On the Use of Optimization for Data Mining: Theoretical Interactions and eCRM Opportunities

Balaji Padmanabhan • Alexander Tuzhilin
The Wharton School, University of Pennsylvania, Philadelphia, Pennsylvania 19104
Stern School of Business, New York University, New York, New York 10012
balaji@wharton.upenn.edu • atuzhili@stern.nyu.edu

Previous work on the solution to analytical electronic customer relationship management (eCRM) problems has used either data-mining (DM) or optimization methods, but has not combined the two approaches. By leveraging the strengths of both approaches, the eCRM problems of customer analysis, customer interactions, and the optimization of performance metrics (such as the lifetime value of a customer on the Web) can be better analyzed. In particular, many eCRM problems have been traditionally addressed using DM methods. There are opportunities for optimization to improve these methods, and this paper describes these opportunities. Further, an online appendix (mansci.pubs.informs.org/ecompanion.html) describes how DM methods can help optimization-based approaches. More generally, this paper argues that the reformulation of eCRM problems within this new framework of analysis can result in more powerful analytical approaches.
(*Data Mining; Optimization; eCRM Applications*)

1. Introduction

Prior research has used optimization methods for solving data-mining (DM) problems (Mangasarian et al. 1990, Vapnik 1995, Fu et al. 2003) and used DM methods for solving optimization problems (Brijs et al. 1999, Campbell et al. 2001). In this survey, we systematically explore how optimization and DM can help one another for certain customer relationship management (CRM) applications in e-commerce, termed *analytical eCRM* (Swift 2002). Analytical eCRM includes customer analysis, customer interactions, and the optimization of various performance metrics, such as customer lifetime value in Web-based e-commerce.

To illustrate how optimization and DM interact in eCRM settings, consider the following two important eCRM problems. The first deals with finding the optimal lifetime value (LTV) of a customer by determining proactive customer interaction strategies resulting in maximal lifetime profits from that customer. Because this LTV optimization problem also relies on the analysis of large volumes of customer data in the eCRM setting, it can be augmented with DM techniques to help improve the LTV model by better estimating its various parameters. The second eCRM problem, primarily in the DM domain, pertains to preprocessing click-stream data as the basis for building DM models, such as purchase prediction models. Zheng et al. (2003) show that inappropriate preprocessing of data can result in significantly worse DM models for critical eCRM problems. Given the nature of click-stream data and the fact that hundreds of derived variables can be created from this data for a user session, it is important to partition

0025-1909/03/4910/1327
1526-5501 electronic ISSN

the click-stream data into an optimal set of sessions and to select an optimal set of derived variables to produce the best DM model from this data. Both these examples show how DM and optimization can interact when solving various eCRM problems.

Understanding interactions between DM and optimization is particularly important in the eCRM context. First, many eCRM problems can be framed as optimization problems, and the domain is characterized by the availability of large volumes of data, often measured in terabytes. In this way, we can explore how patterns generated from DM can be combined with optimization approaches. Second, many eCRM problems deal with online customers interacting with an eCRM system and, therefore, require real-time solutions. For example, wine recommendations should be provided in real time for a customer purchasing wine at an online store. Optimization approaches can be computationally intensive and DM can help in such situations by identifying additional constraints to reduce the search space and, thus, produce faster results. For example, the customer preferences learned from using DM methods could be applied to limit the search space for best wines to white Chablis in a $20–$30 price range. As a result, we may be able to consider far fewer recommendation alternatives, which would ease the task of selecting the best one in real time. Third, many eCRM systems fail to build good relationships with customers, which often results in low satisfaction rates with such systems (Dver 2003). Often, just one bad, or even offensive, experience for a customer can break a relationship with a firm deploying a "bad" eCRM system. Hence, it is crucial to deploy superior performance methods in eCRM applications, and there is the potential to do so by integrating DM and optimization methods into high-performance solutions.

Using the broader context of eCRM applications as motivation, this paper surveys from a theoretical perspective how optimization and DM can be synergistically used. In particular, we present in §2, various eCRM applications and discuss opportunities for optimization. These eCRM applications include (1) maximization of customer LTV, (2) customer analysis, including preprocessing click-stream data and building profiles, and (3) customer interaction methods, including website design and personalization. Many of the eCRM applications discussed in §2 have been traditionally addressed using DM, and we note that there are significant opportunities for optimization to help the DM approaches. We present a theoretical discussion of how optimization can help in various DM problems in §3. The seven DM problems we discuss are: (1) feature selection, (2) active learning, (3) DM model optimization, (4) selection of the best DM model or pattern, (5) classification using mathematical programming, (6) clustering, and (7) rule and constraint discovery. While the interactions between DM and optimization can be bidirectional, a detailed treatment of how DM can help optimization is beyond the scope of this paper. However, an online appendix (at mansci.pubs.informs.org/ecompanion.html) of this paper presents various examples of how DM can help optimization and discusses additional research opportunities. We conclude in §4 by listing several key research opportunities that emerge from this paper. At a general level, in eCRM, these opportunities lie in (1) using DM to help exploit the data better for eCRM problems that are naturally formulated as optimization problems (e.g., selecting the 10 *best* items to recommend), and (2) using optimization for eCRM problems that are more naturally formulated or currently considered as DM problems (e.g., building good predictive models or learning customer patterns from data to build profiles).

2. Optimization Opportunities in eCRM

The purpose of CRM is to identify, acquire, serve, and retain profitable customers by interacting with them in an integrated way across a range of communication channels. Swift (2002) describes analytical eCRM as a four-step iterative process consisting of (1) collecting and integrating online customer data, (2) analyzing this data, (3) building interactions with customers based on this analysis such that certain performance metrics such as LTV are optimized, and (4) measuring the effectiveness of these interactions in terms of these performance metrics. In this section, we will examine how various optimization and DM methods help in

providing better eCRM solutions for these steps, with a focus on steps (2)–(4). We start in §2.1 by studying one of the most important performance metrics—the LTV of a customer. In §2.2, we discuss customer analysis problems and, in §2.3, we discuss customer interaction problems in eCRM.

2.1. Maximizing Lifetime Value

A typical performance metric used in many CRM applications is the LTV of a customer. One of the key questions in CRM is how to develop proactive customer interaction strategies that maximize LTV. This key problem is viewed by some as the "holy grail" of CRM. Substantial work has been done on modeling LTV, mainly in the marketing literature (Schmittlein et al. 1987, Dwyer 1989, Blattberg and Deighton 1991, Gupta et al. 2001, Dreze and Bonfrer 2002).

Traditionally, the problem of estimating LTV is divided into two components: (1) estimating how long a customer will stay, and (2) estimating the flow of revenue from the customer during this period. Estimating the flow of revenue during a customer's lifetime was done with parametric models in the marketing literature. With respect to estimating customer tenure, Schmittlein et al. (1987) provide analytical models that determined whether a customer at any point in time is "active," by identifying a set of qualitative criteria that capture when a customer is more likely to be active. The concept of customer retention is a natural extension of this approach, because it deals with trying to prevent a customer from becoming inactive and, hence, is a proactive method of increasing LTV. Blattberg and Deighton (1991) present a more general framework for interactive marketing for LTV, suggesting the key notion that customers are "addressable" and can be engaged in interaction.

This notion of customer interaction is further explored in Dreze and Bonfrer (2002) in which the problem of optimal communication to maximize LTV is addressed. In their approach, they assume that communications (e.g., mailings) are sent at a fixed time interval θ and the problem is determining the optimal θ. Based on simplifying assumptions regarding the probability of a customer being active at time i to be p^i and on equal expected profit, A, from each communication, they derive value of a customer, V, as a function of θ as

$$V(\theta) = \sum A(\theta) \cdot \left(p(\theta)^i/(1+r)^{\theta i}\right).$$

Dreze and Bonfrer (2002) show that both too little and too much communication can result in a firm's failure to capture adequate value from its customers. This analysis provides useful insights into the LTV optimization problem. However, this work considers only optimal communication frequency with customers and the model does not take into account the existence of data on customer behavior.

