Data Mining and Tree-based Optimization

Robert Grossman*, Haim Bodek & Dave Northcutt **Vince Poor**

Magnify, Inc.
815 Garfield Street
Oak Park, IL 60304
{rlg, haim, dmn}@magnify.com

Dept. of Electrical Eng.
Princeton University
Princeton, NJ 08540
poor@princeton.edu

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

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 (predictive) 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 *predictive* 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 predictive 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 optimization 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 par-

*Robert Grossman is also a faculty member in the Department of Mathematics, Statistics, and Computer Science at the University of Illinois at Chicago.

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 predictive 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.

Problem Description

This section gives a general description of the problem. 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, predictive attributes, and control attributes. The data attributes constitute the underlying data of the problem. Summary attributes are precomputed summaries of the data attributes. The predictive 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 predictive 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

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

by suitably choosing the control attributes r_j.

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:

1. First, we compute the predictive attributes ζ_j using regression trees (Breiman 1984) 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.
2. Second, 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 predictive 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

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

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.

Implementation

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 have developed a data warehouse for large collections of objects specifically designed for parallel and distributed data mining. We are currently testing it 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 (Grossman et al. 1995).

Using PATTERN:Store, we have experimented with various strategies for growing trees in parallel. This work was undertaken by Magnify, Inc. and is implemented in a system called PATTERN:Predict.

Conclusion

In this paper, we have considered a class of problems with the following structure: 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 or predictive attributes. 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. 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 and by presenting a new algorithm to attack them.

Acknowledgments

This work was supported in part by the Massive Digital Data Systems (MDDS) Program, sponsored by the Advanced Research Development Committee of the Community Management Staff.

References

R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, 1996. 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, 307–328. Menlo Park, California: AAAI Press/MIT Press.

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

U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, 1996. 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, 1–34. Menlo Park, California: AAAI Press/MIT Press.

U. M. Fayyad, S. G. Djorgovksi and N. Weir, 1996. 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, 471–493. Menlo Park, California: AAAI Press/MIT Press.

R. L. Grossman, 1996. 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. Forthcoming.

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

R. L. Grossman and H. V. Poor, 1996. Optimization Driven Data Mining and Credit Scoring. In Proceedings of the IEEE/IAFE 1996 Conference on Computational Intelligence for Financial Engineering (CIFEr). Los Alamites, California: IEEE Press.

M. Holsheimer, M. L. Kersten, and A. P. J. M. Siebes, 1996. 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, 447–467. Menlo Park, California: AAAI Press/MIT Press.

J. R. Quinlan, 1986. The Induction of Decision Trees. *Machine Learning* 1: 81–106.