Data Mining and Tree-based Optimization*

H. Bodek and R. L. Grossman†
Magnify, Inc.

H. V. Poor
Princeton University

March, 1996

This is a draft of the paper Robert L. Grossman, Haim Bodek, David Northcutt, and H. Vince Poor, Data Mining and Tree-based Optimization, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, E. Simoudis, J. Han and U. Fayyad, editors, AAAI Press, Menlo Park, California, 1996, pp 323-326.

Abstract

Consider a large collection of objects, each of which has a large number of attributes of several different sorts. We assume that there are data attributes representing data, attributes which are to be statistically estimated from these, and attributes which can be controlled or set. A motivating example is to assign a credit score to a credit card prospect indicating the likelihood that the prospect will make credit card payments and then to set a credit limit for each prospect in such a way as to maximize the over-all expected revenue from the entire collection of prospects. In the terminology above, the credit score is called a statistical attribute and the credit limit a control attribute. The methodology we describe in the paper uses data mining to provide more accurate estimates of the statistical attributes and to provide more optimal settings of the control attributes. We briefly describe how to parallelize these computations. We also briefly comment on some of data management issues which arise for these types of problems in practice. We propose using object

*For additional information, please contact Robert L. Grossman, Magnify, Inc., 815 Garfield, Oak Park, IL 60304, 708 383 7002, 708 383 7084 fax, rlg@magnify.com. This work was supported in part by the Massive Digital Data Systems (MDDS) Program, which is supported by the Community Management Staff in the Department of Defense.

†Robert Grossman is also Director of the Laboratory for Advanced Computing at the University of Illinois at Chicago and a faculty member in the Department of Mathematics, Statistics, and Computer Science.

warehouses to provide low overhead, high performance access to large collections of objects as an underlying foundation for our data mining algorithms.

Keywords: tree-based optimization, parallel classification and regression trees, stochastic optimization, data warehouses

1 Introduction

In this paper, we consider a class of data mining problems that are broadly related to optimization. The goal is to maximize an objective function defined on a large collection of objects by setting certain control attributes of each object. In the problems of interest there is additional structure present: the control attributes are dependent upon other attributes, which are statistical in nature. The problem is difficult due to the large amount of data, the large number of attributes, and the complexity of the data. We use data mining techniques to estimate which attributes influence the control attributes and to estimate these (statistical) attributes from the underlying data attributes of the objects.

A motivating example is to assign a credit score to a credit card prospect indicating the likelihood that the prospect will make credit card payments and then to set a credit limit for each prospect in such a way as to maximize the over-all expected revenue from the entire collection of prospects. Additional details and examples are given in Section 3.

To summarize: we are given a large collection of objects, each of which has a large number of attributes of several sorts. Certain *data* attributes of the objects are given. From these we compute *summary* attributes and estimate *statistical* attributes. The goal is to set the *control* attributes to optimize a given objective function. We use tree-based techniques from data mining to estimate the statistical attributes and to determine which statistical attributes are important for setting the control attributes.

Our over all objective is to derive data mining algorithms that are scalable and data-driven. By scalable we mean algorithms that scale as the number of objects and the number of attributes increases. We are also interested in algorithms which scale as the numerical complexity and data selectivity of a data mining query varies. By data-driven we mean algorithms that do not require sampling, but rather examine all of the data.

To obtain scalable, data driven algorithms, we introduce and exploit three main ideas.

> *Tree-based Optimization.* As the amount of data grows, the number of patterns combinatorially explodes. By focusing on patterns related to the objective function associated with our optimization, we reduce the number of patterns we must examine. We use tree-based techniques as the basis for our optimization algorithms. In particular, one can view the algorithm we use at each leaf of the op-

timization tree as exploiting a different model for a particular subset of the data: we can think of this as tree-based "micro-modeling".

Parallel Trees. We exploit techniques from parallel and high performance computing to reduce the running time of our algorithms. In particular, we partition the data, separately compute the statistical attributes using tree-based techniques for each partition, and then combine the resulting information. We also compute trees in parallel by passing attribute information at each level of the tree between nodes in a parallel computing environment.