In contrast, the DM community studied the LTV problem in the presence of large volumes of customer data (Mani et al. 1999, Rosset et al. 2002). In particular, Mani et al. (1999) predict customer tenure using classical survival analysis methods by building a neural network and training it on past customer data. However, Mani et al. (1999) do not address the problem of computing optimal parameters for LTV models. Similarly, Rosset et al. (2002) compute LTV based on large volumes of customer data by focusing on using DM to estimate customer churn and future revenues. They use statistical and DM methods to estimate future revenues $v(t)$ from the customer and probabilities $S(t)$ that the customer will still be active at various times t in the future. To compute more realistic estimates of $S(t)$, Rosset et al. (2002) group customers into segments and make a simplifying segment homogeneity assumption. This assumption makes it possible to use simple nonparametric estimations of probabilities $S(t)$ for the segments. Once the values $v(t)$ and $S(t)$ are estimated and the LTV values are computed, the next step is to design incentives for the customers that maximize their LTVs. As an example, Rosset et al. (2002) discuss a case in the wireless industry where service providers need to select the best incentive (such as reduced prices, handset upgrade, a free battery) to offer to each customer. In general, this is an optimization problem, although the Rosset et al. (2002) paper determines the best incentive by exhaustive search across a relatively small number of incentives. Because DM methods are used for computing LTV values, and optimization methods determine incentives resulting in highest LTV values, such applications provide examples of how DM methods can help solve optimization problems.

We would like to point out that it is difficult to solve an LTV optimization problem in the presence of large volumes of data, and Rosset et al. (2002) stopped short of doing this. Similarly, Dreze and Bonfrer (2002) studied only an optimization problem and did not deal with large volumes of data. Therefore, developing new algorithms computing optimal LTVs and utilizing optimization and DM methods in the presence of large volumes of data constitutes an important and challenging problem for operations research/management science (OR/MS) researchers.

One way to deal with the complexity of the general LTV optimization problem is to reduce it to simpler types of problems. One such reduction may consider various heuristics producing higher LTV values, rather than attempting to find an optimal solution. For example, we can study customer attrition problems and determine policies that will result in lower attrition rates and, therefore, higher customer LTVs. Another way to deal with this complexity is to seek to optimize simpler performance measures that can serve as proxies for LTV. For example, one can use customer satisfaction rates with various offerings as one such measure. Much of the eCRM literature on customer analysis and interactions discussed below illustrate these two ways of dealing with the complexity of the general LTV optimization problem.

2.2. Customer Analysis in eCRM

Customer analysis includes two main steps in the eCRM context: (1) preprocessing data that tracks various online activities of the customers—this involves starting with individual user clicks on a site and constructing logical user "sessions" and summary variables; and (2) building customer profiles from this and other data. At a general level, a profile is a set of patterns that describe a user. DM is used to learn these patterns from data. Customer profiles are then built from these results.

2.2.1. Preprocessing Click-Stream Data. As argued by Zheng et al. (2003), data preprocessing is a critical step of the knowledge discovery process in eCRM, and the success of most DM methods to a large extent depends on this step. Therefore, any method that directly improves data preprocessing affects DM results. In this section, we examine how optimization can facilitate better data preprocessing of click-stream data.

Most current literature considers heuristic methods for analyzing click-stream data gathered by websites. One of the most important problems is the session identification problem, which determines how to group consecutive clicks into sessions. This is an important business problem because most of the user tracking systems developed by such companies as E.piphany and Blue Martini, provide only ad hoc solutions to the session identification problem.

Some popular session identification methods include session-level characterization (Srivastava et al. 2000, Theusinger and Huber 2000) that aggregates user clicks into sessions, a fixed-length sliding window method (Cooley and Mobasher 1999) that breaks a session into several sliding windows, and different types of clipping methods that break a session into windows of different sizes using various splitting methods (Brodley and Kohavi 2000, VanderMeer et al. 2000, Padmanabhan et al. 2001). Zheng et al. (2003) demonstrate that various session identification methods can produce radically different conclusions derived from the same data. It is therefore important to study optimization-based approaches to preprocessing click-stream data, and we discuss some of the opportunities of how optimization can help DM to do this.

The main reason for preprocessing click-stream data is to build a model, such as one that would predict the likelihood that a current user's session would result in a purchase. Accurate models are crucial for such problems, requiring optimal preprocessing of the click-stream data. To partition the click-stream data into sessions, one needs to specify optimization criteria. This can be done by first identifying "similar" groups of consecutive pages in the click-stream, which can then be partitioned into sessions to maximize intrasession similarities and intersession differences. One way to specify these similarities and differences is to identify the variance of some browsing measure, such as the time spent viewing a page. For any possible "session," let the average time spent per page be μ_i, where i refers to the session number, and let the standard deviation of this measure be σ_i.

An alternative measure is the variance of page content as defined by a set of keywords in each page, i.e., based on how much the keywords in one page differ from those on another page. The problem can then be defined as follows. Given a number of desired sessions n, a minimum and maximum length for each session, and the requirement that only consecutive pages form a session, find an optimal partitioning of the click-stream data into n sessions. Optimality can be defined in many ways, one of which is to achieve dual objectives (maximizing intrasession similarities and intersession differences):

$$\text{minimize} \quad \sum_{i=1}^{n} \sigma_i \Big/ n \quad \text{and}$$

$$\text{maximize} \quad \sum_{i=1}^{n} (\mu_i - \text{avg}(\mu_i))^2 \Big/ n.$$

An opportunity for the optimization community is to determine how to formalize the session optimization approach outlined above, determine good similarity measures, and find optimal solutions for these measures. In this mode of interaction, optimization helps DM by preprocessing data to build better DM solutions. Note that a related use of optimization during preprocessing is the use of optimization models to select the set of variables used to build a DM model. Because this is not specific to eCRM, we discuss this further in §3.1.

2.2.2. Building Customer Profiles. The profiling problem has been studied even before eCRM applications became popular. In particular, Fawcett and Provost (1996) studied this problem within the fraud detection context and used DM to generate rule-based customer profiles to detect fraudulent cellular phone usage behavior. Building rich and accurate profiles of customers based on their transactional histories is also crucial in many CRM applications, including recommendation applications, one-to-one marketing, and personalized Web content delivery. These user profiles can contain (1) factual information about the user, such as demographic and psychographic data, (2) a set of rules capturing behavior of the user, (3) management science models parameterized to particular individuals, and (4) important website visitation patterns, and so on. For example, consider the rule, "if John travels to Los Angeles on business, he stays in expensive hotels." Adomavicius and Tuzhilin (2001) describe how DM methods can be used for learning such rules and for incorporating them into customer profiles.

Because many of the discovered rules can be spurious, irrelevant, or trivial, one of the main problems is how to select an *optimal* set of rules for each customer from the set of rules previously discovered with DM methods. This is crucial in profiling applications that (1) deal with large customer bases where the total number of patterns for all the customers can be in the millions, (2) deal with real-time problems, and (3) need to have only the most important patterns to provide better performance results. In such applications, personalized content needs to be delivered in real time, while the customer is waiting online. Therefore, content delivery decisions should be driven by only a few rules to guarantee real-time delivery and relevant content for the customer (Davis 1998). Consequently, optimization can help customer profiling and other applications to select a small number of the most important patterns from the set of previously discovered patterns.

2.3. Customer Interactions in eCRM

In this section, we will review the problems facilitating better interactions with the customers. In particular, we will focus on website design and on personalization problems.

2.3.1. Website Design. Websites can be viewed as user interfaces to information sources, and it is a challenge to construct efficient websites that provide customers with a good interface. Current DM approaches are based on mining Web logfiles for path traversal patterns and restructuring sites based on the analysis.

In Perkowitz and Etzioni (1998), the problem is defined in terms of automatic construction of index pages based on logfile data, with the goal of constructing index pages that provide users access to information that they are likely to view. The algorithm proposed has four steps: (1) processing the logfiles into user visits, (2) computing co-occurrence frequencies between pages and creating a similarity matrix, (3) creating a graph from this matrix and then

finding cliques in this graph, and (4) creating index pages corresponding to each clique in the graph. The algorithm is evaluated based on how often users simultaneously visit the pages in the index page. Srikant and Yang (2001) propose another interesting approach to the problem of automatically synthesizing Web pages. They address the issue of comparing the "expected location" of various pages with the actual location where the page is, and they propose an algorithm that automatically finds such pairs of Web pages. The main idea is to examine user sessions in the logfiles and identify backtracking points where a user backtracks to a previous page and continues the session. If a number of users backtrack at page p_i and finish their session at page p_j, then this approach considers p_i to be the "expected location" of p_j.

