Web Based Interactive Charts for Exploring Large Datasets

Xuyen On

Thesis

Submitted to the Faculty of Graduate and Postdoctoral Studies

in partial fulfillment of the requirements

for the degree of Master of Science

Computer Science Program

School of Information Technology and Engineering

University of Ottawa

Ottawa, Ontario, Canada

ACKNOWLEDGEMENTS

I would like to thank Dr. Liam Peyton for his patience, guidance and support. This thesis would not have been possible without his help and I am forever grateful for all that he has done. I also owe a lot of thanks to Dr. Timothy Lethbridge and Abdou Wahab, for their guidance and help in writing this thesis. I would also like to acknowledge Patrick Goubran for working with me on the data preparation work and Dr. Gitte Lindgaard from the Carleton University HOT Lab for her contribution to my usability study.

Special thanks to Tom Fazal and Floyd Kelly at Cognos Inc. as well as NSERC who gave me the Industrial Postgraduate Scholarship. Finally, I would like to thank all the students who took part in my usability study at the University of Ottawa.

Abstract

Databases and data warehouses are growing at an increasing rate as storage and processing power becomes cheaper. Finding useful information in these data stores by navigating through gigabytes of data is a challenging task. The problem is that most users cannot remember all of the information as they go through the data. Another problem is that the display area is limited so the user has to scroll to different parts of the screen to see all of the information. Companies and organizations are often spread across many different locations so users must be able to access the data remotely from a web browser that is secure and zero footprint.

This thesis presents a web based visualization tool that addresses these issues. It allows users to explore a multidimensional dataset from the Centers for Disease Control and Prevention (CDC) in Atlanta that contains patient records of vaccinations taken in the U.S.A. We begin by introducing the concepts of data visualization and data warehouses. Then we present a case study that looks at the importance of data preparation for effectively exploring multidimensional data. It also compares the differences and similarities in data preparation for visualization and machine learning.

The main contribution of this thesis is that it shows an implementation for a web based visualization tool that is effective at enabling users to find trends and patterns in large multidimensional datasets. It shows that a zero footprint client is possible by using Scalable Vector Graphics (SVG) and DHTML to generate interactive charts within a web browser. It also shows that users are able to effectively find trends and patterns if they have multiple views of interactive charts with navigation controls such to zoom, pan, and drill into the data.

A usability study was performed with computer science graduate students to determine the effectiveness of the tool. They were asked to complete a set of tasks using the tool to find certain trends or patterns within the CDC data. The results were obtained through interview questions and questionnaires.

It was found that the users were able to complete simple tasks that did not require too much memorization. They had difficulty with tasks which asked them to compare data within a large search space where they had to remember many charts. Multiple views

helped, but other techniques are required. Users wanted the ability to see all of the information at the same time and to have some preprocessing done to narrow the search space.

Table of Contents

1 Introduction 7
1.1 Problem statement 7
1.2 Motivation and Objectives 7
1.3 Thesis Contributions 8
1.4 Organization of Thesis 10
2 Background and Related Work 11
2.1 Basic Techniques for Visualizing Data 11
2.2 Navigation controls 17
2.3 Dynamic Queries 18
2.4 Multiple-View Visualizations 19
2.5 Web-based Charts 21
3 Introduction to Multi-Dimensional Models and OLAP 27
3.1 Multi-dimensional Models 27
3.1.1 OLAP 27
3.1.2 Multidimensional Star Schema 28
3.1.3 Dimension Tables 30
3.1.4 Aggregations 30
4 Case Study : Data Preparation for Visualization versus Data Preparation for Machine Learning 31
4.1 Preparing CDC Dataset 31
4.2 Data Preparation for Machine Learning 32
4.3 Data Preparation for Data Visualization 33
4.4 Visualizing the CDC Dataset 34
4.5 Case Study in Machine Learning 43
4.6 Discussion 48
5 System Architecture 51
5.1 SVG 51
5.2 Web Interface 52
6 Methodology for Usability Study 54
6.1 Test Users 54
6.2 Tasks for Usability Testing 54
7 Results 56
8 Conclusion 63
8.1 Data Preparation 63
8.2 Usability Study 64
9 Future Work 67
10 References 68
A Appendices 73
A.1 Usability Study 73
A.2 Questionnaire 74

List of Figures

Figure 1 Combo Bar and Line Chart 12
Figure 2 Tree Layout for Hierarchical Data 12
Figure 3 SeeSys High Level Treemap 14
Figure 4 Hierarchical Graph Projected onto a Sphere 15
Figure 5 Cube Presentation Model mapped to 2D and 3D crosstabs..... 16
Figure 6 Table Lens 17
Figure 7 Home Finder 18
Figure 8 High Dimensional Projection in XGobi 20
Figure 9 Linked ScatterPlot in XGobi 20
Figure 10 Snap together visualization..... 21
Figure 11 Parallel Coordinates in Vizcraft..... 24
Figure 13 Star Schema Model of CDC Dataset..... 34
Figure 14 Occurrences of Symptoms versus Vaccines 35
Figure 15 Occurrences of Arthritis versus States 36
Figure 16 U.S. Map of Arthritis Reports in patients who have taken LYME vaccine 36
Figure 17 LYME Vaccinations versus States..... 37
Figure 18 US Map of LYME Vaccinations..... 37
Figure 19 Percentage of Arthritis occurrence versus total LYME Vaccinations 38
Figure 20 US Map showing Percentages of Arthritis reports to LYME Vaccinations..... 39
Figure 21 Treemap of Percentages of Arthritis reports to LYME Vaccinations..... 40
Figure 22 Treemap of Agegroups of patients..... 41
Figure 23 Treemap showing Year of Vaccinations 42
Figure 24 Treemap show Sex of Patients..... 43
Figure 25 Most heavily weighted Decision Tree of CDC Data set after boosting..... 45
Figure 26 LYME branch from CDC decision tree with second highest weight after boosting 47
Figure 27 System Architecture of Visualization Tool 51
Figure 28 Screenshot of Visualization Tool..... 53

List of Tables

Table 1 Comparison of Visualization versus Machine Learning 50
Table 2 Results of Questionnaire..... 59

1 Introduction

1.1 Problem statement

There is an exponential trend in the amount of electronic data that is becoming available to us. This is due to decreasing costs in computing power and storage. It is becoming cheaper and cheaper to buy computers with faster processors and larger hard drives. Organizations are finding that their databases and data warehouses are growing at an ever increasing rate. As the data becomes larger, there is a greater need to effectively manage the information.

Finding trends, patterns and associations within the data can be very important for organizations. It can help to identify problem areas, find root causes, and forecast markets. This can be very hard to do, especially when there are many dimensions to explore. One problem is that it is difficult to maintain contextual information. It would help if the user can see the bigger picture when focusing on smaller parts of the search space. Most current commercial products are not effective at presenting context information.

In order to effectively explore through large data sets, the data must be properly prepared. It is computationally expensive to calculate sums and averages in real time. The data has to be structured such that aggregates can be calculated as fast as possible or are pre-computed.

Another consideration is that large organizations are typically spread across many different locations. Installation and maintenance of software can have significant costs because the application can be spread across many computers in different locations. A centralized framework for visualizing large multidimensional data sources is needed to support many users at different locations.

1.2 Motivation and Objectives

As data warehouses continue to grow, users will have a harder time navigating through the data. There is an increasing need for an effective way to visualize multidimensional data so that users can have a better understanding of the content and be able to extract useful information. The objective of this thesis is to develop a framework

for users to be able to do these tasks from a secure thin client that is accessible through the Internet.

Our visualization tool allows a user to explore large multidimensional data sets using web based controls in a zero footprint web browser. The user will be able to quickly and effectively drill into the data through flexible queries and quick updates of interactive charts. This thesis investigates the following issues:

1. Can a user easily find trends and patterns within a multidimensional dataset by drilling into the data and comparing charts using web based navigation controls and multiple views?
2. Are multiple views effective for comparing charts and preserving contextual information for a user while drilling through the data?
3. What navigation controls are possible within a browser using Scalable Vector Graphics (SVG) and DHTML?
4. How should one model data using the star schema for multidimensional analysis?
5. What are the similarities and differences between machine learning and visualization regarding data preparation?
6. What is the importance of user interaction in knowledge discovery?

1.3 Thesis Contributions

The purpose of visualization is to process data in order to make it easier to identify important relationships. An important part of this process is deciding what part of the data needs to be examined. Data has to be readily accessible so that it can be filtered, sorted, and selected dynamically into different views.

This thesis is an investigation of effective methods for finding trends and patterns within multidimensional data sets. We have developed a visualization tool that allows a user to effectively explore large multidimensional datasets within a web browser. It uses the open graphics standard, SVG and DHTML to generate interactive graphs within a web browser. Multiple views are provided to make it easier for the user to find trends and patterns by making comparisons easier and by providing more contextual information.

We also investigate the differences and similarities between data preparation for machine learning and for data visualization, as well as why it is needed. Finally, we perform a usability study to test our hypothesis and give directions for future study.

The results of this investigation should be of interest to large organizations that have many users from different locations who want to look for trends and patterns in large data multidimensional data sources.

1.4 Organization of Thesis

The rest of the thesis is organized as follows:

Chapter 1 introduces the concepts and purpose of data visualization. Common visualization techniques are introduced. The chapter ends with a discussion of why these methods are ineffective.

Chapter 2 looks at related work in visualization of multidimensional data. It discusses the need for dynamic navigation controls and dynamic queries for multidimensional analysis. It also covers recent research in the area of multiple view visualizations and web based visualization tools.

Chapter 3 introduces data warehouse concepts including multidimensional modeling, OLAP, and the star schema.

Chapter 4 is a case study of data preparation for machine learning and data visualization.

Chapter 5 covers the system architecture of the visualization tool. It introduces Scalable Vector Graphics and discusses why it was chosen for our visualization tool. An outline is given to explain how the web interface is organized.

Chapter 6 outlines the methodology for the usability study. The background of the test subjects and the rationale for the user tasks are given.

Chapter 7 presents the results of the usability study.

Chapter 8 gives our conclusions of our case study in data preparation and the usability study.

Chapter 9 gives direction for future work.

Chapter 10 gives a list of references used in this thesis.

2 Background and Related Work

Richard Hamming once said that the purpose of computing is insight, not numbers. The goal of visualization is to gain understanding and insight into data by using our sight [1].

With the recent surge of information within commerce, there is a growing interest in applying visualization techniques to gain insight into business processes. It is becoming increasingly common to use the tools and techniques of visualization to analyze and display large volumes of multidimensional data in order to find useful information [2].

The main purpose of visualization tools is to allow the user to understand complex concepts [3]. One of the most difficult tasks in visualization is to find trends and patterns in large multidimensional datasets. As the number of dimensions increases, the search space becomes larger because the possible number of relationships between the different dimensions increases exponentially. A user can effectively traverse large information spaces if the information is presented visually and the user has dynamic query tools [4]. There have been many attempts to develop graphical widgets that try to reduce errors and improve ease of use but results have been mixed [5][6].

2.1 Basic Techniques for Visualizing Data

A basic way of displaying data sets is to use bar charts and line charts like the ones in Figure 1. This example charts arbitrary values of two different dimensions along a range of years from 1998 to 2001. The bars correspond to one dimension and the line corresponds to another. These charts are effective for comparing simple data sets that only have a few dimensions. It is easy to see that the values of the line chart are about half of bar chart values.

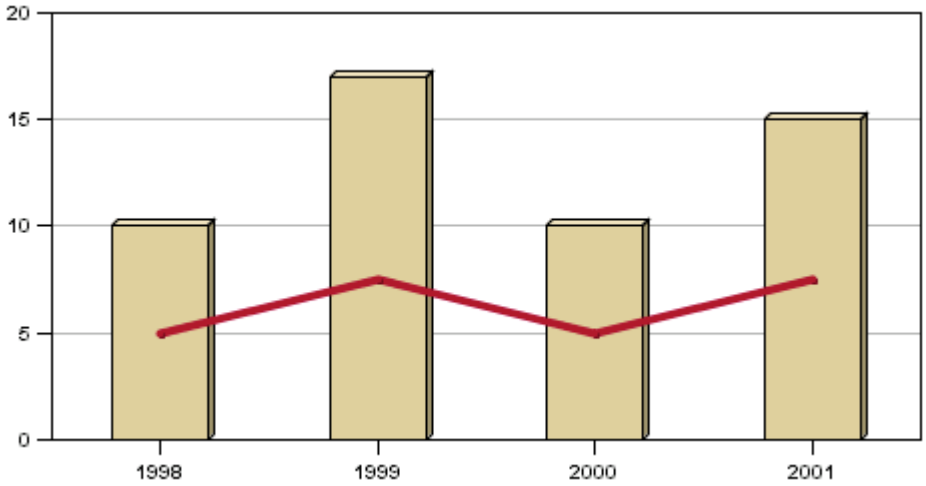

Figure 1 Combo Bar and Line Chart [7]

However, spotting trends and predicting outcomes by looking at static bar charts or line charts is not always effective due to the limited amount of information that can be displayed. This is particularly true when patterns exist in more than a few dimensions.

Hierarchical data are often represented in a tree layout where subsets of data are drawn below the parent data set and are connected by lines as shown in Figure 2.

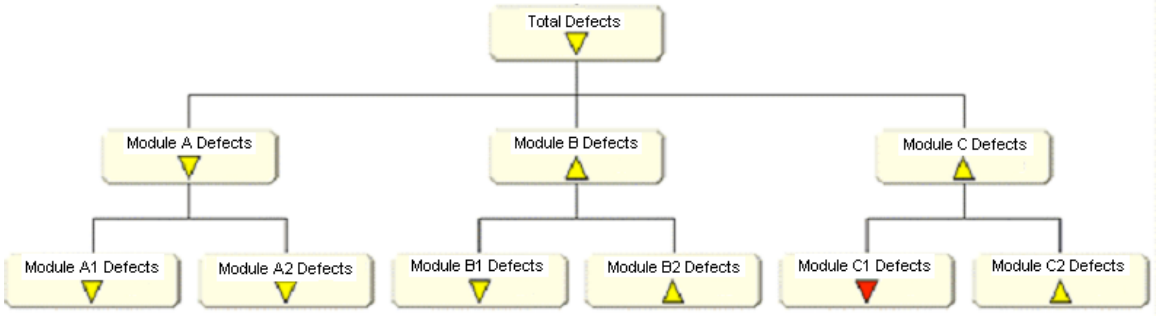

Figure 2 Tree Layout for Hierarchical Data [8]

Data is decomposed hierarchically into modules and sub modules where performance indicators of the parent modules are aggregates of its sub modules. In this example, the tree layout represents the performance metrics for defect rates in a software project. The trends in defect rates are represented by the triangles inside the boxes representing the modules. Triangles pointing up indicate an increasing trend and triangles pointing down indicate a decreasing trend. The colors of the triangles indicate the status of performance.

This gives an added dimension to the chart. Green usually indicates better than expected values, yellow means within expected values, and red means lower than expected values.

While this view provides snapshots of performance, it is hard to identify problem areas because of the limited amount of information that can be shown. Often, the data is aggregated into higher levels to reduce the number of symbols so that everything will fit onto the screen, but this also reduces the detail of information displayed. The user then has to drill down many levels to find detailed data.

One of the main weaknesses of the current visualization systems is the limited amount of information that can be effectively displayed on a screen. Treemaps, originally developed by Shneiderman and Johnson [9], maximize the display area by utilizing all of the available space. Treemaps also preserve the hierarchical structure of the data because subsets of data are nested in the areas of the parent data set.

Baker and Eick [10] from AT&T Bell Laboratories developed a software project visualization tool called SeeSys that uses treemaps to visualize hierarchical data. The display uses nested rectangles whose positions correspond to the hierarchical structure of the components and the size and color correspond to the values of the performance data. Figure 3 shows a high level view of a treemap display where each rectangle represents a component of a software project labeled by a letter of the alphabet.

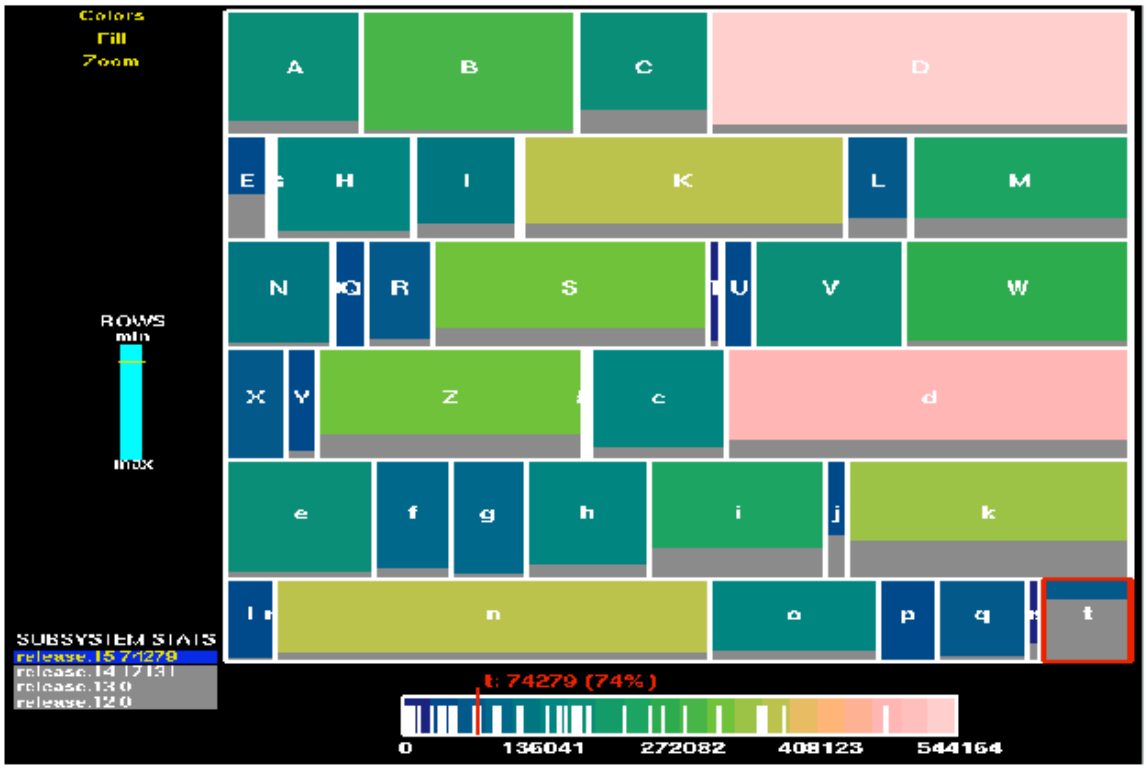

Figure 3 SeeSys High Level Treemap[11]

The size of the rectangles corresponds to the program size of the components in relation to each other and to the whole system. The amount of new code is represented as a smaller grey rectangle filling the bottom of each of the component rectangles. This display visualizes which components have the most amount of code and how much new code has been added. SeeSys provides interactive features, which allow the user to zoom in on individual rectangles of the treemap and to adjust the level of detail of the display. A user can then navigate to different parts of the system. A popup window displays detailed information about a component when the cursor is placed above the corresponding rectangle. There is also an animation feature, which shows how the performance of system progresses with time by showing how the rectangles change in size and color. This allows the user to see trends for code growth, defect rates, complexity, and changes to requirements for the various parts of the system.

