RETAILER PROMOTION PLANNING: IMPROVING FORECAST ACCURACY AND INTERPRETABILITY

MICHAEL TRUSOV, ANAND V. BODAPATI, AND LEE G. COOPER

This article considers the supermarket manager's problem of forecasting demand for a product as a function of the product's attributes and of market control variables. To forecast sales on the stock keeping unit (SKU) level, a good model should account for product attributes, historical sales levels, and store specifics, and to control for marketing mix. One of the challenges here is that many variables which describe product, store, or promotion conditions are categorical with hundreds or thousands of levels in a single attribute. Identifying the right product attributes and incorporating them correctly into a prediction model is a very difficult statistical problem. This article proposes an analytical engine that combines techniques from statistical market response modeling, datamining, and combinatorial optimization to produce a small, efficient rule set that predicts sales volume levels.

JOURNAL OF INTERACTIVE MARKETING VOLUME 20 / NUMBER 3–4 /
SUMMER / AUTUMN 2006
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dir.20068

MICHAEL TRUSOV

is a doctoral candidate at the UCLA Anderson School, 110 Westwood Plaza, Los Angeles, CA 90095; e-mail: michael.trusov@anderson.ucla.edu

ANAND V. BODAPATI

is Assistant Professor at the UCLA Anderson School, 110 Westwood Plaza, Los Angeles, CA 90095; e-mail: abodapat@anderson.ucla.edu

LEE G. COOPER

is Professor Emeritus at the UCLA Anderson School, 110 Westwood Plaza, Los Angeles, CA 90095; e-mail: lee.cooper@anderson.ucla.edu

INTRODUCTION

Since the early 1970s, promotions have emerged to represent the main share of the marketing budget for most consumer-packaged goods (Srinivasan, Pauwels, Hanssens, & Dekimpe, 2004). It is not surprising that numerous studies in the field of marketing research are devoted to promotions. Researchers have examined multiple facets of the promotion phenomenon, such as what, when, and how to promote, optimal length of a promotion, its strategic implications, the persistent effect of promotions, their impact on consumer behavior, and so on. In the business environment, promotion planning is a routine daily task for both retailers and manufacturers; however, the planning task for retailers is quite different from the one faced by manufacturers. Manufacturers—even those with very broad product lines—must develop promotion-planning procedures for a maximum of a couple hundred products. On the other hand, the average supermarket chain carries thousands of items, with some subset of them always being promoted. While there are many important issues that must be addressed in the context of promotion planning, one of the most basic daily issues is the planning of inventory for upcoming sales events. This aspect of promotion planning has received relatively little attention in marketing literature, possibly due in part to the more applied, business-operations nature of the planning process. On the other hand, the techniques traditionally employed by marketing scholars to build forecast models (e.g., econometric methods) are not always well suited for retail-industry-scale systems. Some recent studies, however, bring to light the topic of retail promotion planning and sales forecasting (e.g., Cooper, Baron, Levy, Swisher, & Gogos, 1999; Divakar, Ratchford, & Shankar, 2005). This interest may be attributable to the emergence of new conceptual modeling/computational methods as well as advancements in technologies.

Until the emergence of automated promotion-planning systems, the common practice for retail store managers was to use the "last like" rule in ordering inventory for upcoming promotion events. This meant that they ordered the same quantity of goods that was sold during a similar past promotion (Cooper et al., 1999). While this was the common (and only) approach for decades, advancements in technology offered better ways for managers to handle the planning process. The reliance on the simple "last like" rule became both inefficient and infeasible. Several authors have discussed the advantages of applying market-response models over traditional rule-of-thumb approaches. Indeed, the use of statistical models often results in significant cost savings, as out-of-stock and overstock losses are reduced.

To predict demand for future promotion events, the promotion-forecasting system PromoCast (Cooper et al., 1999) and its extension using data mining (Cooper & Giuffrida, 2000) exploit historical data accumulated from past promotions for each separate SKU in each store within a retail chain. PromoCast uses a traditional market-response model to extract information from continuous variables, and Cooper and Giuffrida (2000) used data mining techniques on the residuals to extract information from many-valued nominal variables such as promotion conditions or merchandise category. The output of the data mining algorithm is a set of rules that specify what adjustments should be made to the forecast produced by the market-response model.

