Data Mining with MineSet: What Worked, What Did Not, and What Might

Ron Kohavi
MineSet Engineering Manager
Silicon Graphics, Inc. M/S 8U-876
2011 N. Shoreline Blvd
Mountain View, CA 94043-1389
ronnyk@engr.sgi.com

August 1, 1998

1 Introduction

At Silicon Graphics, Inc., we have developed a data mining and visualization product called MineSet™(Silicon Graphics 1998, Brunk, Kelly & Kohavi 1997). MineSet first released in early 1996 mostly as a visualization product and then became a full data mining and visualization product late that year with several data mining algorithms based on $\mathcal{MLC}$++ (Kohavi, Sommerfield & Dougherty 1997).

The engineering effort in product development is estimated at over 55 person years, with the engineering team consisting of 17 people right now. During our development we interacted with dozens of customers, taught data mining courses, and used our product on internal databases in our company.

In this paper we detail some things that worked well, some things that did not work as well as we hoped, and some thoughts about the future.

2 What Worked Well

Below we detail that things that we felt worked well for MineSet.

System Architecture MineSet was designed as a client-server architecture as shown in Figure 1. The visualizations are shown on the client while the CPU-heavy analytics and I/O-heavy data transformations are done on the server. The database can reside on the server or in a third tier.

The architectural design worked well because it scales to large datasets. On small datasets, both client and server modules run on the same machine. Unlike products that demo well on 10,000 records but do not scale to large datasets, a client-server architecture provides a growth path from a demo or pilot to more serious knowledge discovery scenarios. While large servers today have tens of gigabytes of memory and hundreds or thousands of disks, client machines (PCs and small workstations) lag by two orders of magnitude. Mining large databases can only be effectively done on a server machine. It also helps to have the database local to the server to avoid long communication delays.

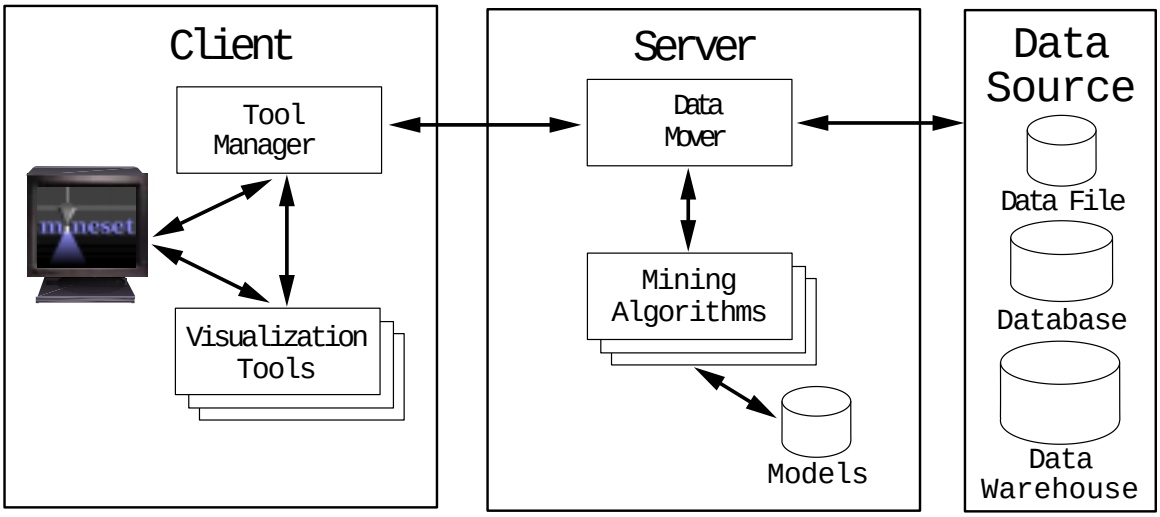

Figure 1: MineSet Architecture

Two Development Libraries MineSet was built on top of two libraries: a visualization library that is based on Inventor/OpenGL and $\mathcal{MLC}$++ for the analytical algorithms. Unlike products that provide isolated tools that were developed independently, MineSet feels like a consistent integrated product where behaviors are similar across the tools, the code reliability can be kept high, and maintenance costs low. The $\mathcal{MLC}$++ library consists of only 102,000 lines of code and 40,000 lines of testers to ensure correctness.

Direct Visualization tools MineSet provides users with direct visualization tools, such as scatterplots, map visualizers, hierarchy visualizers, and statistics visualizers. Figure 2 shows refinancing costs for counties in the US. A picture is worth 1,000 words or, in this case, over 3,000 counties. The human perception system can identify anomalies and patterns much faster in a representative landscape than in a spreadsheet. The visual tools also provide animation capabilities to see trends over time (or other variables).

