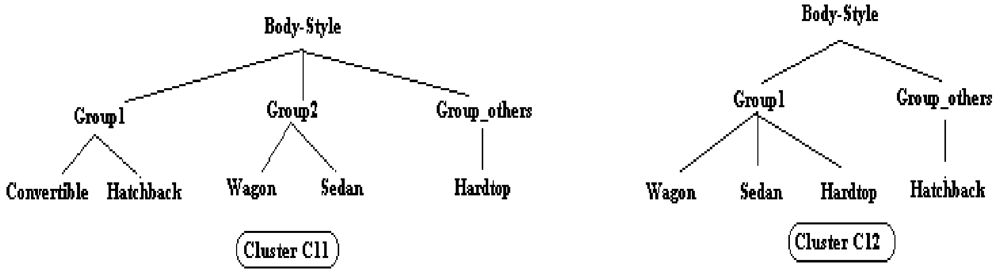


Figure 4.5: Groupings obtained after multidimensional scaling and application of Algorithm 2

We see that there is a difference in the number of groups and the values present in the groups across the two child clusters for the same nominal variable. Interestingly, Hardtop which had no similarities with any other body-style type in cluster C11 shows an affinity with Sedan and Wagon types in the sibling cluster C12. Moreover, Hatchback type becomes an outlier (having no similarities with any other body-style type) in cluster C12. Such unique and sharply contrasting patterns are difficult to obtain by relying on manual exploration alone in high dimensional and high volume datasets.

4.5 Generation of Multidimensional Schema

Advancing to the next step of our methodology, we generate multi-dimensional schema by using the numerical variables as facts and nominal variables as dimensions. The ranked lists of numeric and nominal variables serve as a starting point to constrain the multidimensional space. At this point the appropriate number of important dimensions (nominal variables) and facts (numeric variables) are selected for the creation of multidimensional schema.

For further exploration we analyse the distribution of the top 3 facts over the complete dataset and examine how their average values change over the levels of the cluster hierarchy. Figure 4.6 shows the average values of the top 3 facts taken over the complete dataset for the cluster hierarchy.

It can be seen from Figure 4.6 that the factor loadings based on Eigen values provide a sound basis for identifying variables responsible for the division of data into clusters. The 3 top ranked facts have distributed well into clusters C1 and C2. For instance, the automobiles whose *higwaympg* and *citympg* are lower than the overall average are present in cluster C1 while the automobiles with higher than average values are present in cluster C2. This validates our method of ranking the most significant facts in each cluster at the top. The two highlighted clusters in Figure 4.6, namely C2 and C12, have the overall highest value for highway and city miles per gallon and the lowest value for horse power as compared to all the other clusters in the hierarchy.

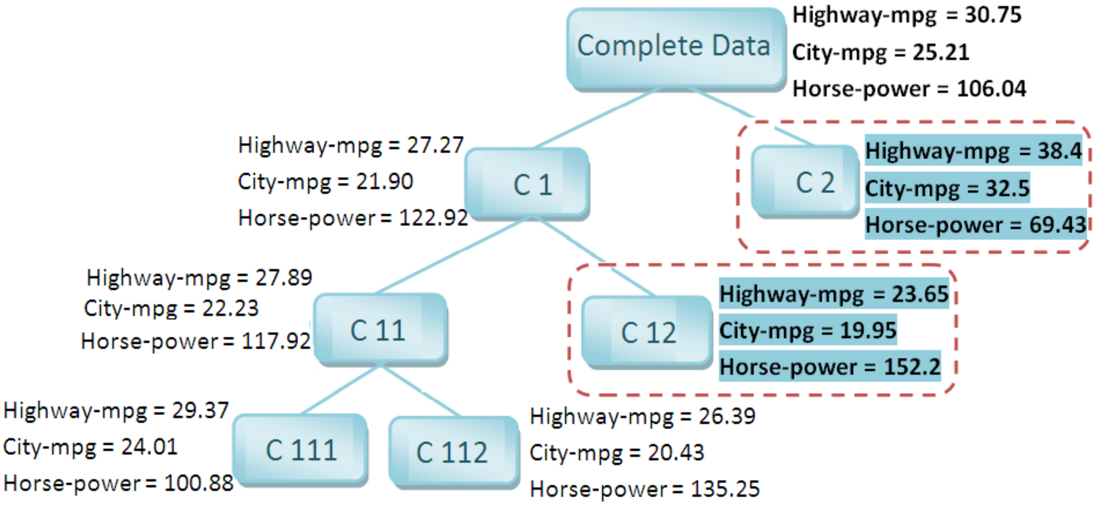


Figure 4.6: Hierarchical tree showing average values of highly ranked facts

Similarly, cluster C12 has the lowest values for city and highway miles per gallon and highest value for horse power. This means that if the user is interested in the analysis of automobiles with respect to the top 3 facts, then these two clusters would be of interest to explore the facts further using the list of ranked dimensions and their dimensional values.

4.6 Informative Data Cube Construction

After creating the multidimensional schema we construct data cubes at different levels in the cluster hierarchy. Figure 4.7 shows the 3 dimensional cubes structure of two clusters (C1 and C11) at different levels of the hierarchy. For easy visualization we have chosen 3 dimensional cube structure in Figure 4.7 though the methodology permits any number of dimensions and facts for cube construction.

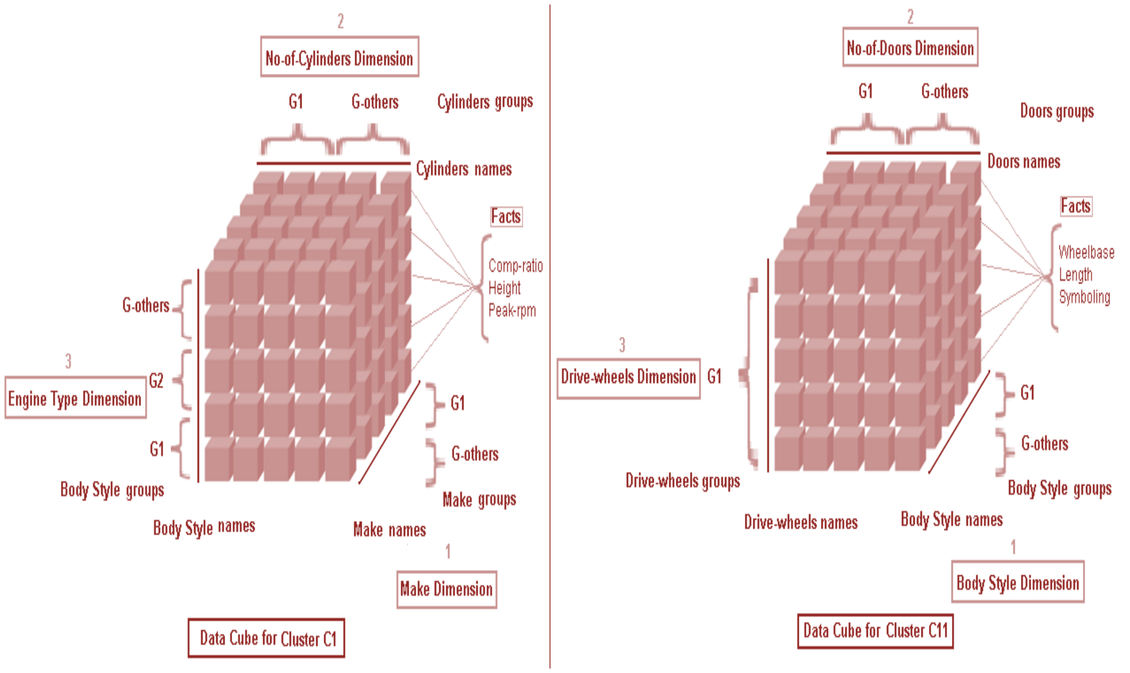


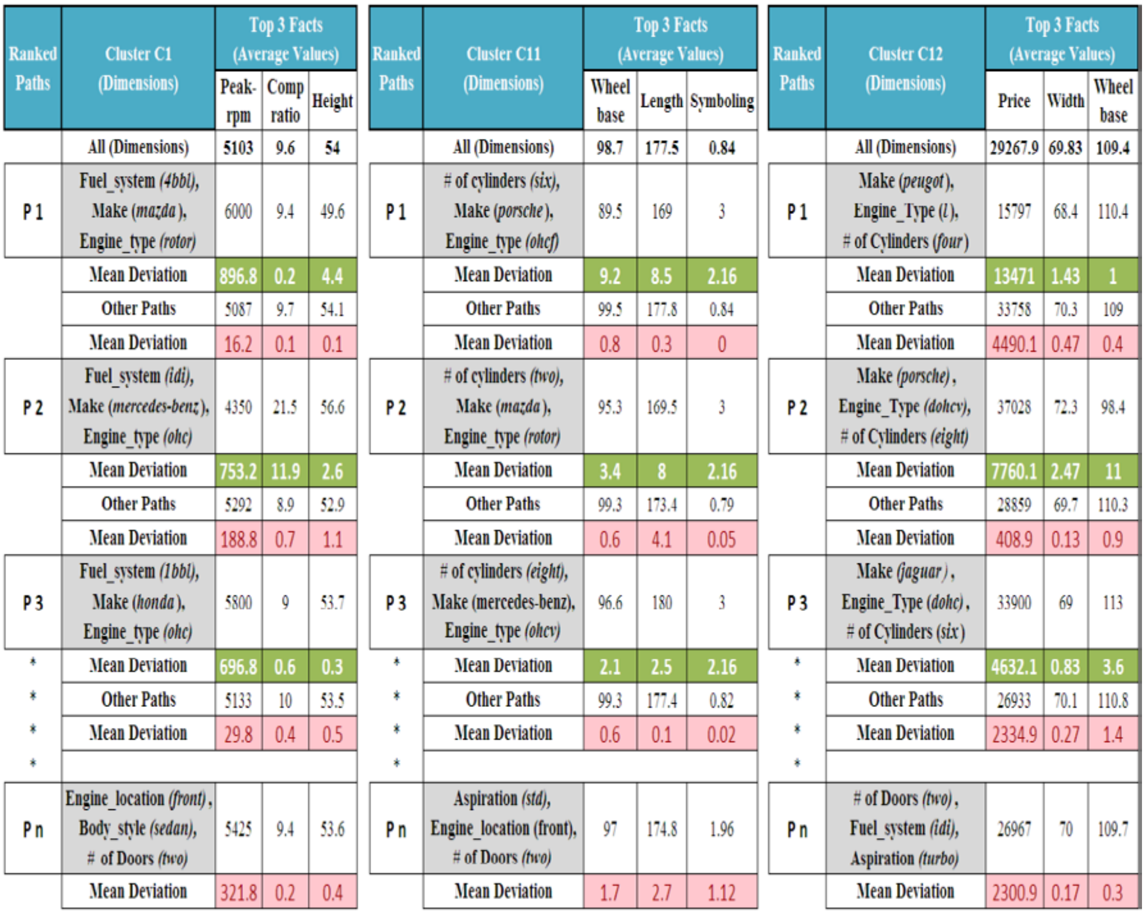
Figure 4.7: Comparison of 3-dimensional cubes at different levels of hierarchy

It can be seen from Figure 4.7 that the top 3 dimensions and facts in data cube C1 are totally absent in data cube C11. Our methodology suggests unique combinations of dimensions and facts for cube exploration using OLAP analysis. For instance, in data cube C1, the three suggested facts namely, *Comp-ratio*, *Height*, and *Peak-rpm* when explored through the ranked dimensions (*Make*, *No-of-Cylinders* and *Engine-type)* would give more significant information as compared to the lowly ranked dimensions and facts. Consider data cube C1 as an example; the three facts (*Comp-ratio, Height* and *Peak-rpm)* can be explored through 22 different *Make* types, 7 different *Engine* types and 6 distinct values for *No-of-Cylinders.* This makes a total of (22 x 6 x 7) 924 data cells in the cube to be explored through standard OLAP analysis.

By grouping together similar dimensional values this search space can be reduced significantly, while enabling meaningful patterns to be discovered. After grouping, we see that there are only 2 groups for *Make*, 3 groups for *Engine-type* and 2 groups for *No-of-Cylinders* dimension, thus yielding only (2x3x2) 12 distinct areas in the data cube for exploration. The grouping not only reduces the search space but also provides groups of dimensional values that have intra-group semantic affinity with each other. Furthermore, outliers for each dimension are highlighted by inserting values that are distant from all others in the *Group-others.* For instance, in the *Make* dimension, Subaru and Porsche are the two types of automobiles grouped in *Group-others* which bear no relation to the other 20 automobiles that were grouped together in *Group1*.

4.7 Exploration of Interesting Cube Regions

In this research, we claim that our proposed methodology provides highly ranked paths that reference interesting cube regions. In order to validate this claim, we took the top 3 ranked paths and compared them with the lowly ranked path suggested by our methodology for each data cluster. Figure 4.8 shows the results of the cube exploration through highly ranked paths for cluster C1, C11 and C12.



Ranked Paths	Cluster C1 (Dimensions)	Top 3 Facts (Average Values)		
		Peak-rpm	Comp ratio	Height
	All (Dimensions)	5103	9.6	54
P 1	Fuel_system *(4bbl)*, Make *(mazda)*, Engine_type *(rotor)*	6000	9.4	49.6
	Mean Deviation	896.8	0.2	4.4
	Other Paths	5087	9.7	54.1
	Mean Deviation	16.2	0.1	0.1
P 2	Fuel_system *(idi)*, Make *(mercedes-benz)*, Engine_type *(ohc)*	4350	21.5	56.6
	Mean Deviation	753.2	11.9	2.6
	Other Paths	5292	8.9	52.9
	Mean Deviation	188.8	0.7	1.1
P 3	Fuel_system *(1bbl)*, Make *(honda)*, Engine_type *(ohc)*	5800	9	53.7
*	Mean Deviation	696.8	0.6	0.3
*	Other Paths	5133	10	53.5
*	Mean Deviation	29.8	0.4	0.5
*				
P n	Engine_location *(front)* , Body_style *(sedan)*, # of Doors *(two)*	5425	9.4	53.6
	Mean Deviation	321.8	0.2	0.4

Ranked Paths	Cluster C11 (Dimensions)	Top 3 Facts (Average Values)		
		Wheel base	Length	Symboling
	All (Dimensions)	98.7	177.5	0.84
P 1	# of cylinders *(six)*, Make *(porsche)*, Engine_type *(ohcf)*	89.5	169	3
	Mean Deviation	9.2	8.5	2.16
	Other Paths	99.5	177.8	0.84
	Mean Deviation	0.8	0.3	0
P 2	# of cylinders *(two)*, Make *(mazda)*, Engine_type *(rotor)*	95.3	169.5	3
	Mean Deviation	3.4	8	2.16
	Other Paths	99.3	173.4	0.79
	Mean Deviation	0.6	4.1	0.05
P 3	# of cylinders *(eight)*, Make (mercedes-benz), Engine_type *(ohcv)*	96.6	180	3
*	Mean Deviation	2.1	2.5	2.16
*	Other Paths	99.3	177.4	0.82
*	Mean Deviation	0.6	0.1	0.02
*				
P n	Aspiration *(std)*, Engine_location (front), # of Doors *(two)*	97	174.8	1.96
	Mean Deviation	1.7	2.7	1.12

Ranked Paths	Cluster C12 (Dimensions)	Top 3 Facts (Average Values)		
		Price	Width	Wheel base
	All (Dimensions)	29267.9	69.83	109.4
P 1	Make *(peugot)*, Engine_Type *(l)*, # of Cylinders *(four)*	15797	68.4	110.4
	Mean Deviation	13471	1.43	1
	Other Paths	33758	70.3	109
	Mean Deviation	4490.1	0.47	0.4
P 2	Make *(porsche)* , Engine_Type *(dohcv)*, # of Cylinders *(eight)*	37028	72.3	98.4
	Mean Deviation	7760.1	2.47	11
	Other Paths	28859	69.7	110.3
	Mean Deviation	408.9	0.13	0.9
P 3	Make *(jaguar)* , Engine_Type *(dohc)* , # of Cylinders (*six*)	33900	69	113
*	Mean Deviation	4632.1	0.83	3.6
*	Other Paths	26933	70.1	110.8
*	Mean Deviation	2334.9	0.27	1.4
*				
P n	# of Doors *(two)* , Fuel_system *(idi)*, Aspiration *(turbo)*	26967	70	109.7
	Mean Deviation	2300.9	0.17	0.3

Figure 4.8: Results of cube exploration through ranked paths (Automobile dataset)

In order to avoid repetition, we present the results for these three clusters. However, the results for all other clusters are consistent and in line with the results presented in Figure 4.8. The right combination of dimensions and their values define a path for navigating in data cubes. This path refers to specific cells in a data cube where the interesting information resides. It can be seen from Figure 4.8 that the top 3 ranked paths consistently give better results as compared to the bottom ranked path, which means that the suggested ranked paths have the potential to reveal interesting information. Interestingness is a subjective term; in this research, we consider the most deviating values from the mean to be the more interesting.

For instance, in cluster C12 the average Price of all the automobiles (across all dimensions) is 29267.9 dollars. If the average Price is calculated across the highly ranked path (P1), which is (Make→Peugeot, Engine_Type→ l and #_of_Cylinders→ four), then the value decreases to 15797 dollars. This shows a deviation of 13471 dollars from the mean.

Similarly, we calculated the deviation of all other paths in the data cube and noted the mean deviation to be 4490.1 dollars only. This reveals that the highly ranked path possesses extremely significant price deviation when compared to all other cells in the data cube. On the other hand, the lowly ranked path (Pn) of cluster C12 consisting of the bottom ranked dimensions and values (# of doors → two, Fuel-system→ idi and Aspiration→ turbo) shows the least deviation (2300.9 dollars) in Price, which is less than all of the deviations covered by P1, P2 and P3. The same holds true for the other two facts present in cluster C12. It clearly shows that the top facts, when analyzed through highly ranked paths, tend to reveal more interesting information than the lowly ranked paths. Moreover, this behaviour is consistent across the three clusters shown in Figure 4.8.

These paths assist users in quickly identifying those cells in a data cube that have the highest deviations from the mean. This is typically the information sought by OLAP analysts who are interested in quickly finding regions among the large search space of data cubes that show large deviations from the norm. Finding such information in a timely manner without the use of automated support may not be feasible in the case of large dimensionality. The inclusion of each dimension exponentially increases the number of cells within a cube. For instance, in the C12 cube, the number of members for the Make dimension is 6, for Engine-type it is 5 and for No_of_cylinders it is 3, thus making up a total of 6 x 5 x 3 = 90 cells to be explored in order to find interesting information. For cube C1 the number is larger, with 21 Make members, 8 fuel-system members and 7 Engine-type members, giving a total of 1176 cells to be explored.

Even in a relatively small cube like C1, it may be next to impossible for a user to navigate through all the right combinations of dimensional members in order to identify the automobiles that, for instance, have the highest deviations for a single fact, say Peak-rpm. The ranked paths that we identify help to resolve this problem. It can be seen from Figure 5.8 that the three highly ranked paths (P1 to P3) suggest 3 unique Make members (mazda, mercedes-benz, honda) out of 21 with 3 unique fuel-system members (4bbl, idi, 1bbl) out of 8 and 2 Engine-type members (rotor, ohc) out of 7 for analyzing the top 3 facts (comp-ratio, height and peak-rpm). It is next to impossible for OLAP users to identify this significant set of dimensions and paths for analyzing any of the facts present in the data cube using a manual cube exploration approach. With our approach, users can pinpoint the cube regions to be explored in order to discover knowledge. Indeed, the assistance via ranked paths of dimensions and facts saves a considerable amount of time and effort and enhances the knowledge discovery process from data cubes.

4.8 Mining Association Rules from Multidimensional Schema

While exploration of interesting regions in cubes via ranked paths is a promising step towards the advancement of knowledge discovery from large datasets, it must be noted that exploration of the data cube via OLAP analysis by itself is limited in finding the associations and correlations that could exist among dimensions. Although our proposed methodology provides a more constrained space to the OLAP user to find patterns, pattern discovery could still be a laborious task. A case can be made that even in the constrained space a user may require some intelligent assistance in order to find hidden associations among the dimensions. On the other hand, an OLAP user after finding a behavioural pattern through OLAP analysis may require further support to drill down to find the dimensional attributes influencing that pattern of interest. In order to deal effectively with such cases, we apply the *Apriori* algorithm for discovering multidimensional association rules.

In order to assess the benefits of the multidimensional schema on knowledge discovery we apply rule mining on multidimensional schema on three clusters C1, C11 and C12 and compare the rule bases generated with those obtained from the original cluster data. Table 4.1 shows the top 10 rules generated for cluster C1 at minimum probability value of 0.4 and minimum importance value of 0.10.

Table 4.1: Rules generated with multidimensional schema

RULES WITHOUT SCHEMA (CLUSTER C1)		
No	**Rules**	**Imp**
R1	Make = alfa-romeo, No Of Cylinders = four → Body Style = convertible	1.410
R2	Make = alfa-romeo, No Of Cylinders = [All] → Body Style = convertible	1.310
R3	Make = porsche, No Of Cylinders = six → Body Style = hardtop	1.134
R4	Make = porsche, No Of Cylinders = [All] → Body Style = hardtop	0.981
R5	Make = plymouth, No Of Cylinders = four → Body Style = wagon	0.493
R6	Make = renault, No Of Cylinders = four → Body Style = wagon	0.493
R7	Make = dodge, No Of Cylinders = four → Body Style = wagon	0.493
R8	Make = volkswagen, No Of Cylinders = four → Body Style = wagon	0.493
R9	Make = mazda, No Of Cylinders = two → Body Style = hatchback	0.483
R10	Make = alfa-romeo, No Of Cylinders = six → Body Style = hatchback	0.362

Based on these thresholds the algorithm produced 47 rules in total. From the rule base produced we see that the two most highly ranked dimensions (*Make* and *No-of-*

cylinders) have a strong tendency to predict the value of the low ranked dimension (*Body-Style)*. Basically, the rules indicate that certain makes of car are distinctive in terms of their body style as some of the Alpha-Romeo models are convertibles whereas the Porsche models tend to come in hardtop version.

Table 4.2 shows the rules generated with the multidimensional structure imposed on cluster C1. We note that while the rules generated are more informative as the *Make* and *No Of Cylinders* terms now refer to groups rather than individual values. The dimensional schema has produced rules that distinguish classes of automobiles from each other. One class belongs to those automobiles whose makes are in group 1, who have x, y and z number of cylinders and are *convertibles*. On the other hand, when automobiles in another *Make* group (group others) are considered the body style tends to be *hardtop*, instead of *convertible*. These rules are more compact, easier to understand and convey more information to an end user than a plethora of rules that cover each and every combination of *Make* and number of *Cylinders*.

Table 4.2: Rules generated with multidimensional schema

RULES WITHOUT SCHEMA (CLUSTER C1)		
No	**Rules**	**Imp**
R1	Make Group = group1, No Of Cylinders Group = group1 →Body Style Name = convertible	1.410
R2	Make Group = group1, No Of Cylinders Group = [All] →Body Style Name = convertible	1.310
R3	Make Group = group_others, No Of Cylinders Group = group1 →Body Style Name = hardtop	1.134
R4	Make Group = group_others, No Of Cylinders Group = [All] →Body Style Name = hardtop	0.981
R5	Make Group = group1, No Of Cylinders Group = group1 →Body Style Name = wagon	0.493
R6	Make Group = group_others, No Of Cylinders Group = [All] → Body Style Name = wagon	0.493
R7	Make Group = group1, No Of Cylinders Group = group_others → Body Style Name = hatchback	0.483
R8	Make Group = group1, No Of Cylinders Group = [All] → Body Style Name = hatchback	0.403
R9	Make Group = group_others, No Of Cylinders Group = group1→ Body Style Name = hatchback	0.362
R10	Make Group = group1, No Of Cylinders Group = [All] → Body Style Name = sedan	0.302

For instance, the first rule generated without schema can be interpreted as: if *Make* of an automobile is [*alfa-romeo*] and *No-of-Cylinders* are [*four*] then the *Body-Style* of that automobile is predicted to be [*convertible*]. On the other hand, the first rule from the schema that utilized the dimensional group level suggests that if *Make* and *No-of-Cylinders* of an automobile belongs to [group1] then *Body-Style* value is [*convertible*]. Here, [group1] of *Make* has 19 distinct values (*peugeot, jaguar, nissan, mercedez-benz,* *saab, mazda, toyota, volvo, honda, alfa-rmero, audi, volkswagon, mitsubishi, isuzu, dodge, plymouth, bmw, mercury, renault)* while *No-of-Cylinders* has 5 distinct values *(six, twelve, eight, four, five)* for [group1]. In other words, Rule1 in Table 4.2 provides much more diverse information as compared to the first rule of Table 4.1 for the same *importance* value.

4.9 Evaluation of Rule Interestingness via Diversity Criterion

In order to quantitatively verify whether the rules produced through our multidimensional design scheme are superior, we evaluate the rule bases on three objective interestingness measures, *Rae*, *CON* and *Hill* that were introduced in section 3.9 of Chapter 3. For a given cluster we rank the rules in descending order of rule Importance and then compare the first *k* (where *k* ranges from 6 to 10, in steps of 1) rules produced against each other on the 3 chosen interestingness measures. Table 4.3 reveals that the multidimensional schema consistently outperforms the raw cluster structure, irrespective of the value of *k* and the level of data granularity.

Table 4.3: Rule interestingness comparison using diversity measures

Cluster Names	Rule sets	NO Schema Rae	With Schema Rae	NO Schema CON	With Schema CON	NO Schema Hill	With Schema Hill
C1	**R1-R6**	0.139	**0.196**	0.168	**0.221**	-3.919	**-3.473**
	R1-R7	0.116	**0.179**	0.154	**0.233**	-4.701	**-3.823**
	R1-R8	0.100	**0.151**	0.142	**0.203**	-5.502	**-4.682**
	R1-R9	0.086	**0.142**	0.132	**0.195**	-6.321	**-5.446**
	R1-R10	0.076	**0.117**	0.122	**0.167**	-7.154	**-6.287**
C11	**R1-R6**	0.246	**0.291**	0.310	**0.396**	-2.784	**-2.079**
	R1-R7	0.206	**0.271**	0.280	**0.393**	-3.454	**-2.272**
	R1-R8	0.173	**0.220**	0.244	**0.473**	-4.270	**-3.126**
	R1-R9	0.152	**0.189**	0.224	**0.303**	-4.927	**-3.755**
	R1-R10	0.151	**0.186**	0.246	**0.317**	-4.965	**-3.791**
C12	**R1-R6**	0.178	**0.203**	0.196	**0.244**	-3.796	**-3.261**
	R1-R7	0.169	**0.186**	0.233	**0.256**	-3.976	**-3.574**
	R1-R8	0.161	**0.181**	0.253	**0.287**	-4.158	**-3.682**
	R1-R9	0.153	**0.174**	0.265	**0.294**	-4.343	**-3.791**
	R1-R10	0.146	**0.170**	0.271	**0.301**	-4.529	**-3.899**

These results thus confirm our qualitative analysis that the use of the multidimensional results in the production of more informative and diverse knowledge to the user in the form of rules.

Finally, we compared the predictive power of rules generated without multidimensional schema with the rules generated from schema. We took *Make* and *No_of_Cylinder* as the input variables in order to predict the *Body_Style* of automobiles. We generated the rules using a randomly chosen subset of 70% of data and then used the remaining 30% to evaluate the predictive accuracy. We ran 10 tests where each test randomly picks unique test data to ensure the predictive accuracy of the generated rules to predict the *Body_Style* of the automobiles accurately. Table 4.4 shows the accuracy percentage for the 10 tests.

From this case study, we note that rule mining performed on the multidimensional schema designed and constructed with the help of hierarchical clustering and multidimensional scaling technique generates diverse rules with greater prediction accuracy. In the next chapter, we discuss the results of second case study which is performed on a larger dataset.

Table 4.4: Rule prediction accuracy

Prediction Tests	Cluster C1		Cluster C11		Cluster C12	
	Schema	**No Schema**	**Schema**	**No Schema**	**Schema**	**No Schema**
Test 1	48.7	43.9	91.4	88.5	100	100
Test 2	48.7	36.5	94.2	85.7	100	100
Test 3	46.3	39	88.5	85.7	100	83.3
Test 4	36.5	41.4	88.5	91.4	83.3	100
Test 5	34.1	43.9	94.2	77.1	100	83.3
Test 6	48.7	43.9	88.5	91.4	100	87.7
Test 7	40.3	33.5	91.2	83.7	83.3	83.3
Test 8	44.7	41	88.5	85.7	100	83.3
Test 9	34.5	36.4	93.5	88.3	83.3	100
Test 10	37.1	31.9	94.2	77.1	83.3	83.7
Average Percentage	**41.96**	39.14	**91.27**	85.46	**93.32**	90.46

Summary

In this chapter, we presented our first case study conducted on a real-world dataset taken from the University of California Irvine (UCI) machine learning repository (Asuncion and Newman 2010), namely Automobile (Schlimmer 1985). The Automobile dataset has a small number of records only 205 has a rich mix of 11 nominal and 16 numeric variables that suits the objectives of our research. This benchmark dataset describes the specification of an automobile in terms of various characteristics, its assigned insurance risk rating and its normalized (financial) losses in use as compared to other automobiles.

We applied Agglomerative Hierarchical Clustering to generate a dendrogram. The calculated threshold for cut-off point for the dendrogram was 0.676, and we cut the dendrogram at this value and considered it to be the last level of our data abstraction hierarchy. There were a total of 5 levels of data abstraction till the cut-off point and the last level had 10 clusters in it. We then ranked the numeric variables in each cluster and found that the ranked lists of numeric and nominal variables have sharp differences across the data hierarchy. In other words, each data cluster has its own unique set of significant numeric and nominal variables.

Our methodology also suggested unique combinations of dimensions and facts for cube exploration using OLAP analysis. For instance, in data cube C1, the three suggested facts namely, *Comp-ratio, Height,* and *Peak-rpm* when explored through the ranked dimensions (*Make, No-of-Cylinders* and *Engine-type)* gave more significant information as compared to the lowly ranked dimensions and facts. Furthermore, the grouping of nominal values not only reduced the search space but also provided groups of dimensional values that have intra-group semantic affinity with each other. Furthermore, outliers for each dimension were highlighted by inserting values that are distant from all others in the *Group-others.* For instance, in the *Make* dimension, Subaru and Porsche are the two types of automobiles grouped in *Group-others* which bear no relation to the other 20 automobiles that were grouped together in *Group1*. The ranked paths suggested by the methodology assisted in quickly identifying those cells in a data cube that have the highest deviations from the mean. This is typically the information sought by OLAP analysts who are interested in quickly finding regions among the large search space of data cubes that show large deviations from the norm.

The case study also revealed association rules generated through schema are more compact, easier to understand and convey more information to an end user than a plethora of rules that cover each and every combination of values of the variables involved in rule mining process. For instance, the first rule generated without schema was: if *Make* of an automobile is [*alfa-romeo*] and *No-of-Cylinders* are [*four*] then the *Body-Style* of that automobile is predicted to be [*convertible*]. On the other hand, the first rule from the schema that utilized the dimensional grouping suggested that if *Make* and *No-of-Cylinders* of an automobile belongs to [group1] then *Body-Style* value is [*convertible*]. Here, [group1] of *Make* had 19 distinct values (*peugeot, jaguar, nissan, mercedez-benz, saab, mazda, toyota, volvo, honda, alfa-rmero, audi, volkswagon, mitsubishi, isuzu, dodge, plymouth, bmw, mercury, renault)* while *No-of-Cylinders* had 5 distinct values *(six, twelve, eight, four, five)* for [group1]. We also quantitatively verified the rules produced through our multidimensional design schema were superior, on three objective interestingness measures. Finally, we compared the predictive power of rules generated without multidimensional schema with the rules generated from schema and found out that the rules generated through schema were not only diverse but also have better prediction accuracy.

Chapter 5

Case Study 2: Adult Dataset

In this chapter, we present our second case study conducted on a much larger dataset as compared to *Automobile* dataset. It is the *Adult* (Kohavi and Becker 1996) dataset which consists of 48,842 records with eight nominal and five numeric variables as shown in Table 5.1. This benchmark dataset was extracted from the US Census bureau website using a data extraction system. More detailed description of this dataset can be found at University of California – machine learning website (Asuncion and Newman 2010).

Table 5.1: Numeric and nominal variables present in *Adult* dataset

Numeric Variables	Nominal Variables	Distinct values in each nominal variable
Age	Sex	2
Hours Per Week	Race	5
Capital Gain	Relationship	6
Capital Loss	Marital Status	7
Final Weight	Work Class	8
	Occupation	14
	Education	16
	Country	41

5.1 Application of Agglomerative Hierarchical Clustering

For the first step of hierarchical clustering, we removed the missing values from the dataset and used 30,162 records (61 % of the total) to generate clusters at multiple levels of data abstraction. After the dendrogram was generated, we calculated the inconsistency coefficient value for determining the cut-off and generate the binary tree using the procedure explained in Algorithm 1. The calculated threshold for cut-off point was 0.623 so we cut the dendogram at this value and considered it to be the last level of our data abstraction. There were a total of 9 levels of data abstraction till the cut-off point and the last level had 18 clusters in it.

Likewise, in the previous case study, we plotted the numeric variables present in each cluster to fix the number of principal components to be extracted. Most clusters were discriminating well on 1 component as depicted in Figure 5.1. The application procedure remains the same as in the previous case study. Figure 5.1 shows the scree plots and it can be seen that the variable in the clusters are discriminated well on 1 component.

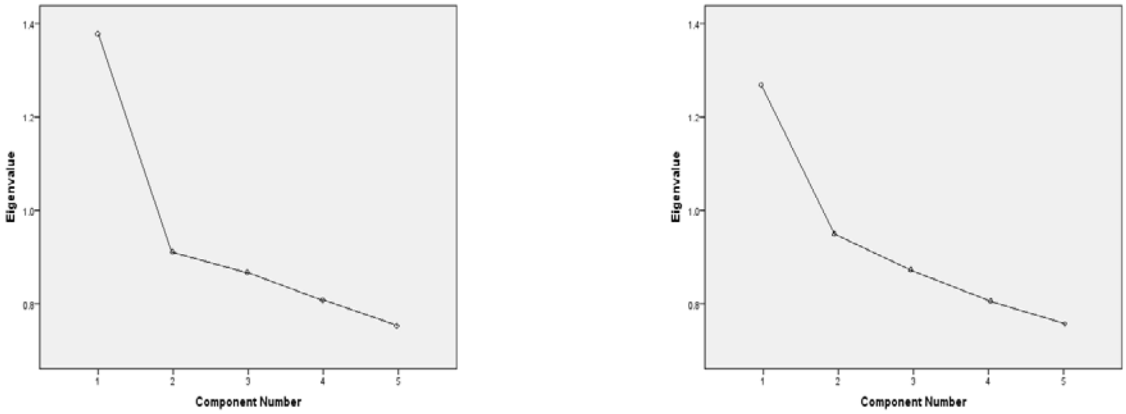
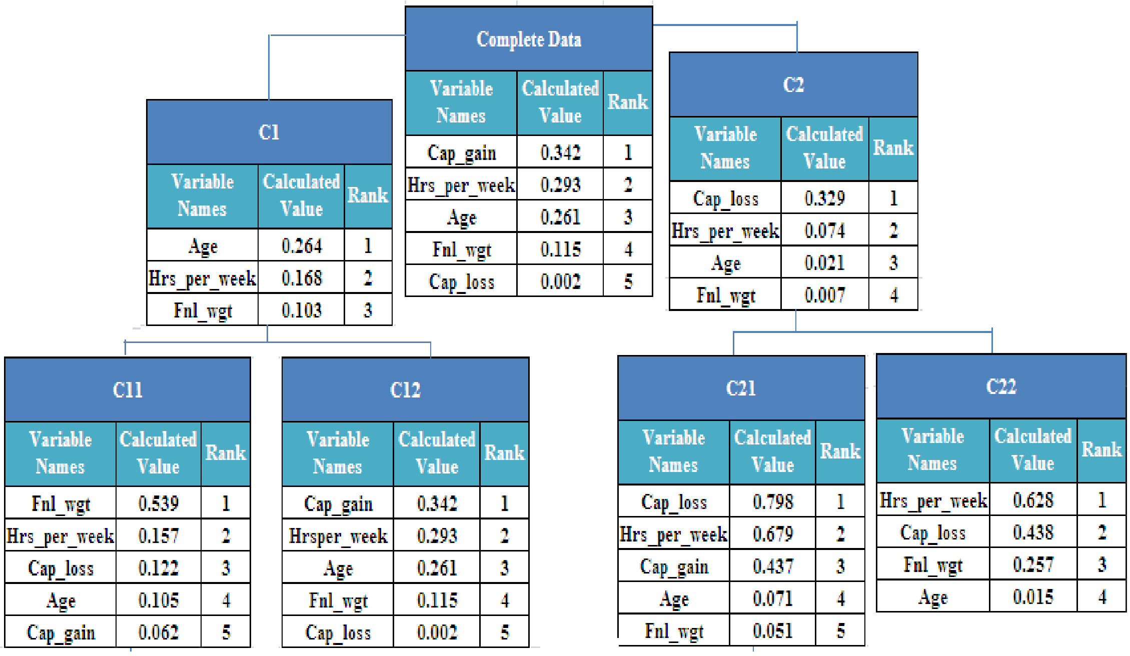


Figure 5.1: Scree plot showing Eigen values of cluster C11 (left) and C12 (right)

5.2 Ranking of Numeric Variables via PCA

We rank the numeric variables present in each cluster and observe that the ranking is unique to each cluster. Similar to the previous case study, sharp differences in rankings for a given variable appear at different levels in cluster hierarchy. The rankings computed for the first few levels of the hierarchy are shown in Figure 5.2.