While it is useful to see how the various parts of the system are performing, it is also important to understand the context of the information. For instance, the bug rate of a component by itself might not be significant but if it is much larger compared to the

components of other subsystems, there might be a problem. Treemaps convey context through its hierarchical layout. It is easy to see which components belong to which parts of the system because the rectangles are laid out hierarchically. Treemaps convey information quickly and effectively because people are able to quickly process spatial information.

The SeeSys visualization tool is able to effectively display metric information of thousands of components using the treemap layout because it maximizes the display area and wastes little space. This makes it easier for the user to identify problem areas because more information can be shown compared to the other display techniques. Navigation features such as changing time dimensions and zooming in on sub modules, makes it easier to track progress, to find trends and patterns, and to identify root causes.

Kreuseler and Schumann [12] developed a flexible framework for Visual Data Mining. They created their Focus+Context technique called the Magic Eye View (Figure 4) to visualize complex hierarchical graphs. The Magic Eye View can represent complex hierarchical data and has the ability to zoom in on particular parts of the graph while still showing contextual information. However, the system does not support dynamic aggregates of data and the user cannot slice and dice through different levels of the tree.

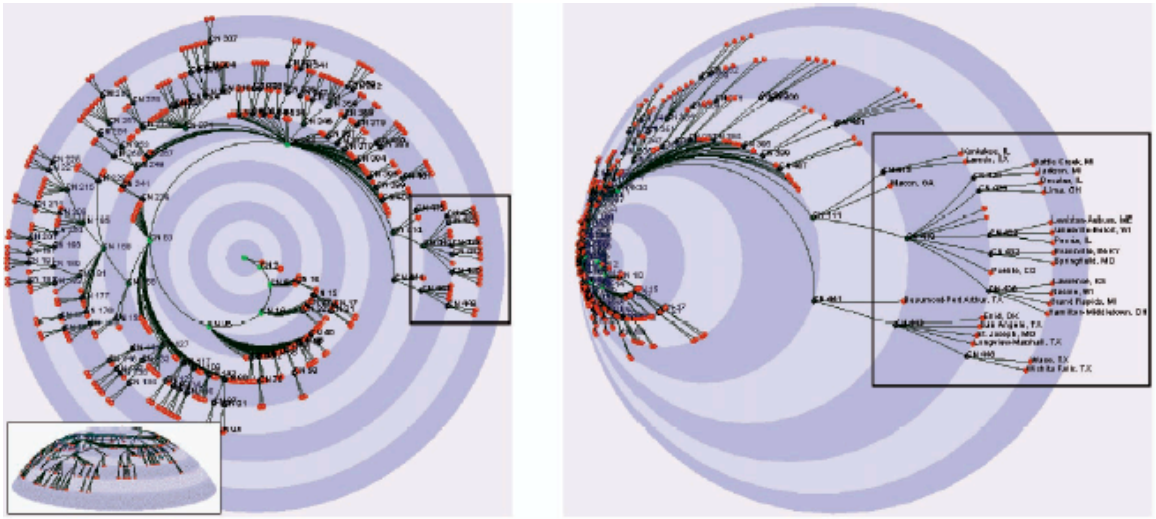

Figure 4 Hierarchical Graph Projected onto a Sphere [12]

Maniatis et al. [13] have developed a flexible OLAP visualization technique by combining their Cube Presentation Model (Figure 5) with the Table Lens cross-tabular presentation model [14]. Their technique simplifies the query and answer retrieval process by applying an algorithm to determine and highlight areas of interest within the display.

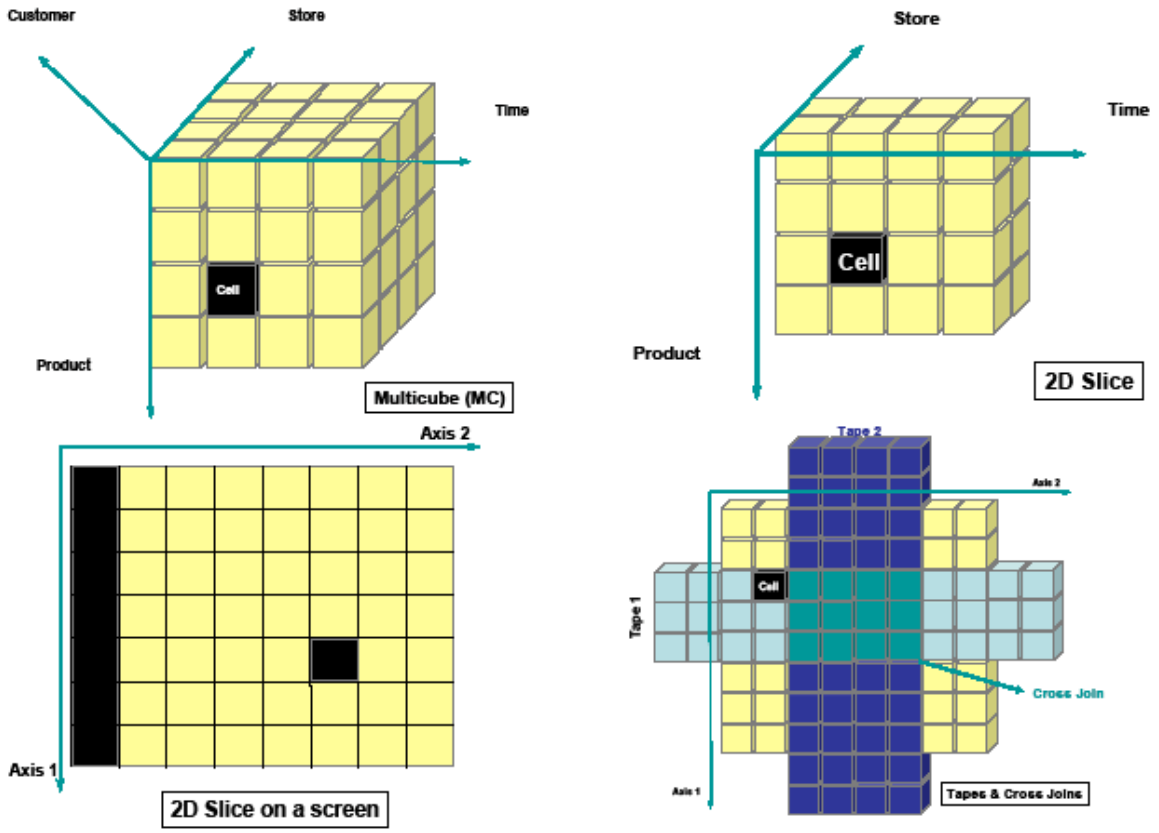

Figure 5 Cube Presentation Model mapped to 2D and 3D crosstabs

The Table Lens allows the user to focus on areas of interest while still providing contextual information by manipulating sections of a crosstab display. The user can zoom in different sections without losing the big picture. Figure 6 shows how the Table Lens works. The selected area will have its rows and columns expanded while the other cells remain the same size.

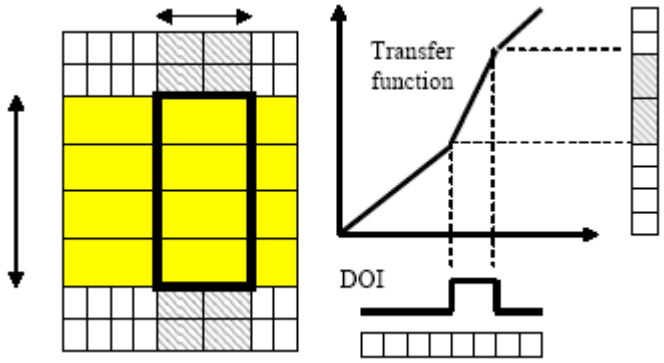

Figure 6 Table Lens

Wietek [15] proposed a dataflow based visual environment called VIOLA (VIsual On-line data Analysis environment). It uses a data flow model that lets the user keep track of his actions while navigating through the data. The user can see what he has done and make small changes to the action history and quickly see the results. VIOLA also uses interactive graphics with multiple views of linked charts to make it easier for the user to explore the data. It uses a multidimensional data model called MADEIRA (Modeling Analysis of Data in Epidemiological InteRActive studies) instead of the star schema. MADEIRA is unique because it captures semantic information as well as aggregate information.

2.2 Navigation controls

Graphs and glyphs alone are not sufficient for information visualization because complex problems such as finding trends and patterns require multidimensional analysis. Two-dimensional displays are inherently limited in the amount of information that they can show. Interactive controls such as zooming and adjustable time displays are needed to effectively visualize multidimensional data [16]. Three-dimensional displays offer an extra degree of freedom to show more attributes, but there are usability problems of navigating through three dimensional data models using two dimensional views and controls.

One way of overcoming this problem is to have interactive controls to select and navigate through different dimensions of the data. The user should be able to interactively change the view by choosing which dimensions to show or hide. Having dynamic controls for navigation makes it much easier to spot trends and future outcomes from the data because of two main reasons. First, the user is able to explore many different

dimensions of the data. Second, the user is able to control the amount of information displayed so that the display does not become too saturated and unintelligible.

2.3 Dynamic Queries

Dynamic queries allow users to quickly and efficiently explore large data sets by having interactive controls to query parameters and by having quick updates of database search results [17,18,19]. An update to a query occurs whenever a control is changed such as a button press or a slider movement. Results are displayed graphically and are updated almost instantaneously. Experiments have shown that dynamic queries are a fast, effective, and easy way of finding trends and spotting exceptions for beginners as well as expert users [20].

Dynamic query user interfaces provide a visual representation of the query and the results. They typically have rapid, incremental, and reversible actions where users can select parameters by pointing and not typing. Some demos of dynamic queries are available from [21].

The HomeFinder tool was one of the first visualization tools to utilize dynamic queries (Figure 7). It allows the user to adjust the level of detail of information provided by moving slide bars and by selecting different buttons. The display is dynamically updated by yellow dots on the screen, which represents houses, as the user changes the search criteria [22].

Figure 7 Home Finder

2.4 Multiple-View Visualizations

One way to make it easier for the user to find the data is to use multiple views. Multiple view visualizations use different views to look at data and help to improve understanding through interaction. They allow users to combine the strengths of multiple visualizations to form a more powerful one. The combined coordinated multiple view visualizations can support a broader range of tasks [23].

People can only remember 7 +/- 2 “chunks” of information in their short term memory [24]. These chunks can be groups of similar items. For example, a set of navigation controls such as zooming, panning, and cropping can be considered as a chunk because they all manipulate objects on the screen for viewing. Another set of controls for file manipulation such as opening, closing, saving, and printing can be considered as a separate chunk because they all affect the file somehow. Multiple views makes it easier for the user to find information because more information is available to the user on the screen so he doesn’t have to store as much in memory. This makes it easier to compare data, as well as finding trends and patterns within large sets of data.

XGobi is a visualization application for multi-dimensional data that was implemented on the X Window platform (Figure 8). It allows the user to explore and compare multidimensional data by focusing, linking, and arranging views that include high-dimensional projections, linked scatterplots, brushing, and matrices of conditional plots. It uses multiple views for comparisons (Figure 9) [25].

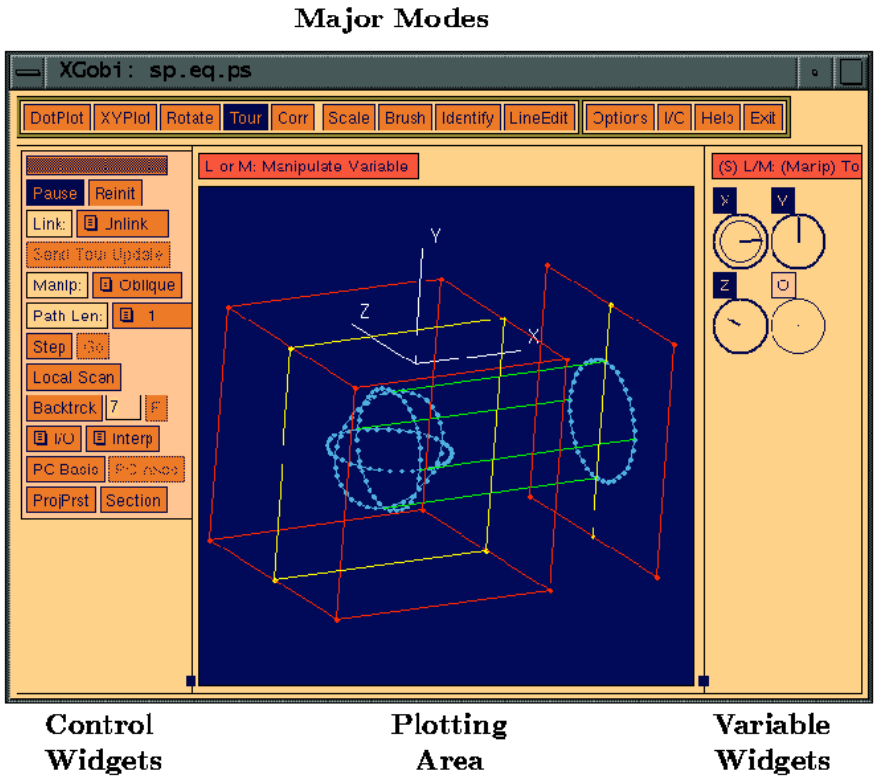

Figure 8 High Dimensional Projection in XGobi

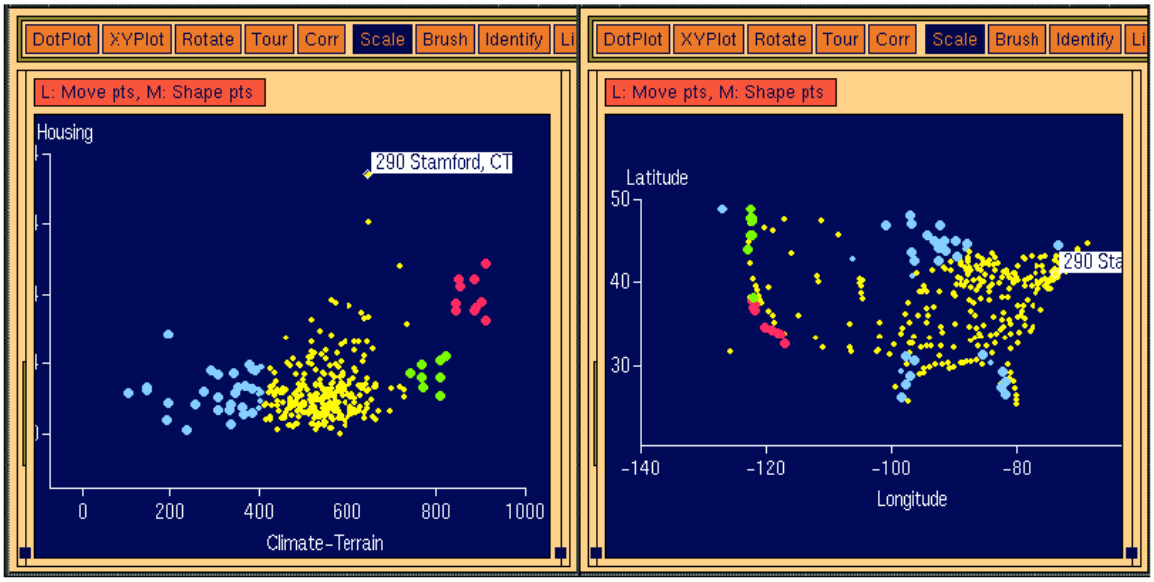

Figure 9 Linked ScatterPlot in XGobi

There are general-purpose visualization tools such as Spotfire [26] but they only offer a partial solution. In many cases custom visualizations are needed, but they are expensive

and time consuming because they require custom programming. There is a need for flexibility in the design and implementation of information visualizations [27]. North and Shneiderman [28] have presented Snap-Together Visualization as a potential solution to this problem. Its conceptual model is based on the relational database model where the results of joins can are visualized. It also supports other features such as brushing, drill down, high level and detailed views, and synchronized scrolling. Visualizations can be easily customized through a simple API.