The purpose of this study is to highlight some of the limitations inherent to the data mining techniques employed in the extension of the PromoCast system and to show how addressing these limitations can improve the accuracy of the forecasts and interpretability of produced knowledge. Specifically, we show that by allowing for set-valued features in the rule syntax (Cohen, 1996), the quality of predictions (in terms of correctly applied forecast adjustments) improves by as much as 50% (The original data mining algorithm produces an 11.84% improvement over the traditional market-response model.) Using this new PromoCast data mining algorithm as an example, we also show that by applying simple optimization techniques to the set of generated rules, the size of the rules space could be significantly reduced without loss of predictive ability. In our sample, we were able to reduce the number of generated rules from 156 to just 21 while preserving the overall accuracy of forecast. The optimized rules set is much more manageable and transparent to a market practitioner.

This article is organized as follows. In the next section, we provide a brief overview of PromoCast, a recent development in the field of automated promotion-forecasting systems. PromoCast has a strong market-performance record, compares favorably against other leading forecasting packages, and combines the strength of traditional market-response models with state-of-the-art data mining algorithms. Limitations of this system that are inherent to the whole class of such models are discussed. Next, we demonstrate how the incorporation of set-valued features into a mining algorithm can significantly enhance promotion-forecasting performance on two measures: scope and accuracy. Results are discussed, and then we present the issue of nonoptimality of rules space generated by the rule-induction algorithm and propose a simple approach which leads to improved interpretability and manageability of produced knowledge. Consequences of rules space optimization in application to other marketing areas are discussed, as are some efficiency issues. We then present our conclusions.

PROMOTION-EVENT FORECASTING SYSTEM–PROMOCAST

The promotion-event forecasting system PromoCast was initially presented to the scientific community by Cooper et al. (1999). For a complete overview and implementation details, the interested reader should consult the original publication. We offer here a brief summary of the PromoCast system, just sufficient to support the idea of our research.

The motivation for PromoCast was to provide a tactical promotion-planning solution for store managers or buyers/category managers. The objective was to supply retailers with knowledge about the amount of stock to order for a promotional event such that it would minimize inventory costs and out-of-stock conditions (two often-conflicting goals). Since the emergence of scanner technology, businesses have been accumulating data on past promotions for each separate SKU in each store within a retail chain. PromoCast was built to take advantage of this huge amount of historical data to predict demand for future promotion events. Figure 1 presents the general schematic of the PromoCast system.

Of particular interest are two key system components: the PromoCast Forecast module and the Corrective Action Generator. The forecast module uses a traditional market-response model to extract information from 67 variables that either come directly from a retailer's historical records (e.g., unit prices, percentage discounts, type of supporting ads) or are being inferred from those records (e.g., baseline sales) (for some examples of traditional market-response models, see Hanssens, 1998). Excluded from the statistical forecaster are nominal variables such as an item's manufacturer and merchandise category. Market-response models are not sufficiently robust to respond to the addition of 1,000 dummy variables for

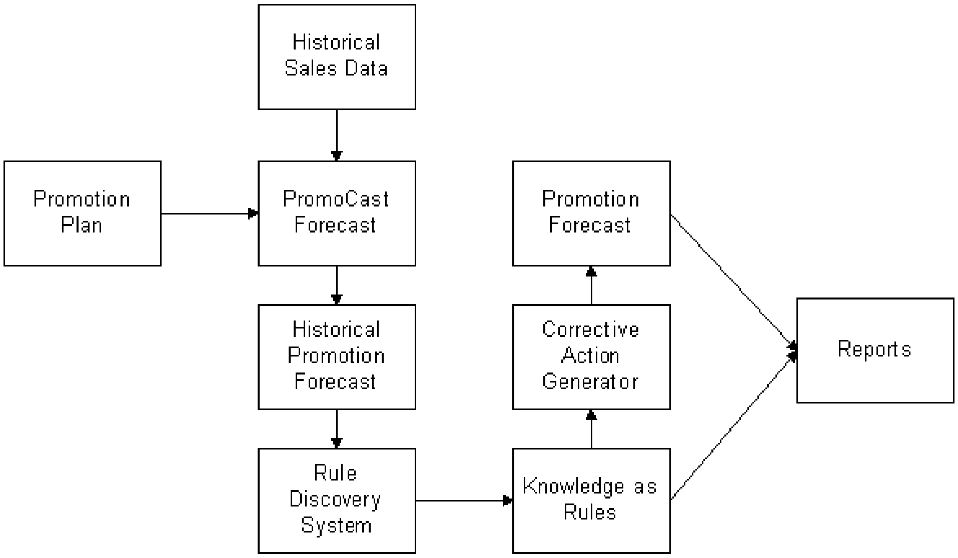

FIGURE 1
PromoCast Design

the manufacturers, 1,200 dummy variables for the merchandise divisions in a grocery store, 95 variables for the store-by-store effects, the possible interactions between these sets of indicators, or the possible interactions with the many other variables in the tactical forecasting model (Cooper & Giuffrida, 2000). As a result, a lot of information would be left in the residuals that would not be incorporated easily into a market-response model.