Object Warehouses. Data driven data mining requires that all of the data be examined. In order to manage very large data sets, we use specialized data management techniques which are optimized to provide low overhead, high performance access to complex data, in contrast to traditional transaction-oriented data management systems. More specifically, our data mining algorithms view the data as a distributed collection of objects, which we manage using an object warehouse specially designed for data mining queries.

Section 2 describes some background ideas and recent related work. In Section 3, we describe the data mining applications of concern to us in more detail. In Section 4, we describe the general problem we consider. Section 5 describes our tree-based optimization algorithm. Section 6 describes some data management issues. Section 7 describes our early experience. Section 8 concludes with a discussion and summary.

2 Background and Related Work

There has been increasing interest recently in developing scalable and parallel data mining algorithms. Agrawal et. al. [2] discuss scalable algorithms for discovering associations. Holsheimer et. al. [11] discuss parallel algorithms for a class of problems which require uncovering relatively few patterns which can cover a database. Fayyad et. al. [5] discuss extracting rules independently from partitioned data using decision trees and then aggregating the results.

Fundamentally, our approach to this class of data mining problems is an outgrowth of the idea of inductive learning [13] and classification and regression trees (CART) [3]. CART is a widely accepted tree-based methodology for performing classification and regression based on adaptation to a training set consisting of attributes vectors paired with correct classifications or function values. There are two basic and significant modifications of CART that are needed in order to apply this notion here. These are

- the development of *optimization* trees; and
- the parallelization of tree-based techniques, such as CART.

The need for an optimization paradigm within the framework of tree-based inference is a corollary to the general philosophy described in the preceding section. In particular, in a totally data-driven situation, the stochastic optimization of control attributes can proceed only from the synthesis of data-driven optimization procedures. Although tree-based inference is not an exclusive approach to this problem, it is certainly an optimum choice from two key viewpoints for the problems of interest here—namely, overall parsimony, which is absolutely necessary in the context of intensive data mining; and the ability to understand clearly the relative significance of the various attributes. Similarly, in order to use data mining techniques based on micromodeling (as opposed to more traditional, and statistically weaker, sampling techniques), the need for a parallelized CART structure is an obvious implication of the data-set sizes of interest here. The technique we propose is related to stochastic programming, such as described in [12].

General approaches to both of the fulfillment of these two requirements are discussed in a recent paper [8] in the context of credit scoring. In particular, optimization trees based on local gradient estimates provided by regression-tree variables are described, as is a distributed algorithm for coordinated growth of classification and regression trees on large partitioned data sets. Each of these ideas finds application in the problems of interest here, as will be evident from the sequel.

3 Some Examples

In this section, we describe three motivating problems involving tree-based optimization. These descriptions of the problems are from [9].

Problem A. Account solicitation and management. In this problem, we are a given a large collection of customers or potential customers, ranging from 10^5 to 10^8. Assume that each customer or prospect has several hundred to a thousand attributes describing them. To be specific, assume that there are one million credit card customers and the goal is to set their credit limit, interest rate and fee structure on the credit card twice a year in order to maximize the life time profit of the customer.

Notice first that there is a natural objective function defined by total life time profit of the collection. Also, notice that there are several types of attributes: 1) data attributes, such as the number, type and amount of purchases over some number of months; 2) summary attributes, which precompute such variables as the average amount of credit used per month, 3) statistical attributes, such as the expected life time revenue of the customer; and 4) control attributes, such as the credit limit, interest rate, and fee of the credit card. The goal is first to use the data attributes to estimate the predictive attributes, and second to use the predictive attributes to set the control attributes in order to optimize the objective function. The role of the precomputed summary attributes is to speed up the computation.

Data mining enters in the following two related manners. Different groups

of customers may require different subsets of data attributes in order to best estimate the predictive attributes. The problem is difficult because of the size of the data set and the number of attributes. With three hundred attributes and a million customers it makes sense to break up the collection of customers into sub-collections specified by attribute sequences with the property that predictive attributes of customers belonging to the same sub-collections are estimated from the same collection of attributes. In other words, *attribute discovery* is an important component in estimating predictive attributes.

In addition, different groups of customers may require different models to set the control attributes. A common approach is to work with a parameterized family of models. In other words, *parameter identification and model discovery* is also an important component in setting the control attributes.