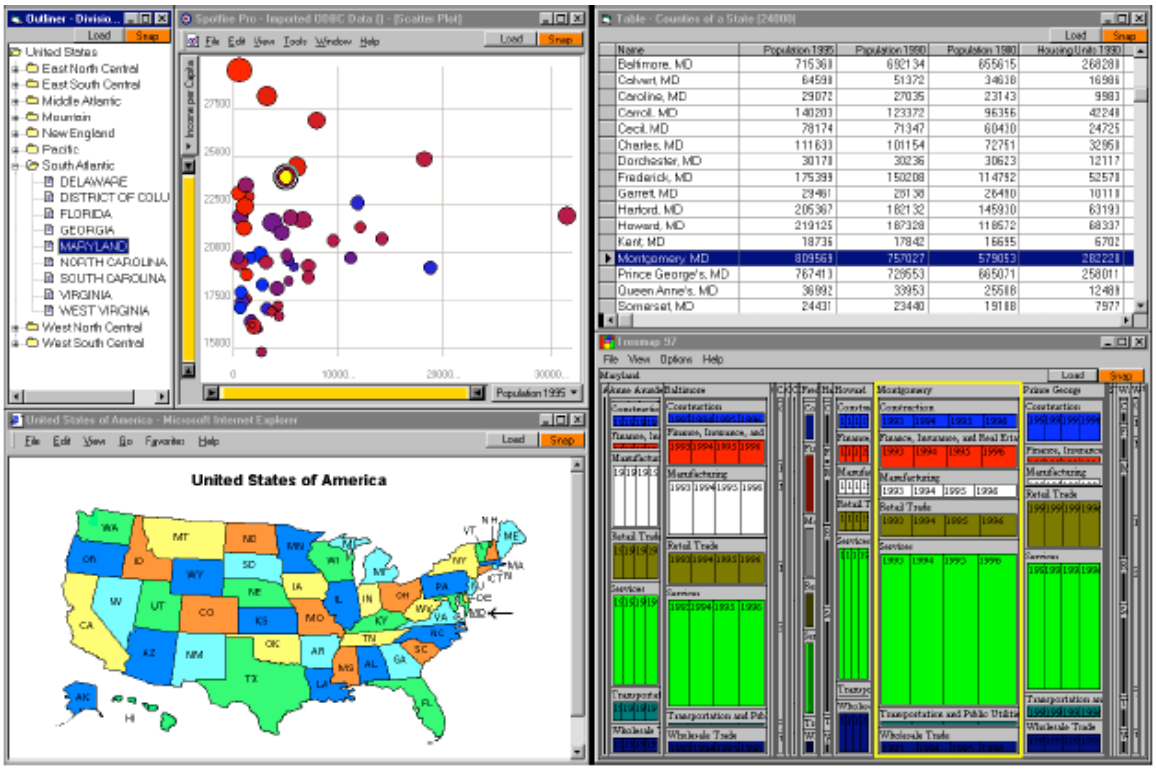

Figure 10 Snap together visualization

2.5 Web-based Charts

Businesses are changing the way they use their software. The traditional approach is to install each instance of an application on individual computers. The problem with this approach is that only one user can access the application at any given time and the application is tied to a single computer. It also costs more to maintain the software because the administrator has to look after many applications on many machines. The client/server architecture was developed to overcome these limitations by centralizing the

most of the application on a single server. The application can be accessed by smaller client programs on different computers which are mainly user interface programs. Centralizing the main application on the server lowers the cost of maintenance because the administrator only has to make changes on a single machine. The recent popularity of the Internet made the client/server architecture a natural choice for businesses to use. Applications can be deployed efficiently and economically because the server and clients can be located anywhere in the world regardless of where one is to the other.

The development of the Common Gateway Interface (CGI) [29] framework for delivering dynamic content and the HTML FORM tag set allowed users to enter and select data through browsers. It started to replace desktop based UIs as the platform for data entry and retrieval. The simplicity of early HTML forms was both their strength and weakness. Early HTML forms consisted of input fields that included text boxes, checkboxes, radio buttons and select lists. All of the processing was done at the server after the user pressed the "SUBMIT" button. The problem with this approach is that there is often a significant pause between pages. The lack of client-side logic meant that any updates required the screen to be redrawn and server had to be frequently accessed. This was fine for basic data entry applications but it was not suitable for more sophisticated data entry and validation applications that require fast screen updates.

During the late 1980's and early 1990's, the client-server architecture became popular and moved some of the logic from the server to client side. Netscape introduced a client side scripting language called Livescript [30] which later became Javascript. Javascript can access FORM elements and can do things like data validation, dynamic formatting, and animation. There have been numerous updates and different versions have appeared. This has led to incompatibilities between browsers of different vendors and even different versions. Currently, Microsoft's Internet Explorer version 5 has almost a 90% market share as of April 2002 [31].

One of the first effective methods of deploying applications through the Internet was to use Java applets. A Java applet is a small application that runs inside a web browser. They behave almost like a regular Java application which means that they can have rich graphics and controls. However, there are limitations. They cannot load libraries or define native methods. They cannot access local files or databases and they do not have

access to certain system properties. These are mainly due to security concerns. The most notable limitation is that browsers require a plug-in before they can run applets. Applets are not completely secure because executable code is sent through HTTP and a number of attacks are possible through this feature. Decompilers can generate source code from the byte code which exposes intellectual property. Finally, applets violate the principle of thin clients because application logic should not be run on the client.

The latest trend in enterprise applications is the "zero footprint" architecture which simply means that the application can be deployed to a web browser without the need for plug-ins. The reason why many organizations are moving towards the "zero footprint" architecture is because of security and ease of maintenance. Plug-ins can have security holes in them that may compromise the data being sent. This prompts for companies to come up with updates or replacements for the plug-ins which may require the administrator's permission. One plug-in may affect another and this may prevent the application from being deployed effectively.

Another concern is that user interfaces still need to have the same functionality and response times in HTML based UIs as those found on standalone applications. The usability of the application significantly affects the productivity of the user.

Goel [32] developed a web based tool called VizCraft (Figure 11) to visualize multidimensional data for aircraft design. It uses applets to display parallel coordinates [33] which represents each attribute as a vertical axis and spaces them evenly out horizontally. The lines that are drawn through the axes represent the data points of a record in the dataset. They are drawn in different colors below to differentiate one record from another. There are two types of axes in parallel coordinates. One type is continuous where data points can exist anywhere within the axis. This type of axis is often used to visualize numerical data where the range of values is continuous. The other type is discrete where data points can only exist at certain places within the axis. These positions are determined by segmenting the axis by the number of possible values the attribute can have. Discrete axes are often used to visualize categorical data where there are only a limited number of possible values.

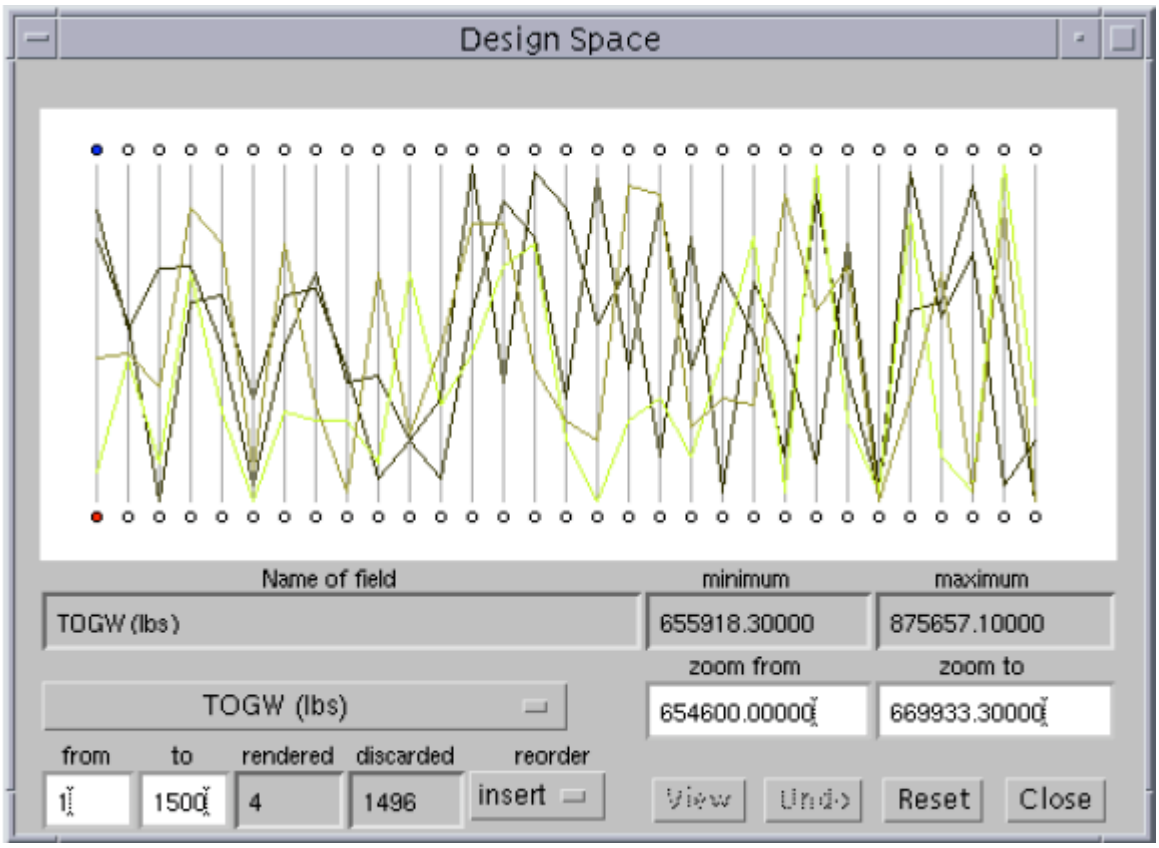

Figure 11 Parallel Coordinates in Vizcraft

The tool was useful for navigating through many dimensions of data which helped the designers to reduce the search space of design parameters by carefully selecting areas of interest. Originally, the designer had to manually change the design parameters, perform the run, and then see the results using a separate plotting package. VizCraft could do it with a few clicks of the mouse. The added benefit of the VizCraft tool was that it encouraged the designers to use the tool more and in new and innovative ways. They tried different parameters more often and that would cause failures which led to the discovery of new bugs in their code.

As good as applets appear to be, there are limitations. Applets have not been successful because of the following reasons [34]

1. Browser incompatibility. Applets behave differently in different browsers. They would look and feel differently or used different API's. This meant testing had to be performed for each different browser.

2. Slow download and startup of applets. Applet classes can take a long time to download. Furthermore, they are treated as regular web files so they can be erased and have to be reloaded each time a page is accessed.
3. Unpredictable behaviour on different operating systems. Applets that work on one system might not necessarily work the same way on another and sometimes it may crash the browser altogether.
4. Need for Plugin. Sun introduced the Java Plugin as a solution for browser problems. This provides consistent behavior across browsers and complete support for the latest APIs. However, one problem is that it is a huge download. Version 1.3 of the Java Plug-in is approximately 5 MB.
5. No standard security model. Explorer and Navigator do not share the same security model for applets. For example, Explorer uses its proprietary Authenticode, and Navigator uses its proprietary Object Signing.
6. Aesthetics. Java applications typically do not have the same level of polish and user friendliness as Windows applications.

2.6 Machine Learning

Machine learning uses induction algorithms for knowledge discovery [35]. This thesis uses the C4.5 [36] decisions tree to generate rules that classify data into frequently occurring groups. The main operation in the decision tree algorithm that splits the data is called information gain. Information gain is the expected reduction in entropy based on a value from a set of all possible values for an attribute [37].

$$\text{Gain}(S,A) = E(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} E(S_v)$$

$\text{Values}(A)$: set of all possible values of attribute A
S_v: subset of S for which A has value v
$|S|$: size of S; $|S_v|$: size of S_v

The basic algorithm for decision tree is given below [38]:

1. Choose an attribute that best differentiates the output attribute values.
2. Create a separate tree branch for each value of the chosen attribute.
3. Divide the instances into subgroups so as to reflect the attribute values of the chosen node.
4. For each subgroup, terminate the attribute selection process if:
    a. All members of a subgroup have the same value for the output attribute, terminate the attribute selection process for the current path and label the branch on the current path with the specified value.
    b. The subgroup contains a single node or no further distinguishing attributes can be determined. As in (a), label the branch with the output value seen by the majority of remaining instances.
5. For each subgroup created in (3) that has not been labeled as terminal, repeat the above process.

The experiments were done using a Java Machine Learning application called WEKA. It implements the C4.5 algorithm in a JAVA package called J48. It also has an attribute filtering feature that determines which attributes give the most information gain. This is useful for reducing the size of the resulting decision tree.

2.7 Usability Studies

Usability studies are needed to gauge how effective a system is for the user. They are typically done by having the user perform predetermined tasks and by observing their reactions. The tasks are designed to test the effectiveness of the system's interface for solving problems such as finding patterns from a large dataset. The results can be obtained through videotape, interview questions, questionnaires or general observations. The results are then compared to a sufficiently large sample size in order to determine their significance.

3 Introduction to Multi-Dimensional Models and OLAP

This chapter introduces the concepts of multi-dimensional modeling and OLAP analysis which will be used later on.

3.1 Multi-dimensional Models

Multi-dimensional modeling is a technique that simplifies the schema of a database by centralizing relationships between dimension tables and fact tables. The benefit of this strategy is that the models are much easier to understand and are faster to query compared to Entity Relationship (ER) models. Dimensional models contains the same information as ER models but are organized for better understandability, and performance. An ER model can be decomposed into multiple dimensional models. The goal for ER models is to minimize data redundancy and to be able to support quick transactions for updates and inserts of data. However, it does not support quick or easy queries. There are typically many tables in the ER model and this makes it hard to form queries because there can be many table joins for a query. It also makes it hard for the user to easily understand the model if there are too many tables in the model because there are just too many relationships between the tables to keep track of.

In contrast to the ER model, multidimensional models are designed to solve complex queries in real time. This means that they need to provide answers quickly. The key to the multidimensional data model is that it enforces simplicity. Simplicity allows users to understand the database and it allows for easier and more efficient navigation [39].

3.1.1 OLAP

Current techniques use spreadsheet type interfaces called crosstabs where values are shown on a grid and the columns and rows can be expanded or collapsed to give the user different levels of detail. These values are then typically shown on a bar or line chart to show how the values change with time. The problem with this type of interface is that the user cannot easily visualize the different dimensions because the user has to go through many steps to generate a chart for just even one dimension. Furthermore, most of these tools are application based which means that it can be only accessed locally on one machine.

The multidimensional data model is composed of logical cubes, measures, dimensions, hierarchies, levels, and attributes. It is an integral part of On-Line Analytical Processing, or OLAP. OLAP is a way of structuring data so that it supports fast analysis of shared multidimensional information [40]. OLAP aggregates metrics and groups them by a defined set of identifiers so users can interactively slice the data and drill down to the details they are interested in.

The process of looking at different dimensions of data by slicing and dicing and drilling up and down the aggregates of data are the main tools in OLAP. OLAP helps in the early stages of the knowledge-discovery process by identifying exceptions and important variables, or finding interactions and associations.

3.1.2 Multidimensional Star Schema

The Star Schema is a relational design that implements a multidimensional model on a relational database. It is made of a single fact table that is joined by many dimension tables. The fact table contains numeric or additive data called facts. The facts are measures of performance. These facts are described by the dimension tables. The dimensions determine how the facts are aggregated and they are usually hierarchical. The most common example is the date dimension, where the facts for a date can be aggregated by years, months, and days. The typical model for the Star Schema is shown in Figure 12.

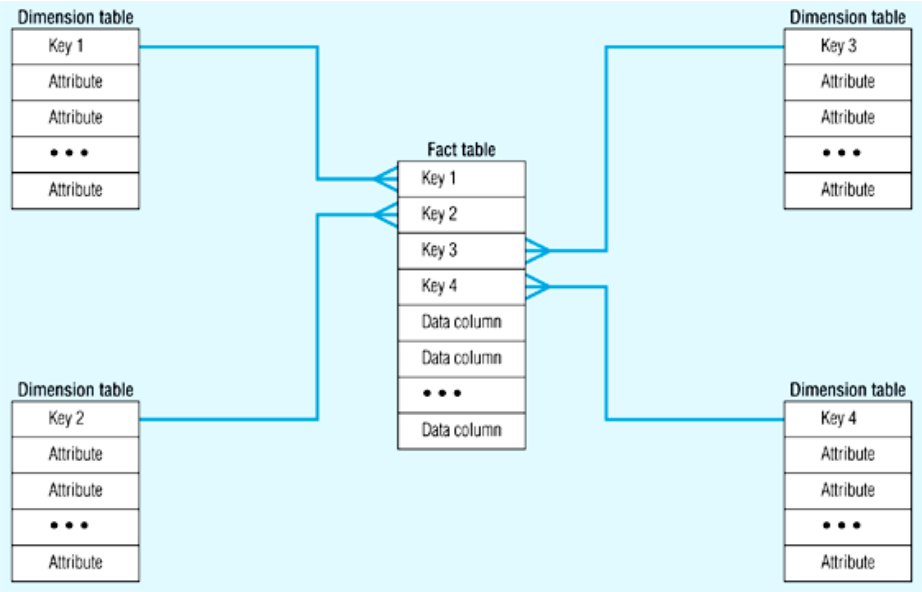

Figure 12 Star Schema

The central table in the star schema model is the fact table which contains measures. These measures are called facts and they are typically numeric and additive across some or all of the dimensions, although they can also be categorical. Fact tables contain a composite primary key, which is composed of several foreign keys (one for each dimension table) and a column for each measure that uses these dimensions.
The facts for the CDC data are the number of reports of different types of symptoms. Each record of patient data has either a 0 to indicate no occurrence or a 1 to indicate an occurrence for each of the symptoms. These values are aggregated and grouped by different dimensions. Each fact table is joined to its respective dimension tables by a set of foreign keys. The dimension table is defined by at least one primary key which maintains referential integrity with the fact table. It contains attributes that can be used to group facts.

The advantage of the Star Schema over an ER schema is that it is very easy to understand, even for non technical business managers. It provides better performance and smaller query times because calculations are pre-computed into aggregates. It is also easily extensible and can handle future changes easily.

3.1.3 Dimension Tables

A star schema stores all of the information about a dimension in a single table. Each level of a hierarchy is represented by a column or column set in the dimension table. A dimension object can be used to define the hierarchical relationship between two columns (or column sets) that represent two levels of a hierarchy; without a dimension object, the hierarchical relationships are defined only in metadata. Attributes are stored in columns of the dimension tables. Dimension tables answer the "why" portion of our question: how do we want to slice the data? For example, we almost always want to view data by time.

3.1.4 Aggregations

Finally, we need to discuss how to handle aggregations. The data in the fact table is already aggregated to the fact table's grain. However, we often want to aggregate to a higher level. For example, we may want to sum reports of symptoms to a monthly or quarterly number. These numbers must be calculated on the fly using a standard SQL statement. The system will be more usable if we can decrease the time required to retrieve higher-level aggregations. This can be achieved by storing higher-level aggregations in the database by pre-calculating them and storing them.