The Corrective Action Generator module addresses this issue. It utilizes data mining technology and a rule-induction algorithm to discover knowledge for forecast adjustments. Knowledge discovery is the process of finding meaningful patterns in data that explain past events so we can use the patterns to predict future events (Berry & Linoff, 2004). In PromoCast, a rule-induction algorithm is used to discover when the information in excluded variables indicates that the forecast should be modified. Once a set of discovered rules is built, these rules are used to adjust the forecast (see Figure 1). The Corrective Action Generator suggests that an offset (positive or negative) be added to the forecasted value to get higher overall accuracy.

The kernel of the Corrective Action Generator module is the Knowledge Discovery using SQL (KDS) data mining algorithm[1] (explained in Technical Appendix,[2] Part A). The KDS algorithm is used in the PromoCast system to discover patterns in excluded variables that explain specific outcomes in residuals. The Corrective Action Generator finds rules such as the following:

```
IF
DCS = "Gelatin" AND
TPR = "Very High" AND
Mfr = "General Foods" AND
Store Node = "Culver City, #231"
THEN
UNDER_12_ = 0,
UNDER_4_11 = 8,
UNDER_3 = 21,
UNDER_2 = 49,
UNDER_1 = 83,
Ok = 15,
 OVER_1 = 7,
 OVER_2 = 1,
 OVER_3 = 0,
 OVER_4_11 = 0,
 OVER_12_ = 0
```

Where the independent variables in the "if" conditions have the following meaning:

- "DCS" stands for the triple "Department-Commodity-Subcommodity," identifying a particular class of merchandise being promoted (e.g., yogurts, gelatins, or prepared dinners).
- "TPR" identifies the level of the Temporary Price Reduction. Promotions usually involve some item-price reduction. Values for this variable have been generalized to a set of five possible discrete values: None, Low, Medium, High, and Very High.
- "Mfr" identifies the manufacturer of the given product.
- "Store Node" allows for store-specific effects.

Errors in the forecast are expressed in a number of cases (i.e., the minimum order quantity for each particular SKU, usually 12 units in a case). For example, an error of −3 means that the statistical forecaster underestimated the sales for that specific promotion by three cases ("UNDER_3" class) while a value of 5 indicates that it overestimated five cases ("OVER_4_11" class).

For example, the previous rule states a clear tendency to underforecast products in the subcommodity "gelatin" for the manufacturer "General Foods" in the Store "231" when a "large" price discount is offered.

The KDS/Noah algorithm is one of the most efficient algorithms developed for large-scale commercial marketing applications to date. Algorithm application to forecasting and direct marketing has been discussed in management and computer science literature (Cooper & Giuffrida, 2000; Giuffrida, Chu, & Hanssens, 2000). The authors and their affiliates conducted extensive benchmark testing of the KDS/Noah algorithm in application to promotion forecasting and demonstrated its superiority to multiple commercially available data mining solutions including SAS EnterpriseMiner, SOMine, CN2, Ripper, Apriory, CBA, and others (Krycha, 1999). The superiority of

[1] "Noah" is one of the variations of the KDS algorithm available in the public domain.

[2] Technical appendices are available from the authors upon request.

the system over other popular algorithms justifies our choice of PromoCast as a benchmark platform.

In conclusion of this section, note that while the Corrective Action Generator strongly contributes to forecast accuracy, the module has certain limitations inherent to the class of employed data mining algorithms. In the following section, we discuss these limitations in depth and show that forecast performance is significantly improved when these limitations are addressed.

SEARCHING FOR "RARE" PATTERNS

As discussed in the previous section, the KDS algorithm discovers patterns in excluded variables to explain specific outcomes in residuals. Specifically, in the example presented earlier, the discovered pattern associated with the outcome of "underforecast" is {DCS = "Gelatin," TPR = "Very High," Mfr = "General Foods," Store Node = "Culver City, #231"}. The discovery of frequent patterns is one of the essential tasks in data mining. To ensure generation of the correct and complete set of frequent patterns, popular mining algorithms rely on a measure of so-called minimum support (Han, Wang, Lu, & Tzvetkov, 2002). For the reader who is not familiar with the idea of minimum support, we may suggest an analogy in statistics—statistical significance. From the statistical perspective, an inference is reliable if based on a large enough number of observations. Thus, a mining algorithm needs to know in how many records a given pattern must appear to be considered an as-a-rule candidate.