Comprehensible Models and search abilities MineSet provides analytical algorithms that build models we have found ways to show visually. Users can interpret the models and interact with them using what-if scenarios. In many cases insight is derived from looking at the model and seeing an interesting pattern or rule. Interesting insight may be derived from a model even if it is inaccurate when used in predictions. In some cases the visualization helps users see their data problems or incorrect assumptions they made.

When large datasets are used, models tend to be large. When a 50,000-node decision tree is built, no one will look at every node. MineSet therefore provides search and filtering abilities on the models. For example, users can search for nodes that have at least some percentage of a given class and at least a minimum level of support in the training set.

MineSet also uses well-known and understood algorithms and not hyped proprietary algorithms that no one but the company's founder can understand.

Drill-down/drill-through Several MineSet tools provide drill-down abilities. Figure 3 shows a drill-down to two specific states. MineSet allows drilling down to records that make up a visual object, i.e., by pointing to a state in a map or a node in a tree, the original records can be shown. Alternatively, that subset of the population can be sent to another tool.

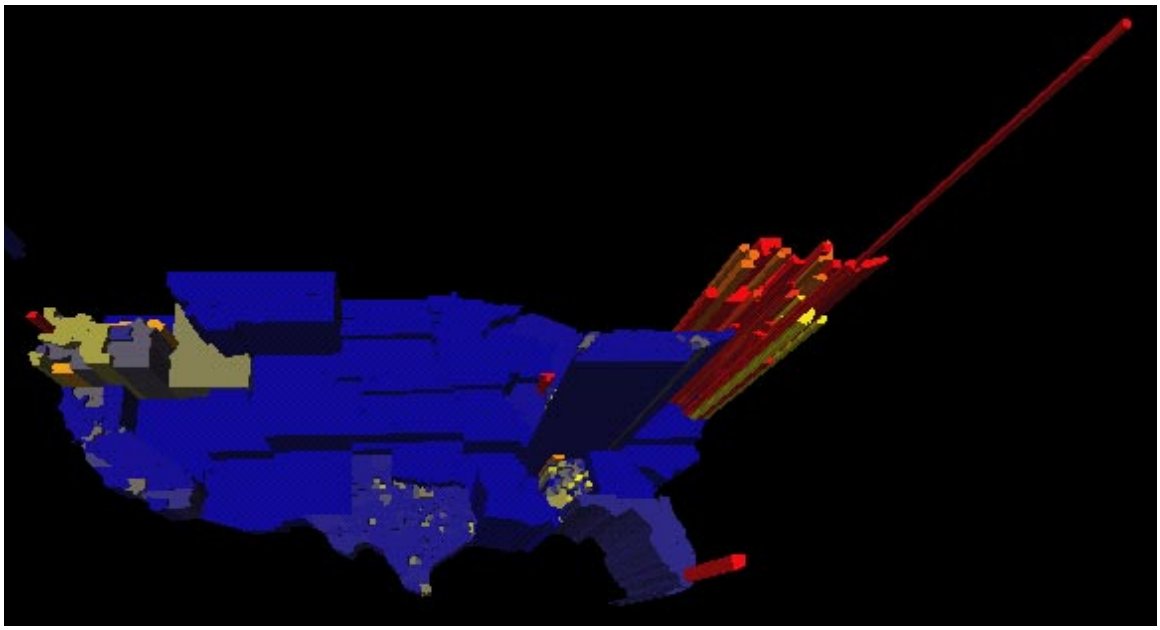

Figure 2: Refinancing costs (mapped to height) for every county based on FIPS codes. Deviations from each state's average are colored from blue (zero deviation) to yellow (0.005) to red (0.01).