Opportunities exist for framing website design problems as optimization problems, and we outline one such method in the online appendix. For example, an objective of a website design can be to maximize navigational simplicity by minimizing the average number of clicks required to get to any page. Further, there may be several constraints on the design. A constraint, motivated by cognitive reasons, may be that the number of links that may be placed on any page should be less than some user-specified threshold value. Yet, another constraint may require the sports and finance pages to be explicitly linked to each other based on input from the marketing department. However, it is difficult to provide user-specific constraints a priori (in many cases, we may not even know these constraints before looking at the data). DM can help to address this problem. First, the website is structured as per the solution to the optimization problem. As the user navigates this site, data on the user's access patterns is gathered. DM can, thus, identify additional constraints using patterns identified by this data. For example, DM can find that most users access the finance and sports pages together. This information can then be used as a constraint. The optimization problem is then solved incorporating the additional constraints. This process can be iteratively done until some user-defined stopping criterion.

This example demonstrates the opportunity for new research to identify similar problems that can be formulated in the context of website design. Note that in this mode, problems in website design will be mainly formulated as optimization problems (e.g., select the optimal set of links to place in a page delivered to a customer). However, DM will play a key role in the specification of the optimization space (e.g., which subsets of links should be in the consideration set), and will also play a part in the specification of the constraints (e.g., DM can be used to discover that for users from a certain domain, links a and b will have to be a part of the dynamically composed page).

2.3.2. Personalization. Another interesting customer interaction issue is the provision of personalized services to customers, including recommendations. Assume that a customer visits an online store (e.g., Amazon.com), and the store wants to recommend k products on the visitor's customized "welcome" page (e.g., 10 books for the customer). Crucially, this decision needs to be made in real time while the customer waits for the requested page. This is an example of a recommender systems problem, and has been extensively studied during the past decade.

Most of the approaches developed for solving this type of a recommendation problem use statistical and DM approaches, and usually try to determine "good" products to recommend to the customer. Various existing recommendation methods were classified by Balabanovic and Shoham (1997) into content-based, collaborative and hybrid approaches and have been reviewed in surveys (Pazzani 1999, Schafer et al. 2001, Adomavicius and Tuzhilin 2003). For example, the collaborative filtering method finds k "nearest neighbors" c_i for customer c (based on various measures that define distances between customers c_i and c, $d(c, c_i)$), takes the products that customers c_i have rated that customer c has not yet consumed, and rates these products based on the average ratings that the k customers c_i have already assigned to them, weighted by the distance measures $d(c, c_i)$. Then, the products with the highest ratings are recommended to customer c. Many proposals have been made for defining distances and weights to compute the weighted ratings (Pazzani 1999).

As argued in Adomavicius and Tuzhilin (2003), the recommendation problem can also be formulated as an optimization problem that selects the *best* items to

recommend to a user. More specifically, if *R:* Users × Items → Ratings is a rating function specifying how much user $u \in$ Users liked item $i \in$ Items, then the recommendation problem determines how to select item i'_u for user u so that

$$\forall u \in \text{Users}, \qquad i'_u = \arg\max_{i \in \text{Items}} R(u, i).$$

Reconsider, for example, that Amazon.com may want to determine which *best* 10 books to put on the customer's "welcome" page. The challenge for this problem is that the rating function is usually partially specified (i.e., not all entries in matrix $R(u, i)$ have known ratings). Therefore, it is necessary to specify how unknown ratings $R(u, i)$ should be estimated from the set of the previously specified ratings. Numerous methods have been developed for estimating these ratings, and the surveys (Pazzani 1999, Adomavicius and Tuzhilin 2003) describe some of these methods.

Once the optimization problem is defined, DM can contribute to its solution by learning additional constraints with DM methods, thereby significantly reducing the search space. For example, in the case of an online wine store, we can learn that a customer usually prefers to buy red inexpensive wines, however, on certain occasions, such as his wife's birthday, he usually buys midrange white Chablis. By storing this information in the customer's profile and invoking it on these special occasions, much tighter constraints on recommendations can be specified.

In this section, we reviewed some of the existing approaches to solving three main analytical eCRM problems: (1) specifying effective performance metrics (such as LTV) and finding optimal solutions based on them, (2) developing effective customer analysis, and (3) customer interaction methods. In general, we described how DM and optimization can help each other to provide better eCRM solutions, pointing out how many of the eCRM problems discussed above have been traditionally addressed using DM. There are significant opportunities for optimization to help the DM approaches to these eCRM problems. More generally, there are significant opportunities for optimization to help various key DM problems, and we discuss these in the next section.

3. How Optimization Can Help Traditional DM Problems

Optimization can contribute to DM in one of two ways: (1) optimization can be a *component* of a larger DM process, or (2) new DM techniques can be built using *entirely* optimization-based methods. For example, optimization as a component of a larger DM technique can be used as a method for determining the selection of the best decision tree out of a set of previously generated decision trees using a genetic algorithm (Fu et al. 2003), and estimating the optimal weights of a multilayer neural network using gradient descent (Rumelhart et al. 1986). Support Vector Machines (SVM) (Vapnik 1995, Burges 1998) fall into the second category, because selection of an optimal separating hyperplane (or a surface) *constitutes* the DM method.

In this section, we consider the following DM problems: (1) feature selection, (2) active learning, (3) DM model optimization, (4) selection of the best DM model or pattern, (5) classification using mathematical programming, (6) clustering, and (7) rule and constraint discovery. We demonstrate how optimization can significantly contribute to DM by developing solutions that are better and more often theoretically sound. Note that the first four problems suggest ways in which optimization can help by being a component of a larger DM process, while the last three problems suggest opportunities to develop new DM techniques using optimization-based methods.

3.1. Feature Selection

When optimization is used to preprocess click-stream data, as described in §2.2.1, optimization is used before DM. In general, optimization can be useful during the preprocessing stage of the DM process (Fayyad et al. 1996) where it can be used to select both an optimal set of features (or attributes) to mine over, and the actual *data* to run the DM models on.

Good feature selection techniques can significantly improve DM algorithms in two ways. First, these techniques can contribute to dimensionality reduction and faster running times. Second, they can contribute to building more accurate models from data. The feature selection problem is defined as selecting a "best" subset of features from a finite space of features,

and has been studied as an optimization problem in Olafsson and Yang (2002) who define the problem in general as

$$\max f_{\text{accuracy}}(\text{Model}, F)$$
$$x_i(\text{the decision variables}) = 1$$
$$\text{if feature } i \text{ is selected, 0 otherwise,}$$
$$i \in F_0 \quad \text{is the set of possible features,}$$
$$\sum x_i \geq \min f \quad \text{and} \quad \sum x_i \leq \max f,$$

where F is the set of features selected, and $\min f$ and $\max f$ are the minimum and maximum number of desired features, respectively.

For specific formulations of the feature selection problem, see the work of Bradley et al. (1998) described in §3.5.1. As mentioned above, an important eCRM application here is the selection of features from preprocessed click-stream data. A single user session can be preprocessed into hundreds of binary (indicator) variables for every page accessed in the session and several continuous variables (e.g., average time spent per page). For a given task (e.g., predicting if this customer will make a purchase in the session), the feature selection problem is to select the most relevant subset of these variables for use in modeling.

3.2. Active Learning

Active learning (MacKay 1992, Cohn et al. 1996, Saar-Tsechansky and Provost 2001) constitutes an approach to selecting the data on which to train DM models, and a good summary of active learning is provided in Saar-Tsechansky and Provost (2001). Mannino and Mookerjee (1999) address a related problem where they optimize the cost of information acquisition for expert systems. By selecting only the data with high utility to the model, active learning aims to minimize the data needed for model building. The usual scenario considered in active learning is that all explanatory variables are known, a current model of the target (dependent variable) exists, but the dependent variable values are often unknown and expensive to acquire. The problem is to determine from which "best" points to acquire this target value. For example, consider the problem of determining user satisfaction at an online store. Determining what content is actually shown to various users is straightforward, and this is automatically recorded by eCRM systems. However, determining user satisfaction is important, but expensive, and may require incurring the cost of conducting surveys. Active learning approaches can be used to determine the best records to get the labels for (i.e., the identity of which customers' data information would be most helpful for satisfaction data).