The minimum support value depends on the current user's goal. The higher the value, the fewer rules are created and the fewer records are classified. The problem of defining minimum support is quite subtle: A too-small threshold may lead to the generation of a huge number of rules whereas a too-large one often may generate no answers (Han et al., 2002). The major consequences of the former are poor scalability and outcome (rules) interpretability. The latter could be paralleled to the famous "broken leg" cue problem described by Meehl (1954), in which highly diagnostic cues that occur infrequently in the data sample are being left out of a statistical model. Next, we demonstrate how arbitrary selection of minimum support may lead to lost knowledge.

We start with the hypothetical example of inventory planning for an upcoming promotion event in the ice cream product category. Let us consider a supermarket which carries 40 different flavors of Ben & Jerry's ice cream. The store database contains historical records on the past promotions and inventory status for all 40 flavors. It happens that most flavors except "cherry" and "coffee" are underforecasted by the statistical model and would result in out of stock before the promotion ends. Assume that the store database contains records for 200 promotion events where none of any particular flavor of Ben & Jerry's ice cream appears more than 30 times. Finally, let's assume that the mining application uses a minimum support level of 50 occurrences. Then the resulting rule for Ben & Jerry's ice cream may look like:

```
IF
DCS = "ICE CREAM" AND
MFR = "BEN & JERRY'S"
THEN
UNDERFORCAST = "10 CASES"
```

Note that flavor attribute does not enter the rule because of insufficient support. The recommendation to adjust the forecast by 10 cases is inferred from the historical records where pattern {DCS = "ICE CREAM," MFR = "BEN & JERRY'S"} appears in association with an underforecast outcome. The same rule can be rewritten as:

```
IF
DCS = "ICE CREAM" AND
MFR = "BEN & JERRY'S" AND
FLAVOR = "ANY"
THEN
UNDERFORCAST = "10 CASES"
```

Note, however, that the following set of rules would produce more accurate forecast for Ben & Jerry's ice cream:

```
IF
DCS = "ICE CREAM" AND
MFR = "BEN & JERRY'S"
FLAVOR = ALL BUT "CHERRY" AND "COFFEE"
THEN
UNDERFORCAST = "10 CASES"

AND
```

IF
DCS = "ICE CREAM" AND
MFR = "BEN & JERRY'S" AND
FLAVOR = "CHERRY" OR FLAVOR = "COFFEE"
THEN
OVERFORCAST = "3 CASES"

This set of rules would be accepted by the original mining algorithm only if each pattern—{DCS = "ICE CREAM," MFR = "BEN & JERRY'S," FLAVOR = "CHERRY"} and {DCS = "ICE CREAM," MFR = "BEN & JERRY'S," FLAVOR = "COFFEE"}—appears in the historical dataset more than 50 times.

This suggests that the intuitive solution for the minimum-support problem is to extend rule syntax with an "OR"-type construct; however, there are certain considerations to keep in mind. A historical promotion-event database for the average supermarket chain may be exceedingly large. Mining for all possible "OR" combinations in data could be a tremendous task even with very powerful computational resources. Thus, algorithm scalability is of high importance. To provide such scalability and also take advantage of highly informative patterns[3] with low support, we propose the following approach. Instead of a brute force search for "OR" patterns in data, the algorithm first generates candidate rules and then attempts grouping among the ones which otherwise would be dropped by the Corrective Action Generator as not meeting the minimum support requirement. In application to our example, instead of considering all possible combinations of 40 different flavors of ice cream (e.g., {"CHERRY" OR "COFFEE"}, {"CHERRY" OR "COFFEE" OR "ORANGE"}, {"STRAWBERRY" OR "VANILLA" OR "ORANGE"}, etc.), we first produce a set of candidate rules, which satisfy a lowered (e.g., to 10 occurrences) minimum-support requirement, and then attempt to combine those patterns contingent on them to be associated with the same forecast outcome (e.g., set of low-support patterns associated with overforecast). Note that in our implementation, we combine rules that share all but one pattern element.[4] As a result, in the ice cream example, the following three forms of grouping are allowed:

- SUBCOMMODITY, MANUFACTURER, {SET OF FLAVORS}
- {SET OF SUBCOMMODITIES}, MANUFACTURER, FLAVOR
- SUBCOMMODITY, {SET OF MANUFACTURERS}, FLAVOR

For business practitioners (e.g., store management), rules produced in this fashion[5] should be intuitively appealing. Indeed, it is consumers' taste preferences which drive the demand for specific flavors. Some flavors of ice cream are more popular than others, and in general, the same price discount for different flavors may produce different sales responses. The same analogy applies to some other product attributes such as packaging size.

In application to the PromoCast Corrective Action Generator, grouping helps in capturing "rare" geographical area effects (e.g., a particular product may require the same forecast adjustment for a certain subsets of stores) or subcommodity effects (e.g., a particular product category requires the same forecast adjustment for a certain group of manufacturers). Implementation details of the algorithm are provided in the technical appendix.

PERFORMANCE TESTING

To test the performance of the proposed approach, we need to establish some benchmarks. We propose that the evaluation procedures consist of the following four steps: First, use the PromoCast Forecast module to produce a forecast based on the market-response model only (not adjusted for categorical variables). Second, use the Corrective Action Generator to "mine" through residual errors and provide corrections for the forecast. Third, compare recommended corrections against the historical data to evaluate the accuracy of the recommendations. Finally, repeat the described procedure using the proposed mining algorithm.

Dataset Description

Historical data obtained from one of the largest U.S. retail chains were used. For the purposes of analysis,

[3] In the data mining literature, these are called high confidence rules.

[4] This restriction was necessitated by the considerations of algorithm scalability.

[5] We use the terms "OR," "grouped," and "composite" interchangeably to denote these type of rules.

we limited our set to 8,195 records, which were randomly selected from the output of the PromoCast Forecast module. A total of 4,195 records were retained for calibration purposes (training set) and 4,000 records were used in the holdout sample. When compared against the actual promotion outcome, 3,184 records in the holdout sample required corrective actions. Of these, 1,217 (30%) were underforecasted by the PromoCast Forecast module, and 1,967 (49%) were overforecasted.

Performance Measures

Recall that the Corrective Action Generator makes adjustments to the demand predictions generated by the Forecast module. Given that our approach applies to the mining module (and, accordingly, does not interfere with the statistical methods of the Forecast module), it is sufficient to compare the performance results of the proposed new version of the Corrective Action Generator with the results produced by the original module. Therefore, the following discussion pertains to the accuracy of corrective actions, but not to the forecast as a whole.

The Corrective Action Generator first identifies records for which matching rules can be found, and then applies adjustment according to these rules. We count the number of records where suggested adjustments are appropriate and penalize for incorrect adjustments to evaluate algorithm performance. Accordingly, the percentage improvement in the forecast should not be interpreted as absolute but rather as improvements in the number of correctly adjusted records. Therefore, for our holdout sample, a reported improvement of 100% would correspond to correct identification and correction of 3,184 records which need adjustment.

RESULTS

The original Corrective Action Generator attempted to correct 917 of 4,000 records found in the holdout sample. Of these 917 records, 647 were adjusted correctly while 270 records received a wrong adjustment. The resulting success rate was 70.48%. Overall improvement in terms of records was 377 of 3,184 (11.84%). The Corrective Action Generator created a total of 71 "simple" rules using a minimum support of 50 records.

The extended version of the KDS algorithm produced additional 85 "grouping" rules, for a total of 156 rules. The Corrective Action Generator with grouping support attempted to correct 2,805 of 4,000 records in the holdout sample. Based on knowledge contained in composite rules, 2,581 adjustments were made; the rest—224 predictions—came from "simple" rules. Note that in the original version of the algorithm, all predictions are based on "simple" rules. The success rate was 78.29%. Overall improvement in terms of records was 1,592 of 3,184 (50.0%). When compared with results produced by the original Corrective Action Generator, we observe significant improvements in performance.

RULES SPACE OPTIMIZATION

In the previous section, we showed how mining and grouping of "rare" patterns significantly improves forecast performance. In this section, we briefly touch on some issues that are common to systems built on rule-induction algorithms. As described earlier, rule-induction algorithms produce knowledge by means of scanning through historical records in search for patterns. Knowledge is formalized in the form of rules in the following format: *pattern* → *prediction*. In a typical real-life application, it is common to have a large number of rules generated by popular data mining algorithms (including the one described in this study). Large numbers of rules tend to have high degrees of redundancy. By redundancy, we mean that multiple rules apply to the same record and produce identical predictions. To clarify this idea, we suggest the following analogy:

> Assume that a certain amount of Knowledge K could be represented by a collection of Rules S. If it is possible to remove some subset of Rules s from the original set S without sacrificing any amount of Knowledge K represented by the remaining set S^*, then we say that the original set S is not optimal or contains redundant rules.