4 Case Study : Data Preparation for Visualization versus Data Preparation for Machine Learning

This thesis uses patient records of vaccinations taken in the U.S.A. It is publicly available from the Centers for Disease Control and Prevention (CDC) [41] in Atlanta, U.S.A. Each patient record lists reported symptoms, the vaccine taken, the date the report, the onset time of symptom, as well as the patient's age, state (location), and sex. We try to find useful information from this data by using two different techniques: machine learning and data visualization. We compare the differences and similarities between these two techniques for data preparation and discuss the results of the study.

4.1 Preparing CDC Dataset

Data preparation for machine learning and data preparation for data visualization are two very different processes. Data preparation is the transformation of raw data into a more usable form to make it easier to analyze. This can include handling missing values, identifying outliers, discretizing values, and decomposing composite data into separate values. We are using a dataset containing records of symptoms found in patients that have taken various vaccinations. The goal is to try and identify any possible links between the symptoms and vaccines and/or other common attributes such as the patient's age, sex, location, vaccination date, and time of symptom's onset. Each record of the dataset has the following information:

1. Event ID – A unique identifier for each record.
2. Sex – Patient is Male, Female, or Unknown.
3. Vaccination Date – Date that patient was vaccinated.
4. Vaccine – Vaccines include: FLU, HEP, PPV, ANTH, TD, SMALL, LYME, FLU&PPV, MMR, VARCEL, HEPA, RAB, MEN, TTOX, TYP, and HEPAB
5. Age – Patient's age.
6. Vaccination Month – The month of the vaccination.
7. Agegroup – Agegroup of Patient.
8. Onset – Date that symptom first reported.
9. List of Symptoms: FEVER, HEADACHE, URTICARIA, DYSPNEA, VOMIT, DIARRHEA, ARTHRITIS, SYNCOPE, ASTHMA, ATAXIA, CONJUNCTIVITIS, TINNITUS, ANAPHYL, EDEMATONGUE, ANA, ALOPECIA, SLE,PERICARDITIS, MYOCARDITIS, SHOCK, COMA, LARYNGITIS - A 0 indicates no presence of symptoms and a 1 indicates presence of symptom.

The information content is sparse because even though there are a total of 13138 records of patient information, only half of the records report one or more symptoms. The number of occurrences of a symptom ranges from 2107 for FEVER, to only 15 for COMA. Machine learning was done using the WEKA Data Mining Tool [42] and visualization was done using the Treemap 4.1 [43] application from the University of Maryland. The dimensional model was implemented on Microsoft Analysis OLAP server [44].

4.2 Data Preparation for Machine Learning

There are numerous machine learning algorithms and techniques but we will only focus on decision trees because rules can be derived from the resulting trees. This makes it easier to identify and group similar patterns. One of the drawbacks of the decision tree algorithm is that tree tends to be very large in size. This is especially true when the data is sparse and there are many attribute values. In order to reduce the size of the trees, we grouped many-valued attributes into categories. For instance, the age of the patient was grouped into 18-29, 30-39, 40-64, 65+, and OTHER. The location of the patients was grouped from individual states into 5 regions NE, NW, SE, SW, and OTHER. The dates had to be decomposed from a single string value into three separate values: day, month and year. We also grouped the records by years and seasons in order to reduce dimensionality. This significantly reduced the size of the trees; however they were still too large because they spanned over several pages. This made it hard to interpret the results because the user has to flip through different pages in order to find all the data. A disadvantage to this approach is that fine grain information is lost. For example, there may be a large occurrence of a symptom in a particular state but this information would be harder to find because the states are grouped into 5 different regions so the user would have to drill into the region to find out the state information. EventID is unique for each record so we do not include it for classification. After filtering out the attributes that contributed the least amount of information, we determined that the following attributes gave the best results: Sex, Region, Year, Agegroup, Vaccine, and Season.

Analyzing all 22 symptoms simultaneously would not be feasible because the tree would be too large, making analysis too difficult. Instead, we focused on one symptom at a time. So for each trial we used the same 6 attributes (listed above) and varied the

symptoms. The makes it harder to determine if there is any relationships between the symptoms because the trials are run independently. A possible solution to this problem is to first perform clustering on the data to determine if there are any links between the different symptoms, vaccines, or patient information and group the similar data into smaller subsets of data for decision tree analysis.

4.3 Data Preparation for Data Visualization

The purpose of data visualization is to organize and present the data in a way that makes it easy for the user to explore the data so that he can find associations, patterns, or identify outliers [45][46]. This is different from machine learning where the data is prepared for a classifier that generates rules or categorizes the data.

Data preparation for visualization tries to provide as much flexibility as possible for the user to navigate through the data. One way to achieve this is to use multi-dimensional modeling. A star schema model for the CDC dataset is shown in Figure 13.

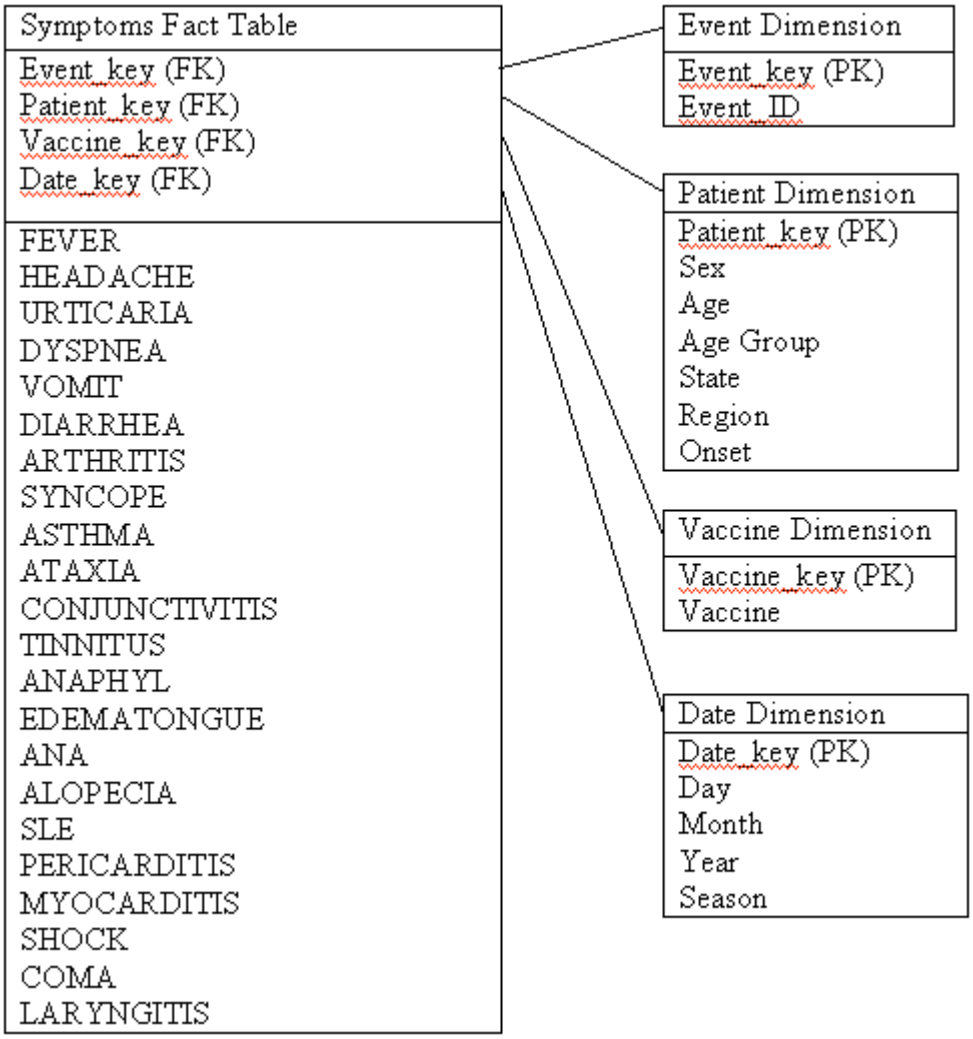

Figure 13 Star Schema Model of CDC Dataset

4.4 Visualizing the CDC Dataset

Some of the techniques used in machine learning, such as clustering and feature selection, can help to reduce the dimensionality of attributes for visualization by grouping the data into smaller sets. There is no point in trying to display data by the EventID since each attribute value is unique so there is no grouping possible. Grouping the dates into seasons and states into regions can aid in the viewing of the data because it is easier to navigate through less data. The difference between the models for data visualization and machine learning is that only the grouped data is used in machine learning whereas the model for data visualization keeps the fine grain information such as the day and month in the date dimension as opposed to keeping only the year information. The user is able to

explore the data through any combination of the attributes in the different dimensions. For example, the data can be filtered first by season and then by month within the season. The dimensional model in Figure 13 lists the symptoms as separate facts but it is also possible to list the symptoms in a separate dimension so that the symptoms themselves can be compared to one another.

The effectiveness of both techniques can be illustrated by the following example. Figure 14 shows the number of instances of each symptom (left column) grouped by the different vaccines (top row).

	Vac																
Data	ANTH	FLU	FLU_PP	HEP	HEPA	HEPAB	LYME	MEN	MMR	PPV	RAB	SMALL	TD	TTOX	TYP	VARCE	Total
Alopecia	8	2	1	16	1	2	0	1	0	1	0	2	1	0	0	0	35
Anaphyl	2	12	1	10	0	2	0	0	3	6	3	1	1	1	1	1	44
Arthritis	12	15	4	26	4	0	103	0	6	10	2	6	3	2	0	0	193
Asthma	7	52	7	15	2	0	4	1	6	14	3	5	4	0	1	0	121
Ataxia	5	15	3	21	2	1	17	0	1	6	3	1	3	1	0	0	79
Coma	0	4	1	3	1	0	0	1	0	1	2	0	1	1	0	0	15
Conjunctiviti	6	13	1	5	1	0	3	1	3	3	2	24	2	0	2	1	67
Diarrhea	26	86	11	42	6	8	7	4	12	23	12	40	5	1	11	6	300
Dyspnea	36	213	22	64	11	6	4	5	17	59	8	105	20	3	3	7	583
Edematongu	2	21	2	8	0	2	0	1	2	3	1	1	4	1	1	1	50
Fever	88	487	171	130	22	20	35	35	81	408	46	331	168	23	18	44	2107
Headache	120	268	46	146	12	13	55	39	43	101	49	250	64	13	6	13	1238
Laryngitis	1	8	2	2	0	0	0	0	0	2	0	2	0	0	0	0	17
Myocarditis	1	2	0	1	0	0	0	0	0	0	0	22	2	0	0	0	28
Pericarditis	1	1	0	5	0	0	0	0	0	1	1	24	1	0	0	0	34
Shock	1	6	1	2	4	1	0	0	0	3	0	0	2	0	0	0	20
Sle	2	1	1	12	0	0	4	0	0	1	0	1	0	0	0	0	22
Syncope	13	62	7	28	4	3	0	15	9	12	2	16	10	2	2	1	186
Tinnitus	17	13	1	5	2	3	9	0	4	1	2	3	4	0	0	0	64
Urticaria	52	279	33	64	9	8	5	12	21	87	26	45	37	5	7	5	695
Vomit	33	140	30	58	9	3	2	8	8	53	16	54	20	2	9	4	449
Total	433	1700	345	663	90	72	248	123	216	795	178	933	352	55	61	83	6347

Figure 14 Occurrences of Symptoms versus Vaccines

There might be a possible relationship between the LYME vaccine and arthritis because arthritis has the highest number of occurrences in LYME out of all the vaccines and also because arthritis is the highest occurring symptom in patients that have taken the vaccination for LYME. If we look at the groupings by region, we see that states in the NE region account for 79 out of 103 reports of arthritis in patients that have taken the LYME vaccine. This is illustrated by Figure 15 and Figure 16 where New Jersey (NJ), Connecticut (CT), Pennsylvania (PA), and New York (NY) have the highest occurrences of arthritis in patients who have taken the LYME vaccine.

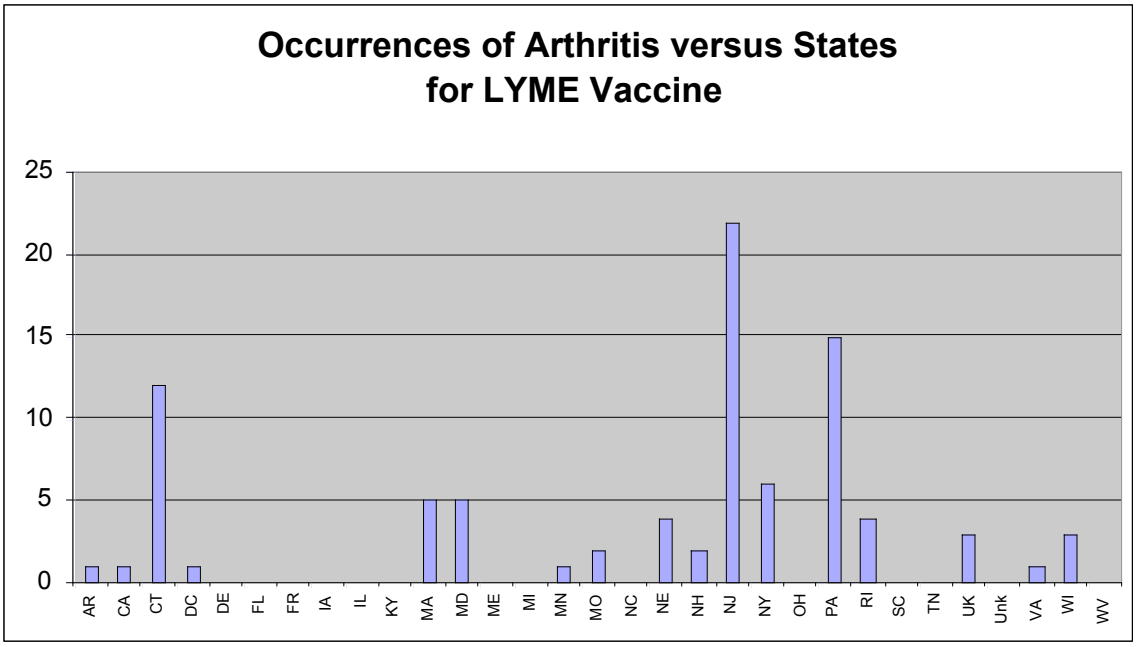

Figure 15 Occurrences of Arthritis versus States

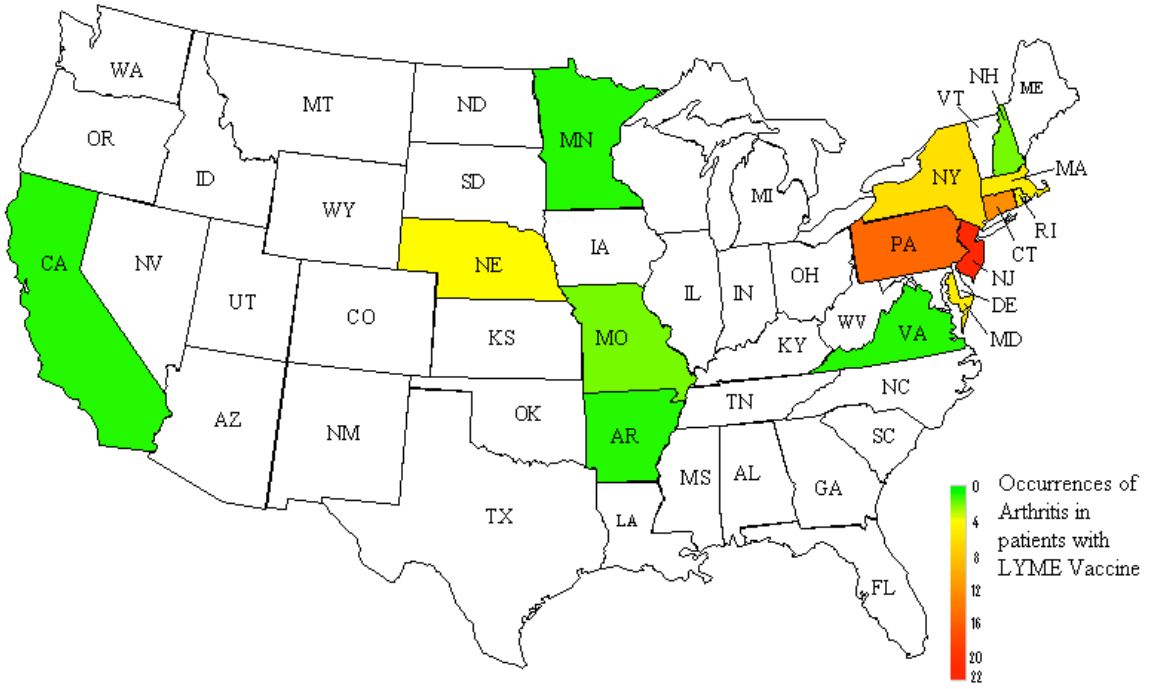

Figure 16 U.S. Map of Arthritis Reports in patients who have taken LYME vaccine

However, this does not tell the whole story. If we look at the pattern of LYME vaccinations across the U.S. as shown by Figure 17 and Figure 18, we see that most of the vaccinations were done in the north eastern states which includes Connecticut (CT),

New Jersey (NJ), New York (NY), and Pennsylvania (PA). These states had the highest number of occurrences of arthritis for patients who took the LYME vaccinations. This is not unusual since it is more likely for states to report higher occurrences if there were more vaccinations.