Problem B. Anomaly detection. In this problem, we have two collections: 1) a real time stream of transactions producing a transaction collection and 2) a collection of customers. All transactions are associated with a customer, but a small percentage, say 1% are fraudulent in the sense they were generated by a third party different than the customer.

As in the first example, there is an objective function and several types of attributes. In this case, the objective function is usually taken to be a weighted average of the number of false positives and false negatives produced by the algorithm. Data attributes include the merchant, customer, and amount associated with the transaction. Summary attributes include the average dollar amount of a transaction during the past three months. Predictive attributes include the probability that a given customer will purchase a high ticket item from a certain merchant category during the next six months. Control attributes include the score assigned to the transaction indicating the probability of fraud or related actions, such as increased monitoring of the customer or merchant, denying the transaction, or calling the merchant.

Attribute discovery is also important in this problem. The merchant category may be the most important predictor for some customers; for others, the amount of the transaction; for still others, the frequency. Model discovery is also important, low usage and high usage customers may be best served by different models.

Essentially the same problem arises when using attribute-base information to look for anomalous or unusual documents within a large collection of documents.

Problem C. Target Assessment. This is a variant of Problem B. We assume that we have tens to hundreds of thousands of targets and a real time stream of information about the targets. The problem is difficult because the information stream is so large that the information must be classified automatically and the target list so large that only a few of the targets at any time may be examined by a human.

The data attributes include the information itself and metadata such as its source, when it was collected, etc. The information is automatically scored in order to attach it to the relevant target and to measure its significance. Both the relevance and significance score are examples of statistical attributes. Finally

control attributes include the action taken, which may range from throwing the information away, to flagging the information and raising the threat posed by the target.

4 Problem Description

This section gives a general description of the problem. For specific examples, see Section 3. We assume that at the logical level, that there is a large collection of objects and that each object has a variety of attributes. The attributes may be primitive data types (such as integers, floats, or character strings) object valued, or collection valued. As illustrated in the examples above, we assume that the objects have several types of attributes: data attributes, summary attributes, statistical attributes, and control attributes. The data attributes constitute the underlying data of the problem. Summary attributes are precomputed summaries of the data attributes. The statistical attributes are random variables that are estimated from the data attributes, while the control attributes are chosen to optimize the objective function.

In more detail, we are given a very large collection of objects $x_1, \ldots, x_N$, where each object x_j has data attributes $x_j[1], \ldots, x_j[n]$. From the data attributes of an object x_j, we precompute summary attribute(s) $\eta_j = e(x_j)$ and estimate the statistical attribute(s) $\zeta_j = f(x_j, \eta_j)$. Finally, we define control attributes $r_j = g(x_j, \eta_j, \zeta_j)$. Here η_j, ζ_j and r_j may all be vector valued. Our goal is to optimize the objective function

$$h = \sum_j \mathrm{Max}_{\mathrm{r_j}} h(x_j, \eta_j, \zeta_j, r_j),$$

by suitably choosing the control attributes r_j.

5 A Tree-based Optimization Algorithm

In this section, we briefly sketch an algorithm we have developed and implemented to solve problems like the ones described above. There is a precomputation, which in part involves training data. The precomputation consists of three steps:

1. First we clean the data and compute the summary attributes η_j. There are no theoretical issues here, but there may be important practical ones, especially for noisy data or incomplete data.

2. In the second step, we compute the statistical attributes ζ_j using regression trees [3] on training data $\mathcal{L}$. We write $\zeta_j = f(x_j, \eta_j)$ and let T_f denote the corresponding tree. Note that there will be several trees in case ζ_j is vector-valued. We describe different strategies for doing this below.