Complete Data

Variable Names	Calculated Value	Rank
Cap_gain	0.342	1
Hrs_per_week	0.293	2
Age	0.261	3
Fnl_wgt	0.115	4
Cap_loss	0.002	5

C1

Variable Names	Calculated Value	Rank
Age	0.264	1
Hrs_per_week	0.168	2
Fnl_wgt	0.103	3

C2

Variable Names	Calculated Value	Rank
Cap_loss	0.329	1
Hrs_per_week	0.074	2
Age	0.021	3
Fnl_wgt	0.007	4

C11

Variable Names	Calculated Value	Rank
Fnl_wgt	0.539	1
Hrs_per_week	0.157	2
Cap_loss	0.122	3
Age	0.105	4
Cap_gain	0.062	5

C12

Variable Names	Calculated Value	Rank
Cap_gain	0.342	1
Hrsper_week	0.293	2
Age	0.261	3
Fnl_wgt	0.115	4
Cap_loss	0.002	5

C21

Variable Names	Calculated Value	Rank
Cap_loss	0.798	1
Hrs_per_week	0.679	2
Cap_gain	0.437	3
Age	0.071	4
Fnl_wgt	0.051	5

C22

Variable Names	Calculated Value	Rank
Hrs_per_week	0.628	1
Cap_loss	0.438	2
Fnl_wgt	0.257	3
Age	0.015	4

Figure 5.2: Ranked lists of numeric variables present in Adult dataset clusters

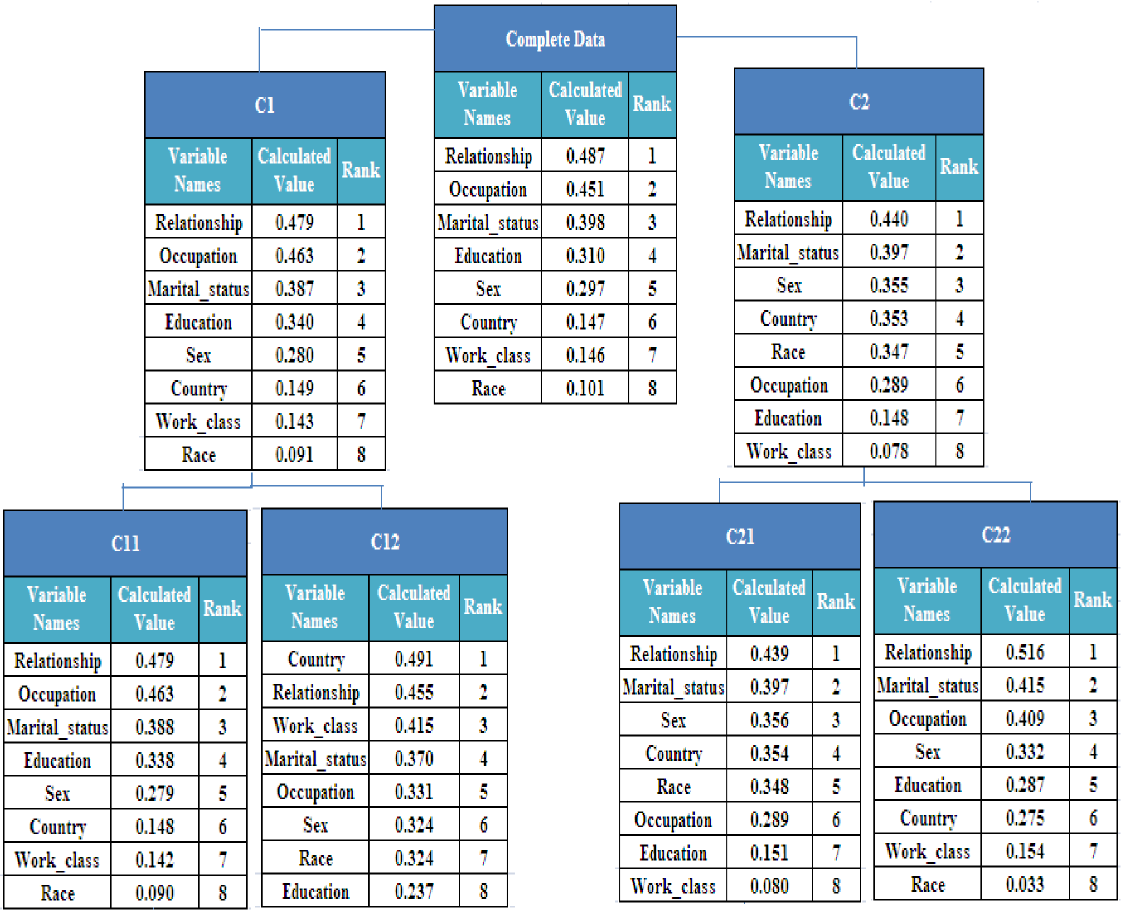
Figure 6.2 shows the ranking of numeric variables in the cluster hierarchy after performing PCA and comparative analysis as explained in section 3.2 of Chapter 3. Interestingly, some clusters such as C1, C2 and C22 have fewer than five numeric variables in each of them. This is due to the fact that the omitted numeric variables have *variance* equal to zero. This is because the Eigen values for variables having zero variance do not exist. We omit such variables because they do not play any role in multidimensional analysis.

5.3 Ranking of Nominal Variables

Similar to case study 1, we rank nominal variables present in each cluster via two separate data analysis techniques in order to achieve the two main objectives of this research; i) identification of interesting cube regions and ii) discovery of diverse association rules.

5.3.1 Application of Multiple Correspondence Analysis (MCA)

After establishing the significant numeric variables in each cluster, we rank the nominal variables in the cluster for finding interesting cube regions. The application procedure remains the same and the nominal variables in each cluster are ranked on the basis of Eigen values from highest to lowest. The rankings of first few levels are shown in Figure 5.3.



Complete Data

Variable Names	Calculated Value	Rank
Relationship	0.487	1
Occupation	0.451	2
Marital_status	0.398	3
Education	0.310	4
Sex	0.297	5
Country	0.147	6
Work_class	0.146	7
Race	0.101	8

C1

Variable Names	Calculated Value	Rank
Relationship	0.479	1
Occupation	0.463	2
Marital_status	0.387	3
Education	0.340	4
Sex	0.280	5
Country	0.149	6
Work_class	0.143	7
Race	0.091	8

C2

Variable Names	Calculated Value	Rank
Relationship	0.440	1
Marital_status	0.397	2
Sex	0.355	3
Country	0.353	4
Race	0.347	5
Occupation	0.289	6
Education	0.148	7
Work_class	0.078	8

C11

Variable Names	Calculated Value	Rank
Relationship	0.479	1
Occupation	0.463	2
Marital_status	0.388	3
Education	0.338	4
Sex	0.279	5
Country	0.148	6
Work_class	0.142	7
Race	0.090	8

C12

Variable Names	Calculated Value	Rank
Country	0.491	1
Relationship	0.455	2
Work_class	0.415	3
Marital_status	0.370	4
Occupation	0.331	5
Sex	0.324	6
Race	0.324	7
Education	0.237	8

C21

Variable Names	Calculated Value	Rank
Relationship	0.439	1
Marital_status	0.397	2
Sex	0.356	3
Country	0.354	4
Race	0.348	5
Occupation	0.289	6
Education	0.151	7
Work_class	0.080	8

C22

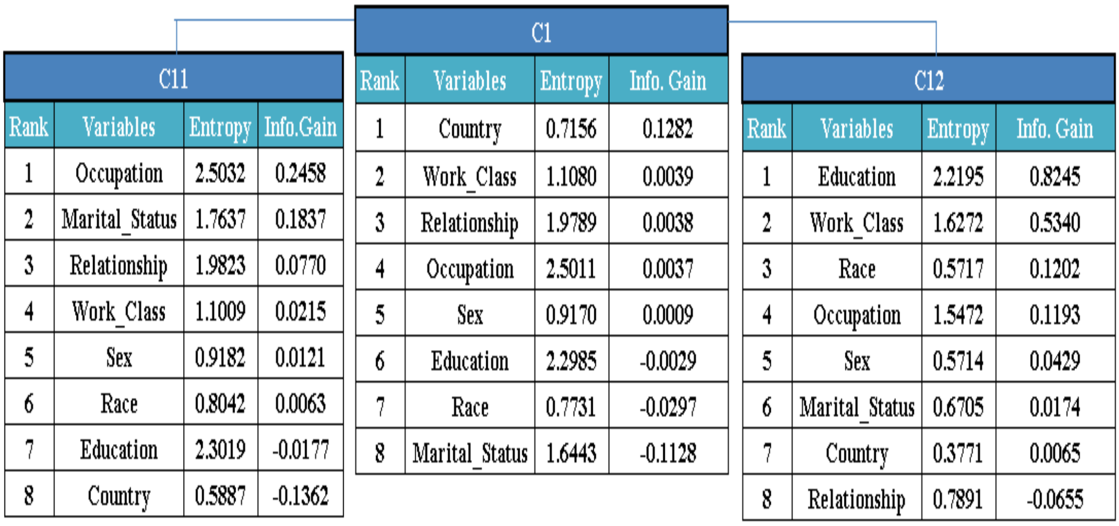
Variable Names	Calculated Value	Rank
Relationship	0.516	1
Marital_status	0.415	2
Occupation	0.409	3
Sex	0.332	4
Education	0.287	5
Country	0.275	6
Work_class	0.154	7
Race	0.033	8

Figure 5.3: Ranked list of nominal variables present in Adult dataset

Similar to the ranking of numeric variables, the nominal variable ranking varies from cluster to cluster, depending on its position in the hierarchy. Each cluster has its own unique set of significant nominal variables.

5.3.2 Analysis of nominal variables via Information Gain

In order to discover diverse association rules, we rank the same nominal variables in each cluster based on information gain measure. Figure 5.4 shows the information gain based ranking for three clusters, at consecutive levels in the hierarchy, namely C1, C11 and C12.



C1

Rank	Variables	Entropy	Info. Gain
1	Country	0.7156	0.1282
2	Work_Class	1.1080	0.0039
3	Relationship	1.9789	0.0038
4	Occupation	2.5011	0.0037
5	Sex	0.9170	0.0009
6	Education	2.2985	-0.0029
7	Race	0.7731	-0.0297
8	Marital_Status	1.6443	-0.1128

C11

Rank	Variables	Entropy	Info.Gain
1	Occupation	2.5032	0.2458
2	Marital_Status	1.7637	0.1837
3	Relationship	1.9823	0.0770
4	Work_Class	1.1009	0.0215
5	Sex	0.9182	0.0121
6	Race	0.8042	0.0063
7	Education	2.3019	-0.0177
8	Country	0.5887	-0.1362

C12

Rank	Variables	Entropy	Info. Gain
1	Education	2.2195	0.8245
2	Work_Class	1.6272	0.5340
3	Race	0.5717	0.1202
4	Occupation	1.5472	0.1193
5	Sex	0.5714	0.0429
6	Marital_Status	0.6705	0.0174
7	Country	0.3771	0.0065
8	Relationship	0.7891	-0.0655

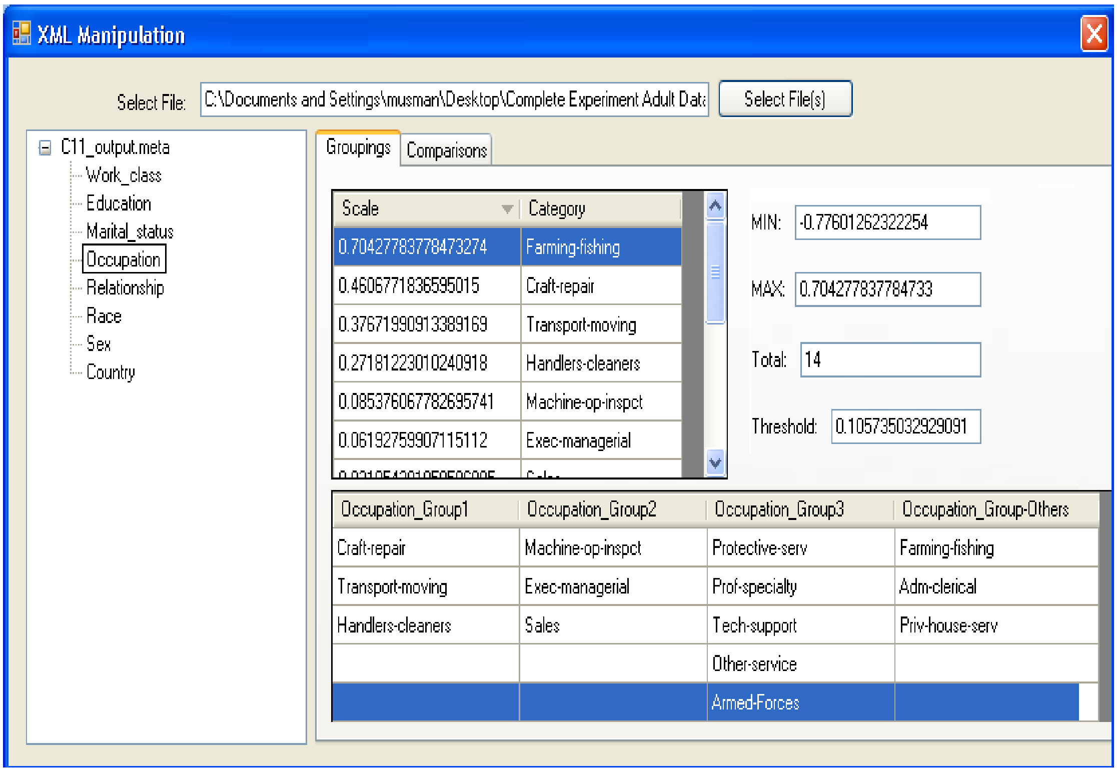
Figure 5.4: Ranking of nominal variables based on Information Gain measure

Similar to numeric variables, the list of ranked nominal variables also show sharp differences as we move from a higher level of data abstraction to a lower level. It can be seen from Figure 5.4 that Country which is top-ranked in cluster C1 is lowest ranked in cluster C11. Similarly, Relationship which is the third most significant variable in C1 took last place in cluster C12. In other words, Country and Relationship had the least randomness in parent cluster C1 but at an immediate lower level in the hierarchy these variables appear to be having the most randomness or impurity.

Similar to our previous case study, we observe the emergence of sharp patterns in both numeric and nominal variables as we go down in the cluster hierarchy. We suggest that these highly ranked variables should be explored as a first choice in a given cluster C1 as they possess more versatile information that defines the split as compared to the lowly ranked variables which play a minimum role in the cluster split.

5.4 Grouping of Nominal Values via Multidimensional Scaling

After ranking the nominal variables, we apply multidimensional scaling and group the values present in each nominal variable using Algorithm 2 presented in Chapter 3. We implemented Algorithm 2 and developed a prototype that shows the grouped nominal values for each nominal variable. Figure 5.5 shows the groupings obtained for variable (Occupation) in C11 in the interface of our developed prototype. The prototype shows the nominal variables present in the cluster in the tree view on the left hand side and displays the different groups and the values present for each group in a data grid for a selected variable.



XML Manipulation

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

Select File(s)

C11_output.meta
- Work_class
- Education
- Marital_status
- Occupation
- Relationship
- Race
- Sex
- Country

Groupings | Comparisons

Scale	Category
0.70427783778473274	Farming-fishing
0.4606771836595015	Craft-repair
0.37671990913389169	Transport-moving
0.27181223010240918	Handlers-cleaners
0.085376067782695741	Machine-op-inspct
0.06192759907115112	Exec-managerial

MIN: -0.77601262322254

MAX: 0.704277837784733

Total: 14

Threshold: 0.105735032929091

Occupation_Group1	Occupation_Group2	Occupation_Group3	Occupation_Group-Others
Craft-repair	Machine-op-inspct	Protective-serv	Farming-fishing
Transport-moving	Exec-managerial	Prof-specialty	Adm-clerical
Handlers-cleaners	Sales	Tech-support	Priv-house-serv
		Other-service	
		Armed-Forces	

Figure 5.5: Groupings achieved for Occupation variable in cluster C11 using the developed prototype

5.5 Generation of Multidimensional Schema

As in the previous case study, we generate a multi-dimensional schema by using the numerical variables as facts and nominal variables as dimensions. The ranked lists of numeric and nominal variables serve as a starting point to constrain the multidimensional space. At this point the appropriate number of important dimensions (nominal variables) and facts (numeric variables) are selected for the creation of the multidimensional schema. As an example, we show the *Age* and *Hrs-per-week* facts for all of the data clusters. Figure 5.6 shows the averages of both these facts across multiple levels of the hierarchy.

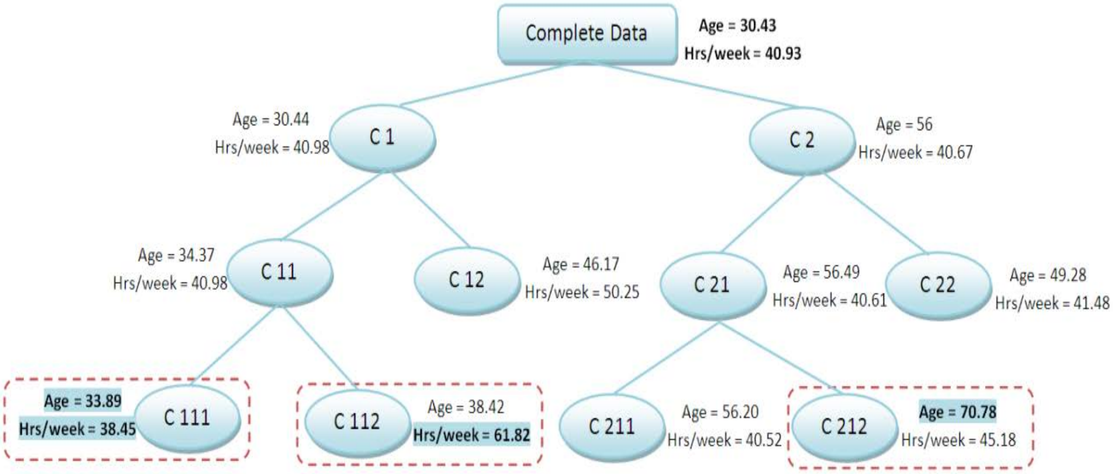
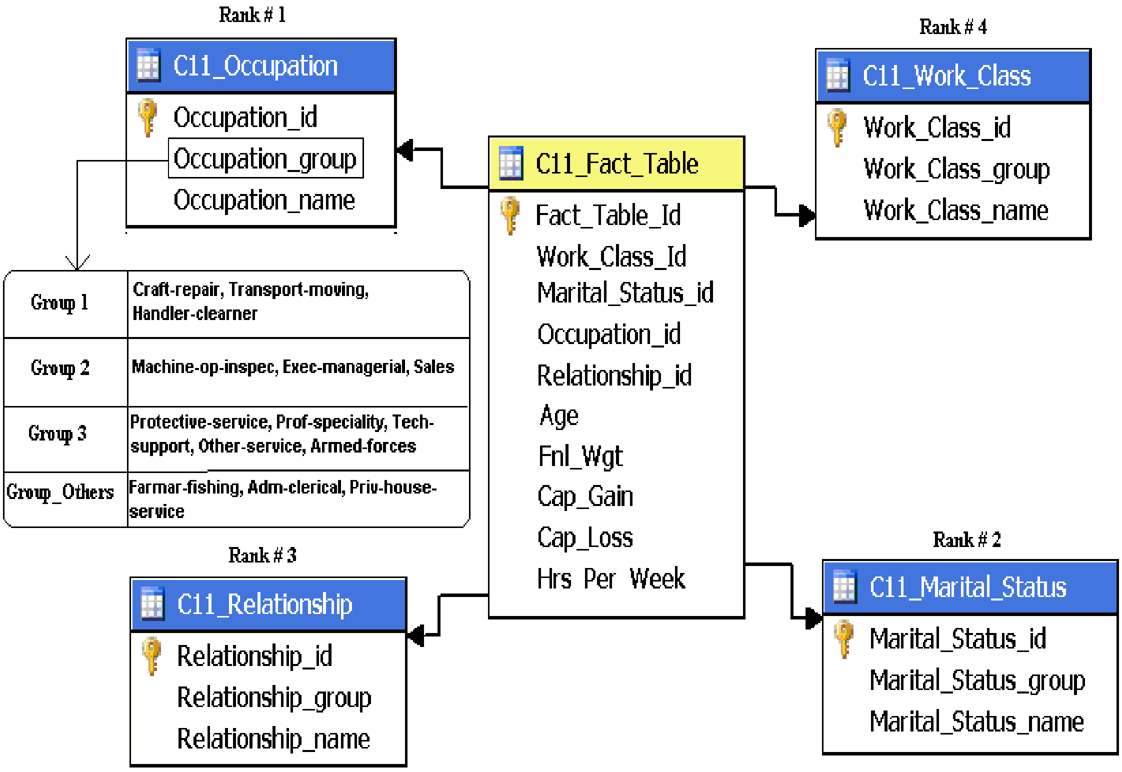


Figure 5.6: Hierarchical tree showing the averages of *Age* and *Hours per week*

We can see from the tree that the average *Age* of people in the complete dataset is 30.43 and they work approximately 40.93 hours per week. However, as we go one level down the hierarchy, we see a sharp difference in the averages as the first cluster C1 has almost the same average, but the second cluster has a much higher average of 56 for *Age*. More interestingly, if we look at cluster C212 we note that the average age here appears to be more than 70. Similarly, cluster C112 has the maximum and C111 has the minimum hours per week value across all data clusters.

By looking at the distinct differences, an analyst would be interested to know more about the effects of other factors such as *Occupation* or *Education* for the people whose age is above 70 but who still work 45 hours or more per week and are present in C212. Another interesting case to be examined is relatively young people working more than 61 hours per week who are also present in C112.

Figure 5.7 shows the schema generated using the top 4 highly ranked dimensions, namely, *Occupation*, *Marital-Status*, *Relationship*, and *Work-Class*.



Group 1	Craft-repair, Transport-moving, Handler-clearner
Group 2	Machine-op-inspec, Exec-managerial, Sales
Group 3	Protective-service, Prof-speciality, Tech-support, Other-service, Armed-forces
Group_Others	Farmar-fishing, Adm-clerical, Priv-house-service

Figure 5.7: Multidimensional schema generated for cluster C11

Moreover, the natural groups obtained in the previous step of our methodology are also depicted for the top ranked *Occupation* dimension. The natural groupings highlight the semantic relationships present in various occupations. For instance, people having occupations like (*craft-repair*, *transport-moving* and *handler cleaner*) tend to have similar behaviour and therefore grouped together in cluster C11. It would be interesting to examine the distribution of fact variables such as *Hrs-per-week* or *Age* associated with these occupation types.

5.6 Informative Data Cubes Construction

In order to explore these ranked dimensions and facts, we construct a 3-dimensional cube using the 3 top ranked dimensions and facts. Each dimension has two levels (group level and value level) in order to navigate in the dimensional hierarchy. Figure 5.8 shows the 3 dimensional cubes structure of two clusters at different levels of the hierarchy.

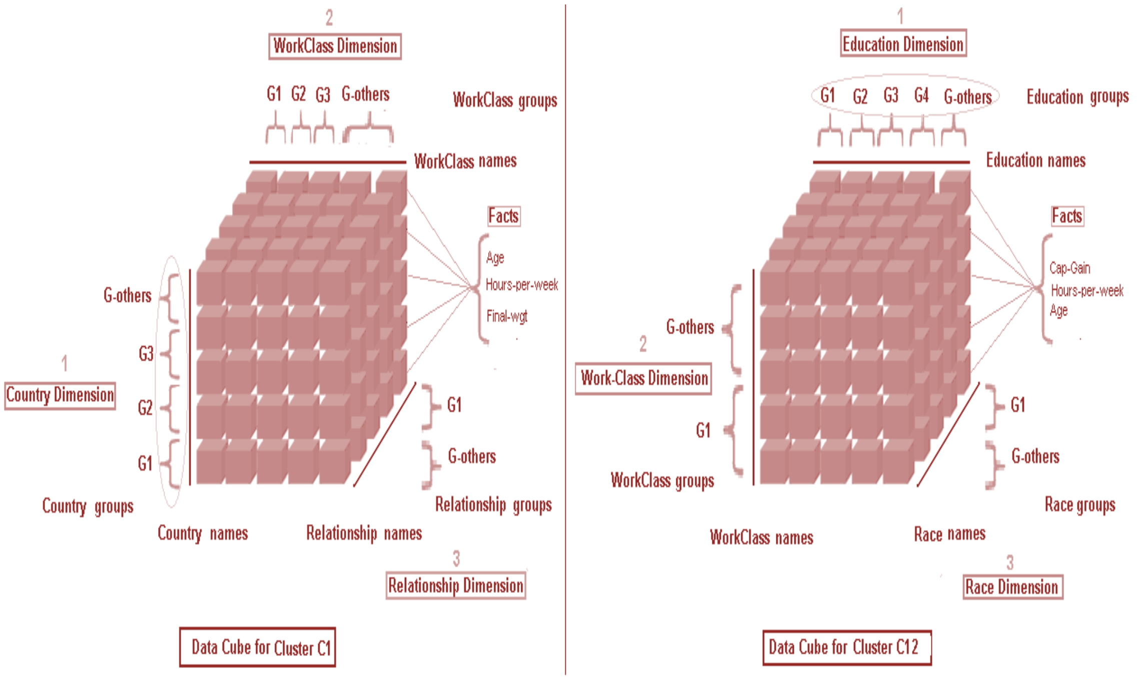
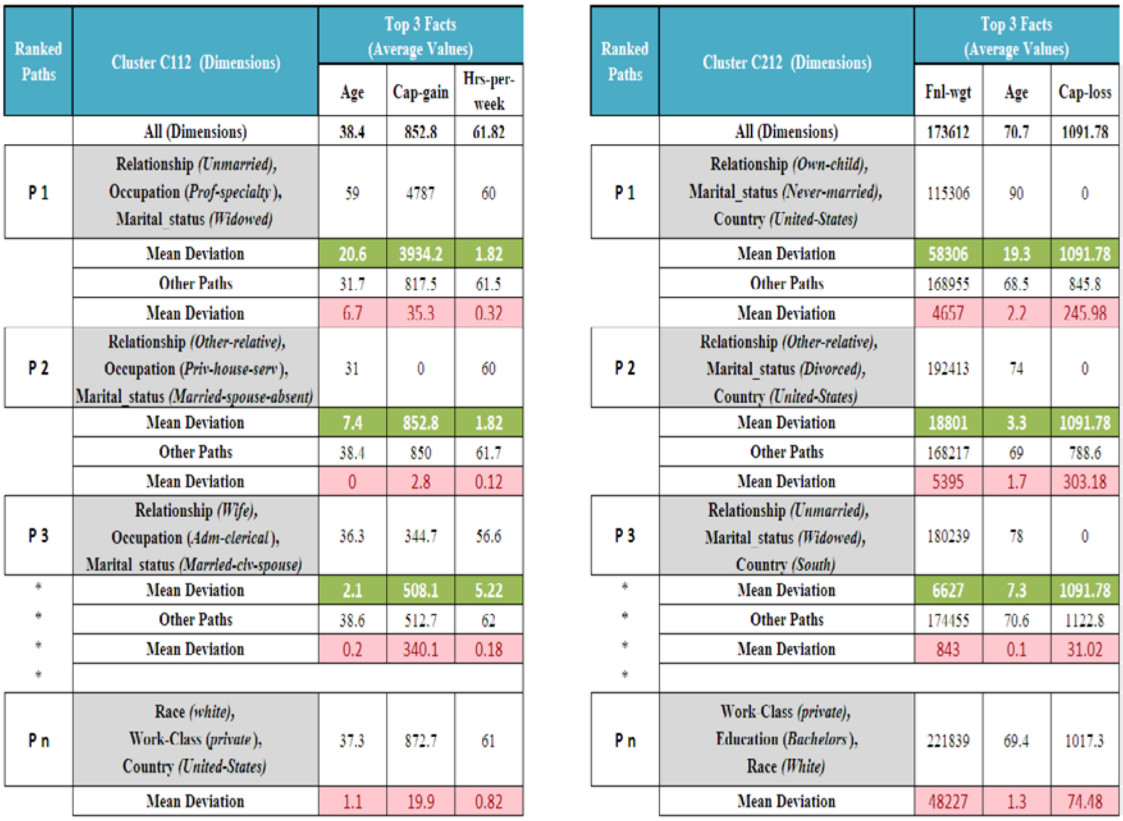


Figure 5.8: Comparison of 3-dimensional cubes at different levels of hierarchy

It can be seen from Figure 5.8 that *Relationship* happens to be the 3rd ranked dimension in cluster C1; however, is totally absent in a lower level cluster C12. Moreover, the *Work-Class* dimension remained at number 2 in rank on both cubes. However, the number of groups within this dimension changed significantly. The *Work-Class* dimension has 4 groups [G1, G2, G3 and G-others] in the cluster C1 cube whereas the same dimension has only 2 groups [G1 and G-others] in the cluster C12 cube which is at a lower level in the cluster hierarchy. Additionally, the values present in each group appear to be entirely different from one another. Such sharp differences give new insights about the underlying data and further relations of these dimensions with the facts can be explored in an OLAP manner.

5.7 Exploration of Interesting Cube Regions

Similar to the previous case study, we took the top 3 ranked paths and compared them with the lowly ranked path suggested by our methodology for each data cluster. Figure 5.9 shows the results of the cube exploration through highly ranked paths for cluster C112 and C212. It is clear from the results in Figure 5.9 that the highly ranked paths in each cluster have a tendency to reveal interesting information about the facts present in the data. For example, we analyze cluster C212 where there were records of older (age 70 plus) people.



Ranked Paths	Cluster C112 (Dimensions)	Top 3 Facts (Average Values)		
		Age	Cap-gain	Hrs-per-week
	All (Dimensions)	38.4	852.8	61.82
P 1	Relationship *(Unmarried)*, Occupation *(Prof-specialty)*, Marital_status *(Widowed)*	59	4787	60
	Mean Deviation	20.6	3934.2	1.82
	Other Paths	31.7	817.5	61.5
	Mean Deviation	6.7	35.3	0.32
P 2	Relationship *(Other-relative)*, Occupation *(Priv-house-serv)*, Marital_status *(Married-spouse-absent)*	31	0	60
	Mean Deviation	7.4	852.8	1.82
	Other Paths	38.4	850	61.7
	Mean Deviation	0	2.8	0.12
P 3	Relationship *(Wife)*, Occupation *(Adm-clerical)*, Marital_status *(Married-civ-spouse)*	36.3	344.7	56.6
*	Mean Deviation	2.1	508.1	5.22
*	Other Paths	38.6	512.7	62
*	Mean Deviation	0.2	340.1	0.18
*				
P n	Race *(white)*, Work-Class *(private)*, Country *(United-States)*	37.3	872.7	61
	Mean Deviation	1.1	19.9	0.82

Ranked Paths	Cluster C212 (Dimensions)	Top 3 Facts (Average Values)		
		Fnl-wgt	Age	Cap-loss
	All (Dimensions)	173612	70.7	1091.78
P 1	Relationship *(Own-child)*, Marital_status *(Never-married)*, Country *(United-States)*	115306	90	0
	Mean Deviation	58306	19.3	1091.78
	Other Paths	168955	68.5	845.8
	Mean Deviation	4657	2.2	245.98
P 2	Relationship *(Other-relative)*, Marital_status *(Divorced)*, Country *(United-States)*	192413	74	0
	Mean Deviation	18801	3.3	1091.78
	Other Paths	168217	69	788.6
	Mean Deviation	5395	1.7	303.18
P 3	Relationship *(Unmarried)*, Marital_status *(Widowed)*, Country *(South)*	180239	78	0
*	Mean Deviation	6627	7.3	1091.78
*	Other Paths	174455	70.6	1122.8
*	Mean Deviation	843	0.1	31.02
*				
P n	Work Class *(private)*, Education *(Bachelors)*, Race *(White)*	221839	69.4	1017.3
	Mean Deviation	48227	1.3	74.48

Figure 5.9: Results of cube exploration through ranked paths (Adult dataset)

According to our methodology the three highest ranked dimensions are *Relationship*, *Marital_status* and *Country*, as they capture the greatest amount of variation in the cluster and hence it would be useful to examine how this sub group of older adults are distributed across different combinations of values taken across these dimensions. However, even for these three highly ranked dimensions there are 9 countries, 6 relationship types and 5 marital status types to choose from, which means that even after constraining the cube to only the highly ranked dimensions a large navigation space still remains to be explored manually.

To resolve this problem, our methodology suggests ranked paths to define data cubes containing distinctive and interesting information. For instance, ranked path P1 highlights the data cell in the cube that has the highest deviation on *Age*. We note that the average age of people whose *Relationship* status is (unmarried) and *Occupation* is (Prof-specialty) has an extreme deviation from the overall mean value of 38.4 registered for all individuals contained in the cluster. Similarly, people having (Priv-house-service) as *Occupation*, and (Married-civ-spouse) as their marital status display the second highest deviation from the mean. Using ranked paths, users can easily determine which particular dimensions chosen from a large number of possibilities exhibit extreme deviation from the average. To take another example, consider the cube defined over cluster C212; even though this cube is defined over just the 3 most highly ranked dimensions, the search space to be explored is still too large for knowledge discovery purposes. This is due to the fact that the *Relationship* dimension has 6 members, the *Marital-Status* dimension has 5 and the *Country* dimension has 10, making up a total of (6 x 5 x 10) = 300 data cells within this cube for exploration. Yet, this is a cluster at a lower level of the data abstraction hierarchy. At higher abstraction levels the situation is even worse; there are 41 countries, 16 education types and 14 occupations, making up a total of 9184 cells to be explored. In order to uncover interesting knowledge in a timely manner, users will need to navigate using the most discriminating countries, education types and occupations suggested by our methodology.