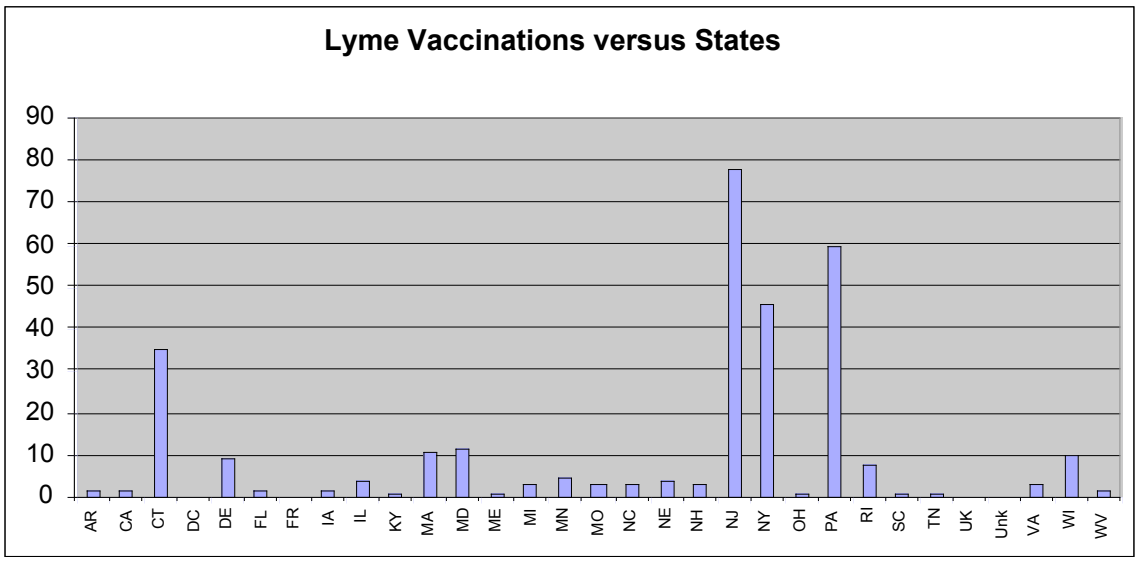

Figure 17 LYME Vaccinations versus States

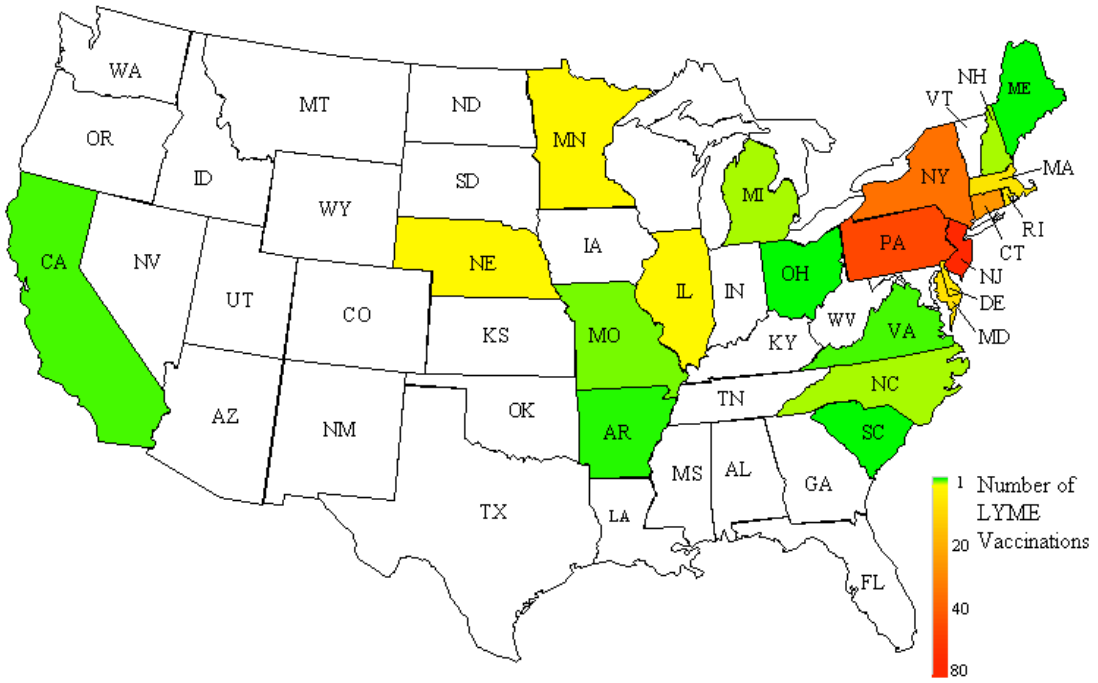

Figure 18 US Map of LYME Vaccinations

It may be clearer to look at the percentages of the number of arthritis occurrences over the total number of LYME vaccinations for each state as shown by Figure 19 and Figure 20. The figures show that Nebraska (NE) has the highest probability of arthritis (100%), followed by New Hampshire (NH) and Missouri (MO). The states with the highest number of occurrences of arthritis (NJ, PA, CT) have smaller probabilities with percentages less than 40%.

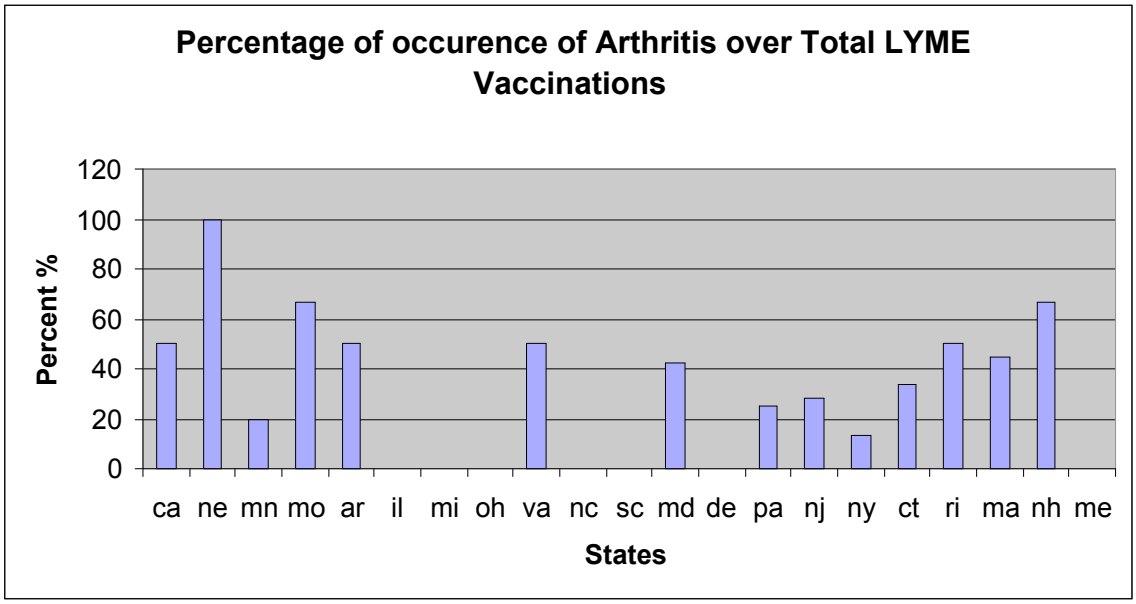

Figure 19 Percentage of Arthritis occurrence versus total LYME Vaccinations

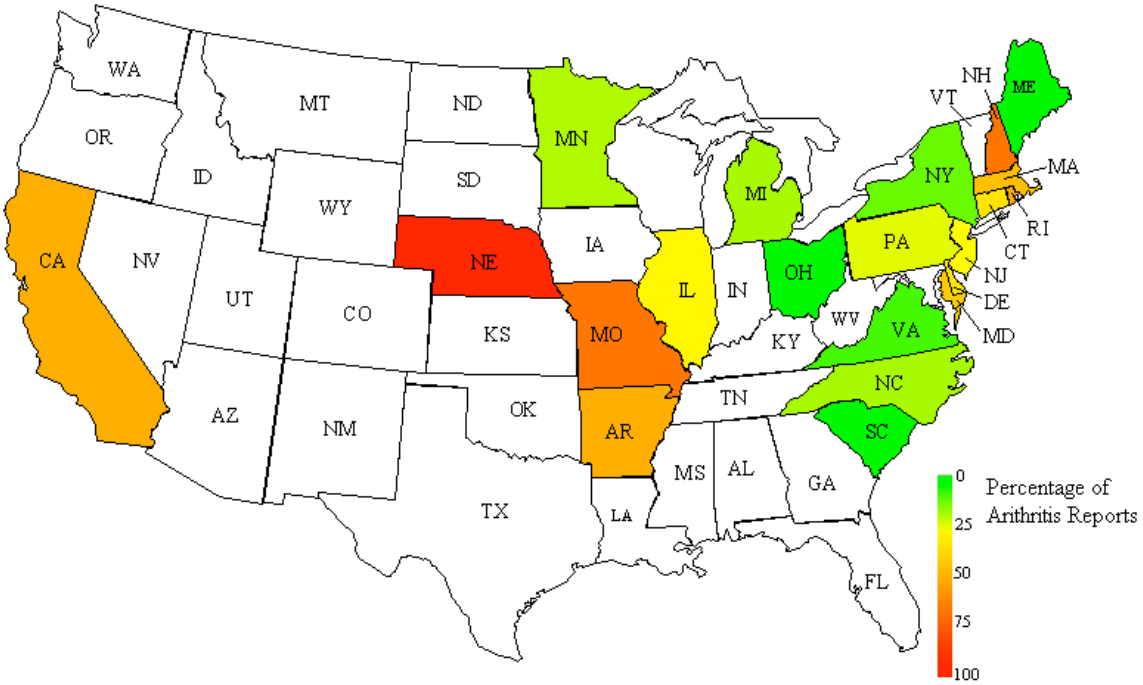

Figure 20 US Map showing Percentages of Arthritis reports to LYME Vaccinations

Figure 19 and Figure 20 may be misleading because they only show the percentages of occurrence of arthritis for each state but they do not show the total number of LYME vaccinations for each state. One way to display both the percentage and the total number of vaccinations is to use a layout technique called a treemap as shown in Figure 21. The size of each rectangle represents the proportion of LYME vaccinations for each state and its color represents the percentage or probability for the occurrence of arthritis. Red represent higher percentages while the lighter green represent smaller percentages. The treemap display shows us that states with the highest number of LYME vaccinations actually have smaller probabilities for arthritis than the states with lower number of LYME vaccinations. This is because the largest rectangles (pa, nj, ny, ct) are more green than the smaller rectangles which means that states with more vaccinations are less likely to report symptoms.

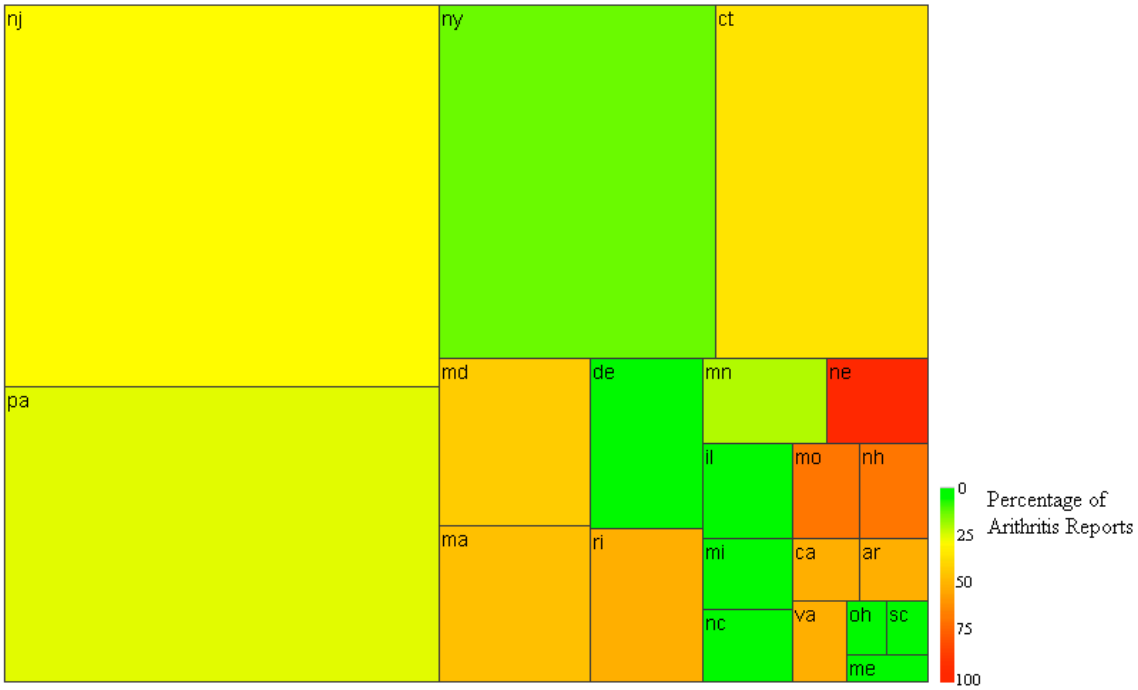

Figure 21 Treemap of Percentages of Arthritis reports to LYME Vaccinations

If we focus on the states with the highest occurrences of arthritis we can find some interesting patterns. Figure 22 is a treemap that shows the distribution of occurrences of arthritis in different age groups. The boundary rectangles that are labeled with state abbreviations represent the states and the nested rectangles represent the age groups. The size of the rectangles represents the number of vaccinations and the color represents the distribution of arthritis occurrences. Red indicate higher occurrences and green indicate fewer occurrences.

It was found that people within the 40-64 agegroup had the most reports of arthritis. This is not surprising because people in this agegroup are more likely to get arthritis due to their age. There was one case in New Jersey of a patient in the 18-29 age group that did get arthritis and was the only one in that age group that received the LYME vaccine so it shows up as indicating 100% probability. However, this is most likely an anomaly. One interesting finding is that while most occurrences of arthritis is reported by patients in the 40-64 agegroup, the occurrences of patients in the 65+ age group is significantly less. One explanation may be that patients that are 65+ may already have had arthritis so there are fewer new occurrences.

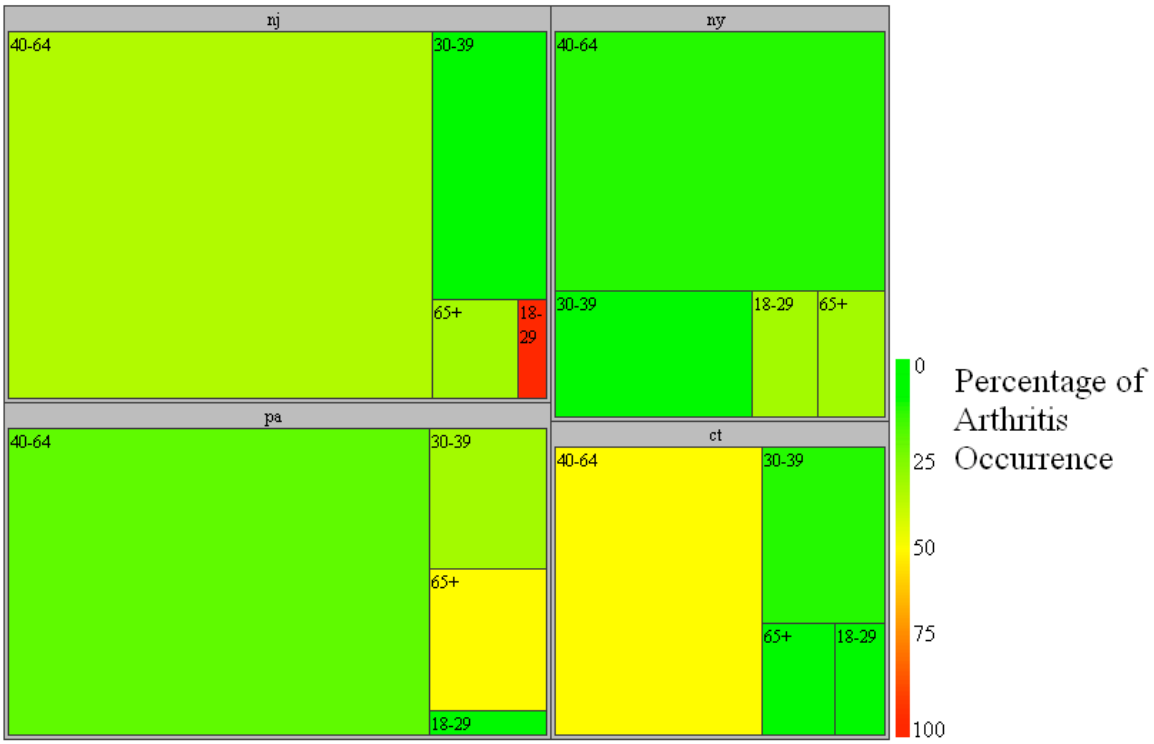

Figure 22 Treemap of Agegroups of patients

Figure 23 is a treemap showing the distribution of the years when the arthritis was reported in patients who have taken the LYME vaccine. It is clear that most occurrences of arthritis occurred in 1999 and 2000. The size of the rectangles represents the proportion of LYME vaccinations and the shade represents the probability of arthritis. We can see that there was one occurrence in 2002 where a patient from New York got arthritis but was also the only one to receive the vaccine. Rhode Island had the highest probability in 2000 followed by Massachusetts for the same year. There were more occurrences in the larger states than the smaller ones but they had smaller probabilities because the larger states also had more vaccinations, so they are greener than some of the smaller states.