The different active learning methods fall into two categories: (1) heuristic based, and (2) optimization based (Hasenjager and Ritter 1999). The "query by committee" approach (Engelson and Dagan 1999, Freund et al. 1997) employs several models, and each model makes its own predictions on the unseen data. The data points chosen are those in which there is maximum disagreement among the models. Optimization approaches employ an objective function, and those data points that optimize this objective function are selected. In Cohn et al. (1996), data points are selected to minimize the overall prediction variance of a model, and Saar-Tsechansky and Provost (2001) select the data according to the variance of bootstrap predictions of class probability estimates. MacKay (1992) proposes a criterion based on information gain and suggests that for Gaussian distributions, this criterion is equivalent to the heuristics that choose data where the current model yields the largest error. Inherently, the active learning problem is an optimization problem (choose the "best" set of data points to build a model), but this has not always been cast as such by prior work. Therefore, it constitutes an opportunity for optimization researchers to do so, as well as suggest new mechanisms for active learning.

3.3. DM Model Optimization

Many DM methods can be directly formulated as optimization problems or can have an optimization component that is a part of the DM problem (Hand et al. 2001). Assume that Λ is the space of the models or patterns parameterized by some set of parameters $\Theta = (\theta_1, \ldots, \theta_d)$. Also, let $S(\theta_1, \ldots, \theta_d \mid D, M)$ be the scoring function specifying how well a specific model (also called "structure" in Hand et al. 2001) M fits data D for the set of parameters Θ. We want to find M and the set of parameters Θ that optimize

the scoring function S for data D. For example, Λ can be the class of decision trees, $\theta_1, \ldots, \theta_d$ can be the parameters controlling the node splitting process, and the scoring function $S(\theta_1, \ldots, \theta_d \mid D, M)$ can be defined in terms of the predictive accuracy of the tree on out-of-sample data. Many other DM problems, such as neural networks and rule discovery problems, can also be formulated as optimization problems in a similar manner. Hence, optimization is a fundamental component of many learning methods used in DM.

As Hand et al. (2001) points out, the solution to this general optimization problem depends on the nature of the parameter space Θ and the scoring function S. If the parameter space Θ and the scoring function S are "simple," we may find a *closed-form* solution to the optimization problem. However, for most of the optimization problems encountered in DM, it is impossible to find closed-form solutions, and the optimal solution needs to be obtained using the following iterative method:

(1) *Initialize.* Choose an initial value for the parameter vector $\Theta = \Theta_0$. Set $i = 0$.

(2) *Iterate.* Compute the value of Θ_{i+1} based on Θ_i using the methods described below.

(3) *Terminate.* Determine if Θ_i approaches a local optimum by examining the distances either between Θ_i and Θ_{i+1} or between $S(\Theta_i)$ and $S(\Theta_{i+1})$.

(4) Repeat steps (1)–(3) for different initial values of Θ_0, and choose the best among the local optima found thus far.

Various parameter estimation methods differ in the way the iterative step 2 is performed. For some methods, the value of Θ_{i+1} can be specified in terms of Θ_i with an equation. For example, gradient descent and similar optimization methods express Θ_{i+1} as $\Theta_{i+1} = \Theta_i + \lambda_i \mathbf{v}_i$, where $\mathbf{v}_i$ is the vector in the parameter space Θ specifying the direction toward the next point (e.g., $\mathbf{v}_i$ can be the gradient of S at Θ_i), and λ_i is a scalar specifying how far we need to go along $\mathbf{v}_i$. For example, weights in multilayer neural networks can be computed using gradient descent. Another example of the iterative procedure constitutes the Expectation Maximization (EM) algorithm (Dempster et al. 1977) that is widely used in various DM applications such as estimating the values of missing data (Little and Rubin 1987), computing state transition matrices in Hidden Markov Models (HMM) (Thrun and Langford 1997), and computing clusters using Gaussian mixture models (Xue and Jordan 1996). One example of utilizing the EM algorithm in eCRM applications is the work of Ypma and Heskes (2002), who use click-stream data and the EM algorithm to learn an HMM, in which the states learned provide important insight into user navigational patterns on the Web.

In some cases, however, it is impossible to express the value of Θ_{i+1} in terms of Θ_i using an equation, and one needs to resort to an algorithm that computes the next value of Θ_{i+1}. This gives rise to the class of combinatorial optimization problems that require heuristic search methods across the space Θ for a solution yielding the optimal score. For example, some of the rule discovery methods generate rules using heuristic search techniques, where the strength of a rule can be defined using one of the many criteria used in DM literature, such as the ones described in Tan et al. (2002). An excellent example of using heuristic search procedures for solving DM problems is the use of tabu search (Glover 1989), which constitutes an iterative greedy search algorithm using memory of past points visited to improve search. In classical gradient descent, the neighborhood of the current solution is searched, and the best point in this neighborhood is chosen as the current best solution. However, this converges to a local optimum. Simulated annealing techniques can be used to move out of local optima and to continue search. Doing so could, however, result in visiting the same points again and getting into cycles. To prevent converging fast into local optima and to prevent getting into cycles, tabu search restricts the selection of the neighborhood of a point by imposing a "tabu" on some subset of points. This concept has been applied in Battiti and Tecchiolli (1995) to train the weights of neural networks, which have been used in building predictive models in eCRM applications. The experiments in Padmanabhan et al. (2001) show that neural networks built on click-stream data outperform several other models of purchase prediction.

As another example of the heuristic search for an optimal solution to a DM problem, consider the problem of selecting the best decision tree using the Minimum Description Length (MDL) principle

(Rissanen 1978). The MDL principle selects a model from a set of all models that minimizes the total number of bits needed to encode the model and the data given the model. Formally, let $D = \{d_1, d_2, \ldots, d_N\}$ be a set of data points and let H be a set of models (hypotheses) describing data D. Also, let C_1 be a coding scheme for hypotheses from H, and C_2 a coding scheme for data D given hypothesis h from H. Then, the MDL principle selects the hypothesis h_0 from H that minimizes the total length of the encoded model and the description of the data given the model

$$h_0 = \arg\min_{h \in H} \left(L_{C_1}(h) + L_{C_2}(D \mid h) \right) \quad (1)$$

taken across different possible coding schemes C_1 and C_2.

For decision tree construction, a tree can first be built using standard DM techniques, such as See5/C4.5 (Quinlan 1993), and then an optimal subtree can be selected that is minimal in the MDL sense, i.e., one that minimizes the total encoding of the tree and the data given the tree (Quinlan and Rivest 1989). This selection is equivalent to pruning the tree and is done to avoid overfitting. MDL has also been used in learning Bayesian networks. Jaronski et al. (2001) use Bayesian networks to understand online audiences in general, and to model stickiness and repeat visits of customers to websites.

3.4. Selection of the Best DM Model or Pattern

Some optimization methods operate on the models or patterns produced by DM algorithms and select the best model or pattern from a set of generated candidates. For example, Kennedy et al. (1997) and Fu et al. (2003) describe how a genetic algorithm selects the best decision tree from a set of decision trees generated by tree induction methods. This selection process is performed in the postprocessing stage *after* the DM algorithm generates decision trees.

Kennedy et al. (1997) present genetic algorithm encoding decision trees as chromosomes, using a variation of one of the standard tree traversal methods. The approach uses the predictive accuracy of decision trees as a fitness function and defines crossover and mutation operations on decision trees and their encodings. Fu et al. (2003) build multiple decision trees from the data by subsampling from the data and building individual decision trees for each sample. This approach then uses genetic algorithms to breed better decision trees using the ideas presented in Kennedy et al. (1997), and reports higher accuracy rates of their approach in comparison to other tree induction methods.

One particular opportunity for using optimization after DM is in selecting the "best" patterns generated by DM algorithms from data. As mentioned in §2.2.2, the problem of building optimal profiles using optimization is an example of this sort of opportunity in eCRM. This problem can be formulated as an optimization problem that selects a set of rules from a discovered set such that they maximize certain objectives such as revenue. There has been little work in DM on this topic, and this is an opportunity for optimization researchers to express these as optimization problems that are formulated and solved *after* the DM step is completed. This would be particularly relevant and useful given the widespread use of rule discovery algorithms in DM.

3.5. Classification

In this section, we review the extensive work on building classifiers using mathematical programming. This work is particularly relevant for eCRM because the domain is characterized by several natural classification problems. For example, we can predict if a customer will make a purchase within a session or not and, thus, classify the customer into "buyer" versus "nonbuyer." Other classification problems include predicting if a customer will be a repeat visitor or will click on an advertisement, or will follow one of several recommendations. The use of optimization methods for learning optimal discriminant functions can be traced all the way back to the work of Fisher (1936). Many researchers have extended this approach across the last 40 years, and more recent surveys of some of these methods can be found in Mangasarian (1997) and Bradley et al. (1999). We now consider linear, nonlinear, and integer programming methods for building classifiers.

3.5.1. Building Classifiers Using Linear Programming. A linear binary classification problem can be

formulated as follows. Given two sets of points **A** and **B** in the n-dimensional real space R^n, the task is to separate them with a hyperplane $\mathbf{w}\mathbf{x}+\gamma=0$. The separable case was studied in Mangasarian (1965) in which **A** and **B** *can* be separated by a hyperplane $\mathbf{w}\mathbf{x}+\gamma=0$ so that the conditions

$$\mathbf{A}\mathbf{w} > \mathbf{e}\gamma, \qquad \mathbf{B}\mathbf{w} < \mathbf{e}\gamma \tag{2}$$

would hold, where **A** and **B** are the $m \times n$ and $k \times n$ matrices of reals, whose rows define sets **A** and **B** of m and k points, respectively, and **e** is a unity vector. In particular, Mangasarian (1965) formulated a necessary and sufficient condition for linear separability of sets **A** and **B** as a positive solution of a certain linear programming problem. Alternative conditions were subsequently formulated in Smith (1968) and Grinold (1972).

In many complex applications, the separability assumption is not realistic because it may be impossible to find a separating hyperplane. To deal with this issue, Mangasarian (1968) separated sets **A** and **B** with a piecewise-linear discriminant function as follows. First, two parallel hyperplanes, $\mathbf{c}\mathbf{x}=\beta$ and $\mathbf{c}\mathbf{x}=\alpha$, are generated by solving the following nonlinear problem:

$$\min_{c,\alpha,\beta}\{-\alpha+\beta \mid \mathbf{A}\mathbf{c}-\mathbf{e}\alpha \geq 0,\ -\mathbf{B}\mathbf{c}+\mathbf{e}\beta \geq 0,\ -\mathbf{e} \leq \mathbf{c} \leq \mathbf{e},\ \|\mathbf{c}\|_2^2 \geq n\}.$$

As Mangasarian (1968) shows, these hyperplanes are the closest ones, which contain between themselves, the intersection of the convex hulls of points in **A** and **B** (note that the nonemptiness of this intersection makes the two sets inseparable). By iteratively repeating this procedure for the subsets of **A** and **B**, which are contained between and lie on the two hyperplanes $\mathbf{c}\mathbf{x}=\beta$ and $\mathbf{c}\mathbf{x}=\alpha$, a piecewise-linear discriminant function is obtained that separates points **A** and **B**. However, the decision problem presented in Mangasarian (1968) is NP-complete (Mangasarian et al. 1990). To deal with this problem, Mangasarian et al. (1990) replace the L_2 norm $\|\mathbf{c}\|_2^2$ in the above minimization problem with the L_∞ norm (the revised term is $\|\mathbf{c}\|_\infty = 1$), and demonstrates that the solution to this problem can be obtained in polynomial time by solving $2n$ linear programs, where n is the (usually small) dimensionality of the pattern space. Then, Mangasarian et al. (1990) use a similar iterative procedure that results in a piecewise-linear function separating sets **A** and **B**, but the resulting solution is more efficient than the one described in Mangasarian (1968).

An alternative solution to the nonseparable case was proposed in Bennett and Mangasarian (1992), where the problem was formulated as finding a hyperplane $\mathbf{w}\mathbf{x}+\gamma=0$ that minimizes the average sum of misclassified points in **A** and **B** (i.e., points that violate conditions of (2)). More precisely, the optimization criterion in Bennett and Mangasarian (1992) is

$$\min_{\mathbf{w},\gamma} \frac{1}{m}\left\|(-\mathbf{A}\mathbf{w}+\mathbf{e}\gamma+\mathbf{e})_+\right\|_1+\frac{1}{k}\left\|(\mathbf{B}\mathbf{w}-\mathbf{e}\gamma+\mathbf{e})_+\right\|_1, \tag{3}$$

where $\mathbf{x}_+$ denotes a vector such that $(\mathbf{x}_+)_i = \max\{\mathbf{x}_i, 0\}$, and $\|\mathbf{x}\|_1 = \sum_i \mathbf{x}_i$ is the L_1 norm. Note that (3) will be zero if conditions (2) are satisfied, as in the separable case. Bennett and Mangasarian (1992) present an efficient solution to this problem.

Building on this work, Bradley et al. (1998) study the feature selection problem that tries to reduce the dimensionality of space R^n so that separation of the two classes still remains meaningful. Bradley et al. (1998) point out that (3) is equivalent to the following formulation:

$$\min_{w,\gamma,y,z}\left\{\frac{e^T y}{m}+\frac{e^T z}{n}\ \middle|\ -Aw+e\gamma+e \leq y,\ Bw-e\gamma+e \leq z,\ y \geq 0,\ z \geq 0\right\}.$$

Bradley et al. (1998) then show how this classification problem can be converted into a "feature selection" problem. The idea is to suppress as many components of w as possible such that the separating hyperplane is of as few dimensions as possible. To do this, Bradley et al. (1998) introduce an extra term $\lambda \in [0, 1)$ and weight the objective function as

$$\min_{w,\gamma,y,z}\left\{(1-\lambda)\left(\frac{e^T y}{m}+\frac{e^T z}{n}\right)+\lambda e^T|w|_*\ \middle|\ -Aw+e\gamma+e \leq y,\ Bw-e\gamma+e \leq z,\ y \geq 0,\ z \geq 0\right\},$$

where $\mathbf{w}*$ denotes a vector, such that $(\mathbf{w}*)_i = 1$ if $\mathbf{w}_i$ is positive and 0 otherwise.

3.5.2. Building Classifiers Using Nonlinear Programming. The separability problem has been approached from a different angle using the nonlinear SVM method (Vapnik 1995, Burges 1998) that builds a classifier in an n-dimensional Euclidean space $\mathbf{R^n}$ as follows. Let $\{\mathbf{x}_i, y_i\}$, $i = 1, \ldots, M$, $y_i \in \{-1, 1\}$ be a set of M labeled training data points, where labels are binary (true/false or $-1/+1$). We will start our discussion of SVM methods with a separable case when there exists a hyperplane $\mathbf{w}\,\mathbf{x} + b = 0$ in $\mathbf{R^n}$ that separates the positive from the negative training examples. Let d_+ and d_- be the shortest distances from the positive and negative points in the training set to the separating hyperplane (see Figure 1). Then, the SVM method seeks to find such a separating hyperplane $\mathbf{w}\,\mathbf{x} + b = 0$ that maximizes the total distance $d_+ + d_-$ called the "margin." The hyperplanes parallel to the separating hyperplane and located from it at the distances d_+ and d_- are denoted as H_1 and H_2, respectively, and are shown in Figure 1. Note that there are no training examples located between these two hyperplanes.

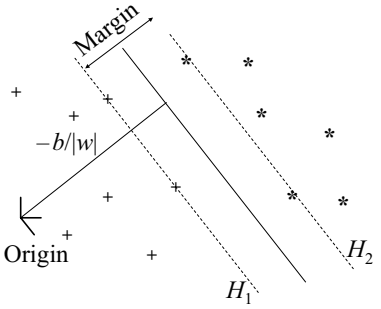
Vapnik (1995) and Burges (1998) show that this problem can be stated as a quadratic programming problem seeking to minimize the L_2 norm of $\mathbf{w}$, i.e., $\|\mathbf{w}\|^2$ subject to the constraints

$$y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0 \qquad \text{for } i = 1, \ldots, M.$$

This constrained optimization problem is solved using standard Lagrangian methods and yields

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i. \tag{4}$$

Figure 1 The Linearly Separable Case for Support Vector Machines



Then, the Wolfe dual maximization problem is (with $\alpha_i \geq 0$)

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j, \tag{5}$$

and the Karush-Kuhn-Tucker conditions yield

$$\alpha_i(y_i(\mathbf{w}\mathbf{x}_i + b) - 1) = 0 \qquad \text{for } i = 1, \ldots, M. \tag{6}$$

We can see from this that if $\alpha_i > 0$, then the point $\mathbf{x}_i$ lies on one of the separating hyperplanes H_1 or H_2, depending on the value of y_i. Such points lying on H_1 and H_2 are called *support vectors* and are the points on the dotted lines in Figure 1. All other training points have $\alpha_i = 0$ because of (6). Notice that only support vectors contribute to the calculation of $\mathbf{w}$ in (4). The value of b can be computed using (6) and (4). Once we trained a SVM by computing $\mathbf{w}$ and b (using (4) and (6)), the SVM classifies a new pattern $\mathbf{x}$ by computing $f(\mathbf{x}) = \mathbf{w}\,\mathbf{x} + b$ and determining the *sign* of $f(\mathbf{x})$.

For the nonseparable case, we can maximize the margin distance $d_+ + d_-$ minus the total distances of all the misclassified points (those points that fell between the separating hyperplanes H_1 and H_2). Vapnik (1995) and Burges (1998) show that the problem can still be formulated as a Wolfe dual quadratic programming problem that maximizes

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j, \tag{7}$$

but this time subject to the additional constraints $0 \leq \alpha_i \leq C$, where C is a parameter chosen by the user that assigns penalties to misclassification errors (larger C means larger errors). The solution is again given by (4), where the summation is taken over all the support vectors ($\alpha_i > 0$).

We have reviewed linear SVMs so far, where separation between classes is done by a hyperplane. Much SVM work focuses on nonlinear SVMs where the separation between classes is done with nonlinear surfaces. This work is surveyed in Vapnik (1995) and Burges (1998).

Although SVMs have been mainly used in the offline applications, one example of using SVMs for classifying Web searches is presented in Chen and Dumais (2000). In particular, their paper describes how SVMs are used for the automatic classification of

users' Web search results into several categories, making it easier to understand the results of search. In this application, an SVM is built to classify text documents into categories.

3.5.3. Building Classifiers Using Integer Programming. Logical Analysis of Data (LAD) is another method for classification proposed in the optimization literature (Crama et al. 1988). It builds a classifier for a binary target variable based on learning a logical expression that can distinguish between positive and negative examples in a data set. In this sense, the problem is similar to the concept learning problem studied in machine learning (Mitchell 1997). The principal difference is that in the logical analysis of data literature, the solutions to the problem are developed by formulating and solving optimization problems; mainly integer programs. Early work in this area was targeted at binary domains. Recent work has extended this target to include numeric predictor attributes by using a process known as "binarization" (Boros et al. 1997). Consider the example from Boros et al. (1997) shown in Table 1.

The data consists of five records of three numeric predictor attributes (A, B, and C) and a binary target. The approach in Boros et al. (1997) uses cutoff values α_A, α_B, α_C to create binary variables X_A, X_B, X_C, where $X_A = 1$ iff $A > \alpha_A$ and X_B, X_C are similarly computed for all points $(A, B, C) \in S^+ \cup S^-$, where S^+ is the set of positive examples (e.g., the first three records in Table 1) and S^- is the set of negative examples. To learn a classifier, the LAD approach views the last four columns of Table 1 as a partially defined Boolean function (pdBf) because it can be viewed as a partially specified truth table (which only specifies the outcome for some values of X_A, X_B, X_C). An *extension* of a pdBf is defined as a Boolean function that is consistent with all the positive and negative examples of the data. An example of such a function for the data in Table 1 is $(X_A \wedge X_B) \vee (X'_A \wedge X'_B) \vee (X_B \wedge X_C)$. The problem then becomes finding such functions. The LAD literature presents methods to find such extensions for various specific classes of Boolean functions (Boros et al. 1995). In particular, for a given pdBf, there can be several consistent extensions and Boros et al. (1997) formulates different integer programs to find the extension with a minimum number of cutoff points for various different classes of functions. Note that the above approach does not distinguish optimal models in training versus test data sets. In DM, however, this is an important distinction because predictive models are required to do well on unseen data. A challenge for optimization researchers building DM algorithms is to explicitly address this issue and to study how optimization over training sets produces optimal models for unseen data.

Table 1 Example Data for the LAD Approach

Original variables			Boolean variables			Target variable
A	B	C	X_A	X_B	X_C	Y
3.5	3.8	2.8	1	1	0	1
2.6	1.6	5.2	0	0	1	1
1.0	2.1	3.8	0	1	1	1
3.5	1.6	3.8	1	0	1	0
2.3	2.1	1.0	0	1	0	0

3.6. Clustering

In §3.5, we focused on the description of mathematical programming methods for building classifiers. In addition, mathematical programming has been used for clustering. In particular, Bradley et al. (1997) formulate a concave minimization problem for finding k clusters in an n-dimensional Euclidean space R^n. Given a set of m points $\mathbf{A} = \{A_i\}_{i=1,\ldots,m}$ and k clusters $\mathbf{C}_l$, $l = 1, \ldots, k$, the problem is to find the locations of such clusters in R^n so that the sum of the minima over l from $\{1, \ldots, k\}$ of the 1-norm distance between each point A_i and the cluster centers C_l is minimized:

$$\begin{aligned} \min_{C,D} \quad & \sum_{i=1}^{m} \min_{l} \{\mathbf{e}^T D_{il}\} \\ \text{subject to} \quad & -D_{il} \le A_i^T - C_l \le D_{il}, \\ & i = 1, \ldots, m; \ l = 1, \ldots, k, \end{aligned}$$

where $D_{il} \in R^n$ is the dummy variable that bounds the components of the difference between point A_i^T and the center C_l and $\mathbf{e}$ is the unity vector. The solution presented in Bradley et al. (1997) reduces this problem to a bilinear program that is solved by using a k-median algorithm.

3.7. Rule and Constraint Discovery

In addition to classification and clustering problems in DM, the discovery of rules in data is another important and popular problem. DM researchers have formulated various rule discovery problems and have suggested methods for discovering rules predominantly based on intelligent searching techniques. For example, consider association rules having the form $A_1, A_2, \ldots, A_N \rightarrow B_1, B_2, \ldots, B_K$, where the LHS and RHS are both conjunctions of binary attributes (*itemsets*). Agrawal et al. (1995) present an efficient algorithm that discovers all association rules with confidence and supports values above some thresholds, where *support* is the number of transactions in which both the LHS and RHS of the rule hold and *confidence* is the strength of the rule computed as *count*(LHS, RHS)/*count*(LHS). There is an interesting relationship between problem formulation and search here. Without the minimum support constraint, the search has to be exhaustive and would generate a huge number of rules. With the minimum support constraint, the problem can be addressed using more intelligent search methods that significantly limit the search space and, therefore, make the problem practical. The opportunity for optimization researchers is to develop new optimization-based search methods that can focus on learning rules faster than traditional DM search methods.

Related work (Hong 1993, Rymon 1994) in machine learning presents an initial approach to exploring this opportunity. The approach minimizes a Boolean logic expression to discover minimal rules from binary data. For example, consider two binary attributes A and B and assume that there is a dependent binary variable C. Let $A = 1$, $B = 0$, and $A = 1$, $B = 1$ represent points corresponding to positive instances of C. This data gives rise to the trivial expression $AB' + AB \rightarrow C$, where the LHS of the rule is in disjunctive normal form. Each term in the LHS corresponds to a conjunction of literals obtained from a single data record. The LHS of the rule is now a Boolean logic expression and logic minimization algorithms (Hong 1993) can be used to simplify it. The LHS simplifies to A and the rule generated is $A \rightarrow C$. In general, simplifying the LHS needs to be efficiently done. See Hong (1993) and Rymon (1994) for examples of algorithms that do this.

In this section, we described how optimization can help solve traditional DM problems. More generally, the interaction between DM and optimization is bidirectional, and there are opportunities for DM to help solve problems that are addressed traditionally using an optimization framework. For example, an opportunity for using DM with optimization is to use DM to learn patterns in data that can, in turn, be used as constraints in a traditional optimization framework. This can ensure that the solutions derived from the optimization approach are consistent with the dominant patterns that emerge from the data and, hence, the solution quality can be improved. Further, using better constraints can help solve the optimization problem faster by appropriately restricting the search space. Consider the content management problem of placing advertisements on Web pages, such as done by DoubleClick. This can be formulated as an optimization problem as pointed out by Geoffrion and Krishnan (2001). Many of the constraints for this problem are about whom to show what types of advertisements. Although the content management problem is an optimization problem, leading CRM vendors do not formulate it as such and, instead, use ad hoc heuristics to determine the selection of content. However, they still use constraints in finding solutions. For example, BroadVision lets domain experts explicitly specify the types of content that should be delivered to various types of users (e.g., users from .com domains should not be shown advertisements with audio during the day). In contrast to this, Blue Martini learns these constraints from data using DM. Before using these discovered constraints to guide content delivery, experts are shown the constraints and decide whether or not to use them. The applications of website design and recommender systems discussed in §2.3 give additional examples of using DM to learn constraints for eCRM problems. While a detailed treatment of how DM can help optimization is beyond the scope of this paper, the online appendix provides additional examples of such interactions.

4. Conclusions

Most firms today recognize the importance of building and maintaining strong relationships with their customers. As firms increasingly rely on their online presence to interact with customers, eCRM will continue to grow in importance. Addressing eCRM problems by utilizing the strengths of both DM and optimization can, therefore, be a useful area for future research. A better understanding of the problems in eCRM and of the possible ways in which DM and optimization can interact, can help exploit the opportunities for synergistically using both approaches in the solution of eCRM problems. To this end, in this paper we (1) surveyed various eCRM applications where such interactions are important, and showed that relatively little work has been done on exploring these interactions in the eCRM context and (2) described different ways in which optimization can help DM by surveying extensive existing literature. More generally, the interactions between DM and optimization are bidirectional and the online appendix presents a treatment of how DM can help optimization.

A strong theme emerging from our survey is the existence of several natural opportunities for interactions between DM and optimization in eCRM and also in general. Optimization can help DM in eCRM applications by developing more systematic methods of preprocessing click-stream data in a manner that optimizes the performance of a DM model subsequently built on the preprocessed data. As pointed out in §2.2.1, there are opportunities for researchers to formulate and solve this optimization problem. As was also pointed out in §2.2.2, many eCRM applications require concise profiles containing the most important information about customers. This is clearly a constrained optimization problem that can be studied by optimization and DM mining researchers. Finally, as pointed out in §3.5, several classification problems exist in eCRM, and relatively few optimization-based DM methods, such as SVM and LAD, have been applied in this domain.

There are also important opportunities beyond eCRM for optimization to help DM. DM methods are only as good as the data they mine and, hence, rigorous preprocessing methods can significantly contribute to building better DM models. Optimization can be particularly effective for developing rigorous active learning and feature selection approaches. Because many DM methods generate a large number of patterns, postprocessing of DM results remains an important problem. Optimization can play a valuable role here. Finally, building new optimization-based algorithms for DM can directly contribute to DM by providing new methods. Various opportunities for optimization to help DM are summarized in Table 2.

Similarly, there are interesting opportunities for DM to help develop better optimization solutions to eCRM problems. As pointed out in §§2.3.1 and 2.3.2, opportunities abound for solving the problems of designing effective websites and generating targeted recommendations by using an optimization-based approach. There are opportunities for DM to help by learning constraints from the data. Both these problems of designing effective websites and generating targeted recommendations are critical in eCRM, and will benefit from further exploration by optimization and DM researchers. Finally, interesting research opportunities exist related to the problem of structuring policies to maximize LTV by exploiting customer data. In general, we believe that DM can play

Table 2 Opportunities for Optimization to Help DM

Method of interaction	eCRM opportunities	General opportunities
Optimization as a DM component	Developing methods for preprocessing click-stream data to optimize performance of eCRM models (§2.2.1) Building optimal customer profiles (§2.2.2)	Developing optimization-based methods for active learning and feature selection (§§3.1 and 3.2) Developing optimization-based methods for selecting best patterns or models generated by DM
Optimization-based algorithms for DM	Applying optimization-based algorithms (e.g., SVM, LAD) to classification problems in eCRM	Developing new optimization-based algorithms for solving DM problems (classification, rule discovery, and so on)

a critical role in identifying relevant variables or learning patterns that can be used within an optimization framework. Such interactions can help build better and faster solutions to problems, but have hardly been explored in the literature thus far.

References

Adomavicius, G., A. Tuzhilin. 2001. Expert-driven validation of rule-based user models in personalization applications. *Internat. J. Data Mining Knowledge Discovery* **5** 33–58.

——, ——. 2003. Recommendation technologies: Survey of current methods and possible extensions. Working paper, Stern School of Business, New York University, New York.

Agrawal, R., H. Mannila, R. Srikant, H. Toivonen, A. I. Verkamo. 1995. Fast discovery of association rules. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, eds. *Advances in Knowledge Discovery and Data Mining*. AAAI Press, Menlo Park, CA.

Balabanovic, M., Y. Shoham. 1997. Fab: Content-based, collaborative recommendation. *Comm. ACM* **40**(3) 66–72.

Battiti, R., G. Tecchiolli. 1995. Training neural nets with the reactive tabu search. *IEEE Trans. Neural Networks* **6**(5) 1185–1200.

Bennett, K. P., O. L. Mangasarian. 1992. Robust linear programming discrimination of two linearly inseparable sets. *Optim. Methods Software* **1** 23–34.

Blattberg, R. C., J. Deighton. 1991. Interactive marketing: Exploiting the age of addressability. *Sloan Management Rev.* (Fall) 5–14.

Boros, E., T. Ibaraki, K. Makino. 1995. Error-free and best-fit extensions of partially defined Boolean functions. Rutcor research report RRR 14–95, Rutgers University, Piscataway, NJ.

——, P. L. Hammer, T. Ibaraki, A. Kogan. 1997. Logical analysis of numerical data. *Math. Programming* **79** 163–190.

Bradley, P. S., U. M. Fayyad, O. L. Mangasarian. 1999. Mathematical programming for data mining: Formulations and challenges. *INFORMS J. Comput.* **11**(3) 217–238.

——, O. L. Mangasarian, W. N. Street. 1997. Clustering via concave minimization. M. C. Mozer, M. I. Jordan, T. Petsche, eds. *Advances in Neural Information Processing Systems*. MIT Press, Cambridge, MA.

——, ——, ——. 1998. Feature selection via mathematical programming. *INFORMS J. Comput.* **10**(2) 209–217.

Brijs, T., G. Swinnen, K. Vanhoof, G. Wets. 1999. Using association rules for product assortment decisions: A case study. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining*, San Diego, CA.

Brodley, C., R. Kohavi. 2000. KDD-cup 2000 organizer's report: Peeling the onion. *SIGKDD Explorations* **2**(2) 86–98.

Burges, C. J. C. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining Knowledge Discovery* **2**(2) 121–167.

Campbell, D., R. Erdahl, D. Johnson, E. Bibelnieks, M. Haydock, M. Bullock, H. Crowder. 2001. Optimizing customer mail streams at Fingerhut. *Interfaces* **31**(1) 77–90.

Chen, H., S. T. Dumais. 2000. Bringing order to the Web: Automatically categorizing search results. *Proc. CHI 2000, Human Factors Comput. Systems*. The Hague, The Netherlands, 145–152.

Cohn, D., Z. Ghahramani, M. Jordan. 1996. Active learning with statistical models. *J. Artificial Intelligence Res.* **4** 129–145.

Cooley, R., B. Mobasher. 1999. Data preparation for mining World Wide Web browsing patterns. *Knowledge Inform. Systems* **1**(1) 5–32.

Crama, Y., P. L. Hammer, T. Ibaraki. 1988. Cause-effect relationships and partially defined Boolean functions. *Ann. Oper. Res.* **16** 299–326.

Davis, O. 1998. Personal communications, October.

Dempster, A. P., N. M. Laird, D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc. B* **39** 1–38.

Dreze, X., A. Bonfrer. 2002. To pester or leave alone: Lifetime value maximization through optimal communication timing. Working paper, Marketing Department, University of California, Los Angeles, CA.

Dver, A. 2003. Customer relationship management: The good, the bad, the future. *Bus. Week* (April 28).

Dwyer, F. R. 1989. Customer lifetime valuation to support marketing decision making. *J. Direct Marketing* **3**(4) 8–15.

Engelson, S., Dagan. 1999. Committee-based sample selection for probabilistic classifiers. *J. Artificial Intelligence Res.* **11** 335–360.

Fayyad, U. M., G. Piatetsky-Shapiro, P. Smyth. 1996. From data mining to knowledge discovery: An overview. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, Cambridge, MA.

Fawcett, T., F. Provost. 1996. Combining data mining and machine learning for efficient user profiling. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining (ACM SIGKDD)*, Portland, OR.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Ann. Eugenics* **7**(2) 179–188.

Freund, Y., H. Seung, E. Shamir, N. Tishby. 1997. Selective sampling using query by committee algorithm. *Machine Learning* **28** 133–168.

Fu, Z., B. Golden, S. Lele, S. Raghavan, E. Wasil. 2003. A genetic algorithm-based approach for building accurate decision trees. *INFORMS J. Comput.* **15**(1) 3–22.

Geoffrion, A. M., R. Krishnan. 2001. Prospects for operations research in the e-business era. *Interfaces* **31**(2) 6–36.

Glover, F. 1989. Tabu search, Part I. *ORSA J. Comput.* **1** 190–206.

Grinold, R. C. 1972. Mathematical programming methods of pattern classification. *Management Sci.* **19** 272–289.

Gupta, S., D. R. Lehmann, J. A. Stuart. 2001. Valuing customers. Working paper, Columbia University, New York.

Hand, D., H. Mannila, P. Smyth. 2001. *Principles of Data Mining*, MIT Press, Cambridge, MA.

Hasenjager, M., H. Ritter. 1999. Active learning in neural networks. Working paper, University of Bielefeld, Bielefeld, Germany.

Hong, S. J. 1995. R-MINI: An iterative approach for generating minimal rules from examples. *IEEE Trans. Knowledge Data Engrg.* **9**(5) 709–717.

Jaronski, W., J. Bloemer, K. Vanhoof, G. Wets. 2001. Use of Bayesian belief networks to help understand online audience. *Proc. Data Mining Marketing Appl. Workshop ECML/PKDD 2001*. Freiburg, Germany.

Kennedy, H., C. Chinniah, P. Bradbeer, L. Morss. 1997. The construction and evaluation of decision trees: A comparison of evolutionary and concept learning methods. D. Corne, J. L. Shapiro, eds. *Evolutionary Computing*. Lecture Notes in Computer Science, Springer-Verlag, 147–161.

Little, R. J., D. B. Rubin. 1987. *Statistical Analysis with Missing Data*. John Wiley and Sons, New York.

MacKay, D. J. C. 1992. Information-based objective functions for active data selection. *Neural Comput.* **4**(4) 590–604.

Mangasarian, O. L. 1965. Linear and nonlinear separation of patterns by linear programming. *Oper. Res.* **13** 455–461.

——. 1968. Multisurface method of pattern separation. *IEEE Trans. Inform. Theory* **14**(6) 801–807.

——. 1997. Mathematical programming in data mining. *Data Mining Knowledge Discovery J.* **1**(2) 183–210.

——, R. Setiono, W. H. Wolberg. 1990. Pattern recognition via linear programming: Theory and application to medical diagnosis. T. F. Coleman, Y. Li, eds. *Large-Scale Numerical Optimization*. SIAM, Philadelphia, PA, 22–31.

Mani, D. R., J. Drew, A. Betz, P. Datta. 1999. Statistics and data mining techniques for lifetime value modeling. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining*, San Diego, CA.

Mannino, M. V., V. S. Mookerjee. 1999. Optimizing expert systems: Heuristics for efficiently generating low cost information acquisition strategies. *INFORMS J. Comput.* **11**(3) 278–291.

Mitchell, T. 1997. *Machine Learning*. McGraw-Hill, New York.

Olafsson, S., J. Yang. 2002. Scalable optimization-based feature selection. *Proc. SIAM Workshop Discrete Math. Data Mining*. Arlington, VA, 53–64.

Padmanabhan, B., Z. Zhiqiang, S. Kimbrough. 2001. Personalization from incomplete data: What you don't know can hurt. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining (ACM SIGKDD)*, San Francisco, CA.

Pazzani, M. 1999. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Rev.* (December) 393–408.

Perkowitz, M., O. Etzioni. 1998. Adaptive web sites: Automatically synthesizing web pages. *Proc. National Conf. Artificial Intelligence*. Madison, WI.

Quinlan, J. R. 1993. C4.5: *Programs for Machine Learning*. Morgan Kaufman, San Mateo, CA.

——, R. L. Rivest. 1989. Inferring decision trees using the minimum description length principle. *Inform. Comput.* **80** 227–248.

Rissanen, J. 1978. Modeling by shortest data description. *Automatica* **14** 465–471.

Rosset, S., E. Neumann, U. Eick, N. Vatnik, Y. Idan. 2002. Customer lifetime value modeling and its use for customer retention planning. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining*. Edmonton, Alberta, Canada.

Rumelhart, D. E., G. E. Hinton, R. J. Williams. 1986. Learning internal representations by error propagation. D. E. Rumelhart, J. L. McClelland, eds. *Parallel Data Processing*, Vol. 1. MIT Press, Cambridge, MA, 318–362.

Rymon, R. 1994. On Kernel rules and prime implicants. *Proc. National Conf. Artificial Intelligence, AAAI*, Seattle, WA.

Saar-Tsechansky, M., F. J. Provost. 2001. Active learning for class probability estimation and ranking. *Internat. Joint Conf. Artificial Intelligence*, Seattle, WA.

Schafer, J. B., J. A. Konstan, J. Riedl. 2001. E-commerce recommendation applications. *J. Data Mining Knowledge Discovery* **5**(1/2) 115–153.

Schmittlein, D. C., D. G. Morrison, R. Colombo. 1987. Counting your customers: Who are they and what will they do next? *Management Sci.* **33**(1) 1–24.

Smith, F. W. 1968. Pattern classifier design by linear programming. *IEEE Trans. Comput.* **4** 367–372.

Srikant, R., Y. Yang. 2001. Mining web logs to improve website organization. *Proc. Tenth World Wide Web Conf.*, Hong Kong.

Srivastava, J., R. Cooley, M. Deshpande, P. Tan. 2000. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations* **1**(2).

Swift, R. 2002. Analytical CRM powers profitable relationships: Creating success by letting customers guide you. *DM Rev.* (February).

Tan, P. N., V. Kumar, J. Srivastava. 2002. Selecting the right interestingness measure for association patterns. *Proc. ACM Internat. Conf. Knowledge Discovery Data Mining*. Edmonton, Alberta, Canada.

Theusinger, C., K. Huber. 2000. Analyzing the footsteps of your customers. *WEBKDD*.

Thrun, S., J. Langford. 1997. Monte Carlo hidden Markov models. Technical report CMU-CS-98-179, Carnegie Mellon University, Pittsburgh, PA.

VanderMeer, D., K. Dutta, A. Datta, K. Ramamritham, S. Navathe. 2000. A. Enabling scalable online personalization on the web. *Proc. ACM Conf. Electronic Commerce*. Minneapolis, MN.

Vapnik, V. N. 1995. *The Nature of the Statistical Learning Theory*. Springer, New York.

Xue, L., M. I. Jordan. 1996. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Comput.* **8**(1) 129–151.

Ypma, A., T. Heskes. 2002. Categorization of web pages and user clustering with mixtures of hidden Markov models. *Proc. Workshop Web Knowledge Discovery Data Mining (WEBKDD 2002)*. Edmonton, Alberta, Canada, 31–43.

Zheng, Z., B. Padmanabhan, S. Kimbrough. 2003. On data preprocessing biases in web usage mining. *INFORMS J. Comput.* **15**(2) 148–170.

Accepted by Arthur M. Geoffrion and Ramayya Krishnan; received December 2001. This paper was with the authors 19 months for 8 revisions.