Depending on the specific application, generated rules could be used either in automatic mode (without human intervention, as is the case with PromoCast) or presented for further analysis to the human agent (e.g., category manager). While in the former case, the number of rules constituting knowledge might be irrelevant (ignoring considerations of additional storage-space requirements and performance issues);

in the latter case, it may directly impact a system's overall usability. Many business settings dictate the necessity for a human expert to conduct the rules screening. Mindless or blind application of automatically generated rules even may have legal consequences—especially in systems where the results of rule application are transparent to the customer, such as dynamic pricing or product recommendations in online retailers. Given the necessity of screening, then, consider the example of a manager who would need to analyze only 20 rules generated by data mining application versus 300 rules. The problem of not optimizing a set of rules (e.g., having 280 redundant rules) is immediately apparent.

We suggest that by applying simple optimization techniques to the set of generated rules, the size of the rules space could be considerably reduced without significant loss of predictive ability. The resulting reduced rule set is much easier for managers to analyze and to perform screening. We demonstrate this idea using the rules set produced by the Corrective Action Generator in application to the promotion-forecast task discussed in the previous section.

Optimization Method

The goal of optimization algorithm is to identify and drop redundant rules because they do not add much value in terms of prediction accuracy. The situation here is very similar to the case of regression with highly correlated variables. There, too, we have a lot of redundancies among the variables because the variables are highly correlated. So, if we drop a particular variable out of the regression, it may not hurt accuracy at all because its effect already may be captured via another variable that it is correlated with. In regression, this is called the best K-subset selection problem, where we choose various values of K from 1 to the maximum number of predictors. Then, for each value of K, we choose the best k variables using a combinatorial heuristic called stepwise regression. We do an almost identical process here, except that instead of identifying the subset of variables, we are identifying the best subset of rules. We choose various values from 1 to the maximum number of rules generated by the data mining algorithm. Then for each value of K, we choose the best k rules using a combinatorial heuristic.

Performance (or predictive ability) of a rules set of size K depends on a quality of rules constituting the set (accuracy in adjustments) and its scope. By scope, we mean the percentage of records in the historical database to which a rules set may be applied. Indeed, having an extremely accurate rules set which is applicable only to very few records, however, is not useful, knowing that most predictions produced by the statistical forecaster (e.g., 75% in our dataset) require some adjustment. The applicability of a rules set is determined by the match between patterns in the rules and patterns found in the dataset. For example, consider the record of the following format:

{DCS = "ICE CREAM," MFR = "BEN & JERRY'S,"
FLAVOR = "CHERRY"}

Let us assume that the data mining algorithm has generated rules shown in Table 1.

TABLE 1 Rules Example

RULE ID	DCS	MFR	FLAVOR	PREDICTION
1	Ice cream	BEN & JERRY'S	Cherry	Under 10 Cases
2	Ice cream	ANY	Cherry	Under 5 Cases
3	Ice cream	BEN & JERRY'S	Any	Under 5 Cases
4	Any	BEN & JERRY'S	Any	Over 3 Cases
5	Any	Haagen-Dazs	Cherry	Under 5 Cases
6	Any	Haagen-Dazs	Cherry	OK
7	Yogurt	Any	Cherry	Over 5 Cases
8	Any	Any	Vanilla	Over 10 Cases

It is easy to see that Rules 1 through 4 can be applied to the record of interest while Rules 5 through 8 are not applicable, as they do not match a pattern in the focal record. Note that Rule 2, for example, can be applied to any manufacturer who produces cherry ice cream, and Rule 3 is applicable to all flavors of BEN & JERRY'S ice cream. The measure of the scope tells us how many records in the calibration dataset to which the particular rule can be applied. Consequently, the scope of a rules set is determined as the combined scope of each individual rule in the set.

To find the best subset of rules, we follow the traditions of statistical modeling by minimizing a combination of error rate (same as maximizing predictive ability) and model complexity (K). The following objective function is proposed:

$$F(K;\lambda) = \begin{bmatrix} \textit{Error rate of} \\ \textit{the best Rule Set} \\ \textit{with K rules} \end{bmatrix} + \lambda \times K$$

where λ is a smoothing parameter that is chosen by managerial judgment. To minimize $F(K; \lambda)$, we use simple combinatorial greedy algorithm, which for each given K and λ searches for the best rules set of size K. Then among all K-optimal rules sets, we choose the best K. The essence of the proposed combinatorial greedy algorithm can be summarized in the following steps:

1. From the set of all generated rules $\{Rule_i\}_{All}$, randomly select a subset of K rules. Call it a Candidate Set—$\{Rule_i\}_c$.
2. Among rules in the Candidate Set, search for the rule $\{Rule_d\}$ which, if dropped from $\{Rule_i\}_c$, causes the smallest deterioration in the objective function. Remove $\{Rule_d\}$ from the Candidate Set.
3. In $\{Rule_i\}_{All}$, search for the rule $\{Rule_c\}$ which, if added to the Candidate Set, produces the largest improvement in the objective function.
4. Repeat Steps 2 and 3 until $Rule_c$ is the same as $Rule_d$.[6]

Next, we use this algorithm to optimize the rules set generated by the Corrective Action Generator.

Optimization Results

As we presented in the section on performance testing, the extended version of the KDS algorithm generated 156 rules (71 simple, and 85 with grouping). This set of rules, when applied to the holdout sample, produced 2,805 predictions from which 2,197 were correct. Our hypothesis is that we should be able to achieve comparable predictive performance with a significantly lower number of rules. To test this hypothesis, we performed multiple runs of the proposed optimization algorithm with different values for smoothing parameter λ (0, 1, 5, and 10). Results for objective function are shown on Figure 2.

For each value of λ, we have found the corresponding optimal value of K (the optimal size of rules set). As such, the optimal K for $\lambda = 1$ is 32 rules, for $\lambda = 5$ is 21 rules, and for $\lambda = 10$ is 19 rules. All runs were performed on a calibration sample. Next, we applied resulting optimal rules sets to the holdout sample. Results are reported in Table 2.

For example, for $\lambda = 5$ and the corresponding optimal rules set of Size 21, the number of correct predictions is 2,199. These results show that compared to nonoptimized rules, the optimal set did equally well in terms of accuracy (Actually, we even gained two records.) At the same time, we were able to "shrink" the size of the rules set by more than seven times (down from 156 to 21).

We summarize the optimization results in Figure 3 by plotting the number of correct predictions as a function

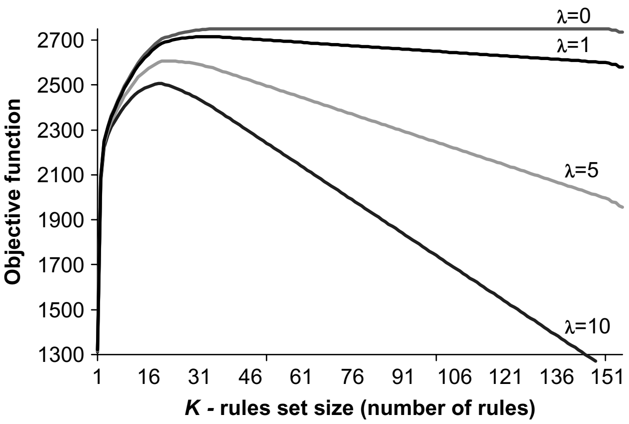

FIGURE 2

Rules Space Optimization

[6] The proposed procedure also should control for possible cyclic behavior.

TABLE 2 Hold-Out Sample Performance With Different Penalty Weights

λ	OPTIMAL RULES SET SIZE	ACCURATE PREDICTIONS
0	35	2,198
1	33	2,198
5	21	2,199
10	20	2,192
15	17	2,192
20	14	2,179
35	5	2,128
50	4	2,128

of the rules set size. The graphs in Figure 3 and Table 2 suggest that with only about 20 rules in the rules set, we are able to achieve prediction results very close to one produced by a full (i.e., 156 rules) rules set.

Rules space optimization also addresses another important problem. It is quite common for product-recommendation systems to use rules produced by human experts. While direct rule incorporation is normally not feasible for retail promotion-forecasting tasks, it is quite likely to be used in smaller scale systems such as automated cross-selling applications. The simplest example of a human-produced rule would be to recommend a tie to the customer who is placing an order for a shirt. When human knowledge in the form of rules enters a recommendation system directly (e.g., on the SKU level), there is always the possibility of human error. The most common problem is the expert's overconfidence (for some examples in behavioral literature, see Klayman, Soll, González-Vallejo, & Barlas, 1999). Both types of rules—the ones produced by the mining system and the ones produced by experts—must be associated with confidence level, which in the case of automated procedures is driven by data from the calibration sample. However, in the case of human generated rules, it is the expert's responsibility to assign the level of confidence. Overconfidence may lead to errors in predictions. Optimization conducted on the calibration set effectively filters out rules that are inefficient in the prediction task. That includes both simply redundant and harmless rules, such as the ones demonstrated in the example mentioned earlier, as well as rules mistakenly proposed by a human expert, which could have the harmful effect of distorting predictions.

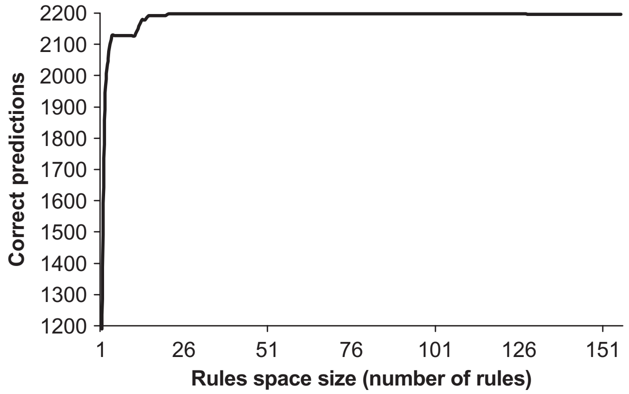

FIGURE 3

Number of Correct Predictions as a Function of the Rules Set Size

CONCLUSION

The PromoCast statistical forecaster coupled with a data mining module is an example of a hybrid system which has different technologies performing functions that they are best suited for. Traditional market-response models have received a considerable amount of attention in marketing and management literature in the past decades. The pace of expansion in this area is somewhat moderate compared to the rapidly advancing field of data mining. This article highlights the importance of revising computational techniques adopted into the management literature from the computer science field, as they might carry the greatest potential for improvements.

REFERENCES

Berry, M.J.A., & Linoff, G.S. (2004). Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management (2nd ed.). Hoboken, NJ: Wiley.

Cohen, W.W. (1996). Learning Trees and Rules With Set-Valued Features. Proceedings of the 8th annual conference on Innovative Applications of Artificial Intelligence, Portland, OR.

Cooper, L.G., Baron, P., Levy, W., Swisher, M., & Gogos, P. (1999). PromoCast™: A New Forecasting Method for Promotion Planning. Marketing Science, 18(3), 301–316.

Cooper, L.G., & Giuffrida, G. (2000). Turning Datamining Into a Management Science Tool. Management Science, 46(2), 249–264.

Divakar, S., Ratchford, B.T., & Shankar, V. (2005). CHAN4CAST: A multichannel, multiregion sales forecasting model and decision support system for consumer packaged goods. Marketing Science, 24(3), 334–350.

Giuffrida, G., Chu, W., & Hanssens, D.M. (2000). Mining Classification Rules From Datasets With Large Number of Many-Valued Attributes. In Lecture Notes in Computer Science (Vol. 1777, pp. 335–349). New York: Springer Berlin/Heidelberg.

Han, J., Wang, J., Lu, Y., & Tzvetkov, P. (2002). Mining Top-K Frequent Closed Patterns Without Minimum Support. Proceedings of the 2002 international conference on Data Mining, Maebashi, Japan.

Hanssens, D.M. (1998). Order Forecasts, Retail Sales and the Marketing Mix for Consumer Durables. Journal of Forecasting, June–July, (17), 327–346.

Klayman, J., Soll, J.B., González-Vallejo, & Barlas, S. (1999). Overconfidence: It Depends on How, What, and Whom You Ask. Organizational Behavior and Human Performance, 79(3), 216–247.

Krycha, K.A. (1999). Übung aus Verfahren der Marktforschung Endbericht [Case Study Growmart: The Effects of Promotional Activities on Sales]. Universität Wien, Institute für Betriebswirtschaftslehre.

Meehl, P.E. (1954). Clinical Versus Statistical Prediction. Minneapolis: University of Minnesota Press.

Srinivasan, S., Pauwels, K., Hanssens, D.M., & Dekimpe, M. (2004). Do Promotions Benefit Manufacturers, Retailers or Both? Management Science, 50(5), 617–629.