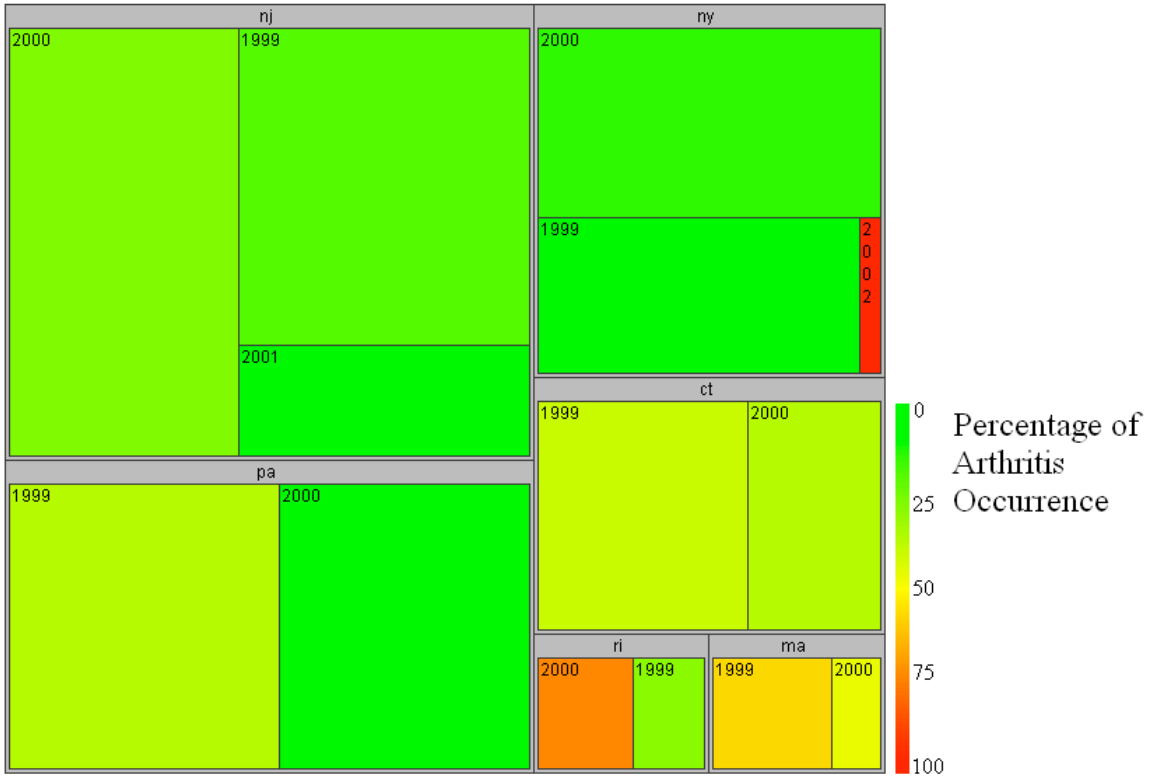

Figure 23 Treemap showing Year of Vaccinations

We wanted to see if the patient's gender showed any patterns but on the whole, the numbers average out to an even approximate ratio of 50/50. However, it is still interesting that some states had significantly higher occurrences in one gender over the other. The results are shown in Figure 24. Connecticut had a lower probability for females than males and there was one case where the patient's gender was unknown but reported as having arthritis. Rhode Island had only 2 males who took the vaccine but both had arthritis whereas the females had significantly fewer occurrences.

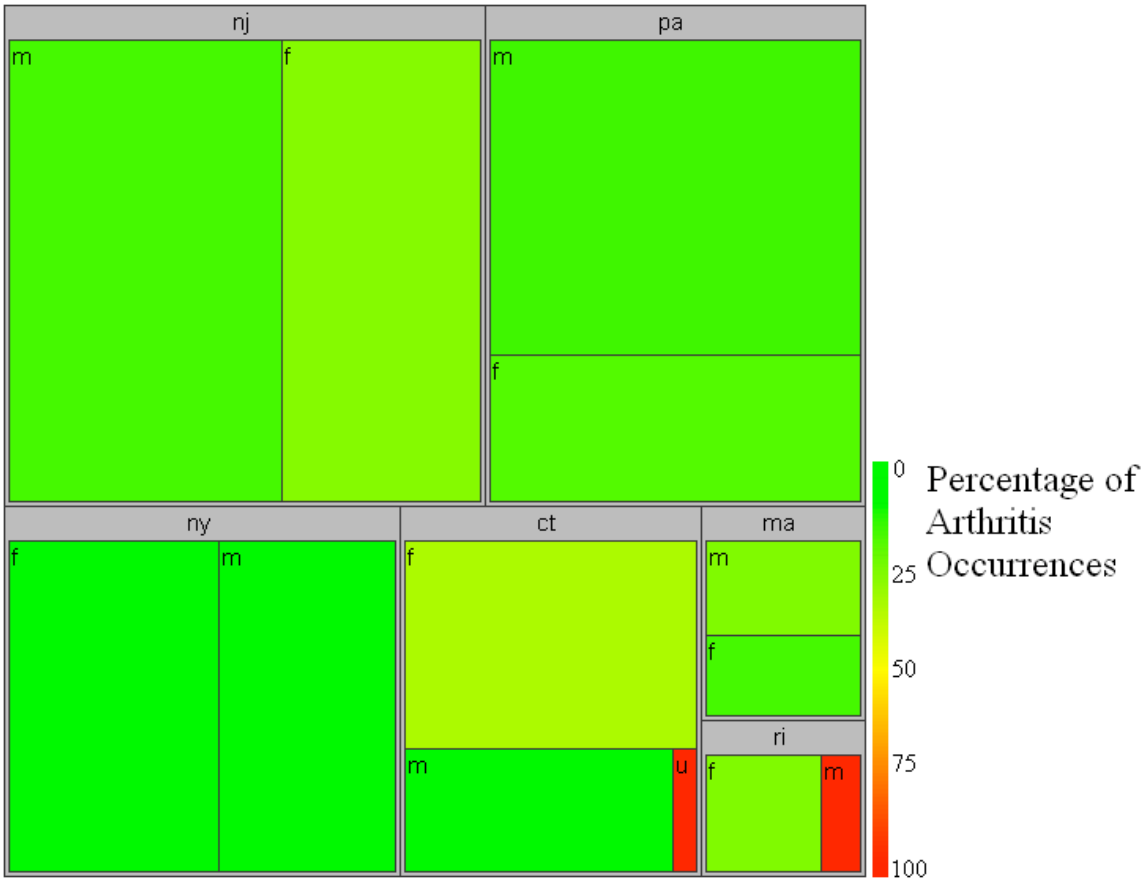

Figure 24 Treemap show Sex of Patients

Multi-dimensional analysis allows for flexible navigation control that allows the user to zoom into subsets of data. This is useful to confirm patterns of data as illustrated by the above examples.

4.5 Case Study in Machine Learning

In this case study, we apply machine learning techniques to try and find interesting trends and patterns in the CDC dataset and to see how the results compare with the visualization techniques used above. Instead of the star schema model, flat files were used to generate decision tree rules using the J48 algorithm in WEKA. The original date format was a composite value that joined the day, month, and year into one string. We separated these values into three separate attributes: day, month, and year. They were then grouped by seasons and years, ignoring the days since it is unlikely that days will show any patterns. We grouped the states into 5 different regions: NE, NW, SE, SW, and OTHER. A filter was applied to determine which of the attributes gave the most

information gain. The following attributes were found to provide the best tree: Sex, Region, Year, agegroup, vaccine, and Season. Finally we focused only on one symptom, Arthritis, in order to keep the decision tree small. We can already see that a lot of the fine grain information is lost. The individual states and dates have been grouped. The star schema model kept all the fine grain information along with the grouped data.

Supervised machine learning algorithms such as decision trees require at least two sets of data. Training data for learning to build the hypothesis or decision tree rules, and testing data to test how accurate the rules are. This is a problem for small data sets because there might not be enough data to build a meaningful decision tree. One way to overcome this problem is to reuse the data for both testing and building the tree; the cross validation method is a standard way to do this [47]. Visualizing data does not suffer from this problem because all of the data is used in building the star schema model.

We encountered a problem using this data set in our initial trial. The generated decision tree had only two branches, and all the values were grouped into one of the branches. We tried different levels of aggregation but often, the trees would still be too large to inspect visually.

In order to overcome this problem we used a technique called Boosting [48]. Initially each record in the dataset is given equal weight. The algorithm is iteratively applied to each subset of data. After each iteration, the records that give the weakest hypotheses have their weights increased. This forces the learner to focus on the misclassified samples. The result is that decision trees with the most weight have the least amount of classification error.

Looking at the decision tree with the most weight (Figure 25), we find a branch (vaccine = LYME) where many examples had traversed.

```
Year <= 2000
|   vaccine = HEPA
|   |   Year <= 1999: 0 (13.19)
|   |   Year > 1999: 1 (35.05/1.01)
|   vaccine = SMALL: 0 (29.94)
|   vaccine = TTOX: 0 (10.15)
|   vaccine = FLU
|   |   Sex = M: 0 (25.88)
|   |   Sex = F
|   |   |   Region = NE
|   |   |   |   Year <= 1999: 1 (79.74/11.67)
|   |   |   |   Year > 1999: 0 (6.09)
|   |   |   Region = SE: 0 (5.07)
|   |   |   Region = NW: 0 (4.57)
|   |   |   Region = SW: 0 (9.64)
|   |   |   Region = OTHER: 0 (7.1)
|   |   Sex = U: 0 (1.52)
|   vaccine = RAB: 0 (18.27)
|   vaccine = LYME: 1 (3379.63/112.15)
|   vaccine = TYP: 0 (5.58)
|   vaccine = FLU_PPV: 0 (11.67)
|   vaccine = MEN: 0 (8.63)
|   vaccine = MMR
|   |   Year <= 1998: 0 (19.79)
|   |   Year > 1998: 1 (34.54/0.51)
|   vaccine = HEPAB: 0 (7.61)
|   vaccine = PPV: 0 (51.76)
(etc…)
```

Figure 25 Most heavily weighted Decision Tree of CDC Data set after boosting

The most heavily weighted tree after boosting gives the most accuracy in terms of classification but it does not give too much information about trends or patterns. The LYME branch indicates that the records were reported at or before the year 2000. The 1 after the colon indicates that the LYME vaccine was taken and the number in the brackets give a relative indication of how many records fit the hypothesis. The first number is an indicator of correctly classified instances, and the second is an indicator of incorrectly classified instances.

The tree with the next highest weight (Figure 26) gives more information. We can see that many reports of arthritis occurred before the year 2000, the vaccine taken was LYME and that the major region of occurrence is in the North East (NE) of the U.S. This is because the sum of all the bracketed numbers under NE with branches having a 1 is greater than the other regions. Looking at the next level, we can see that the age group with the most occurrences is between 40-64 years.

```
Year <= 2000
|   vaccine = LYME
|   |   Region = NE
|   |   |   agegroup = 40-64
|   |   |   |   Sex = M
|   |   |   |   |   Season = FALL: 0 (33.1)
|   |   |   |   |   Season = WINTER: 1 (68.13/14.19)
|   |   |   |   |   Season = SUMMER: 1 (283.86/104.03)
|   |   |   |   |   Season = SPRING
|   |   |   |   |   |   Year <= 1999: 0 (157.05/71.93)
|   |   |   |   |   |   Year > 1999: 1 (87.05/33.1)
|   |   |   |   |   Season = UNKNOWN: 1 (69.07/33.1)
|   |   |   |   Sex = F: 1 (815.68/222.24)
|   |   |   |   Sex = U
|   |   |   |   |   Season = FALL: 1 (0.0)
|   |   |   |   |   Season = WINTER: 1 (0.0)
|   |   |   |   |   Season = SUMMER: 1 (0.0)
|   |   |   |   |   Season = SPRING: 1 (22.71/4.73)
|   |   |   |   |   Season = UNKNOWN: 0 (4.73)
|   |   |   agegroup = 18-29: 0 (111.62/35.97)
|   |   |   agegroup = 65+
|   |   |   |   Season = FALL: 1 (0.0)
|   |   |   |   Season = WINTER: 1 (17.98)
|   |   |   |   Season = SUMMER: 1 (87.05/33.1)
|   |   |   |   Season = SPRING
|   |   |   |   |   Year <= 1999: 1 (40.69/4.73)
|   |   |   |   |   Year > 1999: 0 (14.19)
|   |   |   |   Season = UNKNOWN
|   |   |   |   |   Sex = M: 0 (9.46)
|   |   |   |   |   Sex = F: 1 (17.98)
|   |   |   |   |   Sex = U: 1 (0.0)
|   |   |   agegroup = 30-39
|   |   |   |   Year <= 1999
|   |   |   |   |   Season = FALL
|   |   |   |   |   |   Sex = M: 0 (4.73)
|   |   |   |   |   |   Sex = F: 1 (22.71/4.73)
|   |   |   |   |   |   Sex = U: 1 (0.0)
|   |   |   |   |   Season = WINTER: 0 (4.73)
|   |   |   |   |   Season = SUMMER
|   |   |   |   |   |   Sex = M: 1 (32.17/14.19)
|   |   |   |   |   |   Sex = F: 0 (4.73)
|   |   |   |   |   |   Sex = U: 0 (0.0)
|   |   |   |   |   Season = SPRING: 1 (77.59/23.64)
|   |   |   |   |   Season = UNKNOWN
|   |   |   |   |   |   Sex = M: 1 (22.71/4.73)
|   |   |   |   |   |   Sex = F: 0 (9.46)
|   |   |   |   |   |   Sex = U: 1 (0.0)
|   |   |   |   Year > 1999
|   |   |   |   |   Season = FALL
|   |   |   |   |   |   Sex = M: 1 (17.98)
|   |   |   |   |   |   Sex = F: 0 (4.73)
|   |   |   |   |   |   Sex = U: 1 (0.0)
|   |   |   |   |   Season = WINTER: 0 (9.46)
|   |   |   |   |   Season = SUMMER: 0 (47.29)
|   |   |   |   |   Season = SPRING: 0 (55.81/17.98)
```

```
|   |   |   |   |   Season = UNKNOWN: 0 (0.0)
|   |   Region = SE
|   |   |   Year <= 1999: 0 (23.64)
|   |   |   Year > 1999
|   |   |   |   agegroup = 40-64: 1 (0.0)
|   |   |   |   agegroup = 18-29: 0 (4.73)
|   |   |   |   agegroup = 65+: 1 (17.98)
|   |   |   |   agegroup = 30-39: 1 (0.0)
|   |   Region = NW: 1 (71.93)
|   |   Region = SW: 1 (17.98)
|   |   Region = OTHER
|   |   |   Sex = M
|   |   |   |   agegroup = 40-64: 1 (245.1/47.29)
|   |   |   |   agegroup = 18-29: 0 (9.46)
|   |   |   |   agegroup = 65+: 1 (22.71/4.73)
|   |   |   |   agegroup = 30-39: 1 (40.69/4.73)
|   |   |   Sex = F
|   |   |   |   agegroup = 40-64: 0 (135.27/35.97)
|   |   |   |   agegroup = 18-29: 1 (17.98)
|   |   |   |   agegroup = 65+: 0 (0.0)
|   |   |   |   agegroup = 30-39: 0 (9.46)
|   |   |   Sex = U: 1 (0.0)
```

Figure 26 LYME branch from CDC decision tree with second highest weight after boosting

The results from machine learning agree with the results from visualization. However, we find that it takes more effort to interpret the rules generated from the decision trees. This is because the user has to look at all of the branches, add all the bracketed numbers and compare them with other branches. This would be much harder if we did not group the states into regions, or if the dates were not grouped into seasons. If the user wants to analyze fine grain data, he must manually filter out the data for the desired domain (e.g. for a particular state or month) and run the decision tree algorithm again. Many of the branches can be ignored because we are only interested in the positive case were reports include the arthritis symptom. These branches have 0's, indicating no arthritis reported.

There is also the problem of how the years were separated. We know from visualization that most of the reports of arthritis occurred between 1999 and 2000. This is not easily seen from Figure 26 because the year is split at the top most level and at different lower branches. This makes it very hard to group the reports and see what the range of years is. All of the sub branches with year >= 1999 have to be summed and compared to other years before one can tell which ones have the most reports.

There are many instances of incorrectly classified data in the tree which lowers the accuracy of the rules. It is nearly impossible to get 100% correct classification for many data sets. Small clusters of associated data are especially vulnerable to this problem because it only takes one or two misclassified reports to generate useless or incorrect branches. Visualization does not have this problem because the user is presented with raw data whereas machine learning infers from the data and presents the user with its hypothesis. For example in Figure 16 we see that California has a few reports of arthritis but if one of those reports was incorrectly classified, then California might not show up in the decision tree at all because there were not enough reports to generate a branch. Moreover, these smaller groups with fewer reports are harder to spot because they may be spread over many branches with lower instances of reports. They are much more visible when they are grouped and presented graphically like Figure 16.

4.6 Discussion

We have shown the similarities and differences between data preparation for data visualization and machine learning. Although some techniques can be applied to both methods such as replacing missing values, cleaning data, and categorizing many-valued attributes, there are fundamental differences. This includes how the data is modeled (i.e. star schema versus flat file format). Another difference is that fine grain information can be kept for visualizing but grouped data may be necessary for machine learning techniques to make the results easier to interpret. Unsupervised machine learning algorithms such as decision trees require at least two data sets, one for learning and the other for testing. Data visualization techniques work with the raw data and do not try to infer any relationships between attributes of the data. It is left to the user to make these inferences.

The decision rules generated from machine learning will rarely be 100% accurate. This is especially true for generating rules from small sets of data. Small sets of data may be spread out too thinly along many branches and may not show up at all. However, they can be easily seen when they are grouped and displayed graphically.

Visualization is useful for OLAP analysis where flexibility is needed. The user is able to explore the data and infer their own hypotheses. Machine learning on the other hand is useful for automated processing where the classifier sorts the data for the user. However,

it is nearly impossible in most cases for the classifier to be 100% correct so visualization may be a way of confirming the results of machine learning.

Other differences are that the data model for visualization is very flexible and robust because it is able to support many types of queries. The data model for machine learning is more rigid and fragile. We had to split the data into separate flat files in order to make different queries, one for each vaccine. Fine grain information such as individual states and months versus regions and years was easily visualized and actually showed interesting patterns in the figures. This information was lost when the data was pruned in order to reduce the size of the decision trees. The purpose of preparing and cleaning data for visualization is to make as much data available as possible for the user to explore. The purpose in machine learning is to increase the accuracy of classification. The comparisons are summarized in Table 1.

	Visualization	**Machine Learning**
Automated	No	Yes
Flexibility	Very flexible, can support almost all types of queries	Sensitive to small changes in data and queries
Fine grain data	Retains all data	Fine grain data is lost if tree is pruned or if attributes are grouped
Effort	Large effort needed initially to build model but little effort required to form queries	Large effort needed to prepare data for each different query
Ease of interpretation of results	Generated charts are very easy and intuitive to interpret	Decision trees are often too large making it hard to interpret Have to add values of branches before comparing results of different groups
Data model	Multi-dimensional model	Flat file
Preprocessing	Data is separated into different dimensions and aggregates could possibly be calculated	Data needs to be partitioned into training and test sets
Handling of small data sets	Supports small datasets Can visualize sparse data	Hard to find results for sparse data
Data cleaning	Helps user to visualize more data	Helps classifier to get more accurate results by providing more data
Application to new data	Have to repeat entire process	Can reuse previously learned concepts

Table 1 Comparison of Visualization versus Machine Learning

5 System Architecture

This chapter focuses on the architecture of our visualization tool (Figure 27). The client is Internet Explorer 5 with the Adobe SVG 3.0 plug-in. User input is sent as HTML form data to the Tomcat 4.0 server where it is transformed into a SQL query and sent to Microsoft SQL 2000 to be processed. The processed information is sent back as a JDBC Resultset object to the Tomcat 4.0 server. Tomcat then processes the records in the Resultset and generates charts as SVG graphics within JSP pages which are finally sent back to the client. The chart is finally rendered by the SVG plug-in within Internet Explorer.