3. In the third step, we partition the data using a tree. There are several variants: for simplicity we describe only the simplest. We are given an

objective function $h(x_j, \eta_j, z_j, r_j)$ and partition the data using a regression tree approximation to h computed on the training data $\mathcal{L}$. Here z_j are the instances of the random variable ζ_j arising in the training data. Denote this tree by T_h. It is important to note that when computing the regression tree, *we do not partition using the control attributes.*

Processing a collection (or stream) of objects using the algorithm requires three steps:

1. In the first step, given an object x_j, we use the tree(s) T_f to compute the statistical attributes ζ_j.

2. In the second step, we apply the tree T_h to the object x_j so that x_j is associated with one of the leaves of T_h.

3. In the third step, we compute the control attributes r_j of x_j using the optimization algorithm appropriate for that leaf. The control attributes r_j are defined to optimize the objective function $h(x_j, \eta_j, \zeta_j, r_j)$. That is

$$h(x_j, \eta_j, \zeta_j, r_j) = \text{Max}_r h(x_j, \eta_j, \zeta_j, r).$$

   We are experimenting with several different optimization algorithms to set r_j, including the local gradient algorithms described in [8].

We conclude this section with some remarks.

Loosely speaking, we are using different models and control strategies for each leaf of the tree T_h. Think of this as tree-based micro-modeling. An important advantage is that working with many micro-models that are automatically computed results in a system that is more maintainable that one which uses one or a few models that are derived by a statistician. Traditional approaches typically use a relatively small number of models and thus incur large costs when these models are changed. In contrast, our approach automatically results in a relatively large number of different models, each applying to a relatively small number of cases. This potentially results in more accurate models and decreases the cost when a few models are changed or updated.

6 Data Management Issues

In this section we review some of the relevant data management issues for data-driven data mining. This section is a summary of the material in [9], which contains a fuller discussion of these issues.

Numeric and statistically intensive queries and other complex queries are extremely costly when run against data in databases, especially relational databases in which the data is spread over several tables. The high costs associated with these types of queries basically arises from the fact that traditional databases are optimized for relatively simple data, simple queries, and frequent updates. On the other hand, complex queries on complex data perform best using specialized data management systems that are 1) optimized for frequent reads,

occasional appends, and infrequent updates and 2) joins and other operations on the data are precomputed. These types of data management systems are sometimes called data warehouses.

We assume that the data warehouse views the data as collection of objects and provides low overhead, high performance access to these objects. Although using an object data model instead of a relational data modeling is not strictly necessarily, we have found it convenient for several reasons:

First, data mining often involves attributes that are collection-valued. Examples include a list of a customer's credit card transactions, a list of a customer's credit cards, a list of the cities that the customer has visited during the past eighteen months, etc. Second, data mining often involves working with heterogeneous data. Objects whose attributes are object-valued provides a convenient way to do this. Third, the intermediate data structures that arise when data mining are themselves usually quite complex and more conveniently expressed using an object data model rather than a relational data model, for example.

We also take the position that scalable data mining does not require an object-oriented database, but rather simply an object warehouse. High performance access to collections of objects is achieved by exploiting the access patterns of a typical decision support system: data is most often read, occasionally appended, and rarely updated. An object warehouse can be optimized for working with these types of access patterns. In contrast, a standard relational, object-relational, or object oriented database is optimized for working with short duration transactions.

Perhaps the crucial performance advantage obtained using object warehouses for data mining is the possibility of precomputing many of the intermediate and final data structures that arise in data mining. These include, for example, *summary* attributes which provide a statistical summary of the data attributes and summarize *past* behavior and *statistical* attributes, which as in the examples, above are random variables that can be used to make statistical predictions of *future* behavior. These attributes can be accessed by direct links to the underlying objects they reference or indirectly through specialized index structures.

7 Early Experience

We have developed an object warehouse specialized for data mining applications in parallel and cluster computing environments and used it to mine up to approximately twenty gigabytes of data [7]. We are currently testing the object warehouse on 100 gigabyte size data sets. The object warehouse was developed by Magnify, Inc. and is called PATTERN:Store. An earlier version of this system is described in [6].

Using PATTERN:Store, we have experimented with growing distributed trees in a coordinated fashion [8]. We are currently experimenting with growing distributed trees independently and averaging the results. Typical results are summarized in Figure 2 and Figure 3, which shows for a collection of 10,000

customers the decrease in cost and corresponding accuracy as the number of partitions varies between 1 and 10. Roughly speaking, the cost decreases as expected with the number of partitions, eventually leveling off, while the error increases roughly linearly, but remained quite small for the data we used (less than 1%). We are currently experimenting with mixing the strategies for parallel trees proposed in [8] and that proposed here. This work was undertaken by Magnify, Inc. and is implemented in a system called PATTERN:Predict.

In the full version of this paper, where space permits, we will report on the scalability of our algorithms and their implementations as well as results concerning setting the control attributes for the three problems described in Section 3.

8 Conclusion

The review paper [4] cites a number of current challenges in data mining, including three challenges which are of direct concern in this paper: working with larger data sets, working with high dimensional data, and working with complex data having complex relationships. These same themes were also highlighted in the workshop [10].

In this paper, we limit ourselves to a class of data mining problems with the structure summarized in Figure 1: We are given data attributes. We precompute summary attributes from these. In the examples of interest to us there are hundreds to a thousand data and summary attributes. From the data and summary attributes we use tree-based techniques to estimate statistical variables. Given an objective function, we also use tree-based techniques to discover patterns that are directly relevant to setting the control attributes to maximize a given objective function. This structure arises in a variety of problems ranging from computing scores for credit card prospects to assessing threats in data fusion problems.

By focusing on problems we this type of structure, we can narrow our search to a restricted class of patterns and in this fashion obtain algorithms which can scale to the terabyte scale.

We are interested in algorithms which are data driven in the sense that they examine all of the data. This requires specialized data management techniques. We have developed object warehouses specialized for data mining and used these to develop and test the tree-based optimization algorithms discussed here.

Although this work is preliminary, we feel we have made a contribution by focusing on an important class of problems, by presenting a new algorithm for these types of problems, and by giving preliminary evidence regarding the scalability of our approach.

References

[1] R. Agrawal, T. Imielinski, and A. Swami, Database Mining: A Performance Perspective, *IEEE Transactions on Knowledge and Data Engineering,* Vol-

ume 5, pp 914–925, 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, "Fast Discovery of Association Rules," in *Advances in Knowledge Discovery and Data Mining,* edited U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/MIT Press, pp. 307–328, 1996.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees,* Wadsworth, Belmont, California, 1984.

[4] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From Data Mining to Knowledge Discovery: An Overview," in *Advances in Knowledge Discovery and Data Mining,* edited U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/MIT Press, pp. 1–34, 1996.

[5] U. M. Fayyad, S. G. Djorgovksi and N. Weir, "Automating the Analysis and Cataloging of Sky Surveyes," in *Advances in Knowledge Discovery and Data Mining,* edited U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/MIT Press, pp. 471–493, 1996.

[6] R. L. Grossman, H. Hulen, X. Qin, T. Tyler, W. Xu, "An Architecture for a Scalable, High Performance Digital Library," *Proceedings of the 14th IEEE Computer Society Mass Storage Systems Symposium*, S. Coleman, editor, IEEE, Los Alamites, CA, 1995.

[7] R. L. Grossman, "The Terabyte Challenge: High Performance Data Mining and Data Management," High Performance Computing Challenge Demonstration at Supercomputing 95, San Diego, December 3–7, 1995.

[8] R. L. Grossman and H. V. Poor, Optimization Driven Data Mining and Credit Scoring, in *Proceedings of the IEEE/IAFE 1996 Conference on Computational Intelligence for Financial Engineering (CIFEr),* 1996.

[9] R. L. Grossman, "Early Experience with a System for Mining, Estimating, and Optimizing Large Collections of Objects Managed Using an Object Warehouse," *Proceedings of the Workshop on Research Issues on Data Mining and Knowledge Discovery,* Montreal, Canada, June 2, 1996, to appear.

[10] R. L. Grossman and R. Moore, "Report on the 1996 Workshop on Mining and Managing Massive Data (M3D)," in preparation.

[11] M. Holsheimer, M. L. Kersten, and A. P. J. M. Siebes, Data Surveyor: Searching the Nuggets in Parallel, in *Advances in Knowledge Discovery and Data Mining,* edited U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/MIT Press, pp. 447–467, 1996.

[12] P. Kall and S. W. Wallace, *Stochastic Programming,* John Wiley and Sons, Chichester, 1994.

[13] J. R. Quinlan, "The Induction of Decision Trees," *Machine Learning,* Volume 1, pp 81–106, 1986.