In certain special cases where the distribution of a measure is bimodal, the deviation from the mean may be less interesting to the user via highly ranked paths. However, in such cases the highly ranked paths could either be easily ignored or used for comparative analysis between the two peaks of the data population. For example, in a Census dataset, Hrs-per-week variable may have a bimodal distribution. Suppose this variable shows two extremes (70 and 60 hrs-per-week) respectively in top ranked paths (P1 and P2) for a particular dimension say, Occupation. It would be interesting to identify and compare occupations of people who are working an extreme number of hours per week. However, if one of the extreme paths appears to be obvious to the analyst then that path could be filtered out and paths at a lower level should be explored for discovering interesting knowledge.

5.8 Mining Association Rules from Multidimensional Schema

As done in the previous case study, we applied rule mining on both original cluster data and multidimensional schema for clusters C1, C11 and C12. Similar to the previous case study, we picked the three most impure dimensions (having high entropy values) to generate association rules. Table 5.2 shows the top 10 rules of cluster C12 generated without multidimensional schema at a minimum probability value of 0.4 and minimum importance value of 0.10. Based on these thresholds the algorithm produced 53 rules in total.

Table 5.2: Rules generated without multidimensional schema

RULES WITHOUT SCHEMA (CLUSTER C12)		
No	**Rules**	**Imp**
R1	Education = Bachelors, Occupation = Adm-clerical → Work Class = Local-gov	1.343
R2	Education = [All] ,Occupation = Handlers-cleaners → Work Class = Local-gov	1.298
R3	Education = 10th, Occupation = [All] → Work Class = Local-gov	1.170
R4	Education = HS-grad, Occupation = Adm-clerical → Work Class = Self-emp-not-inc	0.535
R5	Education = Some-college, Occupation = Sales → Work Class = Self-emp-not-inc	0.487
R6	Education = HS-grad, Occupation = Prof-specialty → Work Class = Self-emp-not-inc	0.407
R7	Education = Bachelors, Occupation = Sales →Work Class = Self-emp-inc	0.368
R8	Education = HS-grad, Occupation = Exec-managerial →Work Class = Self-emp-not-inc	0.346
R9	Education = HS-grad, Occupation = Sales →Work Class = Self-emp-inc	0.275
R10	Education = Some-college, Occupation = Exec-managerial →Work Class = Private	0.190

With the same threshold values of support and confidence, we generated the rules from the multidimensional schema of the same cluster, C12. The first 10 rules satisfied the given thresholds and are shown in Table 5.3.

Table 5.3: Rules generated with multidimensional schema

RULES WITH SCHEMA (CLUSTER C12)		
No	**Rules**	**Imp**
R1	Education Group = group1, Occupation Group = group1 → Work Class Name = Local-gov	1.343
R2	Education Group = [All], Occupation Group = group_others → Work Class Name = Local-gov	1.298
R3	Education Group = group_others , Occupation Group = [All] → Work Class Name = Local-gov	1.170
R4	Education Group = group_others , Occupation Group = group1 → Work Class Name = Self-emp-not-inc	0.535
R5	Education Group = group1, Occupation Group = group1 → Work Class Name = Self-emp-not-inc	0.487
R6	Education Group = group1, Occupation Group = group1 → Work Class Name = Self-emp-inc	0.368
R7	Education Group = group_others, Occupation Group = group1 → Work Class Name = Self-emp-inc	0.275
R8	Education Group = group1, Occupation Group = group1 → Work Class Name = Private	0.190
R9	Education Group = [All], Occupation Group = group1 → Work Class Name = Private	0.159
R10	Education Group = group_others , Occupation Group = group_others → Work Class Name = Private	0.154

We observe that the rules generated with the use of the multidimensional schema follow the same trends as in the previous case study. The rules generated through the use of the multidimensional schema are more informative. For example, Rule (R1) without schema predicts Work-Class = [Local-gov] if Education = [Bachelors] and Occupation = [Adm-clerical]. On the other hand, Rule (R1) with schema predicts the same Work-Class value with a diverse set of Education and Occupation values present in [group1] of each dimension.

For instance, [group1] of Education consists of (Doctorate, Masters, Bachelors, Prof-school, Some-college, Assoc-voc) and [group1] of Occupation consists of (Prof-specialty, Exec-managerial, Tech-support, Sales, Adm-clerical, Protective-serv, Other-service, Transport-moving, Machine-op-inspct). These groups provide rich and diverse information to the user while retaining the same importance score of 1.343. If we focus on the Education dimension, we see that rule R1 without schema only suggests Bachelor as the educational level for people who work in Local-government, whereas R1 with the schema suggests a set of values in which Masters and Doctorate qualifications are also present.

5.9 Evaluation of Rule Interestingness via Diversity Criterion

In order to validate the claim that rules generated from schema are more diverse, we performed evaluation of the rules with the *Rae*, *CON* and *Hill* diversity measures, as with the previous Case Study. We took a set of top rules generated without schema and compared it with the same set of rules generated with the use of the schema. Table 5.4 shows the results of this evaluation.

Table 5.4: Rule interestingness comparison using diversity measures

Cluster Names	Rule sets	NO Schema Rae	With Schema Rae	NO Schema CON	With Schema CON	NO Schema Hill	With Schema Hill
C1	**R1-R6**	0.322	**0.337**	0.432	**0.452**	-1.693	**-1.636**
	R1-R7	0.259	**0.295**	0.369	**0.421**	-2.302	**-1.948**
	R1-R8	0.218	**0.277**	0.327	**0.418**	-2.873	**-2.095**
	R1-R9	0.204	**0.270**	0.325	**0.423**	-3.084	**-2.159**
	R1-R10	0.192	**0.264**	0.321	**0.427**	-3.285	**-2.216**
C11	**R1-R6**	0.322	**0.397**	0.432	**0.526**	-1.693	**-1.286**
	R1-R7	0.256	**0.376**	0.365	**0.522**	-2.346	**-1.385**
	R1-R8	0.216	**0.312**	0.324	**0.463**	-2.918	**-1.805**
	R1-R9	0.204	**0.309**	0.325	**0.472**	-3.084	**-1.829**
	R1-R10	0.192	**0.303**	0.321	**0.475**	-3.285	**-1.872**
C12	**R1-R6**	0.164	**0.263**	0.263	**0.371**	-3.017	**-2.248**
	R1-R7	0.217	**0.312**	0.353	**0.454**	-2.437	**-1.703**
	R1-R8	0.172	**0.297**	0.294	**0.454**	-3.339	**-1.800**
	R1-R9	0.142	**0.279**	0.252	**0.441**	-4.261	**-2.110**
	R1-R10	0.136	**0.276**	0.448	**0.443**	-4.755	**-2.141**

It is apparent from Table 5.4 that all the objective measures of diversity show significant improvement for the set of rules generated from the multidimensional schema. Similar to the *Automobile* case study results, the rules generated from the multidimensional schema are more diverse and capable of conveying more interesting knowledge to the user. Furthermore, the prediction accuracy of the rules generated from the schema also appears to be higher when compared to the rules without schema. To validate this claim, we tested the prediction accuracy of the rules generated without schema against the rules generated with schema. We used 30% of the test data from each cluster and ran 10 tests where each test randomly picked unique test data to assess the predictive accuracy of association rules generated. Table 5.5 shows the prediction accuracy for each test, expressed as a percentage.

It is clear from Table 5.5 that the prediction accuracy of the rules generated through the use of the multidimensional schema is higher when compared to the one without schema. Again, we note from this case study on a relatively large dataset, that rule mining performed on the multidimensional schema designed and constructed with the help of hierarchical clustering and multidimensional scaling technique generates diverse rules with greater prediction accuracy.

Table 5.5: Rule prediction accuracies for two sets of rules

Prediction Tests	Cluster C1		Cluster C11		Cluster C12	
	Schema	**No Schema**	**Schema**	**No Schema**	**Schema**	**No Schema**
Test 1	39.7	40.1	40.1	40.1	50	52.2
Test 2	40	40.3	40.3	40	47.7	47.7
Test 3	40.2	39.7	39.7	39.1	56.8	40.9
Test 4	39.5	40	40	39.3	47.7	38.6
Test 5	41.5	39.5	39.5	40.8	43.1	50
Test 6	40	39.7	40.3	39	55.3	40.9
Test 7	41.5	40.1	39.7	39.7	56.8	50.2
Test 8	39.7	39.5	40.1	40	50	39.3
Test 9	39.5	40	40	39.3	47.3	40.9
Test 10	40.2	39.5	41.3	40.8	47.7	47.7
Average Percentage	**40.18**	39.84	**40.01**	39.81	**50.24**	44.84

Summary

In this chapter, we presented our second case study conducted on a much larger dataset as compared to Automobile dataset. It is the Adult (Kohavi and Becker 1996) dataset which consists of 48,842 records with eight nominal and five numeric variables as shown in Table 5.1. This benchmark dataset was extracted from the US Census bureau website using a data extraction system.

We ranked the numeric variables present in each cluster and observed that the ranking is unique to each cluster. Similar to the previous case study, sharp differences in rankings for a given variable appear at different levels in cluster hierarchy. Similar to numeric variables, the list of ranked nominal variables also show sharp differences as we move from a higher level of data abstraction to a lower level. It was observer that *Country* variable which was top-ranked in cluster C1 was lowest ranked in cluster C11. Similarly, *Relationship* variable which was the third most significant variable in C1 took last place in cluster C12. In other words, *Country* and *Relationship* had the least randomness in parent cluster C1 but at an immediate lower level in the hierarchy these variables appear to be having the most randomness or impurity. We suggested that these highly ranked variables should be explored as a first choice in a given cluster as they possess more versatile information that defines the split as compared to the lowly ranked variables which play a minimum role in the cluster split.

Similar to the previous case study, we took the top 3 ranked paths and compared them with the lowly ranked path suggested by our methodology for each data cluster. We observed that the highly ranked paths in each cluster have a tendency to reveal interesting information about the facts present in the data. For example, we analyzed cluster C212 where there were records of older (age 70 plus) people. The ranked path P1 highlighted the data cell in the cube that had the highest deviation on *Age.* We noted that the average age of people whose *Relationship* status is (unmarried) and *Occupation* is (Prof-specialty) has an extreme deviation from the overall mean value of 38.4 registered for all individuals contained in the cluster. Using ranked paths, users can easily determine which particular dimensions chosen from a large number of possibilities exhibit extreme deviation from the average.

As done in the previous case study, we applied rule mining on both original cluster data and multidimensional schema and observed that the rules generated with the use of the multidimensional schema follow the same trends as in the previous case study. The rules generated through the use of the multidimensional schema are more informative. For example, Rule (R1) without schema predicts Work-Class = [Local-gov] if Education = [Bachelors] and Occupation = [Adm-clerical]. On the other hand, Rule (R1) with schema predicts the same Work-Class value with a diverse set of Education and Occupation values present in [group1] of each dimension. For instance, [group1] of Education consists of (Doctorate, Masters, Bachelors, Prof-school, Some-college, Assoc-voc) and [group1] of Occupation consists of (Prof-specialty, Exec-managerial, Tech-support, Sales, Adm-clerical, Protective-serv, Other-service, Transport-moving, Machine-op-inspct). The evaluation results also confirmed that all the objective measures of diversity show significant improvement for the set of rules generated from the multidimensional schema. Similar to the *Automobile* case study results, the rules generated from the multidimensional schema are more diverse and capable of conveying more interesting knowledge to the user. Furthermore, the prediction accuracy of the rules generated from the schema also appeared to be higher when compared to the rules without schema.

Chapter 6

Case Study 3: CoverType Dataset

In this chapter, we present our third case study conducted on a much larger dataset called *CoverType* (Blackard, Dean et al. 1998). This is currently one of the largest datasets in the UCI repository containing 581012 records with 54 variables (42 nominal and 12 numeric) and 7 target classes (Obradovic and Vucetic 2004). This benchmark dataset is used to predict forest cover types from cartographic variables. Forest cover type is basically defined as a descriptive classification of forest land based on occupancy of an area by the tree species present in it. It is a typical real world dataset having an imbalanced class distribution for the cover type variable as shown in Table 6.1.

Table 6.1: Distribution of forest cover types present in *CoverType* dataset

Cover Types (Class name)	*No of records*
Spruce-Fir	211840
Lodgepole Pine	283301
Ponderosa Pine	35754
Cottonwood/Willow	2747
Aspen	9493
Douglas-fir	17367
Krummholz	20510
TOTAL	**581012**

The actual forest cover type for a given observation (30 x 30 meter cell) was determined from US Forest Service (USFS) Region 2 Resource Information System (RIS) database. Independent variables were derived from data originally obtained from the US Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables (wilderness areas and soil types). This study area includes four wilderness areas located in the Roosevelt National Forest of Northern Colorado. More detailed description and background information for this dataset can be found at University of California – machine learning website (Asuncion and Newman 2010).

The main motivation behind choosing this dataset is that it poses extreme challenges to the analysts in finding useful and interesting knowledge from the rich mix of nominal and numeric variables and sheer size of data. To give a glimpse of the difficulties involved in mining association rules from this complex dataset, we present some interesting facts for this dataset discovered by (Webb 2006) which motivated us to mine diverse association rules from large datasets. (Webb 2006) reported the results of this particular dataset and observed that not a single non-redundant rule generated through CoverType dataset was found to be productive. He further explained that this was due to a peculiarity of this dataset which uses 40 mutually exclusive binary variables (SoilType 1 to SoilType 40) to represent which one of 40 soil types predominates in an area. Thus, the most frequent attribute values are values of 0 for individual soil type variables and the most frequent itemsets are sets of these values. Because they are mutually exclusive, for any two of these variables, all associations between these variables must be unproductive. The identified fact that all non-redundant associations for this dataset represented unproductive associations highlighted the dangers of data mining without sound methodologies for discovering meaningful and diverse association rules.

6.1 Application of Agglomerative Hierarchical Clustering

Unlike with the two previous case studies the dataset was sampled prior to the application of hierarchical clustering. Stratified random sampling was used to obtain an unbiased sample in view of the unbalanced nature of the dataset. In this process we divided the records into homogeneous subgroups, defined by the forest cover type variable prior to sampling, thus improving the representativeness of each class in the sample. We used a sample size of 45,000 records to generate clusters at multiple levels of data abstraction. Similar to the previous case studies on smaller datasets, we calculated the inconsistency coefficient value which was 0.519 for determining the cut-off point and generated the binary tree of clusters using the procedure explained in Algorithm 1. There were a total of 8 levels of data abstraction till the cut-off point and the last level had 16 clusters in it.

Although we adopted a stratified sampling method to produce a sample that retains the diversity of classes, the sample size was rather small at approximately 8 % of the total volume of records. As a consequence, overall information loss could be high enough to prevent the accurate depiction of all trends and patterns which exist in the original dataset. In order to accelerate the creation of the dendogram, we used the sampled dataset, instead of the original dataset. Once the binary tree was created, we distributed the original (non-sampled) data by allocating each record to the cluster whose centroid was closest the current record in Euclidean terms. The population of the dendogram with the entire dataset was done in order to alleviate the problem of small volume preventing the identification of certain patterns. Figure 6.1 depicts the first four levels of binary tree after allocation of records for the *CoverType* dataset.

Likewise, as in the previous case studies, we plotted the numeric variables present in each cluster to fix the number of principal components to be extracted. Most clusters were discriminating well on 1 component as depicted in Figure 6.2. The application procedure remains the same as in the previous two case studies. Figure 6.2 shows the scree plots and it can be seen that the data in the clusters are discriminated well on 1 component. The scree plots for the other clusters followed the same trend and were omitted to conserve space.

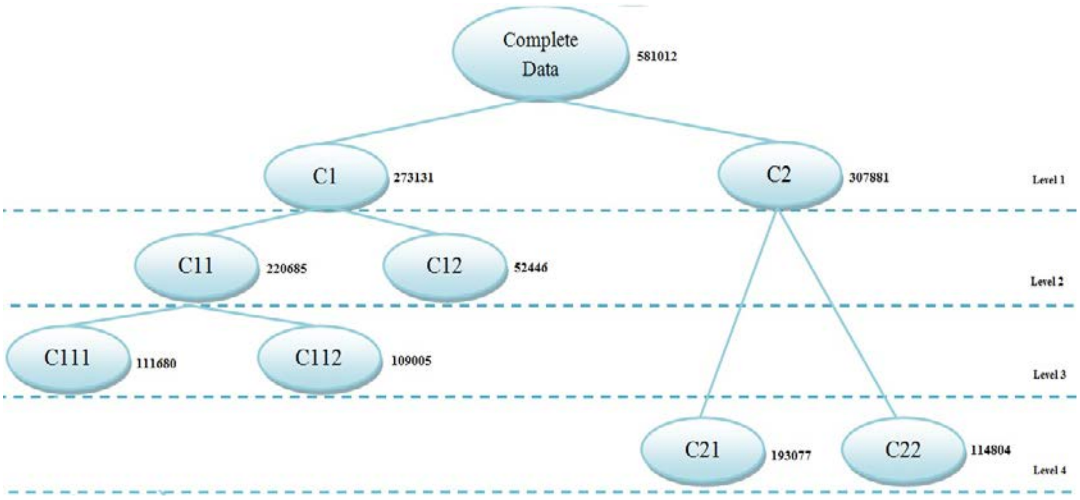


Figure 6.1: Binary tree for CoverType dataset

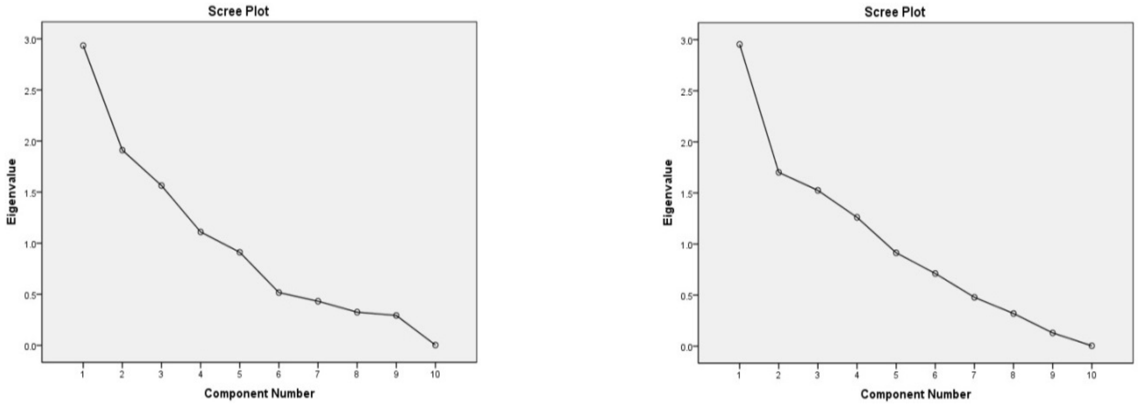
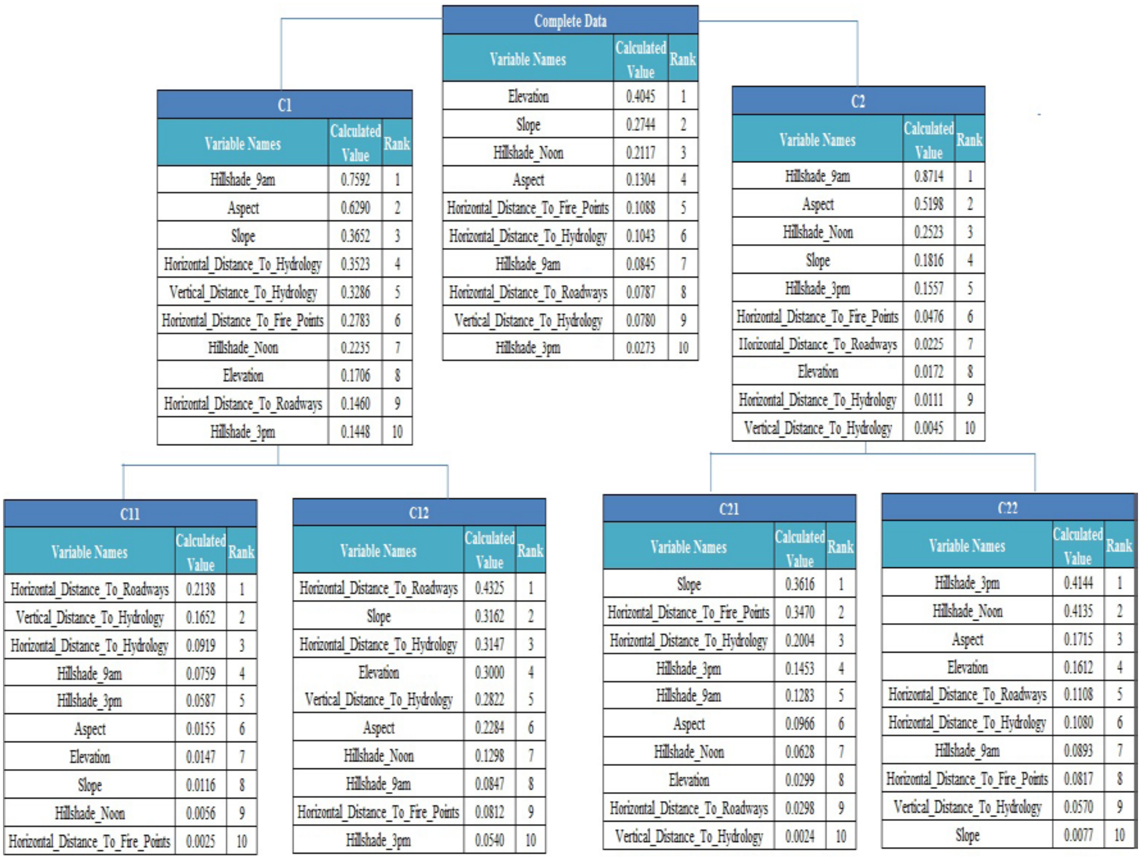


Figure 6.2: Scree plot showing Eigen values of cluster C2 (left) and C21 (right)

6.2 Ranking of Numeric Variables via PCA

We rank the numeric variables present in each cluster and observe that the ranking is unique to each cluster. Similar to the previous case studies, sharp differences in rankings for a given variable appear at different levels in the cluster hierarchy. The rankings computed for the first few levels of the hierarchy are shown in Figure 6.3.

Figure 6.3 shows the ranking of numeric variables in the cluster hierarchy after performing PCA and comparative analysis as explained in section 3.2 of Chapter 3. Unlike in the previous two case studies, we observe from Figure 6.3 that the clusters at the first level of the hierarchy, namely C1 and C2 have the same top 2 highly ranked variables. This is due to the component loadings of *Hillshade_9am* and *Aspect* variables in the four clusters, namely (C11, C12, C21 and C22), responsible for the ranking the two clusters, C1 and C2, at the first level of the hierarchy. These four clusters showed maximum difference in component loading values for *Hillshade_9am* and *Aspect* variables as compared to all other variables involved in the analysis.



Complete Data

Variable Names	Calculated Value	Rank
Elevation	0.4045	1
Slope	0.2744	2
Hillshade_Noon	0.2117	3
Aspect	0.1304	4
Horizontal_Distance_To_Fire_Points	0.1088	5
Horizontal_Distance_To_Hydrology	0.1043	6
Hillshade_9am	0.0845	7
Horizontal_Distance_To_Roadways	0.0787	8
Vertical_Distance_To_Hydrology	0.0780	9
Hillshade_3pm	0.0273	10

C1

Variable Names	Calculated Value	Rank
Hillshade_9am	0.7592	1
Aspect	0.6290	2
Slope	0.3652	3
Horizontal_Distance_To_Hydrology	0.3523	4
Vertical_Distance_To_Hydrology	0.3286	5
Horizontal_Distance_To_Fire_Points	0.2783	6
Hillshade_Noon	0.2235	7
Elevation	0.1706	8
Horizontal_Distance_To_Roadways	0.1460	9
Hillshade_3pm	0.1448	10

C2

Variable Names	Calculated Value	Rank
Hillshade_9am	0.8714	1
Aspect	0.5198	2
Hillshade_Noon	0.2523	3
Slope	0.1816	4
Hillshade_3pm	0.1557	5
Horizontal_Distance_To_Fire_Points	0.0476	6
Ilorizontal_Distance_To_Roadways	0.0225	7
Elevation	0.0172	8
Horizontal_Distance_To_Hydrology	0.0111	9
Vertical_Distance_To_Hydrology	0.0045	10

C11

Variable Names	Calculated Value	Rank
Horizontal_Distance_To_Roadways	0.2138	1
Vertical_Distance_To_Hydrology	0.1652	2
Horizontal_Distance_To_Hydrology	0.0919	3
Hillshade_9am	0.0759	4
Hillshade_3pm	0.0587	5
Aspect	0.0155	6
Elevation	0.0147	7
Slope	0.0116	8
Hillshade_Noon	0.0056	9
Horizontal_Distance_To_Fire_Points	0.0025	10

C12

Variable Names	Calculated Value	Rank
Horizontal_Distance_To_Roadways	0.4325	1
Slope	0.3162	2
Horizontal_Distance_To_Hydrology	0.3147	3
Elevation	0.3000	4
Vertical_Distance_To_Hydrology	0.2822	5
Aspect	0.2284	6
Hillshade_Noon	0.1298	7
Hillshade_9am	0.0847	8
Horizontal_Distance_To_Fire_Points	0.0812	9
Hillshade_3pm	0.0540	10

C21

Variable Names	Calculated Value	Rank
Slope	0.3616	1
Horizontal_Distance_To_Fire_Points	0.3470	2
Horizontal_Distance_To_Hydrology	0.2004	3
Hillshade_3pm	0.1453	4
Hillshade_9am	0.1283	5
Aspect	0.0966	6
Hillshade_Noon	0.0628	7
Elevation	0.0299	8
Horizontal_Distance_To_Roadways	0.0298	9
Vertical_Distance_To_Hydrology	0.0024	10

C22

Variable Names	Calculated Value	Rank
Hillshade_3pm	0.4144	1
Hillshade_Noon	0.4135	2
Aspect	0.1715	3
Elevation	0.1612	4
Horizontal_Distance_To_Roadways	0.1108	5
Horizontal_Distance_To_Hydrology	0.1080	6
Hillshade_9am	0.0893	7
Horizontal_Distance_To_Fire_Points	0.0817	8
Vertical_Distance_To_Hydrology	0.0570	9
Slope	0.0077	10

Figure 6.3: Ranked lists of numeric variables present in CoverType dataset clusters

For instance, the component loadings of *Hillshade_9am* in C11 and C12 are 0.193 and 0.952 respectively which makes the difference equal to 0.759, which is the highest as compared to all the other variables and consequently ranks *Hillshade_9am* at the top in cluster C1. Similarly the *Aspect* variable is ranked 2nd in cluster C2 because the two subsequent clusters, namely C21 and C22, responsible for determining this rank have loading values of 0.605 and 0.085, correspondingly making the difference in loadings equals to 0.519 which is the 2nd highest difference after the highest difference of 0.871 for the *Hillshade_9am* variable. Interestingly, this case study shows that in certain datasets such as Forest Cover, clear delineations between clusters only start to occur at levels 2 and below in the dendogram structure. Despite this, the level 1 delineation is important as the subtle variations in level 1 are a necessary pre-requisite for uncovering more pronounced deviations further down in the cluster hierarchy.

These component loadings basically highlight the degree of variation captured by individual variables in each cluster. If we examine the averages of these two variables then we find more evidence of the variation that is captured by these two variables. For example, in cluster C11 the average *Aspect* of the forest cover types is 71.6 whereas in cluster C12 the average is 255.6 which highlight the significant variation of records present in the two clusters. Therefore, tree species with less *Aspect* values become a part of C11 and vice versa. Similarly, we observe the same high variation in cluster C21 and C22, where cluster C21 has average *Aspect* of 274.6 as compared to the low average *Aspect* of 72.3 in cluster C22.

Interestingly, the most significant variable *Elevation* in the complete un-clustered dataset is amongst the least significant variables at lower levels in the cluster hierarchy. This implies that forest cover types in the complete dataset are split into two separate clusters, primarily on the basis of the *Elevation* and *Slope* variables and marginally on the basis of other variables such as *Vertical_Distance_To_Hydrology* and *Hillshade_3pm.* Similarly, *Horizontal_Distance_To_Roadways* variable which happens to be an insignificant variable in cluster C1 appears to be the most significant in the child clusters C11 and C12. Another strong pattern can be observed in the rankings of cluster C21 and C22, whereby the *Slope* variable is ranked first in terms of significance in one child cluster C21 and whilst being the least significant in the other child cluster C22 at the same hierarchical level.

6.3 Ranking of Nominal Variables

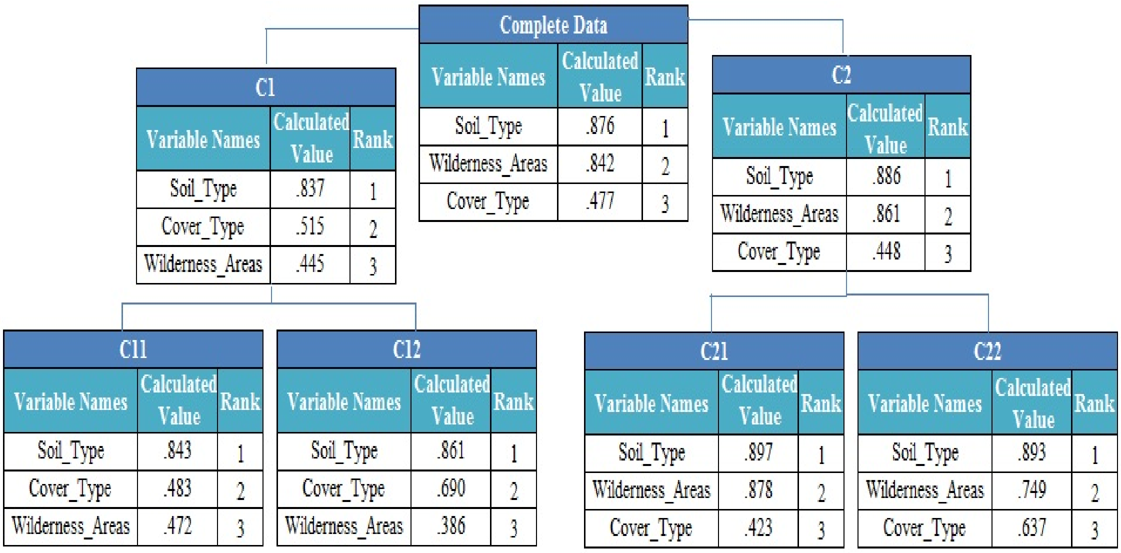
Similar to the previous case studies, we rank nominal variables present in each cluster via two separate data analysis techniques in order to achieve the two main objectives of this research; i) identification of interesting cube regions and ii) discovery of diverse association rules.

6.3.1 Application of Multiple Correspondence Analysis (MCA)

After establishing the significant numeric variables in each cluster, we rank the nominal variables in each cluster in order to find interesting cube regions. The application procedure remains the same and the nominal variables in each cluster are ranked on the basis of Eigen values from highest to lowest. The rankings for the first few levels are shown in Figure 6.4.

The nominal variable ranking varies from cluster to cluster, depending on its position in the hierarchy, except for the Soil_type variable that remains at rank 1 throughout the hierarchy. We note that sibling clusters rank the nominal variables the same. This is due to the fact that the Soil_type variable is a high cardinality variable as compared to the Wilderness_Area and Cover_Type variables. When MCA is applied on the mapped values, the soil type variable captures the highest variation in all of the clusters. The same results are obtained with Entropy based ranking.

However, the information gain measure that drives the Entropy based ranking method is better able to eliminate bias due to high cardinality ranks soil type at the 3rd position in cluster C12, as shown in Figure 6.5. This case study also illustrates relying on only one type of ranking, namely MCA may not be adequate for all kinds of datasets, especially the ones which have these types of complex and dominating variables.



Complete Data

Variable Names	Calculated Value	Rank
Soil_Type	.876	1
Wilderness_Areas	.842	2
Cover_Type	.477	3

C1

Variable Names	Calculated Value	Rank
Soil_Type	.837	1
Cover_Type	.515	2
Wilderness_Areas	.445	3

C2

Variable Names	Calculated Value	Rank
Soil_Type	.886	1
Wilderness_Areas	.861	2
Cover_Type	.448	3

C11

Variable Names	Calculated Value	Rank
Soil_Type	.843	1
Cover_Type	.483	2
Wilderness_Areas	.472	3

C12

Variable Names	Calculated Value	Rank
Soil_Type	.861	1
Cover_Type	.690	2
Wilderness_Areas	.386	3

C21

Variable Names	Calculated Value	Rank
Soil_Type	.897	1
Wilderness_Areas	.878	2
Cover_Type	.423	3

C22

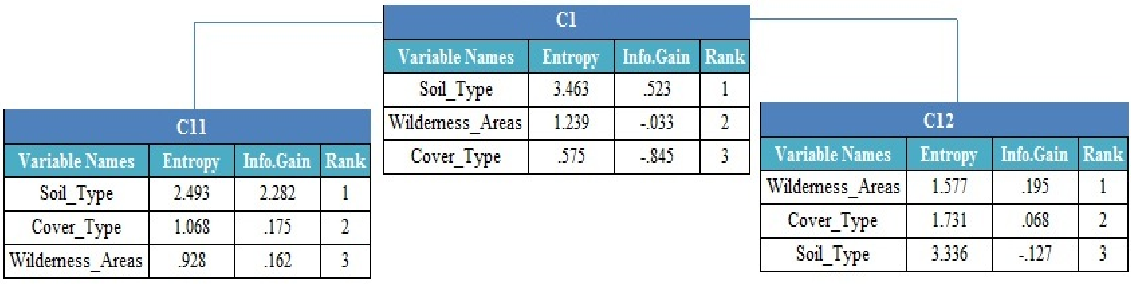
Variable Names	Calculated Value	Rank
Soil_Type	.893	1
Wilderness_Areas	.749	2
Cover_Type	.637	3

Figure 6.4: Ranked list of nominal variables present in Adult dataset

It is important to clarify at this stage that we grouped the 40 binary valued soil type variables present in the original dataset into a single variable named *Soil_Type* with 40 different soil type values such as Soil_Type1, Soil_Type2 and so on. This transformation benefits us in three major ways. Firstly, we are able to identify semantic relationships among different soil types in order to group them using multidimensional scaling and grouping techniques introduced in Chapter 3. Secondly, it eases the exploration of data cubes by taking soil type as a single dimension with 40 distinct categories (values). Thirdly, it allows us to mine and discover underlying associations between multiple soil type values and other nominal variables such as *Wilderness_Area* and *Cover_Type.*

6.3.2 Analysis of nominal variables via Information Gain

In order to discover diverse association rules, we rank the nominal variables in each cluster based on the information gain measure. Figure 7.5 shows the information gain based ranking for three clusters, at consecutive levels in the hierarchy, namely clusters C1, C11 and C12.



C1

Variable Names	Entropy	Info.Gain	Rank
Soil_Type	3.463	.523	1
Wilderness_Areas	1.239	-.033	2
Cover_Type	.575	-.845	3

C11

Variable Names	Entropy	Info.Gain	Rank
Soil_Type	2.493	2.282	1
Cover_Type	1.068	.175	2
Wilderness_Areas	.928	.162	3

C12

Variable Names	Entropy	Info.Gain	Rank
Wilderness_Areas	1.577	.195	1
Cover_Type	1.731	.068	2
Soil_Type	3.336	-.127	3

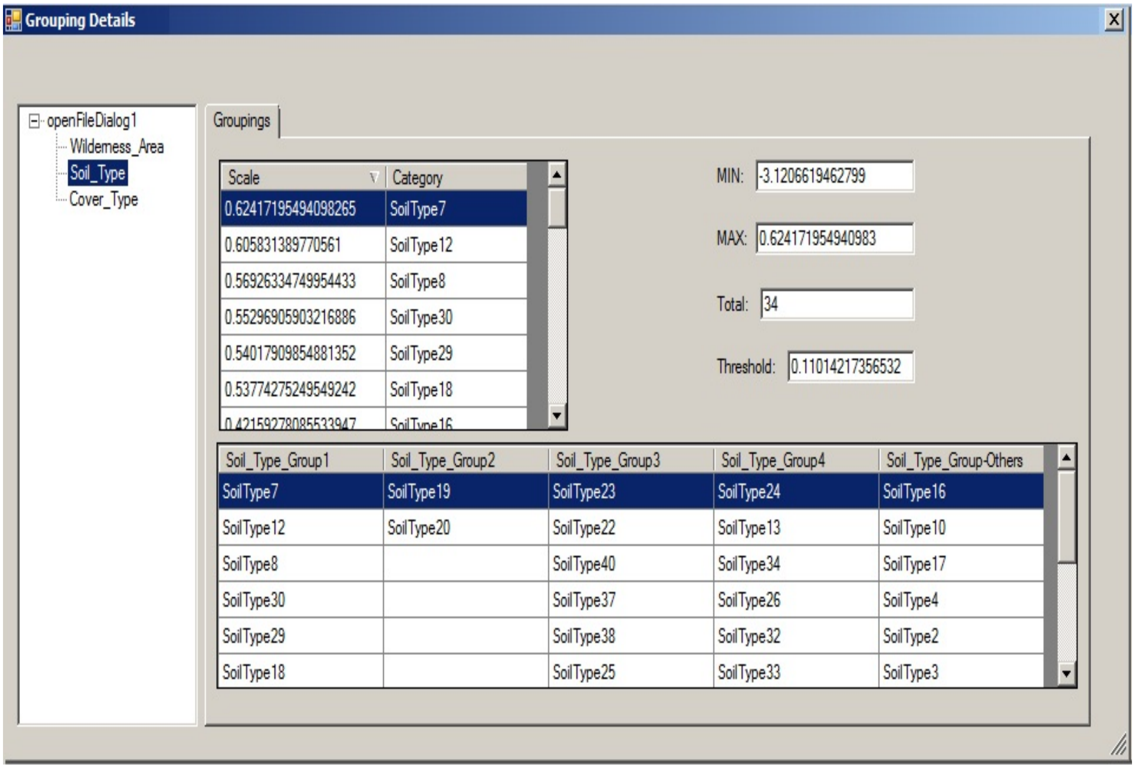
Figure 6.5: Ranking of nominal variables based on Information Gain measure

Similar to numeric variables, the list of ranked nominal variables also show sharp differences as we move from a higher level of data abstraction to a lower level. It can be seen from Figure 6.5 that *Soil_Type* which is top-ranked in cluster C1 is lowest ranked in cluster C12. Similarly, *Wilderness_Area* which is the second most significant variable in C1 took last place in cluster C11. In other words, *Soil_Type* and *Wilderness_Area* had high randomness in parent cluster C1 but at an immediately lower level in the hierarchy these variables appear to be having low randomness or impurity.

Similar to our previous case studies, we observe the emergence of sharp patterns in both numeric and nominal variables as we go down in the cluster hierarchy. We suggest that these highly ranked variables should be explored as a first choice in a given cluster C1 as they possess more versatile information that defines the split as compared to the lowly ranked variables which play a minimum role in the cluster split.

6.4 Grouping of Nominal Values via Multidimensional Scaling

After ranking the nominal variables, we apply multidimensional scaling and group the values present in each nominal variable using Algorithm 2 presented in Chapter 3. Figure 6.6 shows the groupings obtained for the *Soil_Type* variable in C11. The prototype displays the different groups and the values present for each group in a data grid for a selected variable. It can be seen from Figure 6.6 that the *Soil_Type* variable produced five distinct major groups covering the 34 different soil types present in the cluster C11.



Scale	Category
0.62417195494098265	SoilType7
0.605831389770561	SoilType12
0.56926334749954433	SoilType8
0.55296905903216886	SoilType30
0.54017909854881352	SoilType29
0.53774275249549242	SoilType18
0.42159278085533947	SoilType16

MIN: -3.1206619462799

MAX: 0.624171954940983

Total: 34

Threshold: 0.11014217356532

Soil_Type_Group1	Soil_Type_Group2	Soil_Type_Group3	Soil_Type_Group4	Soil_Type_Group-Others
SoilType7	SoilType19	SoilType23	SoilType24	SoilType16
SoilType12	SoilType20	SoilType22	SoilType13	SoilType10
SoilType8		SoilType40	SoilType34	SoilType17
SoilType30		SoilType37	SoilType26	SoilType4
SoilType29		SoilType38	SoilType32	SoilType2
SoilType18		SoilType25	SoilType33	SoilType3

Figure 6.6: Groupings achieved for *Soil_Type* variable in cluster C11

Furthermore, each group represents soil type values which have semantic relationships with other values present in the same group. For instance, Group 2 consists of soil type

19 and soil type 20 which are grouped together and both of these soil types have the same metadata description in the dataset. These soil types belongs to “Typic Cryaguolis and Typic Cryaquolls”, two families of soil types which have the same depth (0 to 4 inches) and same dominant plant (subalpine fir) associations. Moreover, their climate and geographical zones are also identical. It highlights the fact that groups created through the use of multidimensional scaling technique not only have objective similarities but also have likeness in a real-world setting.

6.5 Generation of Multidimensional Schema

As in the previous case studies, we generate a multi-dimensional schema by using the numerical variables as facts and nominal variables as dimensions. The ranked lists of numeric and nominal variables serve as a starting point for selection of an appropriate number of important dimensions (nominal variables) and facts (numeric variables) for the creation of the multidimensional schema.

As an example, we show the *Elevation* and *Slope* fact variables for all of the data clusters. Figure 6.7 shows the averages for these variables across multiple levels of the hierarchy.

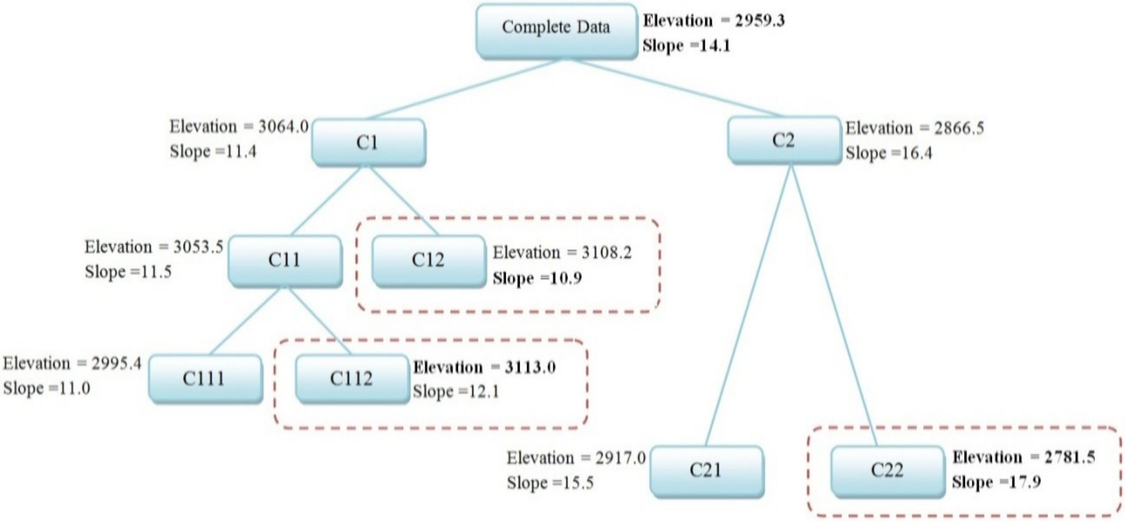


Figure 6.7: Hierarchical tree showing the averages of *Elevation* and *Slope*

We can see from the tree that the average *Elevation* and *Slope* in the complete dataset is 2959.3 and 14.1 respectively. However, as we go one level down the hierarchy, we see a sharp difference in the average emerging as the first cluster C1 covers records which have higher *elevation* and lower *slopes* values while cluster C2 covers the opposite trend of lower *elevation* and higher *slope* values. More interestingly, if we look at cluster C22, we note that it captures forest lands with the lowest average *elevation* and the highest average *slope* values. Similarly, cluster C12 covers the forest land with the lowest *slopes* while cluster C112 covers lands with the highest *elevation*. Thus the clustering covers a spectrum of gradients of forest lands, ranging from low steepness in C112 to steep forest land in C22.

By looking at the distinct differences, an analyst would be interested to know more about the effects of wilderness areas and major tree species in these areas. For instance, it would be interesting to examine the effects of steepness factor by contrasting the predominant tree species grown in cluster C2 with that of cluster C112. Similarly, it would be interesting to investigate which major tree species are associated with which soil types.

Figure 6.8 shows the schema generated for cluster C22 using the three dimensions, namely, *Wilderness Area*, *Soil_Type* and *Cover_Type*.

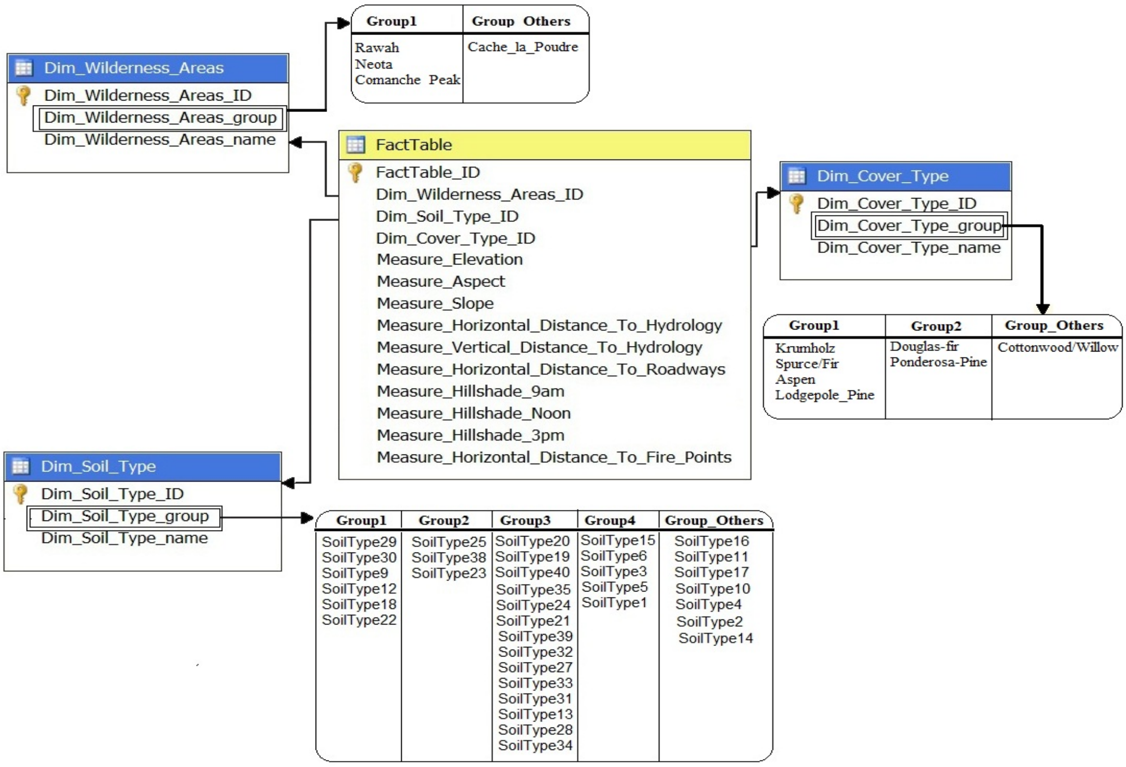


Figure 6.8: Multidimensional schema generated for cluster C22

Moreover, the natural groups obtained in the previous step are also depicted for the top ranked *Soil_Type* dimension. The natural groupings highlight the semantic relationships present in various soil types. For instance, forest land having soil types *(SoilType23, SoilType25 and SoilType38)* tend to have similar behaviour and are therefore grouped together in Group2 of cluster C22.

In the detailed description of the soil types, present on the UCI machine learning website (Asuncion and Newman 2010), it is evident that the above three soil types belong to the *Leighcan family* of soils. According to United States Department of Agriculture (USDA 2012), soils from the *Leighcan family* occur on moraines and consist of residuum and/or till from igneous and metamorphic rock. These soils share common characteristics such as extremely warm climate and stony geographical zones. Additionally, these particular soil types lie in the same climatic zone of *Subalpine*.

In the National Cooperative Soil Survey conducted by US National Resource Conservation Services (NRCS 2012), these zones have (0 to 15%) of slopes, (7000 to 12,000) feet elevation and are derived from gravel deposits of igneous, metamorphic and sedimentary rocks. These similarities among the soil types highlight the fact that groups created through the use of multidimensional scaling technique not only have objective similarities but also have correspondence in a real-life. In this context it would be productive to explore and analyze the distributions of fact variables such as *Elevation* or *Slope* associated with these semantically related groups of soil types, wilderness areas and cover types.

6.6 Informative Data Cube Construction

In order to explore these ranked dimensions and facts, we construct a 3-dimensional cube using the dimensions and top 3 facts. Figure 6.9 shows the 3 dimensional cube structure of two clusters at different levels of the hierarchy.

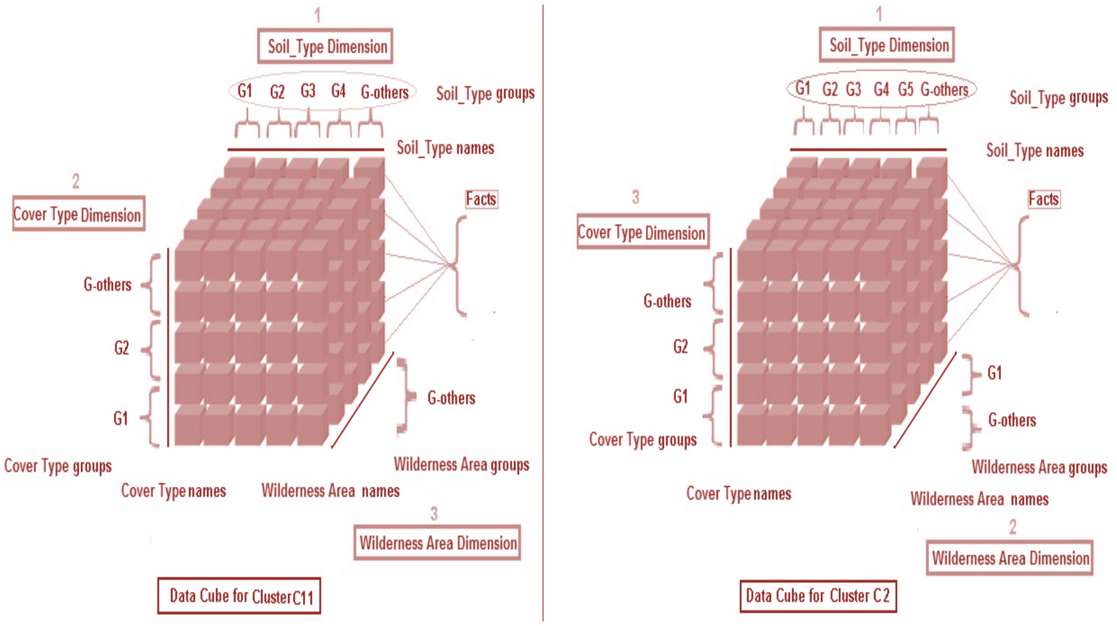
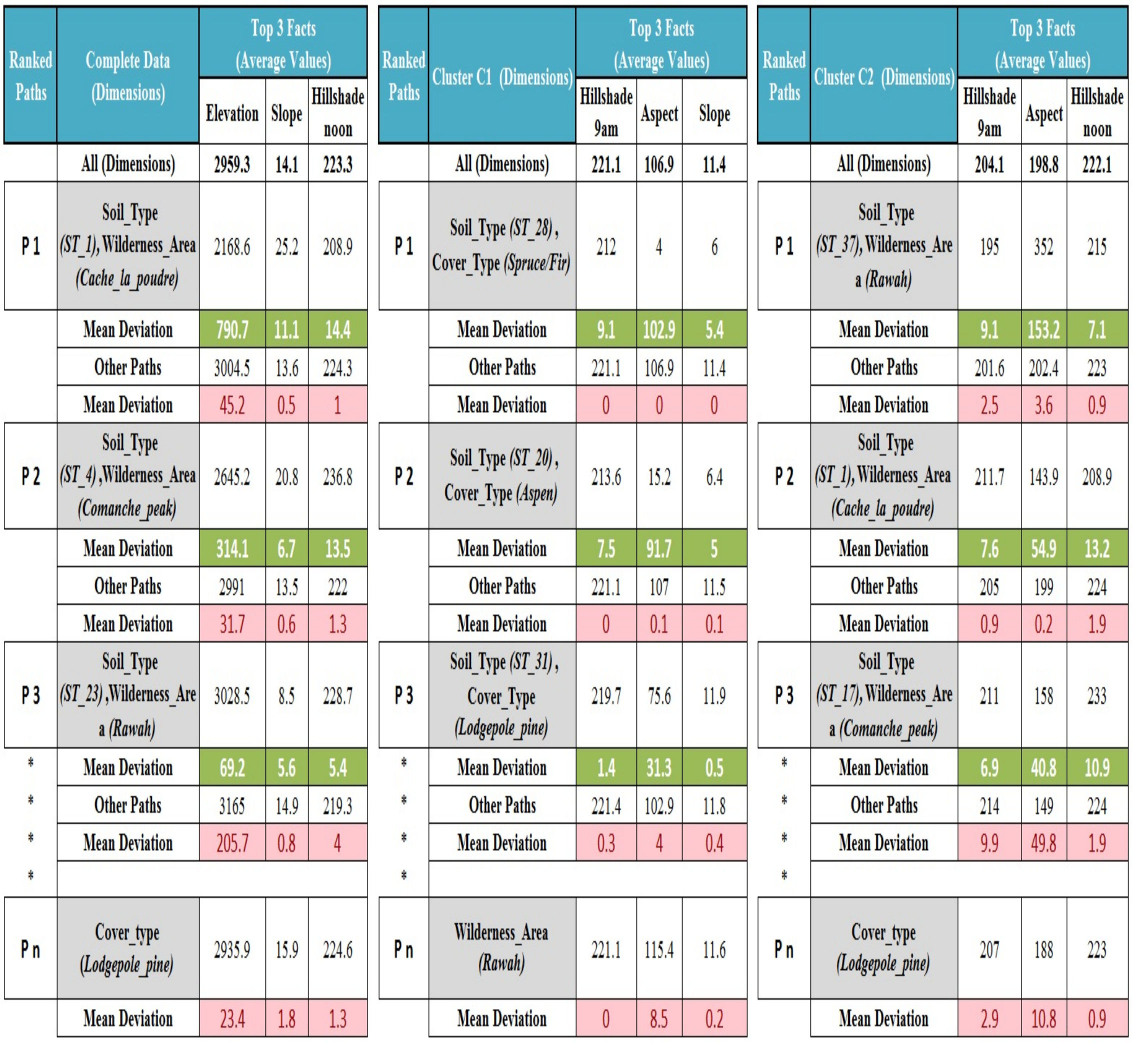


Figure 6.9: Comparison of 3-dimensional cubes at different levels of hierarchy

It can be seen from Figure 6.9 that *Wilderness Area* happens to be the 3rd ranked dimension in cluster C11; however, it holds 2nd position in an upper level cluster C2. Moreover, the *Soil Type* dimension remained at number 1 in rank on both cubes. However, both the total number and composition of groups within this dimension changed between clusters. The *Soil Type* dimension has 5 groups [G1, G2, G3, G4 and G-others] in the cluster C11 cube whereas the same dimension has an extra group [G5] in the cluster C2 cube which is at an upper level in the cluster hierarchy. Similarly, *Wilderness Area* has 2 groups [G1 and G-others] in cluster C2 cube but has only one group of outliers [G-others] in cluster C11 data cube. Additionally, the values present in each group are entirely different from one another. For instance, Group1 of *CoverType* dimension has four values [*Krummholz, Spruce/Fir, Lodgepole Pine* and *Aspen*] in C2 whereas the comparable Group1 in C11 data cube contains only 2 values [*Ponderosa Pine* and *Douglas-fir*]. Such sharp differences give new insights about the underlying data and further relationships of these dimensions with the fact variables can be explored through OLAP analysis.

6.7 Exploration of Interesting Cube Regions

Similar to the previous case studies, we took the top 3 ranked paths and compared them with the lowly ranked path suggested by our methodology for each data cluster. Figure 6.10 shows the results of the cube exploration through highly ranked paths for the complete dataset on clusters C1 and C2. It is clear from the results in Figure 6.10 that the highly ranked paths in each cluster have a tendency to reveal interesting information about the facts present in the data.



Ranked Paths	Complete Data (Dimensions)	Top 3 Facts (Average Values)		
		Elevation	Slope	Hillshade noon
	All (Dimensions)	2959.3	14.1	223.3
P1	Soil_Type *(ST_1)*, Wilderness_Area *(Cache_la_poudre)*	2168.6	25.2	208.9
	Mean Deviation	790.7	11.1	14.4
	Other Paths	3004.5	13.6	224.3
	Mean Deviation	45.2	0.5	1
P2	Soil_Type *(ST_4)*, Wilderness_Area *(Comanche_peak)*	2645.2	20.8	236.8
	Mean Deviation	314.1	6.7	13.5
	Other Paths	2991	13.5	222
	Mean Deviation	31.7	0.6	1.3
P3	Soil_Type *(ST_23)*, Wilderness_Area *(Rawah)*	3028.5	8.5	228.7
*	Mean Deviation	69.2	5.6	5.4
*	Other Paths	3165	14.9	219.3
*	Mean Deviation	205.7	0.8	4
*				
Pn	Cover_type *(Lodgepole_pine)*	2935.9	15.9	224.6
	Mean Deviation	23.4	1.8	1.3

Ranked Paths	Cluster C1 (Dimensions)	Top 3 Facts (Average Values)		
		Hillshade 9am	Aspect	Slope
	All (Dimensions)	221.1	106.9	11.4
P1	Soil_Type *(ST_28)*, Cover_Type *(Spruce/Fir)*	212	4	6
	Mean Deviation	9.1	102.9	5.4
	Other Paths	221.1	106.9	11.4
	Mean Deviation	0	0	0
P2	Soil_Type *(ST_20)*, Cover_Type *(Aspen)*	213.6	15.2	6.4
	Mean Deviation	7.5	91.7	5
	Other Paths	221.1	107	11.5
	Mean Deviation	0	0.1	0.1
P3	Soil_Type *(ST_31)*, Cover_Type *(Lodgepole_pine)*	219.7	75.6	11.9
*	Mean Deviation	1.4	31.3	0.5
*	Other Paths	221.4	102.9	11.8
*	Mean Deviation	0.3	4	0.4
*				
Pn	Wilderness_Area *(Rawah)*	221.1	115.4	11.6
	Mean Deviation	0	8.5	0.2

Ranked Paths	Cluster C2 (Dimensions)	Top 3 Facts (Average Values)		
		Hillshade 9am	Aspect	Hillshade noon
	All (Dimensions)	204.1	198.8	222.1
P1	Soil_Type *(ST_37)*, Wilderness_Area *(Rawah)*	195	352	215
	Mean Deviation	9.1	153.2	7.1
	Other Paths	201.6	202.4	223
	Mean Deviation	2.5	3.6	0.9
P2	Soil_Type *(ST_1)*, Wilderness_Area *(Cache_la_poudre)*	211.7	143.9	208.9
	Mean Deviation	7.6	54.9	13.2
	Other Paths	205	199	224
	Mean Deviation	0.9	0.2	1.9
P3	Soil_Type *(ST_17)*, Wilderness_Area *(Comanche_peak)*	211	158	233
*	Mean Deviation	6.9	40.8	10.9
*	Other Paths	214	149	224
*	Mean Deviation	9.9	49.8	1.9
*				
Pn	Cover_type *(Lodgepole_pine)*	207	188	223
	Mean Deviation	2.9	10.8	0.9

Figure 6.10: Results of cube exploration through ranked paths (CoverType dataset)

We take the top 2 dimensions from each cluster and examine the three highest ranked fact variables to examine forest lands and cover types (tree species). As these highly ranked dimensions capture the greatest amount of variation in the cluster, it would be interesting to see how sub groups of forests are distributed across the different combination of values taken across these dimensions.

However, even for these two highly ranked dimensions there are 41 soil types and 7 cover types or 4 wilderness areas to choose from, which means that even after constraining the cube to only two dimensions a large navigation space still remains to be explored manually. To resolve this problem, our methodology suggests ranked paths to define data cubes containing distinctive and interesting information. For instance, ranked path P1 highlights the data cell in the cube that has the highest deviation on *Elevation* in complete dataset. We note that the average elevation of forest land where *Soil_Type is* (SoilType1) and *Wilderness area* is (Cache_la_pourde) has an extreme deviation from the overall mean value of 2959.3 registered for all cover types contained in the cluster.

It is mentioned in the dataset description page (Blackard, Dean et al. 1998) that Cache_la_pourde is probably more unique than the other wilderness areas, due to its relatively low elevation range and tree species composition of Ponderosa pine, Douglas-fir, and cottonwood/willow.

However, it is extremely challenging to find the relationship of Cache_la_pourde wilderness area with the 40 different soil types. Even if we trim down soil types to only 12 distinct types, considering that in the dataset only 12 unique soil types are associated with Cache_la_pourde, it is still a gruelling task to analyze these 12 soil types in order to identify the one which shows extreme deviation from the average for any given fact variable, say *Elevation*.

Although the first ranked path P1 suggested by our methodology encompasses some common domain knowledge, it still provides additional precise information on soil type which is another dimension to examine known facts. In particular, it was explicit that the elevation for Cache_la_pourde area is low with high slope range but its implicit relationship with the soil types responsible for lowest elevation and highest slope range was not obvious. This previously implicit information can be quickly identified as path P1 pin-points the exact type (Soil Type 1) responsible for the lowest elevation and the highest slope. Analysts can dig out further interesting information by examining the unique characteristics of this particular soil type. For example, it belongs from the Cathedral family of extremely stony soils mostly present in mountain slopes and hills. It covers 2 to 100% of slopes, is at 6200 to 9850 feet of elevation and has a mean annual air temperature of 38 to 50 degrees F.

Similarly in ranked path P2, forests having Comanche_peak as *Wilderness_Area,* and (SoilType4) as soil type, display the second highest deviation from the mean. Again, it is given on dataset description page (Asuncion and Newman 2010) that Comanche Peak area would have a lower mean elevation value. The primary tree species Comanche_Peak has is Lodgepole pine, followed by Spruce/fir and Aspen. However, its association with the corresponding soil type is absent. The ranked path P2 not only highlights the 2nd lowest elevation but also indicates that out of 23 different soil types associated with Comanche_Peak area, Soil Type 4 is responsible for the second lowest elevation value. Soil Type 4 belongs to Vanet series of soils and consists of shallow, well drained, moderately permeable soils.

Using these ranked paths, analysts can not only identify the interesting regions in the data cube for exploration but can also perform comparative analysis. For instance, P1 and P2 for the complete dataset can be compared for the top facts, namely elevation and slope. Both paths suggested low elevation and high slope ranges but their associated soil types were completely different. For Soil Type 4, in ranked path P2, the elevation range is from 7800 to 8500 feet and covers 20 to 40% slopes whereas Soil Type 1 in P1 interestingly shows contrasting values of 6200 to 9850 elevation range and 2 to 100% slope coverage. Moreover, the mean annual air temperature also differs noticeably between the two soil types as Soil Type1 has 38 to 50 degrees F and Soil Type 4 has a tighter range of 42 to 45 degrees F. This new soil type dimension thus enhances the data cube exploration and allows for rich multidimensional analysis.

Our methodology provides these ranked paths in the order: highest to lowest deviation. The lowly ranked path Pn for each cube consists of a combination of dimensional values which show the least deviation captured as compare to all other paths. Using ranked paths, users can easily determine which particular dimensions chosen from a large number of possibilities exhibit extreme deviations from the average.

To take another example, consider the cube defined over cluster C2; even though this cube is defined over just the 2 highly ranked dimensions, the search space to be explored is still too large for knowledge discovery purposes. This is due to the fact that the *Soil_Type* dimension has 40 members and the *Wilderness_Area* dimension has4, making up a total of (40 x 4) = 160 data cells within this cube for exploration. In order to uncover interesting knowledge in a timely manner, users will need to navigate using the most discriminating soil types and wilderness areas suggested by our methodology.

6.8 Mining Association Rules from Multidimensional Schema

As done in the previous case studies, we applied rule mining on both the original dataset as well as the multidimensional schema version for clusters C1, C11 and C12. We chose the two most impure dimensions (having high entropy values) to generate association rules. Table 6.2 shows the top 10 rules for cluster C12 generated without multidimensional schema at a minimum probability value of 0.41 and minimum importance value of 0.10. Based on these thresholds the algorithm produced 84 rules in total.

With the same threshold values, we generated the rules from the multidimensional schema for the same cluster, C12. The first 10 rules satisfied the given thresholds and are shown in Table 6.3. We observe that the rules generated with the use of the multidimensional schema follow the same trends as in the previous case studies. The rules generated through the use of the multidimensional schema are more informative.

Table 6.2: Rules generated without multidimensional schema

RULES WITHOUT SCHEMA (CLUSTER C12)		
No	**Rules**	**Imp**
R1	Soil Type = SoilType3, Wilderness Areas = Cache_la_Poudre → Cover Type = Cottonwood/Willow	1.865
R2	Soil Type = SoilType17, Wilderness Areas = Cache_la_Poudre → Cover Type = Cottonwood/Willow	1.863
R3	Soil Type = SoilType38, Wilderness Areas = Comanche_Peak → Cover Type = Krummholz	1.605
R4	Soil Type = SoilType39, Wilderness Areas = Comanche_Peak → Cover Type = Krummholz	1.573
R5	Soil Type = SoilType40, Wilderness Areas = Comanche_Peak → Cover Type = Krummholz	1.537
R6	Soil Type = SoilType38, Wilderness Areas = Neota → Cover Type = Krummholz	1.486
R7	Soil Type = SoilType40, Wilderness Areas = Neota → Cover Type = Krummholz	1.382
R8	Soil Type = SoilType14, Wilderness Areas = [All] → Cover Type = Douglas-fir	1.020
R9	Soil Type = SoilType2, Wilderness Areas = [All] → Cover Type = Ponderosa-Pine	0.832
R10	Soil Type = SoilType10, Wilderness Areas = Cache_la_Poudre → Cover Type = Ponderosa-Pine	0.823

Table 6.3: Rules generated with multidimensional schema

RULES WITH SCHEMA (CLUSTER C12)		
No	**Rules**	**Imp**
R1	Soil Type Group = Group4, Wilderness Area = Group-Others → Cover Type = Cottonwood/Willow	1.865
R2	Soil Type Group = Group3, Wilderness Area = Group-Others → Cover Type = Cottonwood/Willow	1.863
R3	Soil Type Group = Group1, Wilderness Area = Group1 → Cover Type = Krummholz	1.605
R4	Soil Type Group = [All], Wilderness Area Group = Group-Others → Cover Type = Ponderosa-Pine	1.065
R5	Soil Type Group = Group-Others, Wilderness Area Group = [All] → Cover Type = Douglas-fir	1.020
R6	Soil Type Group = Group4, Wilderness Area Group = Group-Others → Cover Type = Ponderosa-Pine	0.823
R7	Soil Type Group = Group3, Wilderness Area Group = [All] → Cover Type = Ponderosa-Pine	0.814
R8	Soil Type Group = Group5, Wilderness Area Group = Group-Others → Cover Type = Ponderosa-Pine	0.808
R9	Soil Type Group = Group4, Wilderness Area Group = Group1 → Cover Type = Ponderosa-Pine	0.597
R10	Soil Type Group = Group2, Wilderness Area Group = Group1 → Cover Type = Spruce-fir	0.504

These groups provide rich and diverse information to the user while retaining the same importance score of 1.605. If we focus on the Wilderness-Area dimension, we see that rule R3 without schema only identifies Comanche_Peak as a forest land with soil type 38, whereas R3 with the schema identifies a diverse set of wilderness areas in which Rawah and Neota areas are also present.

6.9 Evaluation of Rule Interestingness via Diversity Criterion

In order to validate the claim that rules generated from schema are more diverse, we performed evaluation of the rules with the *Rae*, *CON* and *Hill* diversity measures, as with the previous case studies. We took a set of top rules generated without schema and compared it with the same set of rules generated with the use of the schema. Table 6.4 shows the results of this evaluation.

It is apparent from Table 6.4 that all the objective measures of diversity show significant improvement for the set of rules generated from the multidimensional schema. Similar to the *Automobile* and *Adult* case study results, the rules generated from the multidimensional schema are more diverse and capable of conveying more interesting knowledge to the user.

Table 6.4: Rule interestingness comparison using diversity measures

Cluster Names	Rule sets	NO Schema Rae	With Schema Rae	NO Schema CON	With Schema CON	NO Schema Hill	With Schema Hill
C1	**R1-R6**	0.230	**0.278**	0.277	**0.366**	-2.987	**-2.293**
	R1-R7	0.239	**0.256**	0.342	**0.364**	-2.646	**-2.51**
	R1-R8	0.226	**0.240**	0.34	**0.363**	-2.77	**-2.712**
	R1-R9	0.184	**0.238**	0.288	**0.378**	-3.764	**-2.731**
	R1-R10	0.176	**0.234**	0.292	**0.386**	-4.005	**-2.780**
C11	**R1-R6**	0.275	**0.296**	0.361	**0.394**	-2.338	**-1.884**
	R1-R7	0.247	**0.281**	0.351	**0.402**	-2.644	**-1.999**
	R1-R8	0.228	**0.244**	0.344	**0.369**	-2.892	**-2.393**
	R1-R9	0.187	**0.229**	0.292	**0.364**	-3.884	**-2.573**
	R1-R10	0.161	**0.196**	0.258	**0.327**	-4.81	**-3.131**
C12	**R1-R6**	0.219	**0.284**	0.251	**0.376**	-3.304	**-2.294**
	R1-R7	0.181	**0.278**	0.208	**0.397**	-4.273	**-2.353**
	R1-R8	0.163	**0.263**	0.209	**0.397**	-4.749	**-2.509**
	R1-R9	0.149	**0.225**	0.208	**0.359**	-5.195	**-3.015**
	R1-R10	0.137	**0.205**	0.203	**0.342**	-5.757	**-3.348**

Furthermore, the prediction accuracy of the rules generated from the schema also appears to be higher when compared to the rules without schema. To validate this claim, we tested the prediction accuracy of the rules generated without schema against the rules generated with the use of the schema. We used 30% of the test data from each cluster and ran 10 tests where each test randomly picked unique test data to assess the predictive accuracy of association rules generated. Table 6.5 shows the prediction accuracy for each test, expressed as a percentage.

It is clear from Table 6.5 that the prediction accuracy of the rules generated through the use of the multidimensional schema is higher when compared to the one without schema. Again, we note from this case study on a very large dataset, that rule mining performed on the multidimensional schema designed and constructed with the help of hierarchical clustering and multidimensional scaling technique generates diverse rules with greater prediction accuracy.

Table 6.5: Rule prediction accuracies for two sets of rules

Prediction Tests	Cluster C1		Cluster C11		Cluster C12	
	Schema	**No Schema**	**Schema**	**No Schema**	**Schema**	**No Schema**
Test 1	64.52	64.54	66.28	66.22	64.25	64.15
Test 2	64.47	57.36	66.46	51.75	64.43	53.66
Test 3	64.70	64.45	66.10	51.47	64.22	53.21
Test 4	64.41	57.85	66.41	66.25	64.34	53.65
Test 5	64.50	64.32	66.25	51.81	64.54	64.26
Test 6	64.62	57.37	66.54	51.82	64.43	53.73
Test 7	64.89	57.55	66.42	66.26	64.33	64.27
Test 8	64.70	64.48	66.82	66.31	64.25	64.23
Test 9	64.71	64.22	66.11	51.47	64.62	53.46
Test 10	64.52	64.38	66.41	51.67	64.22	64.14
Average Percentage	**64.60**	61.65	**66.38**	57.50	**64.36**	58.88

Summary

In this chapter, we presented the results of our third case study performed on a large real world dataset namely *CoverType* from ecology domain. As in the previous case studies, this dataset also revealed interesting and previously unknown knowledge. In terms of discovering interesting regions in data cubes, the combination of ranked dimensions namely, *wilderness area* and *soil type* along with important facts such as elevation and slope give interesting insights into various forest cover types. We have shown that through the use of the ranked paths, analysts can pinpoint particular wilderness areas (4 areas) and soils (40 types) which have the highest deviations from the mean without going through the gruelling task of analyzing a large number of paths available for exploration.

The knowledge discovered through the ranked paths was not only aligned to the common domain knowledge but also provided additional dimension to navigate data cubes. In particular, it was explicit that the elevation for *Cache_la_pourde* area is low with high slope range but its implicit relationship with the soil types responsible for lowest elevation and highest slope range was not obvious. This implicit information was easily and efficiently revealed through the top ranked paths suggested by the methodology. It was identified that Soil Type 1 from *Cathedral* family and Soil Type 4 from *Vanet* series of soils are the main types which have lowest elevation and highest slopes in the overall dataset.

More interestingly, when these two soil types were further examined and their characteristics were compared it was observed that their wilderness areas, elevation, slopes and mean air temperature have significant differences. For instance, Soil Type 4 has an elevation range of (7800 to 8500) feet and covers (20 to 40%) slopes whereas Soil Type 1 ranges from (6200 to 9850) feet and has (2 to 100%) slope coverage. Moreover, the mean annual air temperature also differs noticeably between the two soil types as Soil Type1 has (38 to 50 degrees F) and Soil Type 4 has a tighter range of (42 to 45 degrees F).

We also grouped the soil types together through the use of multidimensional scaling technique to form semantically related groups of dimensional values. The values grouped through this technique not only had objective (data driven) similarities but also showed likeness in a real world setting. For instance, Group2 of cluster C11 consists of soil type 19 and soil type 20 which are grouped together and both of these soil types have the same metadata description in the dataset. These soil types belongs to "Typic Cryaguolis and Typic Cryaquolls", two families of soil types which have the same depth (0 to 4 inches) and same dominant plant (subalpine fir) associations. Moreover, their climate and geographical zones are also identical.

Finally, association rule mining also showed that the rules generated through our generated schema were much more diverse and interesting as compared to the rules produced via flat data without the schema structure. The prediction accuracy of the rules to predict cover type has also been tested for both schema and non-schema and the schema structure outperformed the other with marginal difference.

Chapter 7
Scalability Study

In this chapter, we present experiments conducted on synthetic datasets to test the scalability of our proposed methodology. An important issue in our approach is to ensure that the proposed methods do not become a bottleneck in an environment where a large number of records or high dimensionality is present. To address this issue, the focus of this chapter is to show that the each step of the proposed methodology indeed scales with size and dimensionality of data. We have implemented a full-fledged prototype, i.e., for generating synthetic data with various parameters, and have conducted an extensive experimental evaluation to compare the processing time of each step of our proposed methodology. The key variables that we have identified for our scalability study are data size (in terms of number of records) and dimensionality (in terms of number of dimensions/variables). In the following sections, we introduce the experimental setup used to testing scalability and present the results of each methodological step with respect to the key variables.

7.1 Experimental Setup

All experiments conducted for the scalability tests were run on a 64-bit Intel® Core™ i-5 2400 CPU at 3.10 GHz running Windows 7 Operating System with 8GB RAM. The main software used in the experimental study is MATLAB and Microsoft SQL Server 2008 R2. MATLAB has been used to measure the processing time of the first three steps of our methodology namely hierarchical cluster generation, numeric variable ranking and nominal variable ranking while Microsoft SQL Server 2008 R2 has been utilized to create multidimensional schemas, construct informative cubes and generate association rules. In the following sections we show the results of our experiments performed on synthetic datasets with respect to large data size and high dimensionality for each step of our methodology.

7.2 Processing time for hierarchical cluster generation

As explained in Chapter 3 the first step of the proposed methodology is to generate hierarchical clusters at different abstraction levels. Therefore, our first set of experiment aims at the analysis of the impact of data size (number of records) on computation time of the hierarchical cluster generation. We test the influence of increasing the number of records on cluster generation process considering computation time as the main performance indicator. Varying the number of records starting from 200 K to 2000 K, we measure the cluster generation time in seconds. For consistency, we fixed the other variable, namely the number of dimensions to 10 to test the variability in computation time with number of records. Figure 7.1 show the results of our experimentation on synthetic datasets of different sizes.

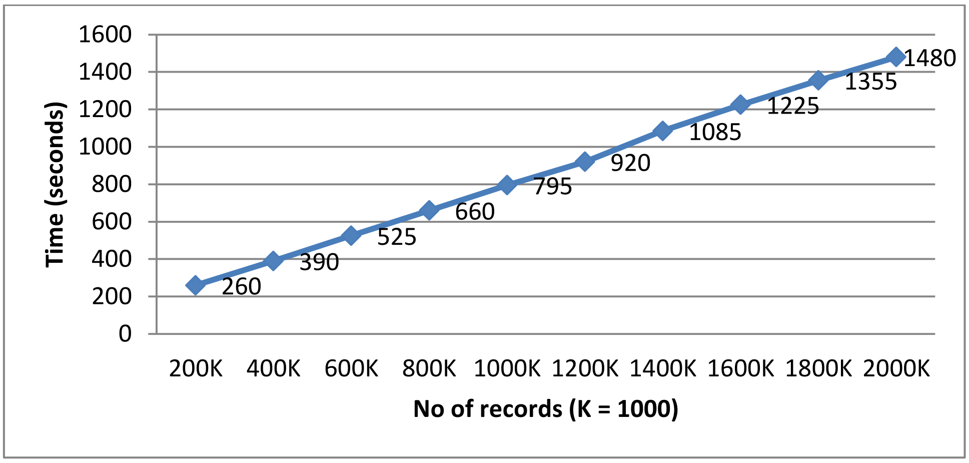


Figure 7.1: Processing time of hierarchical cluster generation w.r.t different data sizes

The y-axis represents the cluster generation time in seconds while the x-axis refers to the number of records in thousands or K. It can be seen from Figure 7.1 that the proposed method of cluster generation which involves the sub-steps of cluster labelling and data allocation at different abstraction levels (see Algorithm 1- Chapter 3) scales well with the increasing number of records. Data size has a proportional or linear effect on computational time.

The second set of experiments aims at the analysis of the impact of second variable which is data dimensionality. We test the influence of dimensions on cluster generation similar to the previous set of experiments by considering the computation time as the main performance indicator. Varying the number of dimensions starting from 20 till 200, we evaluate the processing time in seconds. We fixed the number of records as 10K to test the effect of dimensionality on processing time. The result of the experiment is depicted in Figure 7.2.

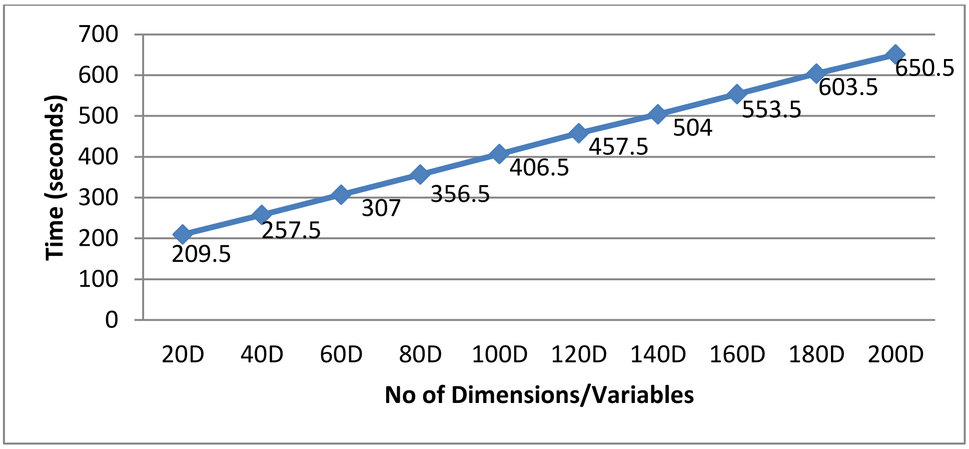


Figure 7.2: Processing time of hierarchical cluster generation w.r.t different dimensions

Here, the x-axis refers to the number of dimensions or variables involved for generating hierarchical clusters at various levels of data abstraction. It is clear from Figure 7.2 that the increase in the number of variables in data effects the cluster generation time in a linear fashion. Similar to the previous set of experiments with large data sizes, it can be observed that cluster generation process also scales well with dimensionality of the data.

7.3 Processing time for ranking numeric variables

In this section, we present the scalability results of the second step of our methodology which is ranking of the numeric variables. Similar to the previous section, we fixed the number of dimensions/variables to 10 and present the results with respect to different data sizes varying from 200K to 2000K. Figure 7.3 shows the results of the experiment.

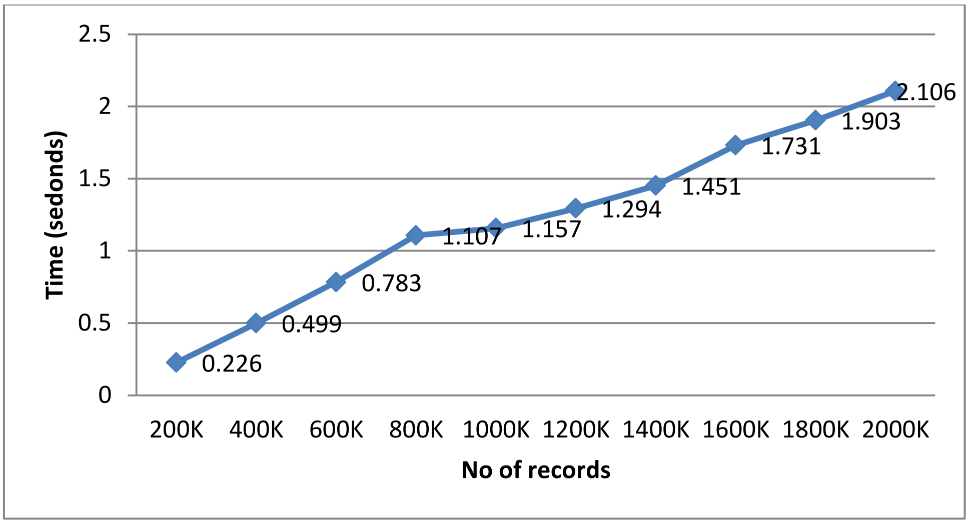


Figure 7.3: Processing time of ranking numeric variables w.r.t different data sizes

It is clear from Figure 7.3 that ranking of numeric variables takes a maximum 2.106 seconds for the extreme case of 2000K data size. Moreover, it shows that PCA which is used for the ranking the numeric variables is a robust method and large data size does not affect the processing time significantly. In fact, over the entire range of data size processing time scales in a sub-linear fashion, as shown in Figure 7.3.

In the second part of the experiment to test the effect of ranking numeric variables on dimensionality, we fixed the number of records to 10K and varied the number of dimensions/variables from 20 to 200 dimensions, in steps of 20. Figure 7.4 shows the results of this experiment. It can be seen from Figure 8.4 that the processing time scales linearly with dimensionality.

However, with increase of dimensionality, we noticed that processing time has higher sensitivity to dimensionality than data size. This is evident from the fact that the gradient in Figure 7.4 is higher than of Figure 7.3. The reason is that when a large number of dimensions are involved then it takes a higher amount of time to compute individual component scores and to compute the item covariance matrix that is needed for the application of PCA.

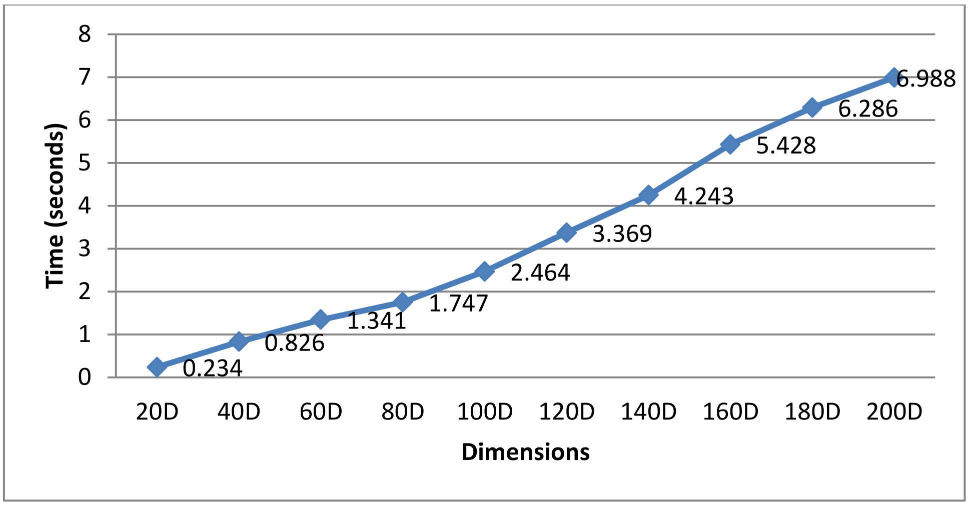


Figure 7.4: Processing time of ranking numeric variables w.r.t different dimensions

7.4 Processing time for ranking nominal variables

Similar to the previous section, we compute the processing time of the third step of our methodology which involves ranking nominal variables. We use the same parameters and firstly measure processing time with respect to different sizes by fixing the number of dimensions to 10. It is important to highlight at this point that for this step, we have used information gain based ranking method to measure the processing time. The reason for choosing information gain instead of Multiple Correspondence Analysis (MCA) based ranking method is that MCA is basically a counterpart of PCA and hence the processing time is more or less similar to PCA whereas the information gain based ranking method utilizes entropy concept which is a completely different method from correspondence analysis techniques. Therefore it is worth investigating and reporting the processing time with respect to our information gain based ranking method. Figure 7.5 shows the results of this experiment.

It is clear from Figure 7.5 that information gain based ranking method is also a robust method as it takes only fraction of a second (0.218) to process 2000K records.

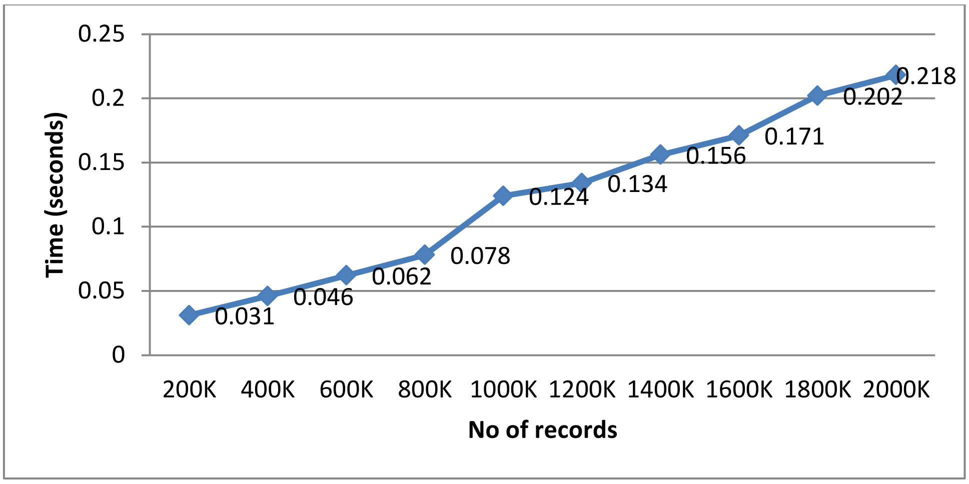


Figure 7.5: Processing time of ranking nominal variables w.r.t different data sizes

If we compare it with correspondence analysis based methods such as PCA/MCA then it can easily be identified that for exactly the same number of records PCA took 2.106 seconds. Figure 7.6 shows the comparison of information based ranking method verses correspondence based ranking method.

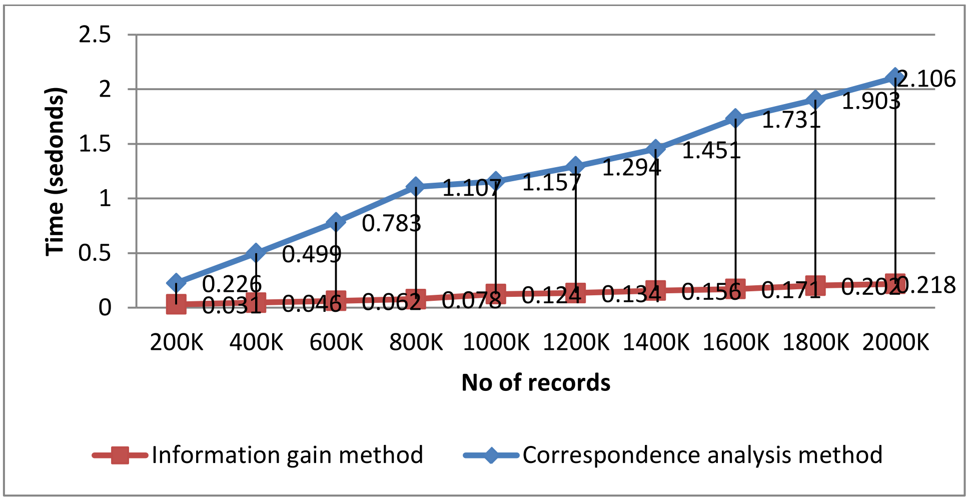


Figure 7.6: Information gain ranking method vs Correspondence analysis based ranking method with respect to different data sizes

We can easily see that there is a significant difference in processing time with respect to different data sizes. It is clear that the information gain based ranking method is much faster than correspondence analysis based methods such as PCA/MCA and should be preferred when faster processing is required.

The second set of experiment tests the scalability with respect to dimensionality. Again, we fixed the number of records to 10K and record the processing time obtained by varying the nominal variables from 20 to 200. Figure 7.7 shows the results of this experiment.

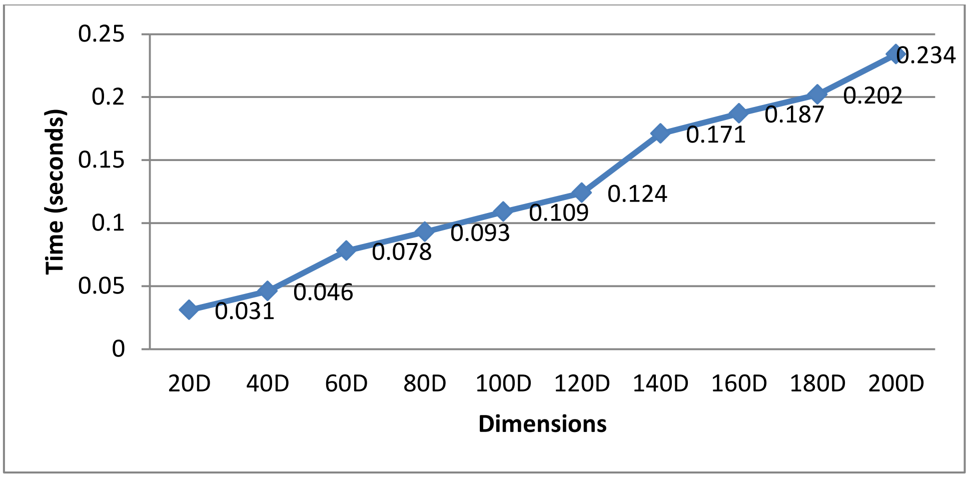


Figure 7.7: Processing time of ranking nominal variables w.r.t different dimensions

It is clear from Figure 7.7 that the information gain based ranking method also scale well with respect to dimensionality. Even for the highest number of dimension the method gives an output in less than a second. Again, if we compare the processing time of information gain method with correspondence based method for high dimensional data then we see that the information gain method supersedes the other. Figure 7.8 shows the results of this comparison.

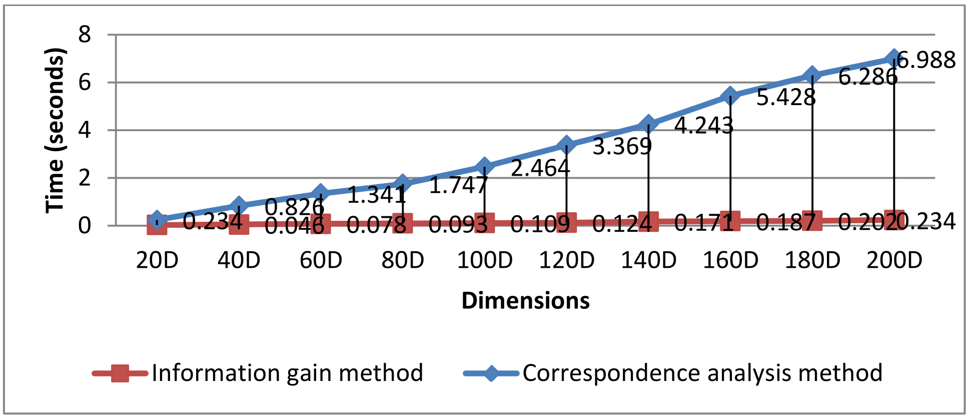


Figure 7.8: Information gain ranking method vs Correspondence analysis based ranking method with respect to different dimensions

It is clear that the information gain based ranking method is significantly more robust than the correspondence analysis based ranking method with respect to scalability. For instance, the processing time for 200 dimensions using correspondence based method is nearly 7 seconds whereas the same number of dimension can be processed in approximately 0.234 seconds using the information gain based ranking method. The results presented in Figure 7.8 clearly show the robustness of information gain based ranking method for nominal variables in terms of both data volume and dimensionality.

7.5 Processing time for multidimensional scaling

The fourth step of our proposed methodology is the application of the multidimensional scaling technique. Similar to previously presented steps, we measured the processing time of this step by varying both data size and dimensionality. Figure 7.9 shows the results of our experiment by varying the number of records whereas Figure 7.10 depicts the results by varying the number of dimensions and fixing the number of records to 10K.

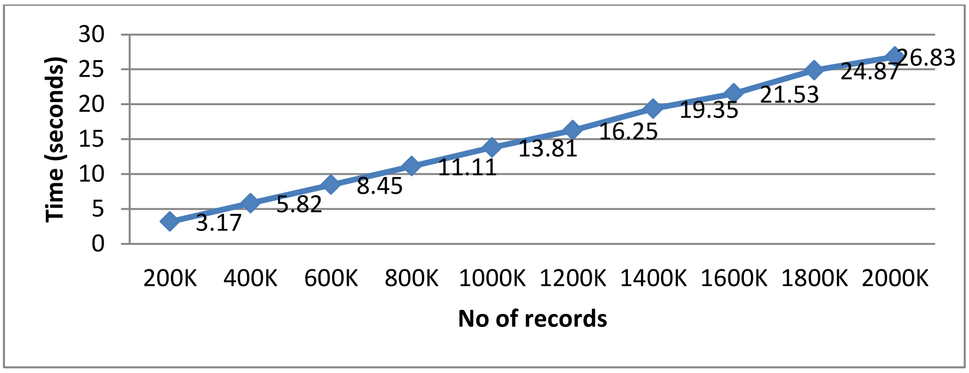


Figure 7.9: Processing time of multidimensional scaling w.r.t different data sizes

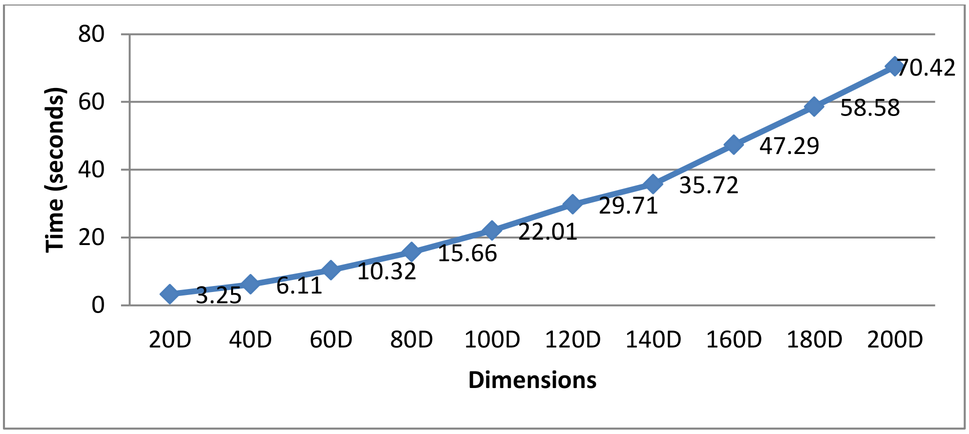


Figure 7.10: Processing time of multidimensional scaling w.r.t different dimensions

Similar to the results of previous steps, it is clear from Figure 7.9 and Figure 7.10 that multidimensional scaling method scales well with data volume and dimensionality of the data. Moreover, the method is more sensitive to dimensionality than data size, in terms of the number of records. For instance, the rate of increase in processing time with increasing number of dimensions is much higher as compared to the rate of increase with respect to increasing number of records.

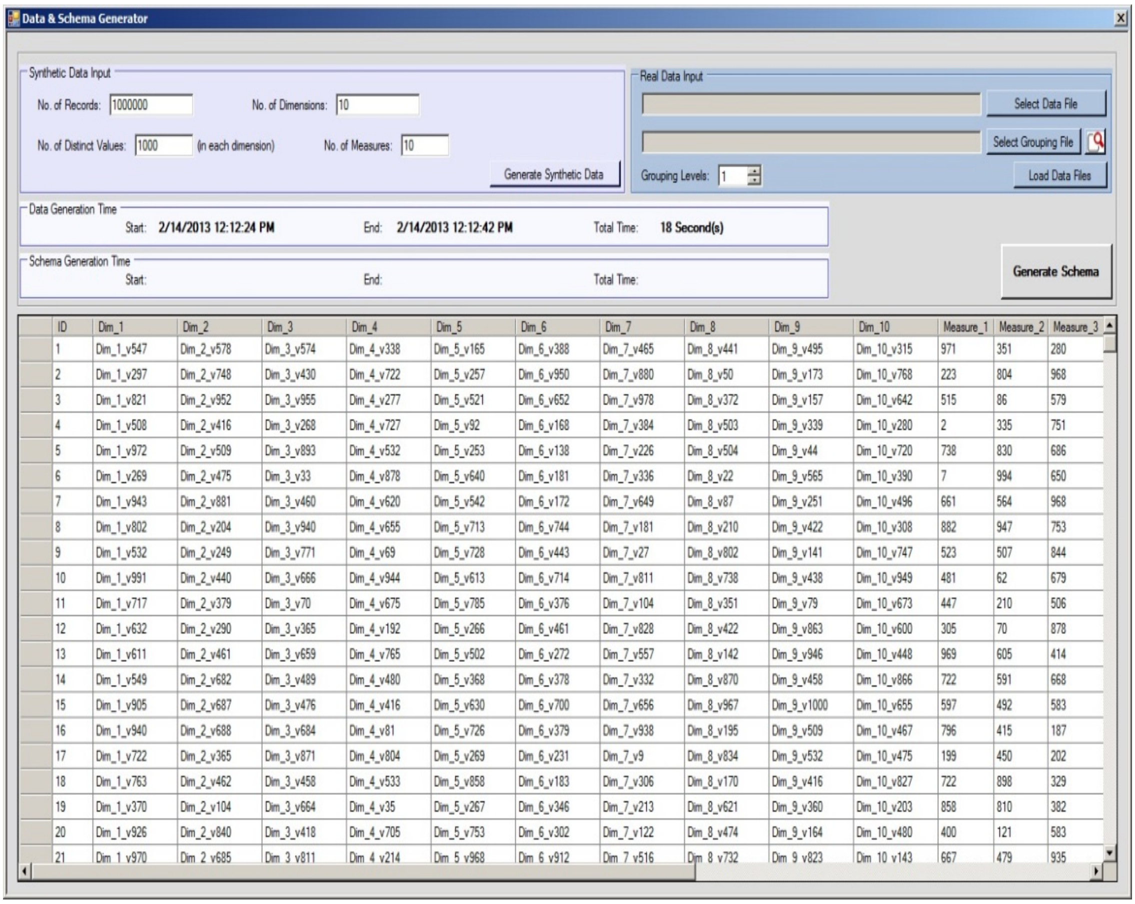
7.6 Processing time of multidimensional schema generation

In this chapter, we presented experiments conducted on synthetic datasets to test the scalability of our proposed schema generation step. We test the scalability of this step in an environment where a large number of records, high dimensionality and a high degree of granularity (dimensional levels) are present. We have implemented a full-fledged prototype, i.e., for automatically generating a schema with various parameters, and have conducted an extensive experimental evaluation using the developed prototype. The additional key variable that we have identified for assessing scalability is dimensional cardinality which plays a vital role in the schema generation process.

7.6.1 Developed Prototype for Evaluation

The schema generation prototype has been developed in Visual Studio.NET 2010 using the C# programming language. Figure 7.11 shows the main interface of our developed prototype. It consists of two data input sections namely *Synthetic Data Input* and *Real Data Input*. In *Synthetic Data Input* the parameters that need to be specified are: number of records, number of dimensions, distinct values taken in each dimension, and the number of measures.

Alternatively, users can input real data in the form of text files and specify dimensional granularity by selecting the appropriate number of grouping levels from the drop down menu in order to automatically generate multidimensional schema. The interface also contains sections where data generation and schema generation time is measured and displayed in seconds.



Synthetic Data Input

No. of Records: 1000000 No. of Dimensions: 10

No. of Distinct Values: 1000 (in each dimension) No. of Measures: 10

Real Data Input

Grouping Levels: 1

Data Generation Time

Start: 2/14/2013 12:12:24 PM End: 2/14/2013 12:12:42 PM Total Time: 18 Second(s)

Schema Generation Time

Start: End: Total Time:

ID	Dim_1	Dim_2	Dim_3	Dim_4	Dim_5	Dim_6	Dim_7	Dim_8	Dim_9	Dim_10	Measure_1	Measure_2	Measure_3
1	Dim_1_v547	Dim_2_v578	Dim_3_v574	Dim_4_v338	Dim_5_v165	Dim_6_v388	Dim_7_v465	Dim_8_v441	Dim_9_v495	Dim_10_v315	971	351	280
2	Dim_1_v297	Dim_2_v748	Dim_3_v430	Dim_4_v722	Dim_5_v257	Dim_6_v950	Dim_7_v880	Dim_8_v50	Dim_9_v173	Dim_10_v768	223	804	968
3	Dim_1_v821	Dim_2_v952	Dim_3_v955	Dim_4_v277	Dim_5_v521	Dim_6_v652	Dim_7_v978	Dim_8_v372	Dim_9_v157	Dim_10_v642	515	86	579
4	Dim_1_v508	Dim_2_v416	Dim_3_v268	Dim_4_v727	Dim_5_v92	Dim_6_v168	Dim_7_v384	Dim_8_v503	Dim_9_v339	Dim_10_v280	2	335	751
5	Dim_1_v972	Dim_2_v509	Dim_3_v893	Dim_4_v532	Dim_5_v253	Dim_6_v138	Dim_7_v226	Dim_8_v504	Dim_9_v44	Dim_10_v720	738	830	686
6	Dim_1_v269	Dim_2_v475	Dim_3_v33	Dim_4_v878	Dim_5_v640	Dim_6_v181	Dim_7_v336	Dim_8_v22	Dim_9_v565	Dim_10_v390	7	994	650
7	Dim_1_v943	Dim_2_v881	Dim_3_v460	Dim_4_v620	Dim_5_v542	Dim_6_v172	Dim_7_v649	Dim_8_v87	Dim_9_v251	Dim_10_v496	661	564	968
8	Dim_1_v802	Dim_2_v204	Dim_3_v940	Dim_4_v655	Dim_5_v713	Dim_6_v744	Dim_7_v181	Dim_8_v210	Dim_9_v422	Dim_10_v308	882	947	753
9	Dim_1_v532	Dim_2_v249	Dim_3_v771	Dim_4_v69	Dim_5_v728	Dim_6_v443	Dim_7_v27	Dim_8_v802	Dim_9_v141	Dim_10_v747	523	507	844
10	Dim_1_v991	Dim_2_v440	Dim_3_v666	Dim_4_v944	Dim_5_v613	Dim_6_v714	Dim_7_v811	Dim_8_v738	Dim_9_v438	Dim_10_v949	481	62	679
11	Dim_1_v717	Dim_2_v379	Dim_3_v70	Dim_4_v675	Dim_5_v785	Dim_6_v376	Dim_7_v104	Dim_8_v351	Dim_9_v79	Dim_10_v673	447	210	506
12	Dim_1_v632	Dim_2_v290	Dim_3_v365	Dim_4_v192	Dim_5_v266	Dim_6_v461	Dim_7_v828	Dim_8_v422	Dim_9_v863	Dim_10_v600	305	70	878
13	Dim_1_v611	Dim_2_v461	Dim_3_v659	Dim_4_v765	Dim_5_v502	Dim_6_v272	Dim_7_v557	Dim_8_v142	Dim_9_v946	Dim_10_v448	969	605	414
14	Dim_1_v549	Dim_2_v682	Dim_3_v489	Dim_4_v480	Dim_5_v368	Dim_6_v378	Dim_7_v332	Dim_8_v870	Dim_9_v458	Dim_10_v866	722	591	668
15	Dim_1_v905	Dim_2_v687	Dim_3_v476	Dim_4_v416	Dim_5_v630	Dim_6_v700	Dim_7_v656	Dim_8_v967	Dim_9_v1000	Dim_10_v655	597	492	583
16	Dim_1_v940	Dim_2_v688	Dim_3_v684	Dim_4_v81	Dim_5_v726	Dim_6_v379	Dim_7_v938	Dim_8_v195	Dim_9_v509	Dim_10_v467	796	415	187
17	Dim_1_v722	Dim_2_v365	Dim_3_v871	Dim_4_v804	Dim_5_v269	Dim_6_v231	Dim_7_v9	Dim_8_v834	Dim_9_v532	Dim_10_v475	199	450	202
18	Dim_1_v763	Dim_2_v462	Dim_3_v458	Dim_4_v533	Dim_5_v858	Dim_6_v183	Dim_7_v306	Dim_8_v170	Dim_9_v416	Dim_10_v827	722	898	329
19	Dim_1_v370	Dim_2_v104	Dim_3_v664	Dim_4_v35	Dim_5_v267	Dim_6_v346	Dim_7_v213	Dim_8_v621	Dim_9_v360	Dim_10_v203	858	810	382
20	Dim_1_v926	Dim_2_v840	Dim_3_v418	Dim_4_v705	Dim_5_v753	Dim_6_v302	Dim_7_v122	Dim_8_v474	Dim_9_v164	Dim_10_v480	400	121	583
21	Dim_1_v970	Dim_2_v685	Dim_3_v811	Dim_4_v214	Dim_5_v968	Dim_6_v912	Dim_7_v516	Dim_8_v732	Dim_9_v823	Dim_10_v143	667	479	935

Figure 7.11: Prototype for generating synthetic data and multidimensional schema

A small sample of either the synthetically generated data or real data loaded by users is shown in the data grid at the bottom of the prototype to give a glimpse of the data before proceeding to the schema generation process. The Generate Schema button generates the appropriate multidimensional schema in the Database Server and loads the corresponding data into dimension and fact tables. The functional code for creating dimension and fact tables and populating the schema is provided at the end of this thesis in Appendix I.

7.6.2 Effect of large data size on schema generation

Our first set of experiments aims at the analysis of the impact of first variable which is the data size (number of records) on execution time of the schema generation process. We test the influence of increasing the number of records on schema generation considering the generation time to be the main performance indicator. Varying the number of records starting from 10 K to 2000 K, we measure the schema generation time in seconds. For consistency, we fixed other variables such as number of dimensions and number of measures to 3 to test variability of the schema generation time with the number of records. Figure 7.12 and Figure 7.13 show the results of our experimentation.

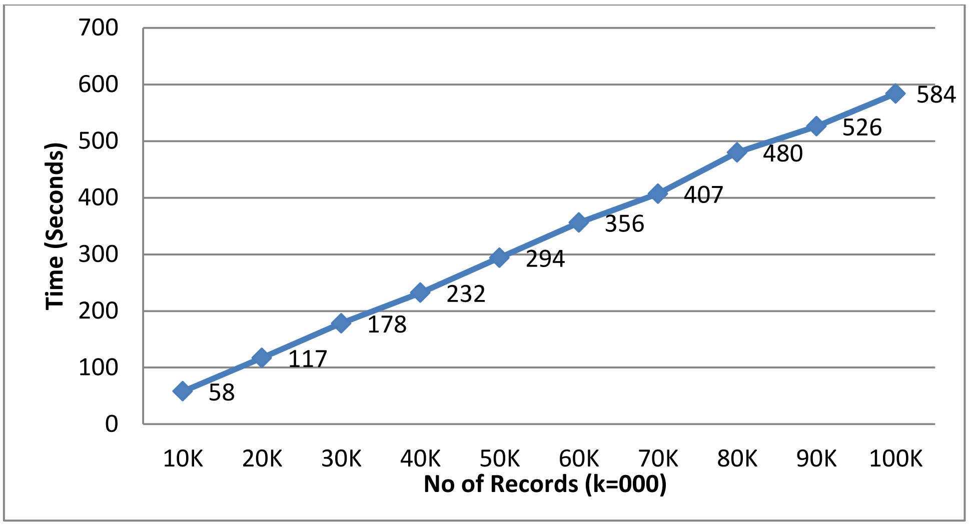


Figure 7.12: Scalability of processing time with Data Size [10K-100K]

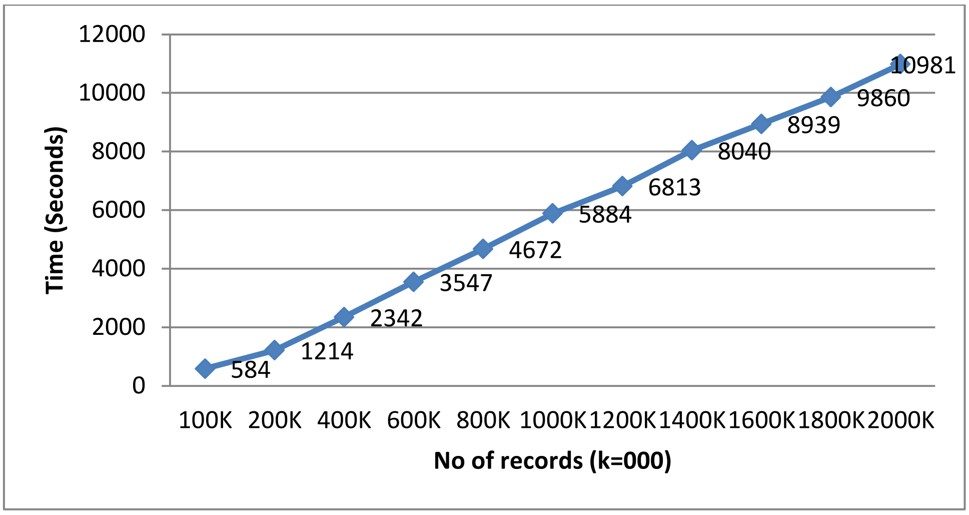


Figure 7.13: Scalability of processing time with Data Size [100K-2000K]

It can be seen from Figures 7.12 and 7.13 that the proposed method scales well with the increasing number of records as the computing time increases linearly.

7.6.3 Effect of dimensionality on schema generation

The second set of experiments aims at the analysis of the impact of the second variable which is schema dimensionality. Varying the number of dimensions starting from 3 to 200, we measure the schema generation time in seconds. We fixed the number of records to 10K to test the effect of increasing dimensions on schema generation time. Figure 7.14 and Figure 7.15 show the results of our experiment. Here, the x-axis refers to the number of dimensions involved for schema generation.

An interesting observation in these experiments is the curse of dimensionality issue. Figure 7.14 and Figure 7.15 confirm that increasing the number of dimensions has a direct impact on schema generation time, and consequently, on the performance of our proposed method.

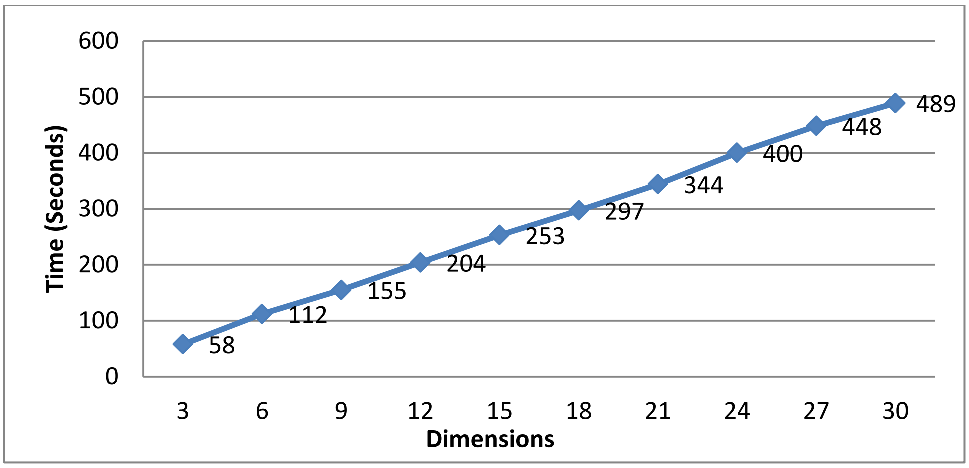


Figure 7.14: Scalability of processing time with dimensionality [3Dims-30Dims]

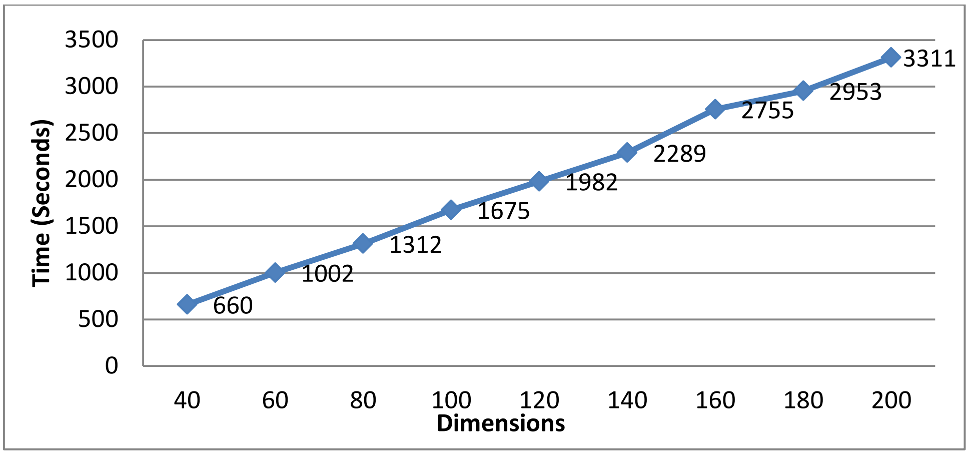


Figure 7.15: Scalability of processing time with dimensionality [40Dims-200Dims]

Figure 7.16 shows the comparison of schema generation time with respect to different number of dimensions. We can see from Figure 7.16 that with each additional dimension the execution time increases significantly.

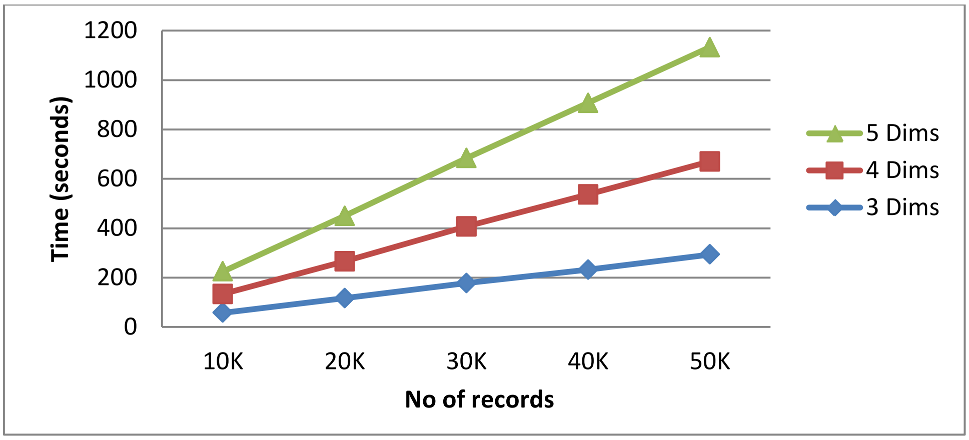


Figure 7.16: Scalability of processing time with data size for varying dimensionality

Figure 7.16 also shows that an increase in the dimensionality causes proportionally greater increases in processing time when the data size is higher, indicating that the two variable are interacting with each other in the determination of processing time. This experiment thus illustrates that dimensionality plays a key role in the determination of schema generation time.

Moreover, if we compare the schema generation time of high dimensional data against that of high data volume, we observe that the impact of adding 3 dimensions is almost equivalent to adding 10,000 data instances. Figure 7.17 presents the relative sensitivity of dimensionality versus data size.

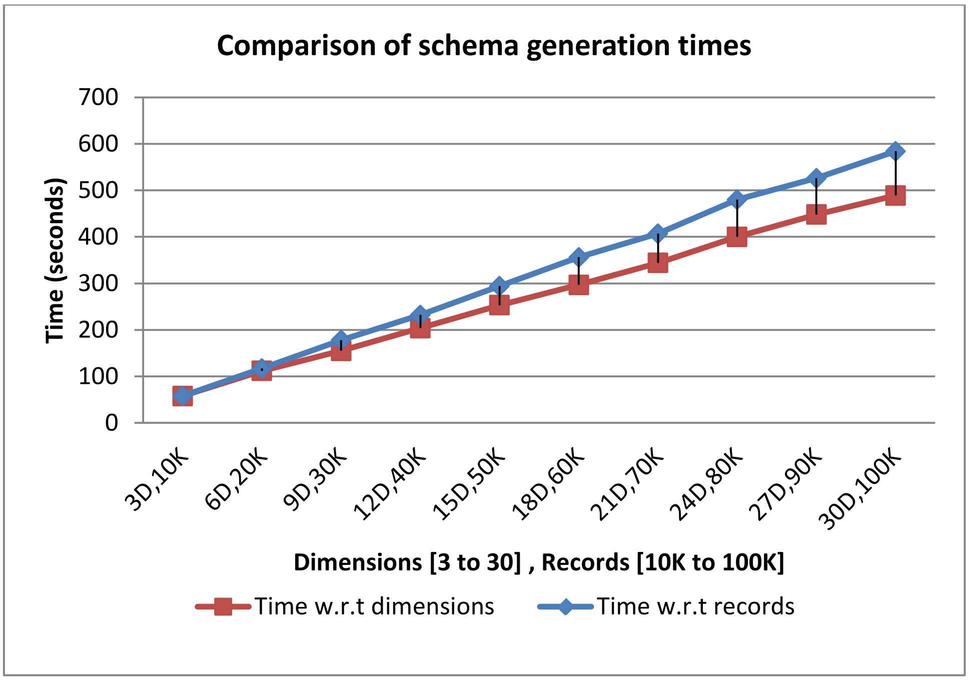


Figure 7.17: Comparative effects of dimensionality and data size on processing time

The dimensionality is increased progressively in intervals of size 3, while the number of records is simultaneously incremented in intervals of 10K in order to assess the relative contributions of dimensionality and data size on processing time. Thus for example at a given data point, say (12D, 40K), the next measurement point (15D, 50K) was generated first by increasing dimensionality by 3 and then proceeding to increase data size by 10K.

From the trajectories of the two curves it is clear in Figure 7.17 that dimensionality is a bigger issue than data size. Moreover, in traditional OLAP applications, cube materialization is the most demanding process (Ribeiro and Weijters 2011) which requires the warehouse designer to select the smallest subset of dimensions that together capture the most meaningful information. Therefore, in our proposed methodology, we stress on using information theoretic measures such as information gain to filter out less informative dimensions, which in turn minimizes the number of views required for materialization. To the best of our knowledge, we are the first to experiment with schema generation with data having over 200 dimensions. Considering the time required for generating schema, our method is clearly robust and scalable to high dimensional data environments.

7.6.4 Effect of cardinality on schema generation

To test the effect of cardinality on schema generation, we run another set of experiments by varying the number of hierarchical levels from 1 to 10 in each dimension to calculate the schema generation time. In these experiments, we fixed the number of records to 10,000 for a 3 dimensional dataset where each dimension consists of 100 distinct values. In addition to the data file, a grouping XML file is also given as input. This grouping file has the grouping information which is utilized by our developed prototype to assign group names at each level of dimensional hierarchy. Figure 7.18 shows the effect of varying the number of hierarchical levels for the 3 dimensional data used for experimentation.

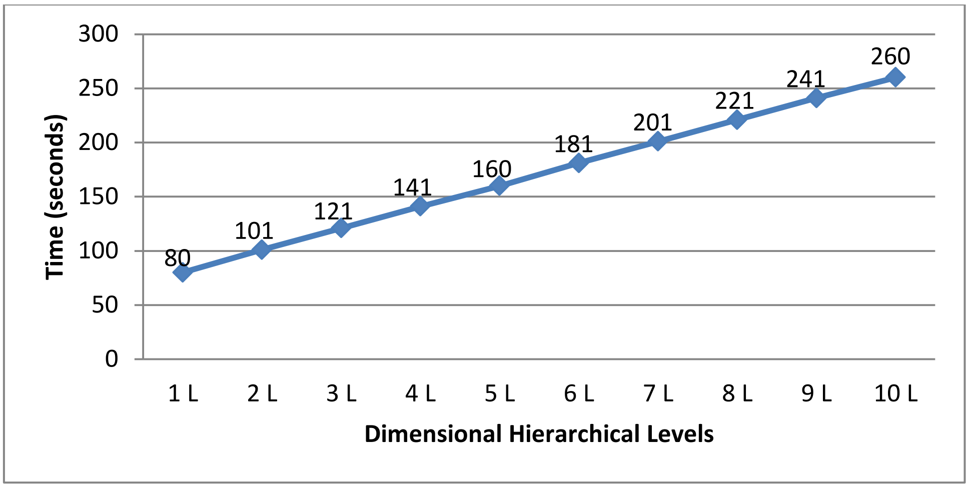


Figure 7.18: Effect of cardinality on processing time

The cardinality is increased 1L (one level) to 10L (ten levels) in the hierarchy. The reason for testing a maximum of ten levels is that in real world data warehousing scenarios, dimensions usually have less than 10 levels in their hierarchy. For example, the most commonly used Time dimension usually consists of seven levels, namely (Year→ Quarter→ Month→ Week→ Day→ Hour→ Minute→ Second) (Han and Kamber 2006) or another common Location dimension is typically defined as follows (Region → Country → State → City → Town → Street) (Malinowski and Zimányi 2008). In general, having more than 10 levels is counter-productive in terms of model comprehensibility in a real world context and this influenced our decision to restrict the number of levels to 10. Similar to results presented in the previous section, the curse of dimensionality issue is also apparent in these experiments.

The results in Figure 7.18 confirms that increasing cardinality or the number of levels in the dimensions has a significant impact on schema generation time. We note that the schema generation time increases by approximately 20 seconds with the addition of each level in the dimensional hierarchy. It highlights the fact that dimensional cardinality directly effects the processing time. The rationale behind this significant increase is the fact that with each additional level in the dimensional hierarchy, schema generation requires two additional tasks. First, it creates an extra column in the dimensional table for the additional level and secondly, inserts the corresponding group names in this column from the uploaded grouping file. However, the linear increase in time shows that our proposed method scales well with the cardinality.

7.7 Processing time of constructing informative data cubes

The sixth step of our methodology is to construct informative data cubes using the automatically generated schema. In this section, we note the cube construction time with respect to both increasing number of records and dimensions. For testing the effect of increasing number of records on cube construction time, we fixed the number of dimensions to 3 and calculated the construction time by varying the number of records. Figure 7.19 shows the results of scalability of processing time with varied data sizes.

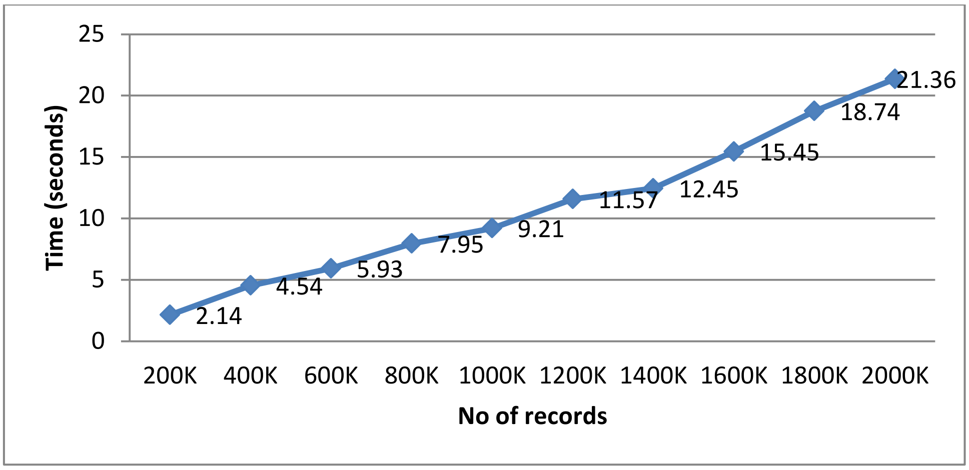


Figure 7.19: Processing time of constructing informative data cubes with varied data sizes

It is clear from Figure 7.19 that the cube construction time is scalable with increasing number of records. We also tested the scalability of this step by varying the number of dimensions from 20 to 200 dimensions in order to ensure that the method scales well with high dimensional data. Figure 7.20 shows the scalability results of cube construction with respect to different number of dimensions.

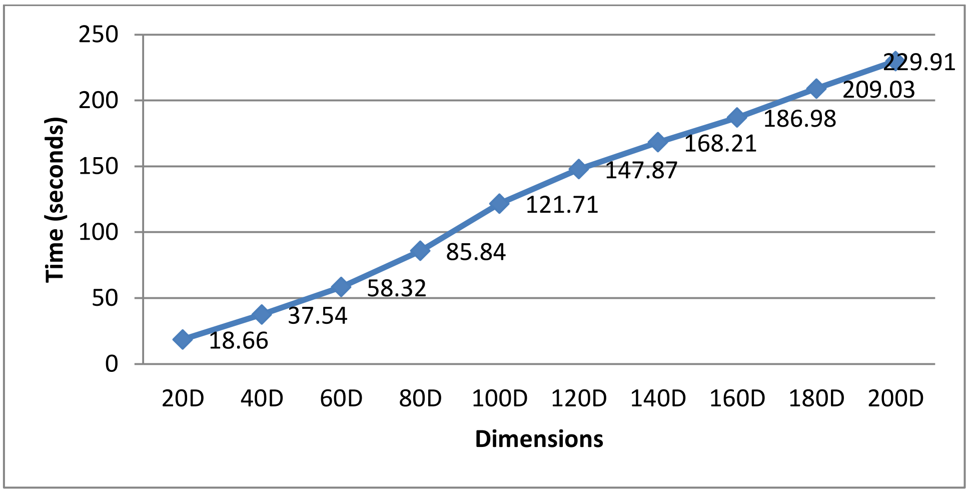


Figure 7.20: Processing time of constructing informative data cubes with varied number of dimensions

Similar to the experiments conducted to test the scalability of schema generation step, the curse of dimensionality issue is also apparent in here. Figure 7.20 confirms that the increasing number of dimensions has a direct impact on cube construction time and, consequently, on the performance of our proposed method.

The difference in gradients of the trajectories in Figures 7.19 (sub-linear) and 7.20 (super-linear), it is apparent that cube generation time is more sensitive to dimensionality than data size. This implies that that a small number of dimensions can have a larger impact on cube generation time as compared to a large number of records.

7.8 Processing time of association rule mining

The final step of our methodology is the application of rule mining algorithm in order to generate diverse association rules. We fixed minimum support to 3% and minimum confidence to 40% for all the experimentation in this step. These are the default values for *support* and *confidence* parameters in MS SQL Server 2008 R2 data mining software and are suitable for large datasets with a large number of distinct items (MacLennan, Tang et al. 2011).

We note the effect of processing time of this final step by varying the two key variables which are consistently used in all our experiments. Firstly, we fixed the number of dimensions to 3 and calculated the processing time by varying the number of records. Figure 7.21 shows the results obtained through our first set of experiment.

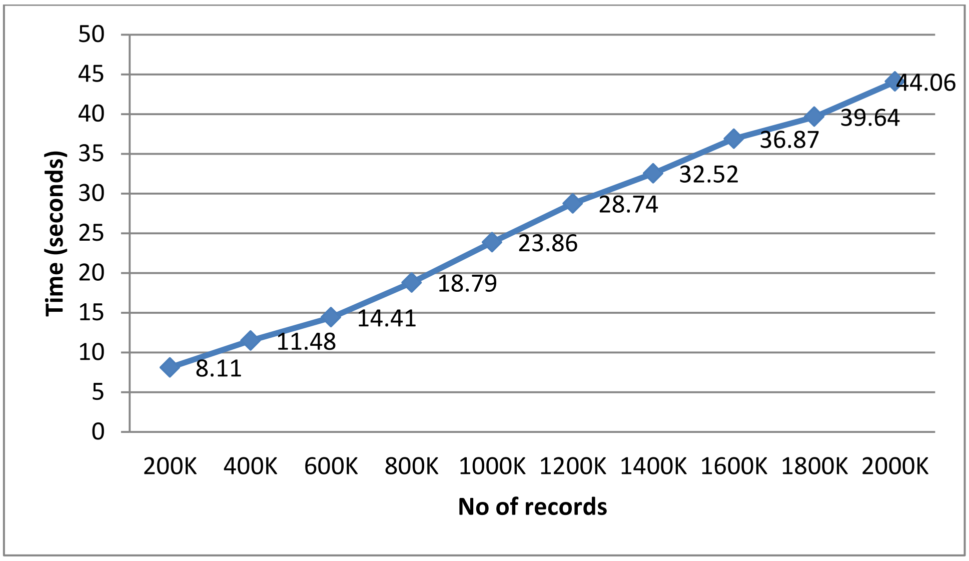


Figure 7.21: Processing time of rule generation with varied data sizes

Secondly, we fixed the number of records to 10K and calculated the processing time by varying the number of dimensions. Figure 7.22 depicts the results by varying the number of dimensions.

Figure 7.21 and Figure 7.22 confirm that rule generation scales well with data volume and dimensionality. However, the results are somewhat different from those obtained with the previous steps of our methodology. Here, we note that scalability with respect to dimensionality is sub-linear as opposed to being super-linear with respect to high data volume.

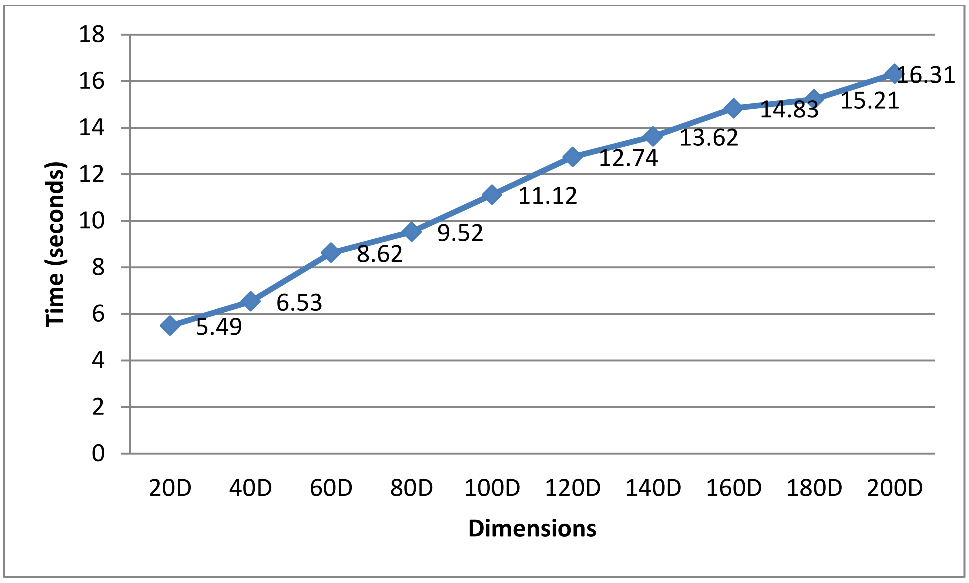


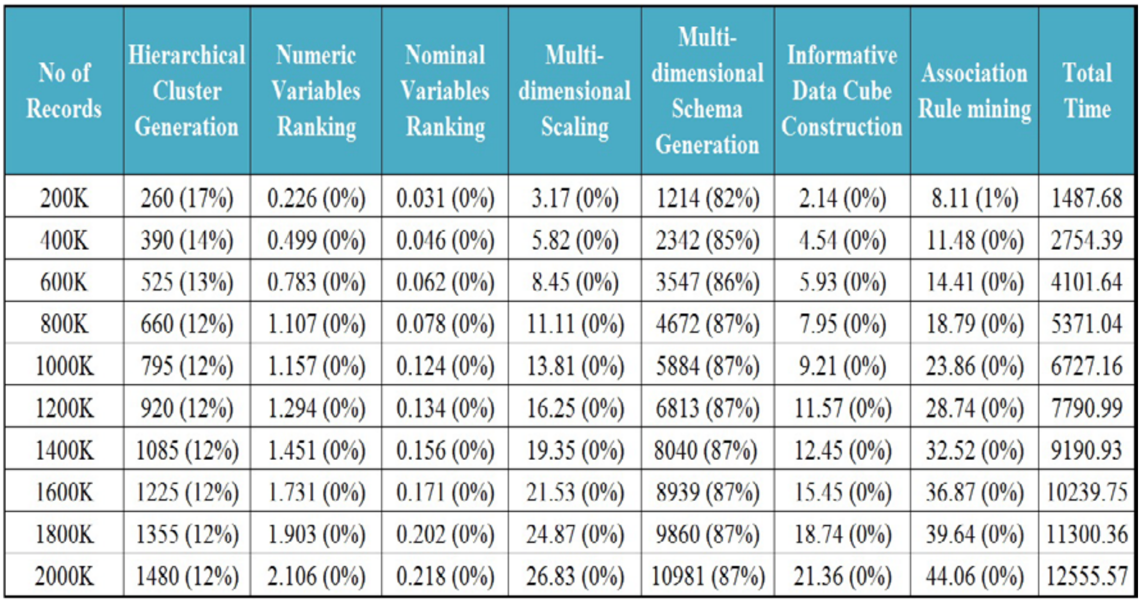
Figure 7.22: Processing time of rule generation with varied dimensions

This is due to the fact that association rule mining involves a time consuming and iterative step of finding frequent and candidate item-sets for a given threshold *support* percentage (3% in our experiment). As association rule mining algorithm works by scanning the complete data set and counting the *support* of each item, processing a large number of records with distinct values takes more time when compared to a high number of dimensions with less number of distinct records. For example, it takes 7.11 seconds to generate rules from a dataset of size 200K having 3 dimensions but lesser time (5.49 seconds) is required to generate rules on a dataset of size of 10K having 20 dimensions. The reason is that the algorithm has to process (200,000 x 3 = 600,000) items for a 3-dimensional dataset of size 200K compared to processing (10,000 x 20 = 200,000) lesser number of items for a 20-dimensional dataset of size 10K. Thus the dimensionality has a smaller (sub-liner) effect as opposed to the larger (super-liner) effect of increasing then number of records for this particular step of our methodology.

7.9 Scalability of the overall methodology

In the previous sections, we presented the scalability results of the individual steps of our proposed methodology whereas this section highlights the combined effect of all the methodological steps on scalability. Table 7.1 summarizes the processing time for individual methodological steps along with the total time required for processing the complete methodology. We varied the data sizes from 200K to 200K and fixed the number of dimension to 10. Each row in Table 7.1 shows the processing time of the steps in seconds for various data sizes. Moreover, the contribution of each step in percentage is shown besides the raw processing time of each step of our methodology.

Table 7.1: Scalability of proposed methodology w.r.t variable data sizes



No of Records	Hierarchical Cluster Generation	Numeric Variables Ranking	Nominal Variables Ranking	Multi-dimensional Scaling	Multi-dimensional Schema Generation	Informative Data Cube Construction	Association Rule mining	Total Time
200K	260 (17%)	0.226 (0%)	0.031 (0%)	3.17 (0%)	1214 (82%)	2.14 (0%)	8.11 (1%)	1487.68
400K	390 (14%)	0.499 (0%)	0.046 (0%)	5.82 (0%)	2342 (85%)	4.54 (0%)	11.48 (0%)	2754.39
600K	525 (13%)	0.783 (0%)	0.062 (0%)	8.45 (0%)	3547 (86%)	5.93 (0%)	14.41 (0%)	4101.64
800K	660 (12%)	1.107 (0%)	0.078 (0%)	11.11 (0%)	4672 (87%)	7.95 (0%)	18.79 (0%)	5371.04
1000K	795 (12%)	1.157 (0%)	0.124 (0%)	13.81 (0%)	5884 (87%)	9.21 (0%)	23.86 (0%)	6727.16
1200K	920 (12%)	1.294 (0%)	0.134 (0%)	16.25 (0%)	6813 (87%)	11.57 (0%)	28.74 (0%)	7790.99
1400K	1085 (12%)	1.451 (0%)	0.156 (0%)	19.35 (0%)	8040 (87%)	12.45 (0%)	32.52 (0%)	9190.93
1600K	1225 (12%)	1.731 (0%)	0.171 (0%)	21.53 (0%)	8939 (87%)	15.45 (0%)	36.87 (0%)	10239.75
1800K	1355 (12%)	1.903 (0%)	0.202 (0%)	24.87 (0%)	9860 (87%)	18.74 (0%)	39.64 (0%)	11300.36
2000K	1480 (12%)	2.106 (0%)	0.218 (0%)	26.83 (0%)	10981 (87%)	21.36 (0%)	44.06 (0%)	12555.57

It is important to clarify at this point the reason for showing the scalability results of only 7 (out of 9) steps of our proposed methodology. Although our methodology has 9 steps in total, 2 of these steps (steps 7 and 9; refer to Chapter 3 for details) do not require computational processing because they are manual knowledge exploration steps.

For example, step 7 of our proposed methodology is a cube exploration step in which users utilize the output of the first 6 steps to discover interesting regions in data cubes via ranked paths. Similarly, step 9 of the methodology allows users to explore the diverse rules generated through the automatically created multidimensional schema. It is also evident from the results present in Table 7.1 that multidimensional schema generation is the most time consuming step of our proposed methodology and this motivated a more detailed scalability study for this step when compared to the other steps of our methodology, such as ranking and multidimensional scaling. Figure 7.23 shows the overall processing time of our proposed methodology with respect to different data sizes.

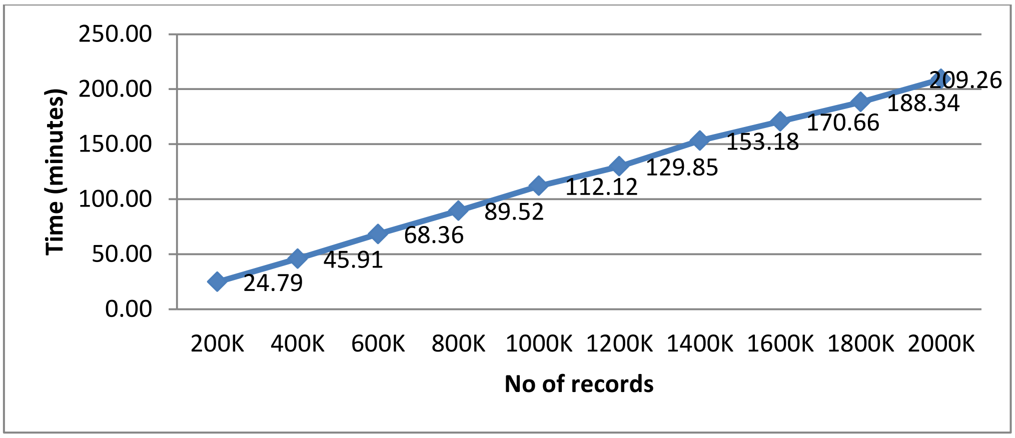
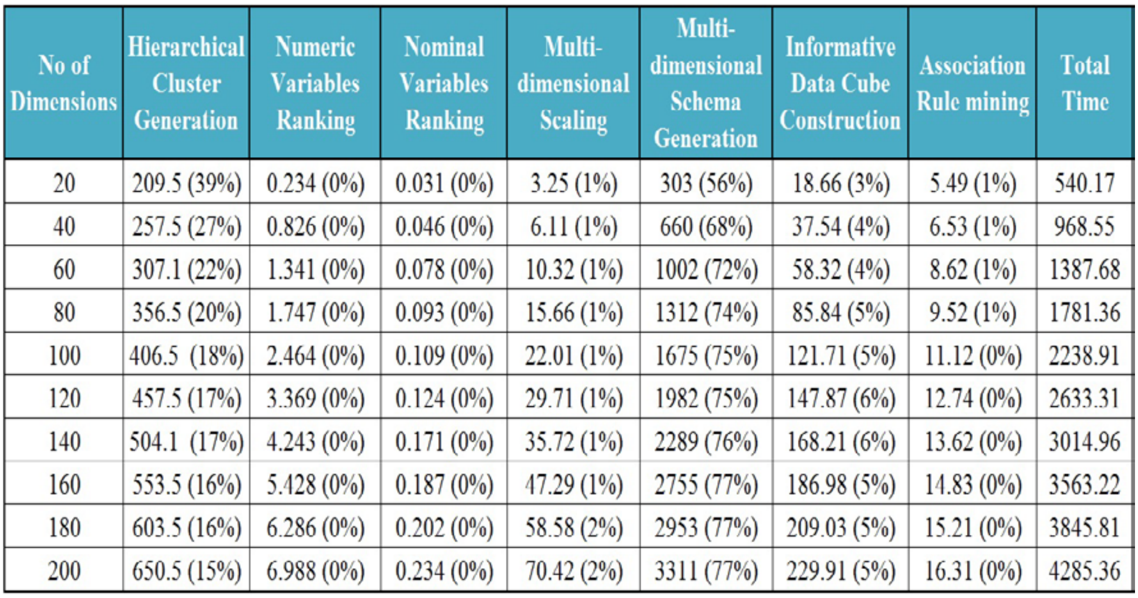


Figure 7.23: Scalability of proposed methodology with respect to different data sizes

We have chosen a larger unit (minutes) instead of (seconds) to represent the processing time on (y-axis). It is clear from Figure 7.23 that our methodology scales well with large data sizes and it only requires 209.26 minutes to discover interesting cube regions and to find diverse association rules from a dataset of size 2000K.

After establishing the scalability of our methodology with large data sizes, we now present the results obtained through the experiments performed with high dimensional data. Table 7.2 summarizes the results of the experiments with variable number of dimensions by fixing the number of records to 10K. Again, it can be seen that the multidimensional schema generation is the most time consuming step. The higher the dimensionality of data the more adverse effect it brings on the processing time.

Table 7.2: Scalability of proposed methodology w.r.t variable dimensions



No of Dimensions	Hierarchical Cluster Generation	Numeric Variables Ranking	Nominal Variables Ranking	Multi-dimensional Scaling	Multi-dimensional Schema Generation	Informative Data Cube Construction	Association Rule mining	Total Time
20	209.5 (39%)	0.234 (0%)	0.031 (0%)	3.25 (1%)	303 (56%)	18.66 (3%)	5.49 (1%)	540.17
40	257.5 (27%)	0.826 (0%)	0.046 (0%)	6.11 (1%)	660 (68%)	37.54 (4%)	6.53 (1%)	968.55
60	307.1 (22%)	1.341 (0%)	0.078 (0%)	10.32 (1%)	1002 (72%)	58.32 (4%)	8.62 (1%)	1387.68
80	356.5 (20%)	1.747 (0%)	0.093 (0%)	15.66 (1%)	1312 (74%)	85.84 (5%)	9.52 (1%)	1781.36
100	406.5 (18%)	2.464 (0%)	0.109 (0%)	22.01 (1%)	1675 (75%)	121.71 (5%)	11.12 (0%)	2238.91
120	457.5 (17%)	3.369 (0%)	0.124 (0%)	29.71 (1%)	1982 (75%)	147.87 (6%)	12.74 (0%)	2633.31
140	504.1 (17%)	4.243 (0%)	0.171 (0%)	35.72 (1%)	2289 (76%)	168.21 (6%)	13.62 (0%)	3014.96
160	553.5 (16%)	5.428 (0%)	0.187 (0%)	47.29 (1%)	2755 (77%)	186.98 (5%)	14.83 (0%)	3563.22
180	603.5 (16%)	6.286 (0%)	0.202 (0%)	58.58 (2%)	2953 (77%)	209.03 (5%)	15.21 (0%)	3845.81
200	650.5 (15%)	6.988 (0%)	0.234 (0%)	70.42 (2%)	3311 (77%)	229.91 (5%)	16.31 (0%)	4285.36

Figure 7.24 depicts the total processing time with respect to different dimensions. Again, we have chosen a larger unit (minutes) instead of (seconds) to represent the processing time on the y-axis.

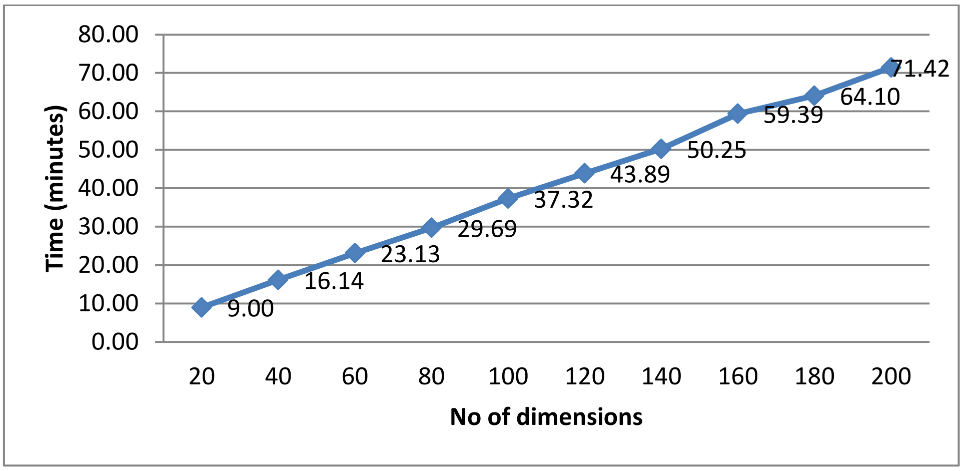


Figure 7.24: Scalability of proposed methodology with respect to different dimensions

Similar to the results obtained for each of the steps, Figure 7.24 confirms that our total processing time scales well with dimensionality. It is not only scalable but also very efficient as it only requires 71.42 minutes to discover the most interesting regions in data cube and to find the diverse association rules from a dataset having 200 dimensions.

Summary

In this chapter, we presented the results of our scalability study performed on synthetic datasets. Considering the processing time as the main performance indicator, this scalability study aimed at the analysis of the effect of two main variables: (i) the size of data and (ii) the dimensionality of data. Firstly, we presented the scalability results of the individual steps of our methodology and secondly we showed that the overall methodology scales well with both data size and dimensionality. We observed that most of the steps of our methodology are robust and their processing time is very low. However, majority of the steps involved in our methodology are more sensitive to high dimensionality compared to data size. We also identified that the multidimensional schema generation step is the most time consuming step in the execution of our methodology.

In order to study schema generation step in detail, we developed a prototype for the automatic generation of multidimensional schema. In our first set of experiments, we analyzed the impact of our first variable which is data size (number of instances) on computation time of the schema generation process. We tested the variability of the processing time for size up to 2000K records by fixing the number of dimensions. The results of our experiments showed that our schema generation method scales well with number of records as the processing time increased linearly.

In our second experiment, we tested the influence of high dimensional data by varying the number of dimensions and fixing the number of records to 10K. We tested up to 200 dimensions and found a linear increase in the schema generation time. Moreover, we observed an interesting issue, called the curse of dimensionality, that increasing number of dimensions has a direct and influential effect on schema generation time and consequently on the performance of our proposed method. The results presented in Figure 7.16 highlighted that with the addition of a single dimension the processing time increases rapidly. Furthermore, we observe a trend (depicted in Figure 7.17) that the impact of adding 3 dimensions is approximately equivalent to adding 10,000 data records. It showed that dimensionality is a bigger issue than data size and this underscores the need for filtering dimensions using methods such as information theoretic measures such as information gain and Eigen value analysis employed in this research.

Finally, we tested the effect of dimensional cardinality on processing time. We varied the number of hierarchical levels to be created by the schema up to 10 levels and again found that increasing cardinality also has a significant impact on schema generation time. We observed that with the addition of each level in the dimensional hierarchy the computation time increases by approximately 20 seconds. However, similar to the previous results, we found that our method also scales well with cardinality. The results of these experiments show that overall our proposed methodology for discovering interesting cube regions and finding diverse association rules scales well with both data volume and dimensionality.

Chapter 8

Thesis Conclusions and Future Work

This thesis has explored the integrated use of data mining, data warehousing and machine learning techniques to enhance knowledge discovery from real world datasets from different application domains. We particularly focused on discovering interesting and diverse knowledge from those application scenarios where there is either no or very limited domain knowledge is available to the analysts. Our case studies demonstrated that useful and interesting knowledge can easily be discovered without excessive reliance on specialized domain knowledge. In this concluding chapter, we summarize and evaluate the main outcomes of this research, discuss the issues that remain open and make suggestions for possible directions of future research.

8.1 Research Achievements

In this section, we outline the research achievements and discuss the extent to which the major objectives of our research were realised. The primary objective of this research was to develop a knowledge discovery methodology that utilizes machine learning and statistical methods to provide automated and data-driven approach in multidimensional schema design and analysis. In terms of design, we identified that fact that very limited research has been conducted in leveraging data mining techniques in the design of data warehouses or multidimensional schema (Sapia, Höfling et al. 1999; Zubcoff, Pardillo et al. 2007; Pardillo, Mazón et al. 2008; Pardillo and Mazón 2010; Usman, Asghar et al. 2010) and these techniques have the potential to offer support in multidimensional schema design process.

In this research, we integrated hierarchical clustering technique with multidimensional scaling in order to support the automated multidimensional schema design. The use of hierarchical data clustering was useful as it highlighted the fact that relationships between numeric and nominal variables changed significantly depending on the granularity of the data. We observed sharp differences in patterns at different levels of data abstraction. For instance, the variables that were ranked higher based on statistical methods such as Principal Component Analysis (PCA) and Multiple Correspondence Analysis (MCA) at one level, appeared to be lowly ranked in an immediate lower level of data hierarchy. We demonstrated that classical statistical methods for data analysis such as PCA and MCA can be successfully used in conjunction with hierarchical clustering to uncover useful information implicit in large multidimensional datasets.

Moreover, the use of hierarchical clustering along with multidimensional scaling technique allowed us to identify and group, at each level of data abstraction, the semantically related values present in each nominal variable. This integrated use of hierarchical clustering and multidimensional scaling revealed the most significant interrelationships that existed between numeric and nominal variables, thus enabled analysts with pathways to explore data in an OLAP manner. The three case studies presented evidence that the interrelationships between nominal and numeric variables are not only significant from both the application and statistical perspectives but could also be discovered without excessive reliance on domain knowledge.

Although our integrated use of techniques showed noteworthy achievements, it can be argued that our methodology deals separately with the numeric and nominal variables. We justified the separate application of these techniques in our methodology (See Chapter 3 for details) and would like to emphasize that none of the existing techniques for mixed data analysis in literature provides the benefits which the integrated use of separate mature techniques provide. For example, a number of proposals appear in literature for clustering mixed data (Luo, Kong et al. 2006; Ahmad and Dey 2007; Hsu, Chen et al. 2007; Hsu and Chen 2007; Tang and Mao 2007; Chatzis 2011; Ji, Han et al. 2012) but the scope of these proposals is limited only to clustering accuracy and quality. None of these proposals provide explicit means to discover further knowledge at different levels of the cluster hierarchy.

Analysts often require more knowledge from large and high dimensional datasets as opposed to merely analyzing a group of values clustered together at various data abstraction levels. Furthermore, these proposals lack the power of analysing semantics among nominal values which could easily be identified through the use of multidimensional scaling technique. Similarly, statistical techniques, though mature cannot be used on their own to promote knowledge discovery. For example, an analyst can apply the multidimensional scaling technique to identify the spread of nominal values on a scale but grouping mechanisms need to be applied so that commonalities of objects within a group can be identified. This also opens up the issue that which set of techniques from these diverse domains of machine learning and statistics work coherently with each other in order to satisfy the ever growing need of analysts and decision makers. Our methodology is a step forward towards the solution of this issue and provides a set of techniques from the machine learning and statistical domains that coherently work together to facilitate the ultimate goal of knowledge discovery from large datasets.

Another objective of this research was to support the generation of cubes of interest at different levels of data abstraction and study the effect of abstraction level on information content. From the literature review on cube design (presented in Chapter 2) we identified that there remains a need for automated support in the design of informative data cubes, especially in domains containing high dimensional data. High dimensional and high volume datasets present significant challenges to analysts in terms of identifying data cubes of interest. Our proposed methodology assisted the warehouse designers and analysts in identifying dimensions and measures of interest in these types of environments. Again, the integrated use of hierarchical clustering and multidimensional scaling made it possible for human analysts to extract knowledge hidden at multiple levels of data abstraction by exploring through ranked pathways provided by our methodology. For instance, in our first case study conducted on the *Automobile* dataset, in data cube C1, when explored through the highly ranked dimensions (Make, No-of-Cylinders and Engine-type) encapsulated patterns whereby variables of interest (fact variables) have much greater deviations from their means as compared to exploration via lowly ranked dimensions. The ranked paths suggested by the methodology assisted in quickly identifying those cells in a data cube that have the highest deviations from the mean. This is typically the information sought by OLAP analysts who are interested in quickly finding regions among the large search space of data cubes that show large deviations from the norm.

Similarly, the third case study conducted on the *Forest Cover Type* dataset also showed that, through the use of the ranked paths, analysts can pinpoint particular wilderness areas (4 areas) and soils (40 types) which have the highest deviations from the mean without undertaking the gruelling task of analyzing a large number of paths available for exploration. The knowledge discovered through the ranked paths was not only aligned to commonly known domain knowledge but also provided precise information on navigation of data cubes.

In particular, it was explicit that the elevation for *Cache_la_pourde* area is low with high slope range but its implicit relationship with the soil types responsible for lowest elevation and highest slope range was not obvious. This implicit information was easily and efficiently revealed through the top ranked paths suggested by the methodology. It was identified that Soil Type 1 from *Cathedral* family and Soil Type 4 from *Vanet* series of soils are the main types which have lowest elevation and highest slopes in the overall dataset. More interestingly, when these two soil types were further examined and their characteristics were compared it was observed that their wilderness areas, elevation, slopes and mean air temperature have significant differences.

Although the ranked pathways suggested by our methodology provided pathways in quickly and efficiently exploring large data cubes there remains a limitation to our approach. The suggested pathways are unable to support customized OLAP queries. For example, the top *k* highest selling items might not be items which have the highest deviation from mean in profit terms. Therefore, our ranked paths do not provide answers to such top *k* type queries because we calculate the absolute difference from mean and it is not necessarily the case that all items which have extreme deviation from mean in profit terms are the top selling items. This situation arises when the *sales quantity* fact variable is not included in the list of highly ranked fact variables, as identified by our methodology. To overcome this problem, the analyst will need to use his/her specialized domain knowledge to augment the list of highly ranked fact variables with the sales quantity variable.

The final objective was to integrate the design and analysis processes in order to extend the capabilities of traditional data exploration methods such as OLAP to discover diverse and meaningful association rules from multidimensional cubes. The main tools used in multidimensional analysis in a data warehousing environment consist of various data aggregation and exploratory techniques that form part of the OLAP suite of methods. While traditional OLAP methods are excellent tools for exploratory data analysis their capability is limited as far as detecting hidden associations between items resident in a large data warehouse.

In order to achieve this objective we generated association rules using the most informative dimensions retained after filtering via information theoretic measures such as Entropy and Information Gain. In the three case studies, the rules generated with our proposed methodology were shown to be more diverse on the *Rae*, *CON* and *Hill* objective diversity measures when compared against an approach that simply generated rules on flat data. Diversity is a common factor for measuring the interestingness of aggregated/summarized data. A diverse rule is interesting because in the absence of any domain knowledge, analysts commonly assume that the uniform distribution holds in summarized data (Geng and Hamilton 2006). According to this reasoning, the more diverse the rule is the more interesting it is. To date 19 diversity measures have been proposed in the literature, 16 of them were proposed by (Hilderman and Hamilton 2001) and the other 3 by (Zbidi, Faiz et al. 2006). However, these measures were used to evaluate the interestingness of database summaries. Summaries are the compact descriptions of raw data at different levels of data abstraction. None of the existing research has utilized diversity measures to evaluate the interestingness of classification or association rules (Geng and Hamilton 2006). To the best of our knowledge, we are the first to evaluate the interestingness of association rules using diversity measures.

A number of other interestingness measures have been proposed in the literature to measure the interestingness of patterns such as novelty, generality, surprisingness, conciseness, peculiarity etc. However, our proposed methodology permits the evaluation of interestingness only through the diversity criterion. The aforementioned interestingness measures are all useful in different scenarios and are sometimes correlated with, rather independent of one another. For example, conciseness often coincides with generality and peculiarity may coincide with novelty. Conversely, some of these measures may have absolute independence, depending on the context or application domain. For example, diversity may have no correlation with novelty. Knowledge which may appear to be diverse does necessarily mean that it is novel. It is extremely challenging to find truly novel patterns/rules from data, thus explaining why novelty has received the least attention in the research community (Geng and Hamilton 2006). Novelty requires the knowledge discovery systems to model everything that the users know explicitly in order to detect what is unknown or novel. In general, it is not feasible for users to specify all knowledge quickly and consistently in a system. Therefore, the proposed methodology utilized probability-based objective measures for evaluation since such measures do not involve modelling user expectations in advance which is an extremely difficult task in areas where domain knowledge is either limited or not available.

However, the probability based measures neither takes into account the context of the application domain nor the goals and background knowledge of the user. The objective measures only involve the probabilities of the antecedent of the rule, the consequent of the rule, or both and to represent the generality, correlation and reliability between the antecedent and consequent of the rules (Geng and Hamilton 2006). Nonetheless, novelty remains a major criterion in the evaluation of interesting rules and discovery of diverse rules have the potential to complement the identification of novel rules. For example, the rules discovered through diversity measures could be analyzed by the domain users as a first step and only those diverse rules which appear to be novel according to user's analysis could be identified and retained as a second step. As it is widely accepted that no single measure is superior to others or suitable for all applications, thus the aforementioned two step approach is a reasonable way to discover diverse and novel rules. However, the determination of rule novelty was out of the scope of this research.

Our case studies revealed association rules generated through the use of the schema are more compact, easier to understand and convey more information to an end user than a plethora of rules that cover each and every combination of values of the variables involved in rule mining process. For instance, the rules generated with schema suggested a group of distinct values for each input variable as opposed to a large number of rules containing a single value for each input variable.

It is clear from the discussion presented in this section that our proposed methodology achieved the following main objectives of this research.

- Provided automated and data-driven support for the design and construction of multidimensional schema.
- Generated cubes of interest at different levels of data abstraction and identified the effects of data abstraction levels on information content.
- Identified at each level of data abstraction, the most significant interrelationships that exist between dimensions (nominal variables) and facts (numeric variables)
- Discovered diverse and meaningful association rules from multidimensional cube structure at various levels of data abstraction.

8.2 Benefits of the automated approach over the traditional domain based approach

In this section we give a general discussion on the application of the proposed automated methodology versus application of the traditional manual method of data analysis. 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 useful knowledge.

Without the presence of this automated methodology, data warehouse designers have to rely heavily on domain knowledge to model dimensions and dimensional hierarchies. Moreover, the manually designed schema may be unable to highlight the natural grouping of values which are interesting and worth exploring using an OLAP tool. For instance, *Country* could be taken as a dimension by a human data warehouse designer and one meaningful way of grouping countries is to assign countries based on geographical regions such as *Asia*, *Europe*, and *Africa*. However, with the application of the automated method as proposed by this research can result in the *Country* dimension taking on completely different semantics. For example, the countries *Australia*, *Mexico* and *Spain* were grouped together by Gross Domestic Product (GDP) in the underlying data. This gives a non-traditional, yet, data semantic driven scheme for dimensional design that may not be apparent to data warehouse designers. Other similar examples were also discussed in detail in the three case studies presented in this thesis.

We believe that the proposed methodology 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. Thus knowledge discovery requires not just domain expertise but also the use of automated aids that provide additional insights through the use of data driven methods.

8.3 Conclusions

The research represented in this thesis was motivated by the observation that integrated use of data mining and warehousing techniques are gaining rapid momentum as the core technology for knowledge discovery from large datasets in the business world and beyond. On one hand, the emergence of novel application domains with minimum domain knowledge availability motivates this integrated approach. On the other hand, it is not feasible to simply thread in the result of one technique into the other as the seamless integration requirement goes well beyond a simple merger of techniques. For example, if we take the integration of hierarchical clustering and PCA for ranking the numeric variables, a simple combination of the two techniques will not work as it is not possible to simply apply PCA on the current generation of clusters in order to rank them. The raw component scores obtained through PCA does not indicate the importance of a variable in a given cluster. It is the *difference* in component scores between consecutive levels in the hierarchy that reveals the relative importance or ranking of the variables.

Likewise, MCA maps the nominal variables onto principal axes, but simply using this mapping alone is not sufficient to understand the semantics amongst the potentially large number of values present in the nominal variables. In order to make sense out of such large nominal values, a grouping mechanism is required so that those values that are of local proximity to each other and share properties in common are grouped together. Again, it is clear that it is not just a case of simply applying the different techniques in some sequential order without doing any intelligent pre-processing beforehand. Such pre-processing plays a vital role in discovering useful information.

Thus for such integration to occur in a seamless manner a suitable methodology is required for coupling data mining and machine learning techniques into one coherent mechanism for supporting data warehouse design. The work presented an attempt to reduce the gap between the capabilities of integrated systems and the design requirements imposed by emerging knowledge discovery applications.

In the introductory chapter, we explained the general complexity of integrating data mining and machine learning techniques by pointing out that most of the integrated use of techniques for knowledge discovery had the following main assumptions. Firstly, prior work assumed that data analysts could identify set of informative dimension and data cubes based on their domain knowledge. Unfortunately, situations exist where such assumptions are not valid. These include high dimensional datasets where it is very difficult or even impossible to predetermine which dimensions and which cubes are the most informative. Secondly, it restricts the application of prior methodologies to only those domains where such domain knowledge is available. However, a knowledge discovery system should be able to work in ill-defined domains (Nkambou, Fournier-Viger et al. 2011) and other domains where no background knowledge is available (Zhong, Dong et al. 2001). Thirdly, majority of the work done in the past focused on the discovery of knowledge from pre-existing data warehouses whereas very limited work exists in utilizing mining techniques to design a multidimensional model that supports knowledge discovery.

The overall power of such integrated methodologies for knowledge discovery is determined by the interplay between the design and analysis steps. Thus, any improvement in design will in turn lead to enhanced knowledge discovery.

8.4 Future Work

The results presented in this thesis improve the knowledge discovery process by the fusion of data mining, data warehousing and machine learning technologies. The proposed methodology addresses the requirement for enriching the knowledge discovery a step closer by integrating the stand-alone analysis methods and overcoming their individual limitations of comprehensive analysis. However, the set of methods and techniques proposed by our methodology is by no means exhaustive.

The use of some potentially useful methods was not explored due to the limited timeframe of our work whereas others were excluded deliberately as they did not fit or would have expanded the scope of our resulting methodology. The body of work in this research can be extended by connecting it to the other related research areas, such as Online Analytical Mining (OLAM), real-time or temporal data warehousing, or by extending the current methods and their applications of this research in diverse application domains. In the latter category, a number of promising directions for future research can be identified.

Hierarchical Clustering

We have not considered any other clustering methods except agglomerative hierarchical clustering. It would be interesting to apply and compare other hierarchal clustering methods, especially methods which have a tendency to outperform traditional agglomerative clustering in terms of clustering accuracy and performance such as *constrained agglomerative algorithms* (Zhao and Karypis 2002) and *Dynamically Growing Self-Organizing Tree (DGSOT)* algorithm (Khan, Awad et al. 2007).

Variable Ranking

We have only utilized Principal Component Analysis (PCA) technique for ranking the numeric variables. It would be interesting to explore the use of alternative ranking methods for numeric variables such as *Fisher score* (Tsuda, Kawanabe et al. 2002), *Linear Support Vector Machine* (Tong and Chang 2001), *Discriminant Analysis* (Klecka 1980) etc. which have also proved to be effective in various application domains. Likewise, we have only considered Multiple Correspondence Analysis (MCA) and Information Gain measure to rank nominal attributes. It would also be useful to explore the use of other methods such as *Factor Analysis* (Lawley and Maxwell 1971) and *Categorical Principal Components* (Linting, Meulman et al. 2007) which is a non-linear approach that supports the use of nominal variables and is capable of handling and discovering non-linear relationships between variables. Use of these methods can extend the applicability of our methodology in those datasets where non-linear relationships exist which could not be identified through linear PCA. Another possible direction for future work would be to find an automatic method for detecting that certain fact variables could in fact be used to define dimensions. For instance, the Age variable is inherently numeric but can be used as a dimension variable by discretizing its value into distinct age ranges. Here the challenge would be to find the optimal partitioning strategy for discretization. Apart from the commonly used equal-width and equal-frequency strategies, newly proposed approaches based on entropy (Han and Kamber 2006) have been shown to outperform the more naïve approaches and these are worth exploring.

Visual exploration of diverse association rules

Association rule mining algorithms typically give a textual list showing simple IF-THEN statements for the association rules and very little research has been conducted in the area of developing visualizers for effective visualization of association rules. Our work can be extended in terms of representing the diverse association rules using some form of visual representation such as *rule-focusing methodology* (Blanchard, Guillet et al. 2007) which is an interactive methodology for the visual post-processing of association rules. It allows users to explore large sets of rules freely by focusing his/her attention on limited subsets. This approach relies on rule interestingness measures, on a visual representation, and on interactive navigation among the rules. A visual representation can improve user understanding of the level of diversity manifested by different sets of rules. For example, the framework proposed by (Liu, Hsu et al. 1999) could be adopted. The proposed framework has an interestingness analysis component and a visualization component. We could use our diversity measures to evaluate the rule interestingness and the visualization component could then be used to visually explore the diverse rules produced. The key strength of the Liu et. al visualization component is that from a single screen, the user is able to obtain a global, yet detailed picture of various interesting aspects of the discovered rules. Enhanced with color effects, the user can easily and quickly focus his/her attention on the more diverse rules. This powerful component can be integrated with our methodology in the post-processing of rules.

Support for deeper dimensional levels for richer information

In this research the dimensions that we design support two level hierarchies, with the first level consisting of groups and the second consisting of individual values within each group. A promising direction for future research would be to explore the use of deeper hierarchies, as it was shown in the case studies that the dimensional structure was responsible for capturing rich information. Our grouping algorithm (Algorithm 3-presented in Chapter 3) can be extended in recursive manner to accommodate multiple levels in the dimensional hierarchy. For example, the current algorithm automatically calculates a threshold for grouping the values at first level of dimensional hierarchy; we can keep decrementing this threshold until the groups formed at the first level split and divide into multiple groups at a lower level of hierarchy. The level in dimensional hierarchy where the division occurs could be labelled as second level and the same procedure could be repeated for obtaining multiple levels in the hierarchy. Conversely, the threshold can also be incremented in a nesting manner. For instance, we can keep incrementing the automatically calculated threshold until the groups formed at the first level merge together at a higher level of dimensional hierarchy. The current level of grouping achieved through Algorithm 3 could be called as a base level (instead of first level) and from this base level, both deeper (divisive manner) and higher (nesting manner) levels of dimensional hierarchies could be obtained using the proposed extensions that we have just described.

Wider applicability and interesting application domains

We also intend to test the methodology with complex datasets from the biological, medical and engineering domains to further test the performance and scalability of the proposed methodology. In addition to the above mentioned directions for future work, we would also like to highlight two other interesting applications of our work. Firstly, it would be productive to extend the methodology to handle data streams. In the context of data stream research, taming the multidimensionality of real-life data streams in order to efficiently support *OLAP analysis/mining tasks* is a critical challenge (Cuzzocrea 2009). It would be very useful to adapt our proposed methodology to stream data and automatically build schema in order to adapt the process of discovery of cubes of interest in the face of concept changes that occur in data stream environments.

Secondly, an issue not addressed at all in this research is privacy preserved data mining. The concept of diversity has been used in privacy preserved data mining in a completely different manner to our usage of it. (Machanavajjhala, Kifer et al. 2007) proposed a powerful privacy criterion called *l*-diversity and showed that the traditional *k-annoymized* datasets have some subtle and severe privacy problems. First, an attacker can discover the values of sensitive attributes when there is little diversity in those sensitive attributes and secondly, the *k-anonymity* strategy does not guarantee privacy from attackers possessing background knowledge. In order to avoid such attacks the diversity based proposal inserts a diverse set of values for each tuple in sensitive datasets. This insertion of such diverse sets of values makes it impossible for the attackers to violate privacy based on their background knowledge. Similarly, we see a possible future direction for our current research in the data privacy domain. The association rules produced through our methodology can be leveraged to introduce diversity in the datasets to avoid attackers identifying private information based on background knowledge.

References

Abdelbaki, W., R. Ben Messaoud, et al. (2012). "A Neural-Based Approach for Extending OLAP to Prediction." Data Warehousing and Knowledge Discovery: 117-129.

Abdi, H. and D. Valentin (2007). "Multiple correspondence analysis." Encyclopedia of measurement and statistics: 651-657.

Agrawal, R., T. Imieliński, et al. (1993). Mining association rules between sets of items in large databases, ACM SIGMOD Record. **22:** 207-216.

Ahmad, A. and L. Dey (2007). "A k-mean clustering algorithm for mixed numeric and categorical data." Data & Knowledge Engineering **63**(2): 503-527.

Asuncion, A. and D. J. Newman (2010). UCI machine learning repository [http://archive.ics.uci.edu/ml] Irvine, CA: University of California, School of Information and Computer Science.

Becue-Bertaut, M. and J. Pages (2008). "Multiple factor analysis and clustering of a mixture of quantitative, categorical and frequency data." Computational Statistics & Data Analysis **52**(6): 3255-3268.

Ben Messaoud, R., S. Loudcher Rabaseda, et al. (2007). "OLEMAR: An Online Environment for Mining Association Rules in Multidimensional Data."Advances in Data Warehousing and Mining, IGI Global, 2, 1-35.

Bernstein, A., F. Provost, et al. (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**(4): 503-518.

Blackard, J. A., D. Dean, et al. (1998). The forest covertype dataset http://archive.ics.uci.edu/ml/datasets/Covertype.

Blanchard, J., F. Guillet, et al. (2007). "Interactive visual exploration of association rules with rule-focusing methodology." Knowledge and Information Systems **13**(1): 43-75.

Borg, I. and P. J. F. Groenen (2005). Modern multidimensional scaling: Theory and applications, Springer Series in Statistics Second Edition.

Chatzis, S. P. (2011). "A fuzzy c-means-type algorithm for clustering of data with mixed numeric and categorical attributes employing a probabilistic dissimilarity functional." Expert Systems with Applications **38**(7): 8684–8689.

Chung, S. M. and M. Mangamuri (2005). "Mining association rules from the star schema on a parallel NCR teradata database system."International Conference of Information Technology: Coding and Computing (ITCC'05), Nevada.

Cordes, D., V. Haughton, et al. (2002). "Hierarchical clustering to measure connectivity in fMRI resting-state data." Magnetic resonance imaging **20**(4): 305-317.

Cox, M. A. A. and T. F. Cox (2008). "Multidimensional scaling." Handbook of data visualization: 315-347.

Cuzzocrea, A. (2009). CAMS: OLAPing Multidimensional Data Streams Efficiently. Data Warehousing and Knowledge Discovery. T. Pedersen, M. Mohania and A. Tjoa, Springer Berlin Heidelberg. **5691:** 48-62.

D'Agostino, R. B. and M. A. Stephens (1986). Goodness of Fit Techniques, M. Dekker, New York.

Dori, D., R. Feldman, et al. (2008). "From conceptual models to schemata: An object-process-based data warehouse construction method." Information Systems **33**(6): 567-593.

Doring, C., C. Borgelt, et al. (2004). Fuzzy clustering of quantitative and qualitative data. IEEE Annual Meeting of the Fuzzy Information Processing NAFIPS'04.

Fayyad, U., G. Piatetsky-Shapiro, et al. (1996). "The KDD process for extracting useful knowledge from volumes of data." Communications of the ACM **39**(11): 27-34.

Fred, A. L. N. and A. K. Jain (2005). "Combining multiple clusterings using evidence accumulation." Pattern Analysis and Machine Intelligence, IEEE Transactions on **27**(6): 835-850.

Geng, L. and H. J. Hamilton (2006). "Interestingness measures for data mining: A survey." ACM Computing Surveys (CSUR) **38**(3): 9.

Goil, S. and A. Choudhary (2001). "PARSIMONY: An infrastructure for parallel multidimensional analysis and data mining." Journal of parallel and distributed computing **61**(3): 285-321.

Goodall, D. W. (1966). "A new similarity index based on probability." Biometrics **22**(4): 882-907.

Greenacre, M. J. (1991). "Interpreting multiple correspondence analysis." Applied Stochastic Models and Data Analysis **7**(2): 195-210.

Hahn, K., C. Sapia, et al. (2000). Automatically generating OLAP schemata from conceptual graphical models. Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP, ACM.

Han, J. (1998). "Towards on-line analytical mining in large databases." ACM Sigmod Record **27**(1): 97-107.

Han, J. and M. Kamber (2006). Data mining: concepts and techniques, Morgan Kaufmann.

Hilderman, R. J. and H. J. Hamilton (2001). "Knowledge discovery and measures of interest."Kluwer Academic, Boston, MA.

Hsu, C.-C., C.-L. Chen, et al. (2007). "Hierarchical clustering of mixed data based on distance hierarchy." Information Sciences **177**(20): 4474-4492.

Hsu, C.-C. and Y.-C. Chen (2007). "Mining of mixed data with application to catalog marketing." Expert Systems with Applications **32**(1): 12-23.

Hsu, C. C., C. L. Chen, et al. (2007). "Hierarchical clustering of mixed data based on distance hierarchy." Information Sciences **177**(20): 4474-4492.

Hsu, C. C. and Y. P. Huang (2008). "Incremental clustering of mixed data based on distance hierarchy." Expert Systems with Applications **35**(3): 1177-1185.

Huang, Z. (1997). Clustering large data sets with mixed numeric and categorical values. Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining, World Scientific, Singapore**:** 21–34.

Inselberg, A. and B. Dimsdale (1991). Parallel Coordinates. Human-Machine Interactive Systems. A. Klinger, Springer US**:** 199-233.

Ji, J., X. Han, et al. (2012). "A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data." Knowledge-Based Systems **30**: 129-135.

Jinwook, S. and B. Shneiderman (2002). "Interactively exploring hierarchical clustering results [gene identification]." Computer **35**(7): 80-86.

Jolliffe, I. T. (2002). Principal Component Analysis, Springer New York.

Kamber, M., J. Han, et al. (1997). Metarule-guided mining of multi-dimensional association rules using data cubes. In Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97), 207-210.

Kaya, M. and R. Alhajj (2003). "Integrating fuzziness with OLAP association rules mining." Machine Learning and Data Mining in Pattern Recognition: 65-81.

Khan, L., M. Awad, et al. (2007). "A new intrusion detection system using support vector machines and hierarchical clustering." The VLDB Journal **16**(4): 507-521.

Klecka, W. R. (1980). Discriminant analysis, SAGE Publications, Incorporated.

Koh, Y. S., R. Pears, et al. (2011). "Automatic Item Weight Generation for Pattern Mining and its Application." International Journal of Data Warehousing and Mining (IJDWM) **7**(3): 30-49.

Kohavi, R. and B. Becker. (1996). "Adult dataset." from http://archive.ics.uci.edu/ml/datasets/Adult.

Kosara, R., F. Bendix, et al. (2006). "Parallel Sets: interactive exploration and visual analysis of categorical data." Visualization and Computer Graphics, IEEE Transactions on **12**(4): 558-568.

Kumar, N., A. Gangopadhyay, et al. (2008). Navigation Rules for Exploring Large Multidimensional Data Cubes. Data Warehousing and Mining: Concepts, Methodologies, Tools, and Applications, IGI Global**:** 1334-1354.

Lawley, D. N. and A. E. Maxwell (1971). Factor analysis as a statistical method, Butterworths London.

Le Roux, B. and H. Rouanet (2009). Multiple correspondence analysis, Sage Publications, Inc. 2009.

Li, C. and G. Biswas (2002). "Unsupervised learning with mixed numeric and nominal data." IEEE Transactions on Knowledge and Data Engineering **14**(4): 673-690.

Linting, M., J. J. Meulman, et al. (2007). "Nonlinear principal components analysis: introduction and application." Psychological methods **12**(3): 336.

Liu, B., W. Hsu, et al. (1999). Visually Aided Exploration of Interesting Association Rules. Methodologies for Knowledge Discovery and Data Mining. N. Zhong and L. Zhou, Springer Berlin Heidelberg. **1574:** 380-389.

Liu, Z. and M. Guo (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 IEEE. **3:** 146-151

Luo, H., F. Kong, et al. (2006). "Clustering mixed data based on evidence accumulation." Advanced Data Mining and Applications **4093**: 348-355.

Machanavajjhala, A., D. Kifer, et al. (2007). "l-diversity: Privacy beyond k-anonymity." ACM Transactions on Knowledge Discovery from Data (TKDD) **1**(1): 3.

MacLennan, J., Z. Tang, et al. (2011). Data mining with Microsoft SQL server 2008, Wiley Publishing, Inc. ISBN: 978-0-470-27774-4.

Malinowski, E. and E. Zimányi (2008). "Introduction." Advanced Data Warehouse Design: 1-16.

Mansmann, S. (2009). "Extending the OLAP Technology to Handle Non-Conventional and Complex Data." PhD thesis, University of Konstanz, Germany).

McCane, B. and M. Albert (2008). "Distance functions for categorical and mixed variables." Pattern Recognition Letters **29**(7): 986-993.

Messaoud, R. B., S. L. Rabaséda, et al. (2006). Enhanced mining of association rules from data cubes, *DOLAP '06 Proceedings of the 9th ACM international workshop on Data warehousing and OLAP.* ACM, 11-18.

Milenova, B. L. and M. M. Campos (2002). Clustering large databases with numeric and nominal values using orthogonal projections. In Proceedings of the 29th Conference on Very Large Databases (VLDB), Berlin, Germany.

Milenova, B. L. and M. M. Campos (2002). O-cluster: scalable clustering of large high dimensional data sets. Proceedings of 2002 IEEE International Conference on Data Mining (ICDM'02).

Nestorov, S. and N. Jukic (2003). Ad-hoc association-rule mining within the data warehouse. Proceedings of the 36th IEEE Annual Hawaii International Conference on System Sciences, IL.

Nestorov, S. and N. Jukic (2003). Ad-hoc association-rule mining within the data warehouse, In Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS 2003), 232–242.

Ng, E. K. K., A. W. C. Fu, et al. (2002). Mining Association Rules from Stars, IEEE Computer Society.*In IEEE International Conference on Data Mining (ICDM)*, 322–329.

Nguyen, T. M., A. M. Tjoa, et al. (2005). "Data warehousing and knowledge discovery: A chronological view of research challenges." Data warehousing and knowledge discovery: 530-535.

Nkambou, R., P. Fournier-Viger, et al. (2011). "Learning task models in ill-defined domain using an hybrid knowledge discovery framework." Knowledge-Based Systems **24**(1): 176-185.

NRCS. (2012). "Official Soil Series Description - LEIGHCN series, National Cooperative Soil Survery, USA."

Obradovic, Z. and S. Vucetic (2004). Challenges in Scientific Data Mining: Heterogeneous, Biased, and Large Samples, Technical Report, Center for Information Science and Technology, Temple University, Chapter 1, pp.1-24 Ohmori, T., M. Naruse, et al. (2007). "A New Data Cube for Integrating Data Mining and OLAP."

Ohmori, T., M. Naruse, et al. (2007). A New Data Cube for Integrating Data Mining and OLAP. Data Engineering Workshop, 2007 IEEE 23rd International Conference on. 896-903.

Ordonez, C. and C. Zhibo (2009). "Evaluating Statistical Tests on OLAP Cubes to Compare Degree of Disease." Information Technology in Biomedicine, IEEE Transactions on **13**(5): 756-765.

Palopoli, L., L. Pontieri, et al. (2002). "A novel three-level architecture for large data warehouses* 1." Journal of Systems Architecture **47**(11): 937-958.

Pardillo, J. and J. N. Mazón (2010). "Designing OLAP schemata for data warehouses from conceptual models with MDA." Decision Support Systems **3**(1): 51-62.

Pardillo, J., J. N. Mazón, et al. (2008). "Model-driven metadata for OLAP cubes from the conceptual modelling of data warehouses." Data Warehousing and Knowledge Discovery: 13-22.

Pardillo, J., J. Zubcoff, et al. (2008). Applying MDA to integrate mining techniques into data warehouses: a time series case study. Mining Multiple Information Sources MMIS 08. Las Vegas**:** 47-53.

Peralta, V., A. Marotta, et al. (2003). Towards the Automation of Data Warehouse Design. 15th Conference on Advanced Information Systems Engineering, short paper proceedings (CAISE FORUM)

Pighin, M. and L. Ieronutti (2008). "A Methodology Supporting the Design and Evaluating the Final Quality of Data Warehouses." International Journal of Data Warehousing and Mining (IJDWM) **4**(3): 15-34.

Poole, J. and D. Mellor (2001). Common Warehouse Metamodel: An Introduction to the Standard for Data Warehouse Integration, John Wiley & Sons, Inc.

Psaila, G. and P. L. Lanzi (2000). Hierarchy-based mining of association rules in data warehouses, In Proceedings of the 2000 ACM symposium on Applied computing-Volume 1, 307-312.

Ribeiro, J. T. S. and A. J. M. M. Weijters (2011). Event Cube: Another Perspective on Business Processes. On the Move to Meaningful Internet Systems: OTM 2011. R. Meersman, T. Dillon, P. Herreroet al, Springer Berlin Heidelberg. **7044:** 274-283.

Rosario, G. E., E. A. Rundensteiner, et al. (2004). "Mapping nominal values to numbers for effective visualization." Information Visualization **3**(2): 80-95.

Sapia, C., G. Höfling, et al. (1999). On supporting the data warehouse design by data mining techniques. In Proc. GI-Workshop Data Mining and Data Warehousing, 63.

Sarawagi, S. (2001). "iDiff: Informative Summarization of Differences in Multidimensional Aggregates." Data Mining and Knowledge Discovery **5**(4): 255-276.

Sarawagi, S., R. Agrawal, et al. (1998). Discovery-driven exploration of OLAP data cubes. Advances in Database Technology — EDBT'98, Springer Berlin Heidelberg. **1377:** 168-182.

Schlimmer, J. C. (1985). "Automobile dataset" Retrieved 20 june, 2012, from http://archive.ics.uci.edu/ml/datasets/Automobile.

Seo, J., M. Bakay, et al. (2004). "Interactively optimizing signal-to-noise ratios in expression profiling: project-specific algorithm selection and detection p-value weighting in Affymetrix microarrays." Bioinformatics **20**(16): 2534-2544.

Seo, J., M. Bakay, et al. (2003). Interactive color mosaic and dendrogram displays for signal/noise optimization in microarray data analysis. Proceedings of the International Conference on Multimedia and Expo.461-464

Song, I. Y., R. Khare, et al. (2008). Samstar: An automatic tool for generating star schemas from an entity-relationship diagram. Conceptual Modeling-ER. Springer Berlin Heidelberg, 522-523.

Tang, W. and K. Z. Mao (2007). "Feature selection algorithm for mixed data with both nominal and continuous features." Pattern Recognition Letters **28**(5): 563-571.

Tjioe, H. C. and D. Taniar (2005). "Mining association rules in data warehouses." International Journal of Data Warehousing and Mining **1**(3): 28-62.

Tong, S. and E. Chang (2001). Support vector machine active learning for image retrieval. Proceedings of the ninth ACM international conference on Multimedia. Ottawa, Canada, ACM**:** 107-118.

Tryfona, N., F. Busborg, et al. (1999). starER: A conceptual model for data warehouse design. Proceedings of the 2nd ACM International Workshop on Data Warehousing and OLAP (DOLAP), ACM.

Tryfos, P. (1998). Methods for business analysis and forecasting: text and cases, Wiley.

Tsaipei, W. (2011). "CA-Tree: A Hierarchical Structure for Efficient and Scalable Coassociation-Based Cluster Ensembles." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **41**(3): 686-698.

Tsuda, K., M. Kawanabe, et al. (2002). "Clustering with the Fisher score." Advances in Neural Information Processing Systems **15**: 729-736.

Tuzhilin, A. and G. Adomavicius (2002). Handling very large numbers of association rules in the analysis of microarray data. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM.

Uguz, H. (2011). "A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm." Knowledge-Based Systems **24**(7): 1024-1032.

USDA. (2012). "Keys to soil taxonomy." OSD, Official Soil Series Descriptions.

Usman, M. and S. Asghar (2011). "An Architecture for Integrated Online Analytical Mining." Journal of Emerging Technologies in Web Intelligence **3**(2): 74-99.

Usman, M., S. Asghar, et al. (2009). A Conceptual Model for Combining Enhanced OLAP and Data Mining Systems. INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference on.

Usman, M., S. Asghar, et al. (2010). Data Mining and Automatic OLAP Schema Generation. Proc. of Fifth International conference on Digital Information Management (ICDIM), Canada.

Usman, M. and R. Pears (2010). "Integration of Data Mining and Data Warehousing: A Practical Methodology." International Journal of Advancements in Computing Technology **2**(3): 31 - 46.

Usman, M. and R. Pears (2011). Multi Level Mining of Warehouse Schema. Networked Digital Technologies. Eds. S. Fong, Springer Berlin Heidelberg. **136:** 395-408.

Usman, M., R. Pears, et al. (2013). "A data mining approach to knowledge discovery from multidimensional cube structures." Knowledge-Based Systems **40**(0): 36-49.

Webb, G. I. (2006). Discovering significant rules. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, 434-443.

You, J., T. Dillon, et al. (2001). An integration of data mining and data warehousing for hierarchical multimedia information retrieval. Proceedings of 2001 International Symposium of Intelligent Multimedia, Video and Speech Processing. Hong Kong , China, IEEE**:** 373-376.

Zbidi, N., S. Faiz, et al. (2006). "On mining summaries by objective measures of interestingness." Machine learning **62**(3): 175-198.

Zhao, Y. and G. Karypis (2002). Evaluation of hierarchical clustering algorithms for document datasets. Proceedings of the eleventh international conference on Information and knowledge management. McLean, Virginia, USA, ACM**:** 515-524.

Zhen, L. and G. Minyi (2001). A proposal of integrating data mining and on-line analytical processing in data warehouse. In Proceedings of International Conferences on Info-tech and Info-net, ICII 2001 - Beijing.

Zhong, N., J. Dong, et al. (2001). "A hybrid model for rule discovery in data." Knowledge-Based Systems **14**(7): 397-412.

Zhu, H. (1998). On-line analytical mining of association rules, Master's thesis, Simon Fraser University, Burnaby, British Columbia, Canada.

Zubcoff, J., J. Pardillo, et al. (2007). "Integrating clustering data mining into the multidimensional modeling of data warehouses with UML profiles." Data Warehousing and Knowledge Discovery: 199-208.

Appendix I

C # code of the developed prototype for synthetic data and schema generation.

Create Database Function

```
    public void CreateDatabase(string dbName)
    {
        String str = "";
        SqlConnection myConn = new SqlConnection("Server=WT405A-002WCW\\DWSERVER;Integrated security=SSPI; database=master");

        str = "CREATE DATABASE " + dbName + " ON PRIMARY " +
            "(NAME = " + dbName + ", " +
            "FILENAME = 'E:\\" + dbName + ".mdf', " +
            "SIZE = 10MB, MAXSIZE = 100MB, FILEGROWTH = 10%) " +
            "LOG ON (NAME = " + dbName + "_Log, " +
            "FILENAME = 'E:\\" + dbName + "_log.ldf', " +
            "SIZE = 10MB, " +
            "MAXSIZE = 100MB, " +
            "FILEGROWTH = 10%)";

        SqlCommand myCommand = new SqlCommand(str, myConn);

        try
        {
            myConn.Open();
            myCommand.ExecuteNonQuery();
        }
        catch (System.Exception ex)
        {
            MessageBox.Show(ex.ToString(), "Alert", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        finally
        {
            if (myConn.State == ConnectionState.Open)
            {
                myConn.Close();

            }
        }
    }
```

Schema Generation Function

```
    private void btnGenerate_Click(object sender, EventArgs e)
    {
        if (txtDistinctValues.Text.Trim() != "" && txtNoOfDimensions.Text.Trim() != "" &&
            txtNoOfMeasures.Text.Trim() != "" && txtNoOfRecords.Text.Trim() != "")
        {
```

```
        lblStartTime.Text = DateTime.Now.ToString();
        Int32 totalRows = Convert.ToInt32(txtNoOfRecords.Text.Trim());
        Int32 totalDimensions = Convert.ToInt32(txtNoOfDimensions.Text.Trim());
        Int32 totalDistinctValues = Convert.ToInt32(txtDistinctValues.Text.Trim());
        Int32 totalMeasures = Convert.ToInt32(txtNoOfMeasures.Text.Trim());
        ArrayList dimensions = new ArrayList();
        ArrayList distinctValues = new ArrayList();
        ArrayList measures = new ArrayList();

        // Generate dimension column names
        GenerateDimensions(ref dimensions, totalDimensions);

        // Generate measure column names
        GenerateMeasures(ref measures, totalMeasures);

        // Generate distinct values
        GenerateDistinctValues(ref distinctValues, totalDistinctValues);

        // Generate dataset
        GenerateDataset(dimensions, measures, distinctValues, ref ds, totalRows);

        // Set time
        lblEndTime.Text = DateTime.Now.ToString();
        TimeSpan ts = Convert.ToDateTime(lblEndTime.Text.Trim()) -
Convert.ToDateTime(lblStartTime.Text.Trim());
        lblTimeTaken.Text = ts.TotalSeconds.ToString() + " Second(s)";

      }
      else
        MessageBox.Show("Please enter all input values.");
    }


    private void CreateDatabaseSchema(DataSet ds)
    {
      string dbName = "Generated_SCHEMA";

      // create db
      CreateDatabase(dbName);

      SqlConnection myConn = new SqlConnection("Server=WT405A-
002WCW\\DWSERVER;Integrated security=SSPI; database=" + dbName);
      myConn.Open();

      // create dimension tables
      GenerateDimensionTables(ref myConn, ref ds);

      // create fact table
      GenerateFactTable(ref myConn, ref ds);

      myConn.Close();
    }
```

Grouping File Reading Function

```
    private ArrayList GetGroupingList(XmlDocument xDoc, string dimName)
    {
      ArrayList list = new ArrayList();
    try
    {
        dgvGroupingData.DataSource = null;

        XmlNode root = xDoc.DocumentElement;
        XmlNode dimension = root.SelectSingleNode("//dimension[@name='" + dimName + "']");

        if (dimension != null)
        {
            DataGridViewColumn col1 = new DataGridViewColumn();
            col1.HeaderText = "Scale";
            col1.Name = "Scale";

            col1.CellTemplate = new DataGridViewTextBoxCell();
            DataGridViewColumn col2 = new DataGridViewColumn();
            col2.HeaderText = "Category";
            col2.Name = "Category";
            col2.CellTemplate = new DataGridViewTextBoxCell();
            DataSet dsTemp = new DataSet();
            DataTable dt = new DataTable("tab");
            dt.Columns.Add(new DataColumn("Scale", typeof(double)));
            dt.Columns.Add(new DataColumn("Category", typeof(string)));
            ArrayList scales = new ArrayList();
            ArrayList cats = new ArrayList();
            XmlNode node = dimension;
            node = node.FirstChild;

            while (node != null)
            {
                scales.Add(Convert.ToDouble(node.Attributes[0].Value));
                cats.Add(node.InnerText);
                node = node.NextSibling;
            }
            //scales.Sort();
            for (int j = 0; j < scales.Count; j++)
            {
                DataRow row = dt.NewRow();
                row["Scale"] = scales[j].ToString();
                row["Category"] = cats[j].ToString();
                dt.Rows.Add(row);

            }
            DataRow[] sortedRows = dt.Select("", "Scale DESC");

            double max = Convert.ToDouble(sortedRows[0][0]);
            double min = Convert.ToDouble(sortedRows[sortedRows.Length - 1][0]);
            double thresh = max - min;
            thresh = thresh / sortedRows.Length;
            list = CreateGroups(thresh, dimName, sortedRows);

        }
    }
    catch(Exception ex)
    {
        MessageBox.Show("GetGroupingList => " + ex.Message);
    }

    return list;
}
```

Group Creation Function

```
private ArrayList CreateGroups(double thresh, string currentDimName, DataRow[] sortedRows)
{
    ArrayList tableGrp = new ArrayList();

    try
    {
        ArrayList table = new ArrayList();
        ArrayList group = new ArrayList();
        ArrayList groupGrp = new ArrayList();
        int j = 0;
        string prevValue = "", grpVal = "", prevGrpVal = "";

        for (int i = 0; i < sortedRows.Length; i++)
        {

            if (group.Count == 0)
            {
                group.Add(currentDimName + "_Group" + j.ToString());
                groupGrp.Add(currentDimName + "_Group" + j.ToString());

                if (i == 0)
                {
                    group.Add(sortedRows[i][0].ToString());
                    prevValue = sortedRows[i][0].ToString();
                    groupGrp.Add(sortedRows[i][1].ToString());
                    prevGrpVal = sortedRows[i][1].ToString();
                }

                j++;
            }
            else
            {
                double val = 0;

                val = Convert.ToDouble(prevValue) -
                      Convert.ToDouble(sortedRows[i][0].ToString());

                if (val < thresh)
                {
                    group.Add(sortedRows[i][0].ToString());
                    prevValue = sortedRows[i][0].ToString();
                    groupGrp.Add(sortedRows[i][1].ToString());
                    prevGrpVal = sortedRows[i][1].ToString();

                    if (i == sortedRows.Length - 1)
                    {
                        table.Add(group);
                        tableGrp.Add(groupGrp);
                    }
                }
                else
                {
                    table.Add(group);
                    tableGrp.Add(groupGrp);
                    group = new ArrayList();
                    groupGrp = new ArrayList();
                    group.Add(currentDimName + "_Group" + j.ToString());
                    groupGrp.Add(currentDimName + "_Group" + j.ToString());
                    j++;
                    group.Add(sortedRows[i][0].ToString());
                    prevValue = sortedRows[i][0].ToString();
                    groupGrp.Add(sortedRows[i][1].ToString());
                    prevGrpVal = sortedRows[i][1].ToString();

                    if (i == sortedRows.Length - 1)
                    {
                        table.Add(group);
                        tableGrp.Add(groupGrp);
                    }
                }
            }
        }

        CreateOtherGroupColumn(ref table, ref tableGrp, currentDimName);
        ReorderGroupNames(ref table, ref tableGrp);

    }
    catch(Exception ex)
    {
        MessageBox.Show("CreateGroups => " + ex.Message);
    }

    return tableGrp;
}
```

Populating Data Grid Function

```
private void PopulateGroupGrid(ArrayList table, ArrayList tableGrp)
{
    try
    {
        // ---------------------  Value GRID ---------------------
        dgvGroupVals.Rows.Clear();
        dgvGroupVals.Columns.Clear();

        // create columns
        for (int i = 0; i < table.Count; i++)
        {
            ArrayList item = (ArrayList)table[i];
            DataGridViewColumn col1 = new DataGridViewColumn();
            col1.HeaderText = item[0].ToString();
            col1.Name = item[0].ToString();
            col1.CellTemplate = new DataGridViewTextBoxCell();
        }

        if (table.Count != 0)
        {
            // --------------------- Group GRID ---------------------

            dgvGroupVals.Rows.Clear();
            dgvGroupVals.Columns.Clear();

            // create columns
            for (int i = 0; i < tableGrp.Count; i++)
            {
                ArrayList item = (ArrayList)tableGrp[i];
                DataGridViewColumn col1 = new DataGridViewColumn();
                col1.HeaderText = item[0].ToString();
                col1.Name = item[0].ToString();
                col1.CellTemplate = new DataGridViewTextBoxCell();
                int colInd = dgvGroupVals.Columns.Add(col1);
                dgvGroupVals.Columns[colInd].Width = 140;
            }

            int index1 = dgvGroupVals.Rows.Add();

            //fill rows
            for (int i = 0; i < tableGrp.Count; i++)
            {
                ArrayList item = (ArrayList)tableGrp[i];

                for (int j = 1; j < item.Count; j++)
                {
                    if (dgvGroupVals.Rows.Count < item.Count - 1)
                        index1 = dgvGroupVals.Rows.Add();

                    dgvGroupVals.Rows[j - 1].Cells[i].Value = item[j];
                }
            }
        }


    }
    catch (Exception ex)
    {
        MessageBox.Show("PopulateGroupGrid => " + ex.Message);
    }
}
```

Storing Group Names Funciton

```
private void ReorderGroupNames(ref ArrayList table, ref ArrayList tableGrp)
{
    try
    {
        int k = 1;

        for (int i = 0; i < tableGrp.Count; i++)
        {
            ArrayList group = (ArrayList)tableGrp[i];
            ArrayList values = (ArrayList)table[i];

            if (group.Count > 0)
            {
                string dimName = group[0].ToString();

                if (!dimName.Contains("_Group-Others"))
                {
                    string[] splittedName = dimName.Split('_');
                    string newName = "";

                    for (int j = 0; j < splittedName.Length - 1; j++)
                    {
                        newName += splittedName[j] + "_";
                    }

                    newName += "Group" + k.ToString();
```

```
                        group[0] = newName;
                        values[0] = newName;
                        tableGrp[i] = group;
                        table[i] = values;
                        k += 1;
                    }
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("ReorderGroupNames => " + ex.Message);
        }
    }
```

Creation of Group-Others Function

```
    private void CreateOtherGroupColumn(ref ArrayList table, ref ArrayList tableGrp, string
currentDimName)
    {
        try
        {
            if (table.Count != 0)
            {
                ArrayList valueToDelete = new ArrayList();
                ArrayList groupToDelete = new ArrayList();
                ArrayList otherGroupTable = new ArrayList();
                ArrayList otherGroupGroups = new ArrayList();

                for (int i = 0; i < tableGrp.Count; i++)
                {
                    ArrayList group = (ArrayList)tableGrp[i];
                    ArrayList values = (ArrayList)table[i];

                    if (group.Count == 2)
                    {
                        valueToDelete.Add(values);
                        groupToDelete.Add(group);

                        if (otherGroupGroups.Count == 0)
                        {
                            string[] dimName = group[0].ToString().Split('_');
                            string newname = dimName[0];

                            for (int j = 1; j < dimName.Length - 1; j++)
                            {
                                newname += "_" + dimName[j];
                            }

                            otherGroupGroups.Add(newname + "_Group-Others");
                            otherGroupTable.Add(newname + "_Group-Others");
                        }

                        otherGroupGroups.Add(group[1].ToString());
                        otherGroupTable.Add(values[1].ToString());

                    }
                }
                for (int i = 0; i < groupToDelete.Count; i++)
                {
                    table.Remove(valueToDelete[i]);
                    tableGrp.Remove(groupToDelete[i]);
                }

                tableGrp.Add(otherGroupGroups);
                table.Add(otherGroupTable);
            }
            else
            {
                if (dgvData.Rows.Count != 0)
                {
                    ArrayList otherGroupTable = new ArrayList();
                    ArrayList otherGroupGroups = new ArrayList();
                    otherGroupTable.Add(currentDimName + "_Group-Others");
                    otherGroupGroups.Add(currentDimName + "_Group-Others");
                    otherGroupTable.Add(dgvData.Rows[0].Cells[0].Value);
                    otherGroupGroups.Add(dgvData.Rows[0].Cells[1].Value);
                    tableGrp.Add(otherGroupGroups);
                    table.Add(otherGroupTable);
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("CreateOtherGroupColumn => " + ex.Message);
        }
    }
```

Fact Table Creation and Insertion Function

```
    private void GenerateFactTable(ref SqlConnection myConn, ref DataSet ds)
    {
        try
        {
            string qry = "CREATE TABLE FactTable (FactTable_ID INTEGER IDENTITY (1, 1) PRIMARY KEY NOT NULL, ";

            for (int i = 1; i < ds.Tables[0].Columns.Count; i++)
            {
                string tableName = ds.Tables[0].Columns[i].ColumnName;

                if (tableName.Contains("Dim"))
                    qry += tableName + "_ID INTEGER";
                else
                    qry += tableName + " INTEGER";

                if (i != ds.Tables[0].Columns.Count - 1)
                    qry += ", ";
            }

            qry += ")";

            SqlCommand myCommand = new SqlCommand(qry, myConn);
            myCommand.ExecuteNonQuery();

            // insert data
            string insertQry = "INSERT INTO FactTable(";
                for (int i = 1; i < ds.Tables[0].Columns.Count; i++)
                {
                    string tableName = ds.Tables[0].Columns[i].ColumnName;

                    if (tableName.Contains("Dim"))
                        insertQry += tableName + "_ID";
                    else
                        insertQry += tableName;

                    if (i != ds.Tables[0].Columns.Count - 1)
                        insertQry += ", ";
                }

                insertQry += ") VALUES(";

                for (int j = 0; j < ds.Tables[0].Rows.Count; j++)
                {
                    string vals = "", dec = "";

                    for (int k = 1; k < ds.Tables[0].Columns.Count; k++)
                    {
                        string colName = ds.Tables[0].Columns[k].ColumnName;

                        if (colName.Contains("Dim"))
                        {
                            // generate sub query
                            dec += " declare @" + colName + " varchar(50); SET @" + colName + " = (SELECT "
+ colName + "_ID FROM " + colName + " WHERE " + colName + "_name = '" +
ds.Tables[0].Rows[j][k].ToString() + "' ); ";
                            vals += "@" + colName;
                        }
                        else
                            vals += ds.Tables[0].Rows[j][k];

                        if (k != ds.Tables[0].Columns.Count - 1)
                            vals += ", ";
                    }

                    vals += ")";
                    SqlCommand insertCommand = new SqlCommand(dec + insertQry + vals, myConn);
                    insertCommand.ExecuteNonQuery();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("GenerateFactTable => " + ex.Message);
            }
        }
```

Dimension Tables Creation and Insertion Function

```
        private void GenerateDimensionTables(ref SqlConnection myConn, ref DataSet ds)
        {
            try
            {
                XmlDocument xDoc = new XmlDocument();

                for (int i = 0; i < ds.Tables[0].Columns.Count; i++)
                {
```

```
                string tableName = ds.Tables[0].Columns[i].ColumnName;

                if (tableName.Contains("Dim"))
                {
                    string qry = "CREATE TABLE " + tableName + " " +
                                    " (" + tableName + "_ID INTEGER IDENTITY (1, 1) PRIMARY KEY NOT NULL, ";

                    for (int ind = 0; ind < Convert.ToInt32(nudGroupingLevel.Value); ind++)
                    {
                        qry += tableName + "_group_lvl_" + (ind + 1).ToString() + " varchar(50) ";

                        if (ind != Convert.ToInt32(nudGroupingLevel.Value) - 1)
                            qry += ", ";
                    }

                    qry += ", " + tableName + "_name varchar(50) )";

                    SqlCommand myCommand = new SqlCommand(qry, myConn);
                    myCommand.ExecuteNonQuery();

                    if (txtGroupingFile.Text.Trim() != "")
                    {
                        xDoc.Load(txtGroupingFile.Text);
                    }

                    // insert data
                    for (int j = 0; j < ds.Tables[0].Rows.Count; j++)
                    {

                        string insertQry = "IF (SELECT Count(*) FROM " + tableName + " WHERE " + tableName + "_name = '" + ds.Tables[0].Rows[j][i].ToString() + "' ) = 0 BEGIN INSERT INTO " + tableName + " (" + tableName + "_name, ";

                        for (int ind = 0; ind < Convert.ToInt32(nudGroupingLevel.Value); ind++)
                        {

                            insertQry += tableName + "_group_lvl_" + (ind + 1).ToString();

                            if (ind != Convert.ToInt32(nudGroupingLevel.Value) - 1)
                                insertQry += ", ";
                        }

                        insertQry += ") VALUES('" + ds.Tables[0].Rows[j][i].ToString() + "',";

                        for (int ind = 0; ind < Convert.ToInt32(nudGroupingLevel.Value); ind++)
                        {
                            insertQry += "'" + GetGroupValue(xDoc, tableName.Replace("Dim_", ""), (ind + 1), ds.Tables[0].Rows[j][i].ToString()) + "' ";

                            if (ind != Convert.ToInt32(nudGroupingLevel.Value) - 1)
                                insertQry += ", ";
                        }

                        insertQry += ") END";
                        SqlCommand insertCommand = new SqlCommand(insertQry, myConn);
                        insertCommand.ExecuteNonQuery();
                    }
                }
            }
            }
            catch (Exception ex)
            {
                MessageBox.Show("GenerateDimensionTables => " + ex.Message);
            }
        }
```

Reading Grouping Information Function

```
        private string GetGroupValue(XmlDocument xDoc, string dimensionName, int groupingLevel,
string valueToCompare)
        {
            string groupVal = "";
            ArrayList objGroupingList = new ArrayList();
            objGroupingList = GetGroupingList(xDoc, dimensionName);

            if (objGroupingList != null && objGroupingList.Count > 0) // get group value
            {
                for (int k = 0; k < objGroupingList.Count; k++)
                {
                    ArrayList innerList = (ArrayList)objGroupingList[k];

                    if (innerList.Contains(valueToCompare))
                    {
                        groupVal = innerList[0].ToString().Trim();
                        break;
                    }
                }
            }
            else
                groupVal = "";

            return groupVal;
        }
```

Synthetic Dataset Generation Function

```
        private void GenerateDataset(ArrayList dimensions, ArrayList measures, ArrayList distinctValues,
ref DataSet ds, Int32 totalRows)
        {
            DataTable dt = new DataTable("tab");
            DataColumn colPK = new DataColumn("ID");
            dt.Columns.Add(colPK);

            // add dimension columns
            for (int i = 0; i < dimensions.Count; i++)
            {
                DataColumn col = new DataColumn(dimensions[i].ToString());
                dt.Columns.Add(col);
            }

            // add measure columns
            for (int i = 0; i < measures.Count; i++)
            {
                DataColumn col = new DataColumn(measures[i].ToString());
                dt.Columns.Add(col);
            }

            int id = 1;

            // fill data
            for (Int32 i = 0; i < totalRows; i++)
            {
                DataRow row = dt.NewRow();
                row["ID"] = id++;

                for (int j = 1; j < dt.Columns.Count; j++)
                {
                    if (dt.Columns[j].ColumnName.Contains("Dim"))
                    {
                        row[dt.Columns[j].ColumnName]        =        dt.Columns[j].ColumnName        +
GetRandomValue(distinctValues);
                    }
                    else
                        row[dt.Columns[j].ColumnName] = random.Next(0, 1000);
                }

                dt.Rows.Add(row);
            }

            ds.Tables.Add(dt);
            dgvData.DataSource = ds;
            dgvData.DataMember = "tab";
            btnGenerateSchema.Enabled = true;
        }

        private string GetRandomValue(ArrayList distinctValues)
        {
            int index = random.Next(0, distinctValues.Count);
            return distinctValues[index].ToString();
        }

        private void GenerateDistinctValues(ref ArrayList distinctValues, int totalDistinctValues)
        {
            for (int i = 0; i < totalDistinctValues; i++)
            {
                distinctValues.Add("_v" + (i + 1).ToString());
            }
        }

        private void GenerateMeasures(ref ArrayList measures, int totalMeasures)
        {
            for (int i = 0; i < totalMeasures; i++)
            {
                measures.Add("Measure_" + (i + 1).ToString());
            }
        }

        private void GenerateDimensions(ref ArrayList dimensions, int totalDimensions)
        {
            for (int i = 0; i < totalDimensions; i++)
            {
                dimensions.Add("Dim_" + (i + 1).ToString());
            }
        }
        public static void ShuffleInPlace(ArrayList source)
        {
            Random rnd = new Random();

            for (int inx = source.Count - 1; inx > 0; --inx)
            {
                int position = rnd.Next(inx);
                object temp = source[inx];
                source[inx] = source[position];
                source[position] = temp;
            }
        }
```

Schema Generation Time Calculation Function

```
        private void btnGenerateSchema_Click(object sender, EventArgs e)
        {
            lblSchemaStartTime.Text = DateTime.Now.ToString();
            CreateDatabaseSchema(ds);
            lblSchemaEndTime.Text = DateTime.Now.ToString();
            TimeSpan ts = Convert.ToDateTime(lblSchemaEndTime.Text.Trim()) -
Convert.ToDateTime(lblSchemaStartTime.Text.Trim());
            lblSchemaTimeTaken.Text = ts.TotalSeconds.ToString() + " Seconds";
        }

        private void btnBrowseDataFile_Click(object sender, EventArgs e)
        {
            openFileDialog1.FileName = "";

            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                txtDataFile.Text = openFileDialog1.FileNames[0];
            }
        }

        private void btnBrowseGroupingFile_Click(object sender, EventArgs e)
        {
            openFileDialog1.FileName = "";

            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                txtGroupingFile.Text = openFileDialog1.FileNames[0];
            }
        }

        private void btnLoadDataFiles_Click(object sender, EventArgs e)
        {
            if (txtDataFile.Text.Trim() != "")
            {
                lblStartTime.Text = DateTime.Now.ToString();

                // load data file
                StreamReader rdr = new StreamReader(txtDataFile.Text.Trim());
                DataTable dt = new DataTable("data");
                Boolean colsCreated = false;
                string nextLine = "", line = "";

                while (rdr.Peek() > -1)
                {
                    if (nextLine == "")
```

```
                        line = rdr.ReadLine();
                    else
                    {
                        line = nextLine;
                        nextLine = "";
                    }

                    if (!colsCreated)
                    {
                        nextLine = rdr.ReadLine();
                        CreateColumns(ref dt, line, nextLine);
                        colsCreated = true;
                    }
                    else
                    {
                        string[] splittedLine = line.Split('\t');

                        {
                            DataRow dr = dt.NewRow();

                            for (int j = 0; j < dt.Columns.Count; j++)
                            {
                                string groupingVal = "";

                                if (dt.Columns[j].ColumnName.Contains("Dim"))
                                {

                                }

                                if (groupingVal.Trim() == "")
                                    dr[dt.Columns[j].ColumnName] = splittedLine[j];
                                else
                                    dr[dt.Columns[j].ColumnName] = splittedLine[j] + "  -  " + groupingVal;
                            }

                            dt.Rows.Add(dr);
                        }
                    }
                }

                ds.Tables.Add(dt);
                dgvData.DataSource = ds;
                dgvData.DataMember = "data";
                btnGenerateSchema.Enabled = true;

                // set time
                lblEndTime.Text = DateTime.Now.ToString();
                TimeSpan ts = Convert.ToDateTime(lblEndTime.Text.Trim()) -
Convert.ToDateTime(lblStartTime.Text.Trim());
                lblTimeTaken.Text = ts.TotalSeconds.ToString() + " Seconds";
            }
            else
                MessageBox.Show("Please select data file.");
        }
```

Column Creation Function

```
        private void CreateColumns(ref DataTable dt, string line, string nextLine)
```

```
    {
        string[] splittedLine = line.Split('\t');
        string[] splittedNextLine = nextLine.Split('\t');

        for (int i = 0; i < splittedLine.Length; i++)
        {
            string colName = "";

            if (i == 0)
                colName = "ID";
            else
            {
                Double val = 0;
                Double.TryParse(splittedNextLine[i].Trim(), out val);

                string name = splittedLine[i].Trim().Replace("-", "_");

                if (val != 0)
                    colName = "Measure_" + name;
                else
                    colName = "Dim_" + name;
            }

            DataColumn col = new DataColumn(colName);
            dt.Columns.Add(col);
        }
    }
```

View Grouped Values Function

```
    private void btnViewGrouping_Click(object sender, EventArgs e)
    {
        if (txtGroupingFile.Text.Trim() != "")
        {
            Form1 frm = new Form1(txtGroupingFile.Text.Trim());
            frm.ShowDialog();
        }
        else
            MessageBox.Show("Please select meta file.");
    }
}}
```