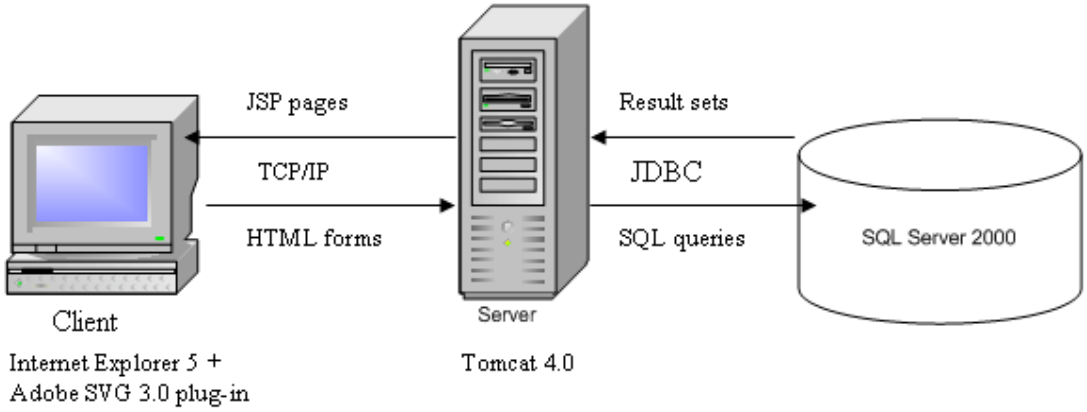

Figure 27 System Architecture of Visualization Tool

5.1 SVG

Scalable Vector Graphics (SVG) is an XML grammar for Web graphics from the W3C. Advantages of SVG over other graphic standards are that it is open source, it is vector based, it requires little bandwidth and finally it provides interactive controls and rich graphical features.

The closest competitor to the SVG standard for interactive graphics for browsers is Macromedia's Flash. Flash provides many of the same features that SVG has and it is already incorporated into Internet Explorer 5. However it is not an open standard and although the plug-in is freely available, development of Flash pages requires expensive commercial tools that must be purchased. There are also licensing concerns if users want to deploy the application on many servers.

We choose to use SVG because it is open and freely available. It will provide a zero footprint user interface platform for enterprise applications because upcoming browsers such as Internet Explorer 6 and Mozilla will have native support for SVG. This can significantly reduce operating costs for large organizations that want to have large scale deployments for their applications. Another advantage of using SVG is that it requires less bandwidth to operate than other products such as Flash because it is a much more compact framework. This means that response times are reduced which leads to better usability.

There is a new standard for developing GUI's using SVG called SVGUI. It is a Graphical User Interface framework written for SVG using ECMAScript. This project is currently in the design phase. The core code includes a W3C DOM implementation, Core and Events. These interfaces are extended to form the SVGUI DOM. A standard set of HTML like controls are being developed. They include: push button, radio button, checkbox, popup menus, text box, sliders, and scroll bars. Some possible features may also include: windows, database enabled widgets, tabs, and tree viewers [49].

5.2 Web Interface

A screenshot of the tool can be seen in Figure 28. The user interface is made of four HTML frames. Two of the frames on the left provide the user controls and the two frames on the right contain the charts. The top left frame contains a select list for every dimension of the data set. The user is able to select any combination of attributes within each list and updates occur automatically after each selection. This provides a quick and easy way to navigate through the data with minimal interaction. The bottom left frame has other controls which determines if the top or bottom chart is updated, which dimension to chart and which type of chart to display.

The tool supports three different types of charts. It uses the bar chart because it is the most general type of display that is common in many visualization applications and because most users are familiar with its simple graphical concepts. Larger values have higher bars while smaller values have shorter bars. The tool also uses a US Map that contains the major states of the U.S.A. because the data contains geographical data. It may be useful to compare symptoms between different parts of the country. The color of each state corresponds to its relative number of occurrences of symptoms. States with

fewer occurrences are colored green while states with more occurrences are colored red. Finally, a treemap is used because it is effective for displaying many attributes simultaneously in a limited area. The size of the rectangles within the treemap corresponds to its relative value. The larger the attribute value is, the larger the rectangle becomes. A highlight feature was provided to make it easier for the user to track a few attributes values among many.

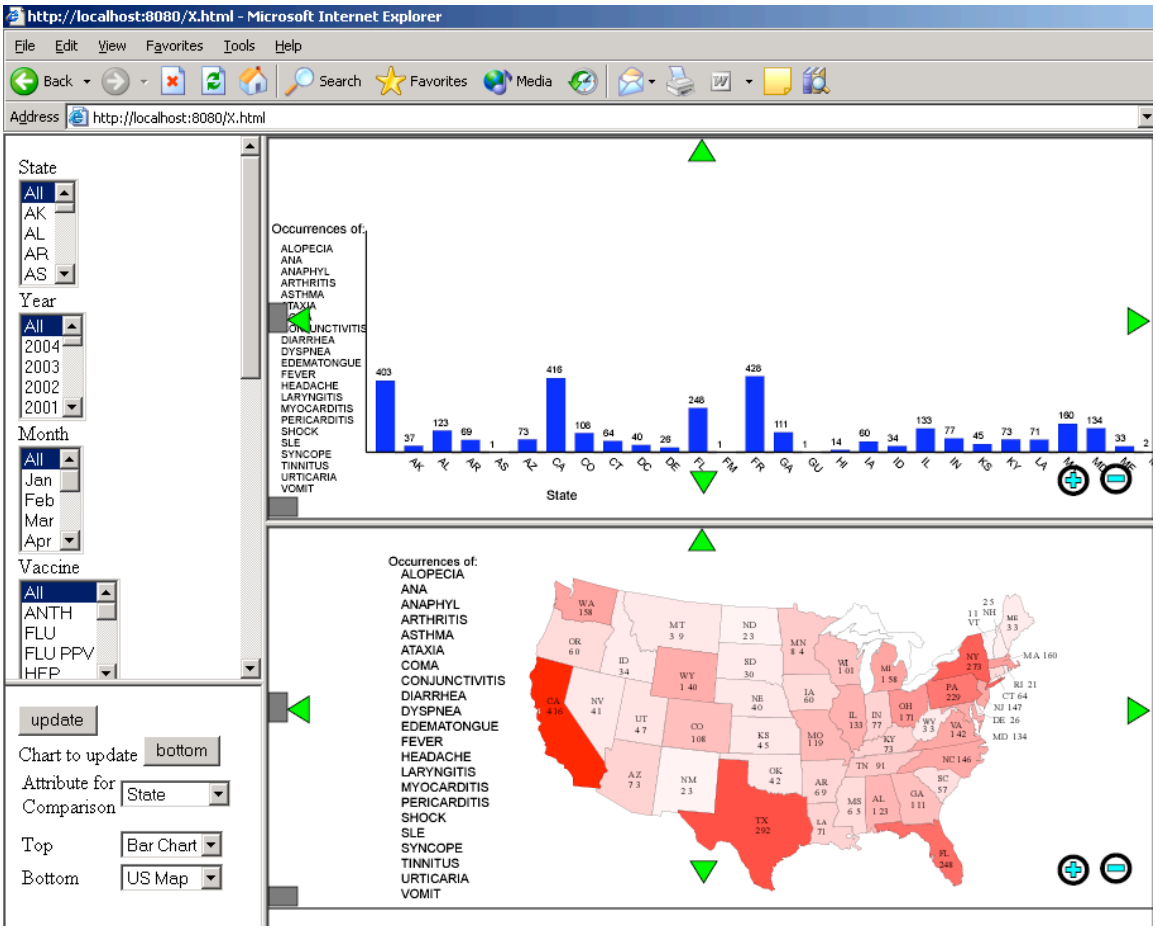

Figure 28 Screenshot of Visualization Tool

6 Methodology for Usability Study

We used graduate students from Computer Science for our study. We followed the general principles of usability testing which includes developing and writing tasks for the test subjects and recording the results through questionnaires and interview questions. Our study was approved by the University of Ottawa's Research Ethics Board.

6.1 Test Users

Our test group consisted of 13 Masters Students in Computer Science at the University of Ottawa. This is sufficient according to [50] for the results to be significant. Two of the test subjects had previous experience with visualization applications and three others have done work with machine learning and data warehousing. All test subjects were considered equal because they were all new to the tool and all were given the same introduction and tutorial. The tasks did not require any specialized knowledge and were general enough that anyone could understand them.

6.2 Tasks for Usability Testing

Task 1

The user was asked to find the years which had the highest occurrence of the symptom URTICARIA. They were asked to try the bar chart first, then the US map, then the treemap. This was done to see how effective each of the different types of charts were for finding patterns.

Afterwards, they were shown the highlight feature which colors particular sections of the treemap bright yellow. The purpose was to see if it helped the user spot trends while drilling through dimensions using the treemap display.

Task 2

The user was asked to find the three states with the largest total number of reports of symptoms. The next step was to compare the charts of the three states that had the highest number of reports of symptoms to see if there were any similarities or differences. The user was allowed to only use one chart first. Then they were asked to use two charts for the same task.

This was done to see if having multiple views helped the user to compare charts.

Task 3

The user was asked to find out which vaccine has the highest number of reports for arthritis. Then the user was asked to find which agegroup had the highest number of reports for the vaccine with the highest number of reports for arthritis and compare it with the total number of patients who have taken any of the vaccines and have reported arthritis. The user was also asked to compare it to the total number of patients who have taken the LYME vaccine and have shown any symptoms, not just arthritis. Finally the user was asked to find other symptoms that have similar occurrences as arthritis for patients who have taken the LYME vaccine.

The goal was to see if the user could effectively drill down into the data and identify trends and patterns within subsets of data.

Task 4

The final task had the user comparing different charts for Onset data and to find which vaccines have similar occurrences.

The objective was to see if the user could navigate through a large search space and use the multi-view display in order to find charts with similar patterns.

7 Results

Task 1

All the test subjects were able to identify that WY and AL have the highest occurrences of UTICARIA but only 3 of the subjects were able to identify the initial rising then falling trend. Most of the users preferred the US Map, then the bar chart, then the treemap in that order. However, they found treemaps to be the best chart to find trends because it could display all of the attribute values on the screen without having to scroll whereas the bar chart display could not fit all of the values onto the screen without scrolling. Nearly all users agreed that the highlight feature was helpful in find trends and patterns in treemaps because it helps them track values of interest.

Most users found it hard to spot trends with the treemap because the size of the rectangles did not change in size with respect to changes within a dimension. The size of the rectangles in a treemap corresponds to the relative values of the attributes. For example, it was easy to spot the relative differences between different states but it was much harder to spot the differences when comparing states between different years.

Some users noticed that symptoms were highest in 2002-2003. All users identified that WY, AL, CA were the highest occurring states. It was easy for them to identify WY and AL because these states were significantly higher than the others. Most users could not keep track of the states in the treemap without the highlight feature. Some users said that this task would be much easier if they could see many simultaneous windows to compare from.

Task 2

All the users were able to identify NY, CA, and TX as the states with the largest occurrences of symptoms. Of the 13 test subjects, 8 users preferred using the bar chart, 3 preferred the treemap, and 2 preferred the US Map when asked to find similarities and differences between the charts. The users that preferred the treemap chart stated the main reason being that they were able to see all of the data on the screen without having to scroll or zoom out and that it was easy to compare the sizes of the rectangles. The 2 users that preferred the US Map said they did not like the bar chart because they had to scroll through the dimension to see all of the data and they did not like the treemap because it was an unfamiliar concept. Most of the users however, preferred to use the bar chart

because its simplicity. All the users immediately knew how to read the bar chart and the US Map but only a few users were able to use the treemap effectively. They preferred using a familiar interface over new type of interface even though it required more effort.

When asked to compare the charts of the 3 states with the highest occurrences of symptoms, 10 out of 13 users said that looking at 2 charts simultaneously made the task significantly easier than looking just at 1 chart. In addition, most users were able to spot small differences between charts when both charts were shown simultaneously.

One significant finding for the US Map was that a few users noticed the larger states first regardless of its color and intensity. This indicates that size has a significant impact on a user's attention.

Almost all users stated that TX and CA were similar because the values for FLU and SMALL vaccines were higher than the other vaccines. A few users stated that NY and CA were similar because both had higher values for the SMALL vaccine.

Some of the users stated that having 2 charts made it easier to compare data but having more charts would be even better. Another suggestion was to overlay the charts to make comparisons easier. Only 2 out of 13 users said that being able to see 2 charts at the same time did not help them for this task.

Task 3

All users found that patients who have taken the LYME vaccine had the highest number of reports for arthritis. They also found that the agegroup that had the highest number of reports among these patients was 40-64 years. When asked how these patients compare with other patients who have taken other vaccines or who have reported other symptoms, almost all users found that the patients who are in the 40-64 agegroup have reported higher occurrences of all symptoms and not just arthritis. Most of the users came to the conclusion that there was no significant contribution of the LYME vaccine to the arthritis symptom and that patients that were in the 40-64 agegroup were more likely to report any kind of symptom and not just arthritis. Only one user stated that LYME was a significant contributor because he observed that patients who have taken the LYME vaccine represent a large proportion of the patients who have reported arthritis as a symptom. Another user stated that patients who were in the 40-64 agegroup were more likely to get arthritis.

All the users used one chart to drill through different dimensions of that data even though they had the option of using two. It was also noted that 11 of the 13 users chose to use the bar chart to compare data. Three users stated that this task was hard to do regardless of which chart was used. Although, it was observed that other users also had a hard time drilling through the data even though they did not explicitly state it. The main reason was they were not able to remember the settings and options selected from previous views of the charts.

Some of the users noted that there were too many parameters to remember and suggested to limit the number of options or controls for the GUI and not to display too many select lists. One possibility to solve this problem is to hide the controls for certain attributes if they do not contribute useful information to the data. This would require some intelligence and preprocessing of the data.

Task 4

When the users were asked to compare the onset profiles for all the different symptoms, the idea was to see if they could successfully navigate through a large search space and to be able to spot patterns or similarities and differences between many charts using the tool. Most users were able to notice that the general trend is that most vaccines have early onset. Sameday onset was usually the highest followed by 1-6 days onset, and then 7-14 days onset. The following days did not show any significant trends. Except for two, all users only used one frame to compare charts even though they were able to use two.

It was noted that all users found this task to be difficult because they were not able to remember all the charts. Two users did not complete the task. Of the ones that attempted to find similar charts, no two answers matched. Each user came up with different answers. One user said that Urticaria and vomit had similar profiles. Another user said that Dyspnea, Edemantongue and Fever, Headache have similar profiles.
Other results included: Myocarditis and Coma, Anaphyl and Asthma, Alopecia and Ana. Four of the users suggested that the task would be easier if some of the work was already done for them such as clustering similar chart together and that being able to see many charts at the same time would also greatly help.

The inconsistent results of this task shows that the tool was not effective for finding patterns or grouping similar charts. Users stated that the tool was adequate for visualizing the data but the task was too difficult because the search space was too large and they were not able to remember all the charts when going through all the symptoms.

Questionnaire

Table 2 shows the results of the questionnaire given to the users.

User	A:Treemap useful	B:Bar Chart useful	C:US Map useful	D:Highlighting useful	E:Navigation easy
1	3.5	4.5	3	4	4
2	2	4	5	5	4
3	3	5	4	3	4
4	4	4	4	5	4
5	3	3	5	4	4
6	3	5	4	4	5
7	4	5	5	5	5
8	3	4	4	4	4
9	4	4	2	2	4
10	5	4	5	5	3
11	5	5	2	4	4
12	4	5	5	5	5
13	2	5	4	4	4
Average	3.5	4.4	4.0	4.2	4.2

Table 2 Results of Questionnaire

Each column labeled by a letter contains the results of a question. There are weighted on a scale of 1 to 5 where 1 means to strongly disagree with the statement and 5 means to strongly agree. The statements and the averages of the responses are listed below. The letters corresponds to the columns in Table 2.

1 – Strongly disagree
5 – Strongly agree

A. I found the treemap to be useful: 3.5/5
B. I found the bar chart to be useful: 4.4/5
C. I found the US Map to be useful: 4.0/5
D. I found the highlight feature to be useful: 4.2/5
E. I found the navigation to be easy: 4.2/5

The results show that most users found the bar chart to be the most useful in general followed by the US Map and the treemap. The reason given by the users was that they

were familiar with bar charts and it was easy to compare attribute values. It was easy read attribute values by comparing the height of the bars. However, some of the users said that the chart would have been much easier to use if they did not have to scroll to see all of the data or if all of the data was visible on the same screen.

The US Map was the second choice by many users because it was also a familiar charting technique and it provided geographic data whereas the other charts did not. But it was only useful for looking at one dimension, states. One weakness that it suffers from the other charting techniques was that it was hard to distinguish between attributes that had similar values because they had a hard time distinguishing one shade of color from another.

Treemap was the least popular type of chart in the study because most of the users were new to the concept and because it required effort to learn how it worked. A few of the users already knew the treemap concept and they preferred to use it over the other charts. This was especially true when they would have to use the scroll feature on the bar chart. The main complaint that users had with the treemap was having to keep track of rectangles as they drilled through different screens because many of the rectangles looked alike and were the same color, white. A highlight feature was provided to overcome this problem. This feature would let the user color a rectangle bright yellow to differentiate it from the others. Once this feature was enabled, most of the users found it much easier to keep track of the rectangles and to find patterns as indicated by the result of question D. of the questionnaire (4.2 out of 5).

Finally the user was asked to assess the overall effectiveness and usefulness of the tool's overall user interface. The result show 4.2 out of 5 which indicates that the tool was found to be very useful by almost all the users. Only one user scored the tool 3 out of 5, indicating that it was only moderately useful.