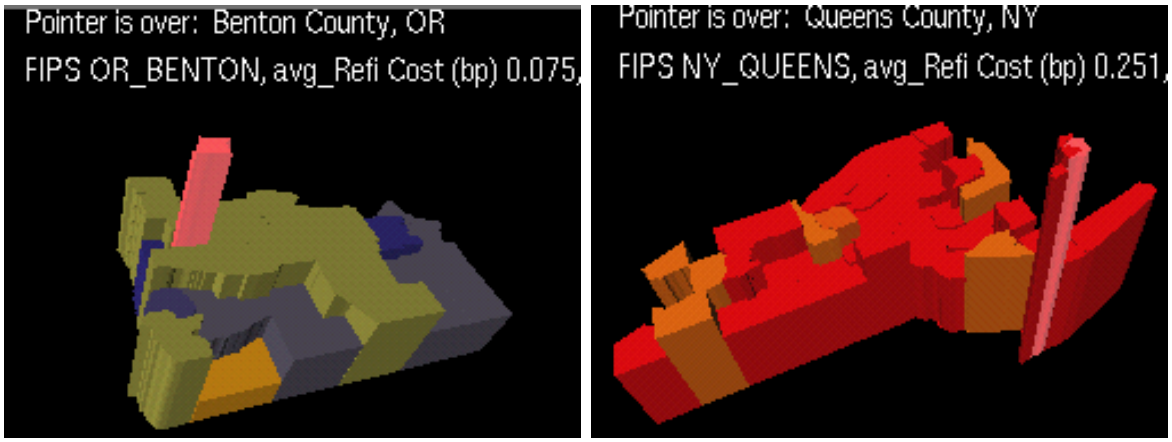

Figure 3: Refinancing costs for Oregon (left) and NY (right). Differences between neighboring counties are on the order of hundreds of dollars: enough to deter some people from refinancing.

Defaults everywhere MineSet provides defaults for most options. Users can get started by choosing an algorithm and hitting the "Go!" button. As users gain experience, they may set defaults differently, but the learning curve is less steep when reasonable defaults are automatically chosen.

3 What Did Not Work or is Missing

Below we detail that things that we felt did not work as well as we hoped or that are missing. Many of the missing features require significant research.

Data Transformations This is the hardest part of the KDD process: getting the data into a minable form. MineSet supports basic transformations such as adding columns, removing columns, sampling, and even transposes. It is common to see users doing dozens of transformations inside MineSet. However, in many projects, especially in early stages, there still needs to be a Perl script writer or an SQL writer. This area is not well understood and commercial Query Builders do not support needed operations.

KDD project management MineSet provides session management and allows saving and restoring sessions. However, there is no system for keeping track of the overall KDD process (Brachman & Anand 1996). Users end up with dozens of models and session files.

Meta Knowledge MineSet has no meta-knowledge database. In many cases processes could be simplified by knowing basic properties of data, goals, and tasks.

Database loads MineSet has one of the best integrations with commercial database systems with native interfaces to Oracle, Sybase, and Informix. Sadly, database systems today are optimized for OLTP (On-Line Transaction Processing) and are not geared for reading a whole database quickly. Moreover, we are able to store files in a fast binary format that is faster to load (e.g., strings are already pre-hashed). For large files, reading time may be significant and our binary files are faster by a factor of three. Note that the integration with a database is provided, but users typically dump the database to a flat file and continue from there.

APIs MineSet provides the ability to launch operations in batch mode and through configuration files, but does not supply a full API (Application Programming Interfaces). This has limited the ability of system integrators to tailor MineSet to their needs.

Deployment There are two types of deployments: the model and the knowledge. The model is relatively easy to deploy in MineSet. Sharing knowledge is harder. How does one share discoveries with others in the organization? Does everyone in the organization need to install MineSet? We wrote some prototype visualizations in Java but there were too slow. We have some prototypes in VRML but they are not general and also slow. VRML is not the answer for all visualizations (Akeley 1998). This remains an open problem.

SGI MineSet runs only on Silicon Graphics hardware. This was a restriction because it limited our market (but increased hardware sales). We are now porting MineSet to other platforms, starting with Windows.

Time Series All the analytical tools in MineSet are based on the standard assumption that records are independently and identically distributed (i.i.d.). Many datasets have a time attribute that is special. Can we utilize this attribute more effectively?

Attribute types: Text, pictures MineSet recognizes the basic types: strings, integers, floats, bins, dates. Long strings are better handled with text mining algorithms. Can we mine text?

4 Some Thoughts

Below we describe some issues that are open and highly debatable.

Scaling out of Core Most data mining algorithms (including those in MineSet) are implemented to run in memory and degrade ungracefully if the data and data structures do not fit in memory. (A notable exception is SPRINT (Shafer, Agrawal & Mehta 1996).) With the growth of operating systems and apps to 64-bit, we see little need for scaling out of core (but we do see the need to generate 64-bit versions of existing algorithms). Memory prices are dropping and with them systems are being built with large memories. Large commercial systems are now seen with dozens of gigabytes of memory.

Time-wise, the difference between running in core and out of core is about two orders of magnitude and getting worse (algorithms need to do multiple scans but disk read times are not improving as fast as memory sizes). Thus, if mining a few gigabytes in core takes a few hours today, mining that same size out of core may take five days. We believe users will not tolerate this.

Sampling The law of large numbers works. Sampling is a fine solution for initial data exploration. Significant patterns of interest will show up in reasonably sized samples. Only in rare cases will users wish to see patterns with miniscule support. In those cases it is more likely that users will start the mining on a small filtered sample (which may be the result of a previous drill-down operation).

No Business Case for Software Only DM Companies There is no clear business case for a software-only data mining company *today*. This will change once the market grows and the question is how many of the 75 data mining companies that exist today will exist in a few years.

With all the hype surrounding data mining, it is hard to make the business case for a profitable company that sells a horizontal data mining software tool such as MineSet today. The Meta Group estimated the size of the data mining market by year 2000 will be $8.4 billion (Meta Group 1997) with 8% of the market share related to macromining tools such as MineSet ($655 million). The estimates seem to be very optimistic now that two years have passed. Recent estimates for the data mining software market in 1997 are around $50 million.

MineSet exists at Silicon Graphics because it leverages a lot of hardware. Other companies are doing system integration or several types of professional services/consulting.

Anytime algorithms Sometimes a model is completely wrong and users can quickly tell that. For example, a very common case is to have a perfect or near perfect predictor for the label in a classification problem. When you look for customers that churned, the root of the decision tree may be customer number (if it is zero, they churned). When we tried to classify large versus small sales at Silicon Graphics, we found that the root of the decision tree was amount of tax paid. Significant time was spent waiting for the complete tree to be built, only to realize that we had a near perfect predictor that we needed to remove. Anytime algorithms (Boddy & Dean 1989) show *some* results quickly and then improve. The idea is an extension of the "effort knob" mentioned in Thearling (1998).

Process Wizards As data mining will be available to more business users, we need to provide more help on the processes and tasks. This can be achieved by developing wizards, templates, or building custom applications where the data mining component is hidden.

Interfaces to the rest of the process Mining is only one part of the KDD process. The data mining tools need to provide tighter integration with data cleansing tools, reporting tools (e.g., OLAP tools), and post processing (e.g., campaign management). In some cases, the data mining could serve to help these other parts. Specifically, data cleansing tools could benefit from tigher integration with data mining tools.

5 Conclusion

We have discussed what works well in MineSet, what did not work as well, and some thoughts.

Howard Frank from DARPA said that predicting the future is easy; Getting it right is the hard part. We predict that in the next couple of years, many data mining companies will go bankrupt or change their business model to be more solution oriented rather than focusing on horizontal technology. Recent examples of this trend include DataMind and HyperParallel. We believe that horizontal products like MineSet can only exist in large companies where the product leverages other parts of the business (consulting, hardware, database sales).

References

Akeley, K. (1998), Riding the wave.
`http://www.sgi.com/developers/marketing/forums/akeley.html`.

Boddy, M. & Dean, T. (1989), Solving time-dependent planning problems, *in* N. S. Sridharan, ed., 'Proceedings of the Eleventh International Joint Conference on Artificial Intelligence', Vol. 2, Morgan Kaufmann Publishers, Inc., pp. 979–984.

Brachman, R. J. & Anand, T. (1996), The process of knowledge discovery in databases, *in* 'Advances in Knowledge Discovery and Data Mining', AAAI Press and the MIT Press, chapter 2, pp. 37–57.

Brunk, C., Kelly, J. & Kohavi, R. (1997), MineSet: an integrated system for data mining, *in* D. Heckerman, H. Mannila, D. Pregibon & R. Uthurusamy, eds, 'Proceedings of the third international conference on Knowledge Discovery and Data Mining', AAAI Press, pp. 135–138.
`http://www.sgi.com/Products/software/MineSet`.

Kohavi, R., Sommerfield, D. & Dougherty, J. (1997), 'Data mining using $\mathcal{MLC}$++: A machine learning library in C++', *International Journal on Artificial Intelligence Tools* **6**(4), 537–566.
`http://www.sgi.com/Technology/mlc`.

Meta Group (1997), Data mining market trends: A multiclient study.

Shafer, J., Agrawal, R. & Mehta, M. (1996), Sprint: a scalable prallel classifier for data mining, *in* 'Proceedings of the 22nd International Conference on Very Large Databases (VLDB)'.

Silicon Graphics (1998), *MineSet User's Guide*, Silicon Graphics, Inc. http://mineset.sgi.com.

Thearling, K. (1998), Some thoughts on the current state of data mining software applications. DS*, Jan 13.