What features did you like about the system?

The majority of users found that having a multiview to display 2 different charts helped them to find patterns because they did not have to rely on their memory as much. They also liked the ability to view different types of charts simultaneously because the information could be viewed differently. The navigation features for zooming in/out, and

panning was found to be useful by 8 out of the 13 users. The general OLAP type interface coupled with the charts was found to be useful by several users. Almost all of the users liked the highlight feature for the treemap because it made it much easier to track data when drilling through different screens. The users commented that they liked the quick updates of the tool which made it easier to navigate through the different dimensions and the fact that it was web based because it is easily deployed.

What features did you not like about the system?

Many of users complained that they had to scroll through the bar chart in order to get at all of the information and this made it harder to compare charts. Another complaint was that the treemap was not initially intuitive and it was difficult to read the labels because the rectangles were too small and sometimes letters were clipped. Some users found that there were too many options and controls in the user interface which made it harder to navigate through the data.

The US Map was only useful for showing one dimension of the data.

What improvements do you think could be made?

One suggestion was that the task of comparing many charts could be simplified if some preprocessing, such as clustering, classification, or filtering was done for the user. Another suggestion was to have scripting ability so that certain tasks can be automated. Most users noted that it would have been even better if the tool could show many charts at a time and not just have 2. An alternative to this idea was to overlay different charts over one another. The treemap chart could be improved if the size of the rectangles were not just dependent along one dimension. There were a few suggestions on how the selection lists could be improved. One idea was to have the ability to move selected and unselected items between two lists. Another idea was to hide or collapse the lists since having many selection lists makes it harder to find items of interest.

Was it hard to find the information? Why?

Most of the users found that the tool provided a good interface for all the tasks. However, one user commented that some charts did not support certain kinds of analysis

and that having specialized charts would be useful. For example, finding similar charts among many was found to be hard by most of the users.

The most common problem among the users was that it was hard to keep track of all of the data while drilling through different dimensions and going through many screens because they were not able to remember all of it.

The treemap proved to be difficult to use for most of the users because they were not familiar with the concept and because the size of the rectangles were only dependent on the current display not on previous ones. This made it very hard to spot trends and patterns.

Which tasks were easier and which ones where harder and why?

Users found the easiest tasks to be the ones that only required one chart. They also found it easy to do simple tasks such as identifying the highest value of an attribute. They had a much harder time with tasks which required them to keep track of more than a few charts. This was the case when they were asked to find similar charts among many and when they were asked to explore through many different dimensions of data to look for trends and patterns.

We find that the type of chart used for visualization is less important than having navigation controls to drill into the data and having multiple views for comparing charts. Although most users did not like using the treemaps initially, they were able to use it effectively once they learned the concept and how it worked. We learned from the case study in data preparation of the CDC data set that preparing data for machine learning requires a lot of effort because each query requires customized processing. The initial effort to prepare data for visualization was great, but once done, much less effort was required. This was because the star schema model is flexible and can support the aggregate functions required for exploring the data. We also found that preparing data for machine learning is a delicate process because small changes in the data can dramatically affect the results. Data preparation for visualization is much more robust because it supports many types of queries for drilling through many dimensions of data.

8 Conclusion

This chapter presents the findings of our research in developing a web based visualizing tool for large multidimensional datasets.

8.1 Data Preparation

Our case study in data preparation for machine learning and data visualization has shown the importance of preparing data for visualization and how it is significantly different from the process of preparing data for machine learning.

The purpose of preparing data for machine learning is to process the data such that the machine learning algorithm is able to extract as much useful information as possible by either classification or generating rules and associations. This involves cleaning the data which include such things as removing bad data, inserting missing values, and transforming data from one format into another. Sometimes part of the data is used by the machine learning algorithm to for it to learn a concept. The rest of the data is used to test the concept.

We used decision trees for machine learning because rules can be derived from the resulting trees. It was necessary to group the data in order to reduce the number of attribute values. Otherwise, the decision tree would have too many branches. In particular, grouping the states into 5 different regions and the dates into seasons significantly reduced the number of branches in the decision tree. We found that dimensions with many valued attributes resulted in very large decision trees which were unreadable. As a result, it was very difficult to generate rules because the data was too spread out. However, once the many valued attributes were grouped, the resulting decision tree only had two branches and all of the values fell into one branch. In order to overcome this limitation, we used another machine learning technique called boosting which generates many decision trees. Each decision tree is given a weight and a voting scheme determines the final outcome.

This whole process would have to be repeated if we wanted to focus on different parts of the data. In general we find that machine learning requires a lot of data manipulation and is a very delicate process because a small change in the data can have a very large impact on the results. It is also very iterative because it is usually a trial and error process in order to obtain useful information. There is a trade off for generating small decision

trees because the results are more meaningful and easier to manage but find grain information is lost when attribute values are grouped.

Generating a decision tree with approximately 10,000 records took several seconds to complete on a 2.8 GHz Pentium 4. However, this is a relatively small data set and the C4.5 algorithm is not scalable. The time required would exponentially increase because the records are stored cumulatively in memory. This increases processing time as well as memory requirements because the sample size increases with each iteration.

Data visualization requires a different approach to data preparation. The data must support quick queries and be flexible enough to answer many types of queries. We applied the multidimensional model by using the star schema for our data for this purpose.

The visualization study confirmed the results of the machine learning study. However, the user had much more flexibility in navigating through the data and was able to find much more detailed information like fine grain data. Although the data preprocessing for data visualization required a lot of effort initially, subsequent analysis was much easier because the star schema is flexible enough to support many types of queries. Data mining required manual preprocessing to support just one type of query. Furthermore, fine grain information was lost because attribute values were grouped into smaller sets.

8.2 Usability Study

We have developed web based tool that integrates a rich graphical user interface using SVG with a flexible star schema data model. It provides multiple views of charts that make it easier for the user to remember and compare data. This allows users to easily find trends and patterns in large multi-dimensional data sets.

The results of the usability study for our visualization tool shows that it is effective for navigating through large data sets with many records but users were only able to navigate through small search spaces and do simple tasks that do not require lots of memorization. Users were able to easily complete tasks when all of the information was visible to them or when they only had to remember one or two charts. Most of the users were able to complete all the tasks using the navigation controls provided by the web interface but they require more help for complicated tasks that involves a lot of

memorization such as finding similar charts from a set of many charts. Users also had problems remembering their steps when drilling through many levels of the data and comparing charts so they were not able to easily retrieve previous views of the charts. The result was that they lost focus and easily forgot what the charts looked like.

The questionnaires indicate that they found the controls to be adequate although they thought that some of the tasks were too difficult regardless of the navigation tools provided. These tasks required the user to keep track of many charts as they navigate through different dimensions trying to find trends and patterns. Having two frames for multiple views helped the users to remember some of the information but more frames or more views would have made the tasks easier. However, this also brings up the issue of how many views should be supported before the user is overwhelmed by the number of charts displayed or when the charts become too small to read. An alternative solution is to overlay the charts but there is a limit to the number of layers before the display becomes too dense with information to be usable.

Many of the users suggested that they needed some form of help to complete tasks which required them to remember many charts. Machine learning techniques such as clustering and classification could be used to reduce the search space by identifying similar groups of data or areas of interest.

The bar chart was the most used type of chart in the tasks because all the users were familiar with the concept. However, more effort was required for certain tasks because all of the data could not be displayed on screen at the same time. The user would have to use the graphical navigation controls to zoom, scroll and pan in order to see all of the bar chart. This would distract the user's attention from remembering previous values so it would take them longer to complete the tasks. In contrast, the treemap was able to display all of the information on the screen at the time but most users preferred not to use it because they were unfamiliar with concept and it took some effort for them to learn it. There were a few users who where already familiar with how treemaps worked and others who picked up the concept quickly. These users preferred to use the treemap over the bar chart, especially when they had to scroll through a display on the bar chart. Many users preferred to use the US Map whenever the task looked at the state dimension. These

users liked the US Map because it was also a simple and familiar concept and because it provided information about location whereas the other charts do not.

Users had their own personal preferences for different types of charts for different tasks and some charts were more popular than others. However, most users stated that they liked having the option of using different types of charts and would like to see even more types of charts supported in the tool.

Using 13 test users is enough to identify most of the usability problems based on the findings of Jakob Nielsen who found that on average, 10 users found 80% of the problems in a user interface [51]. It would have been interesting to use medical professionals as test users, but it was not essential for our usability tests since the experiment did not require medical knowledge, and the tasks were general enough that they could have been applied to almost any type of data. All of the test users had similar backgrounds and they all were given the same introduction and training for the tool.

9 Future Work

Currently the Adobe SVG plugin viewer does not allow for communication across HTML frames. This limits the interactivity of the user interface because any mouse event in one frame will not have an effect in another frame. This means that there can be no coordination between charts or controls in different frames on the client slide. All information must be passed to the server first. This places more work on the server because of the added bandwidth and processing requirements. Further research is required to determine other possible solutions, such as using servlets or Flash, and to see how effective they are.

The tool could be optimized for even quicker queries by pre-computing aggregates and supporting an aggregate navigator. The tool has tested to support small groups of users but the framework can be extended and be made scalable to support many more users. Some possible approaches to this include adding connection pooling, load balancing across multiple servers, and caching results.

10 References

1 McCormick, B.H., T.A. DeFanti, M.D. Brown (ed), Visualization in Scientific Computing, Computer Graphics Vol. 21, No. 6, November 1987.

2 K.W. Brodlie, L.A. Carpenter, R.A. Earnshaw, J.R. Gallop, R.J. Hubbard, A.M. Mumford, C.D. Osland, P. Quarendon (eds), Scientific Visualization, Techniques and Applications, Springer-Verlag, 1992.

3 Card, S., Mackinlay, J., Shneiderman, B., Readings in Information Visualization: Using Vision to Think, Morgan Kaufmann, 1999.

4 Shneiderman, B., Designing the User Interface: Strategies for Effective Human-Computer Interaction: Second Edition, Addison-Wesley Publ. Co., Reading, MA, 573 pages, 1992.

5 Iarchitech Inc., URL : http://www.iarchitect.com/mshame.htm, December 2004.

6 Drop down menus – use sparingly, URL : http://www.useit.com/alertbox/20001112.html, December 2004.

7 Visual Mining Inc., URL : http://www.visualmining.com/examples/graphs.html, January 2004.

8 Prodacapo Inc., URL: http://www.prodacapo.com/solutions/bsc-ss.asp, January 2004.

9 Brian Johnson and Ben Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In IEEE Visualization '91 Conference Proceedings, p. 284-291, San Diego, California, October 1991.

10 Baker, M.J., and Eick,S.G., Space-Filling Software Visualization. Journal of Visual Languages and Computing, 6: p. 119-133, 1995.

11 SeeSys: Space-Filling Software Visualization, URL : www.cs.umd.edu/class/spring2001/ cmsc838b/presentations/Jaime_Spacco/SeeSys.ppt, March 2005.

12 M. Kreuseler, H. Schuman, Information visualization using a new Focus+Context Technique in combination with dynamic clustering of information space, Proceedings Workshop on New Paradigms in Information Visualization and Manipulation, pp 1-5, 1999.

13 A. Maniatis, P. Vassiliadis, S. Skiadopoulos, Y. Vassiliou. Advanced Visualization for OLAP. In Proc. ACM 6th International Workshop on Data Warehousing and OLAP (DOLAP 2003), New Orleans, Louisiana USA, November 7, 2003.

14 Peter Pirollo, Ramana Rao: Table Lens as a Tool for Making Sense of Data. Proceedings of the AVI ’96 Workshop, Gubbio, Italy, June 1996.

15 Wietek, F.: Modelling Multidimensional Data in a Dataflow-Based Visual Data Analysis Environment. Proceedings 11th Conference on Advanced Information Systems Engineering (CAiSE*99), Springer, 1999.

16 I. Herman, G. Melancon, and M. Marshall. Graph visualization and navigation in information visualisation: a survey. IEEE Transactions on Visualization and Computer Graphics, 6(1): p. 24-43, 2000.

17 Ahlberg, C. and Schneiderman, B., Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays, Proc. ACM SIGCHI, p. 313-317, 1994.

18 Eick, S., Data Visualization Sliders, Proc. User Interf. Softw. Techn., p. 119-120, 1994.

19 Shneiderman, B., Dynamic Queries for Visual Information Seeking, IEEE Softw., 11, p. 70-77, 1994.

20 Ahlberg, C. and Wistrand, E., IVEE: An Information Visualization and Exploration Environment, Proc. IEEE Info. Vis., p. 66-73, 1995.

21 Information Visualization and Exploration Environment (IVEE) Development AB, URL : http://www.ivee.com/, December 2004.

22 Williamson, C. and Shneiderman, B., The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System, Proc. ACM SIGIR, p. 339-346, 1992.

23 Baldonado, M., Woodruff, A., Kuchinsky, A., “Guidelines for Using Multiple Views in Information Visualization”, Proc. ACM Advanced Visual Interfaces 2000, 2000.

24 Miller, G.A., The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review, 63, p. 81-97, 1956.

25 D. F. Swayne, D. Cook, A. Buja . XGobi: Interactive Dynamic Data Visualization in the X Window System, Journal of Computational and Graphical Statistics 7 (1), 1998.

26 Ahlberg, C., Wistrand, E., “IVEE: An Information Visualization and Exploration Environment”, Proc. IEEE Information Visualization 1995, p. 66-73, 1995.

27 Keim D. A., Lee J. P., Thuraisinghaman B., Wittenbrink C.: Database Issues for Data Visualization: Supporting Interactive Database Exploration , Proc. Workshop on Database Issues for Data Visualization, Atlanta, GA, 1995, Springer Lecture Notes, 1996.

28 North, C., Shneiderman, B., “Snap-Together Visualization: A User Interface for Coordinating Visualizations via Relational Schemata”, Proc. ACM Advanced Visual Interfaces 2000, p. 128-135, 2000.

29 The Common Gateway Interface, URL : http://hoohoo.ncsa.uiuc.edu/cgi/overview.html, December 2004.

30 JavaScript, How did we get here? URL : http://www.oreillynet.com/pub/a/javascript/2001/04/06/js_history.html, December 2004.

31 Browser stats, URL : http://www.thecounter.com/stats/2002/April/browser.php, December 2004.

32 A. Goel, "Visualization in Problem Solving Environments", Digital Library and Archives, ETD etd-061899-113642, Virginia Tech, VA, June 1999.

33 A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. In Proceedings of IEEE Visualization '90, p. 360-375, Los Alamitos, CA, October 1990. IEEE Computer Society Press, 1990.

34 Applets: Still Essential to Java, URL : http://www.javaworld.com/javaworld/jw-12-2000/jw-1201-soapbox.html, December 2004.

35 Glossary of Terms, URL : http://ai.stanford.edu/~ronnyk/glossary.html, March 2005.

36 Quinlan, J.R. C4.5: Programs for machine learning. California: Morgan Kaufmann 1992.

37 Decision Trees Connectionist and Statistical Language Processing, URL : http://homepages.inf.ed.ac.uk/keller/teaching/connectionism/lecture9_4up.pdf, March 2005.

38 http://grb.mnsu.edu/grbts/doc/manual/J48_Decision_Trees.html, March 2005.

39 R. Kimball. The Data Warehouse Toolkit. John Wiley, 1996.

40 What is OLAP? URL : http://www.olapreport.com/fasmi.htm, November 2004.

41 Centers for Disease Control and Prevention, URL : www.cdc.gov, December 2004.

42 WEKA 3: Data Mining Software in JAVA, URL : http://www.cs.waikato.ac.nz/ml/weka, December 2004.

43 Treemap, URL : http://www.cs.umd.edu/hcil/treemap, December 2004.

44 Microsoft SQL Server Analysis Services, URL : http://www.microsoft.com/sql/evaluation/bi/bianalysis.asp, December 2004.

45 Card, S., Mackinlay, J., Shneiderman, B., Readings in Information Visualization: Using Vision to Think, Morgan Kaufmann, 1999.

46 Data Mining, URL : http://www.siggraph.org/education/materials/HyperVis/applicat/data_mining/data_mining.html, October 2004.

47 R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, p. 1137-1143. San Mateo, CA: Morgan Kaufmann, 1995.

48 R. E. Schapire, “A brief introduction to boosting,” in Proc. 16th Int. Joint Conf. Artificial Intell., 1999.

49 SVGUI Project, URL : http://svgui.sourceforge.net/, November 2004.

50 Jakob Nielsen, Usability Engineering, Boston: AP Professional, c1993.

51 How to Conduct a Heuristic Evaluation, URL : http://www.useit.com/papers/heuristic/heuristic_evaluation.html, March 2005.

Appendices

A.1 User Task List

Task 1:
Which states have the highest occurrences of the symptom URTICARIA and during which years are the occurrences the highest?
Try bar chart first.
Try US chart.
Try treemap.
Try using highlight feature.
How easy it was to determine the relative number of occurrences of the symptoms from the states with the highest values to the other states using the bar chart, US chart, and treemap?

Task 2:
Find 3 states with the largest number of occurrences of symptoms for all criteria.

Using only 1 chart, try to find out which vaccines had the highest number of reports of symptoms for each of the three states.
Try it again using 2 charts.

What are the similarities and differences in the number of occurrences (look at the profile for occurrences) in the states with the highest occurrences of symptoms?

Task 3:
Find out which vaccine has the highest number of reports for arthritis.

For this subset, which agegroup has the highest number of reports?

How does this compare with the total number of patients who have taken any of the vaccines and have shown arthritis?

How does it compare with the total number of patients who have taken LYME vaccine but have shown any of the symptoms, not just arthritis?

Task 4:
Look at the different onset profiles of the symptoms. What does the profile for onset look like for all symptoms? Which vaccines have different profiles than the others?

A.2 Questionnaire

What features did you like about the system?

What features did you not like about the system?

What improvements do you think could be made?

Was it hard to find the information? Why?

Which tasks were easier and which ones where harder and why?

1 – Strongly disagree
5 – Strongly agree

I found the treemap to be useful:
I found the bar chart to be useful:
I found the US Map to be useful:
I found the highlight feature to be useful:
I found the navigation to be easy: