Chapter 11

Data Mining for Individual Consumer Models and Personalized Retail Promotions

Rayid Ghani, Chad Cumby, Andrew Fano, and Marko Krema

Contents

11.1 Introduction ................................................................. 202
11.2 Related Work .............................................................. 204
11.3 Data ........................................................................... 204
11.4 Individual Consumer Modeling ..................................... 205
  11.4.1 Shopping List Prediction ........................................... 205
    11.4.1.1 Machine Learning Methods ............................. 206
    11.4.1.2 Evaluation .................................................... 208
    11.4.1.3 List Prediction Experiments ............................ 209
    11.4.1.4 Fixing Noisy Labels ....................................... 210
  11.4.2 Identifying and Predicting Behaviors ....................... 211
    11.4.2.1 Basket-Size Variance ....................................... 211
    11.4.2.2 Pantry-Loading or Hoarding Attribute ............... 212
    11.4.2.3 Brand Loyalty ............................................... 212
    11.4.2.4 Individualized Product Substitutions ................ 212
    11.4.2.5 Behavioral Categories .................................... 213
    11.4.2.6 Price Sensitivity ............................................ 213
    11.4.2.7 Price Efficiency ............................................ 214

201
11.1 Introduction

Retailers have been collecting large quantities of point-of-sale data in many different industries. Loyalty card programs at many grocery chains have resulted in the capture of millions of transactions and purchases directly associated with the customers making them. Despite this wealth of data, retailers have not been effective at providing individualized interactions and promotions to consumers. This is not to say that there has been no work attempted on retail transaction data. Research in mining association rules [2] has led to methods to optimize product assortments within a store by mining frequent itemsets from basket data [5]. Customer segmentation has been used with basket analysis in the direct marketing industry for many years to determine which customers to send mailers to. Additionally, a line of research based on marketing techniques developed by Ehrenberg [8] seeks to use a purchase incidence model with anonymous data in a collaborative filtering setting [9].

Traditionally, most of the data mining work using retail transaction data has focused on approaches that use clustering or segmentation strategies. This is usually done to overcome the data sparseness problem and results in systems that are able to overcome the variance in the shopping behaviors of individual customers, while losing precision on any one customer. The author believes that with access to the massive amounts of data being captured, and the relative high shopping frequency of a grocery store customer, one can develop individual consumer models that are based on only a single customer’s historical data. This does not imply that previous approaches using clustering or association rules are inferior, but the goal in this chapter is to explore a new area that targets building individual models. Of course, in many situations, detailed historical data may not be available for every customer, in which case some form of generalization would be necessary and clustering approaches would be valuable.

A major reason that this area has not been more prominent in retail data mining research is that in the past there has been no individual channel to the customer for brick and mortar retailers. Direct mail is coarse grained and not very effective as it requires the attention of customers at times when they are not shopping and...
may not be actively thinking about what they need. Coupon-based initiatives given at checkout time are seen as irrelevant because they can only be delivered after the point of sale and often discarded by customers right away. However, with the advent of PDAs (personal digital assistants), in-store kiosks, and shopping cart-mounted displays such as the model shown in Figure 11.1, retailers are in a position now to deliver personalized information to each customer as they navigate through the store.

Given the large amounts of data being captured by retailers and the emergence of personal devices that customers will have access to while shopping in retail stores, the challenge is to create applications and techniques that can learn patterns of behaviors for individual customers and then enable interactions that are highly personalized. This chapter describes work on predicting shopping lists for each customer as they enter a store. This shopping list predictor serves as an anchor on which to base further interactions. If one can predict what the customer is going to buy today, the next step is to predict customer behavior with respect to specific products and then use these predictions to “say” something about each product. We use customer purchase data from a retailer to create individual consumer models to detect and predict the behaviors of customers with respect to their shopping. Using list prediction as a base, these consumer models enable the retailer to provide customers with individual and personalized interactions as they navigate through the retail store. The Shopping Assistant system is one aspect of the system that uses the

Figure 11.1 Cart-mounted display device.
predicted shopping list as a starting point for interacting with the customer. Instead of promoting random products, one uses the items on the predicted list to deliver personalized promotions. Each promotion and interaction with the customer is based on some attribute of the customer model that is learned for the customer. The major contributions of this work are as follows:

1. Shopping list prediction
2. Combining knowledge-based techniques with statistical and learning algorithms to build individual consumer models that capture different aspects of shopping behavior
3. Using these models to offer individual promotions to customers
4. Intelligent promotion planning: industrializing the use of the models by creating a set of tools that enable retailers to offer personalized promotions using visualization, interaction, simulation, and optimization techniques

The case study presented in this chapter deals with the retail industry and enabling personalized interactions within a store. In general, we believe that utilizing transaction data, learning individual behavioral models of customers (or entities in the data) using data mining techniques, and then optimizing higher-level objective functions based on these individual models is applicable in a variety of areas.

11.2 Related Work

There has been previous work on creating personal assistants for shopping. For example, the IBM Easi-Order system [4] and a system developed at Georgia Tech [12] use PDAs to display personalized grocery information to each shopper before and during their shopping trips. In the first system, a list is developed on the PDA, then sent to the store to be compiled and picked up. In the second, the PDA was used as an aid during the shopping trip to show locations and information on items in a list. The 1:1Pro system described in [1] was designed to produce individual profiles of customer behavior in the form of sets of association rules for each customer, which could then be restricted by a human expert. Theoretically, these profiles could then be used to develop personalized promotions and predict certain purchases. However, there have been no significant efforts in the data mining community to create behavioral models for individual customers using real-world retail store data and then use those models to automatically create personalized promotions based on the higher-level business goals of the retailer and manufacturer.

11.3 Data

In our study, we use transactional data provided to us by a grocery store chain. The data spans a period of two years and contains purchase data for all customers in ten of their stores. This results in a dataset consisting of about 700,000 customers
and 12.5 million transactions. Because the objective was to construct very detailed models, we decided to sample the dataset to get a smaller set of representative customers.

From the overall set, 22,000 customers shopped between 20 and 300 times, which was judged to be the legitimate population for whom to predict lists. We use this entire dataset and create individual consumer models for each customer. For the shopping list prediction, we sampled this population to produce a dataset of 2200 customers with 146,000 associated transactions. Because the number of transactions for each customer follows a power law, uniform random sampling to select 10 percent of the customers would result in a sample skewed toward customers with a small number of transactions. To get a representative sample, we first split the population into deciles along three attributes: (1) total amount spent, (2) total number of transactions, and (3) \(\frac{\text{transactions}}{\text{amount spent}}\). For each set of deciles, 10 percent of the data was selected with uniform probability from each decile. The 10 percent samples obtained for each attribute were found to be statistically similar to the other two (details omitted), and the final sample used was taken from total amount spent.

11.4 Individual Consumer Modeling

Our high-level goal is to create individual models for each consumer using only that consumer’s own historical transaction data. These models are used to predict customer behavior and provide customers with individual and personalized interactions. The interactions we enable consist of a shopping list that is first presented to the customer as he enters the store and identifies himself. The second aspect of the interaction is providing promotions or offers for products on the shopping list as the customer navigates the store. This section first describes work on the creation of the shopping list predictor (classifier) and then describes the attributes that constitute the consumer model and are inferred for each customer. More details on predicting shopping lists can be found in [7].

11.4.1 Shopping List Prediction

We formally frame the shopping list prediction as a classification problem, describe the algorithms and methodology behind the system, and present results on the grocery dataset described in Section 11.3. Immediate advantages of the shopping list prediction task include a useful reminder for consumers of items they might otherwise have forgotten, which directly result in reclaimed revenues for the retailer.

Our results show that one can predict a shopper’s shopping list with high levels of accuracy, precision, and recall. For retailers, the result is not only a practical

* We have conducted more experiments on a larger dataset and found similar results.
system that increases revenues by up to 11 percent, but also enhances customer experience and loyalty by giving retailers the tools to individually interact with customers and anticipate their needs, giving personalized promotions of the types we will introduce.

We define a set $C$ of customers, a set $T$ of transactions made by those customers, and a fixed set $P$ of product categories bought by these customers equivalent to those normally used on shopping lists. Within $T$ and $P$ we define for each $c \in C$ sets $T_c \subseteq T$ and $P_c \subseteq P$, consisting of the transactions of each customer $c$ and the product categories bought by customer $c$, respectively. For each transaction $t \in T_c$, our task then becomes to output a vector $y \in \{0, 1\}^{|P_c|}$ where $y_i = 1$ if for a given order of all categories in $P_c$, customer $c$ bought $p_i \in P_c$ in transaction $t$, and where $y_i = 0$ if customer $c$ did not buy $P_i$. We can then formulate the overall problem as $|P_c|$ binary classification problems for each customer and derive a separate classifier for each.

### 11.4.1.1 Machine Learning Methods

To approach the shopping list prediction task, we employ classification methods from machine learning. As discussed previously, the problem of predicting the overall assortment of categories purchased $y$ can be broken down into $|P_c|$ individual binary classifications. Each class can be thought of as a customer and product category pair. If the dataset consists of $|C|$ customers and an average of $q$ categories bought by each customer, one can construct $|C| \times q$ classes (and as many binary classifiers). For each of these classes $y_i$, a classifier is trained in the supervised learning paradigm to predict whether that category will be bought by that customer in that particular transaction. Here we present a series of examples of the form $(x, y)$, where $x$ is a vector in $\mathbb{R}^n$ for some $n$, encoding features of a transaction $t$, with $y_i \in \{0, 1\}$ representing the label for each example (i.e., whether or not the category corresponding to $y_i$ was bought).

We experimented with two kinds of machine learning methods to perform this task. First we trained decision trees (specifically using C4.5 [14]) to predict each class label. Next we tried several linear methods (Perceptron[15], Winnow[11], and Naive Bayes) to learn each class. These linear methods offer several advantages in a real-world setting, most notably the quick evaluation of generated hypotheses and their ability to be trained in an online fashion.

In each case, a feature extraction step preceded the learning phase. Information about each transaction $t$ is encoded as a vector in $\mathbb{R}^n$. For each transaction, we include properties of the current visit to the store, as well as information about the local history before that date in terms of data about the previous four transactions. The assumption here is that examples and their labels are not independent, and that one can model this dependence implicitly by including information about the previous visits. This tactic is similar to methods in Natural Language Processing.
(NLP) for tasks such as part-of-speech tagging, where tags of preceding words are used as features to predict the current tag [16].

The features included in example \((x^{t'}, y^{t'})\) about transaction \(t^j\) are:

1. Number of days at \(t^j\) since product category was \(p_i\) bought by that customer. We call this the replenishment interval at \(t^j\).
2. Frequency of interval at \(t^j\). For each category \(p_i\) we build a frequency histogram per customer for the interval at purchase binned into several ranges (e.g., three to five days, seven to nine days). This histogram is normalized by the total number of times items in that category were purchased.
3. The interval range into which the current purchase falls. These are the same ranges as cited above.
4. Day of the week of the current trip.
5. Time of day for the current trip broken down into six four-hour blocks.
6. Month of the year for the current trip.
7. Quarter of the year for the current trip.

We also include all of the above attributes for the previous four transactions, \(t^{j-1}, t^{j-2}, t^{j-3}, t^{j-4}\) in \((x^{t'}, y^{t'})\). Additionally, we include four additional features with respect to each transaction in the local history:

1. Whether category \(p_i\) was bought in this transaction
2. The total amount spent in this transaction
3. The total number of items bought in this transaction
4. The total discount received in this transaction

Note that the previous four features are only used for the local history of the current transaction and not for the current transaction itself. Because we are predicting the products bought for the current transaction when the customer enters the store, we obviously do not have access to these features.

In the case of decision tree learners, the above is the entire set of features used. For the set of linear classification methods utilized, it is often difficult to learn a linear separator function using a relatively low-dimensional feature space such as we have constructed. By combining basic features, effectively increasing the dimensionality of each example vector \(x\), we increase the chance of learning a linear function that separates all the positive and negative examples presented. Once again, this tactic is similar to those used to learn classifiers in NLP contexts where combinations of words such as bi-grams and tri-grams are used as features in addition to the basic words.

Therefore for the linear methods, several combinations of the basic features listed above were added to each example to improve learnability. For each numbered feature type above, we combine it with those of the same type in the customer’s previous four transactions (local history). For example, feature 4 (day of the week for the current transaction) is combined with feature 4 of the previous
transaction to produce a new feature. For the set-valued feature types above such as 4, boolean features are instantiated for each value (e.g., one feature per day). The combinations of these features used are simple boolean conjunctions. For the feature types corresponding to continuous valued attributes such as feature 2, we create a single real-valued feature. To create combinations of these features, we use a nonlinear transformation.

An additional set of methods explored in this experiment took the form of several hybrid methods. As discussed below, due to the large number of output classes we are trying to predict over all customers, we would like to evaluate the performance of our prediction strategies in aggregate with a single measure. However, as we treat each class as independent of each other for a given transaction (a simplifying, albeit untrue assumption), different classification methods can be used for different classes. This is our hybrid approach. In the experiments, we combined a baseline predictor with the various learned classifiers in the following fashion. If the category being predicted is within the top \( n \) (for some \( n \)) categories by purchase frequency for a given customer, then we predict positive, otherwise we predict according to the output of a given learned predictor.

### 11.4.1.2 Evaluation

The problem of predicting grocery shopping lists is an interesting learning problem because of the sheer number of classes that must be predicted. Abstracting from the product level (around 60,000 products) to the level of relatively specific categories useful for grocery lists reduces this number to some degree. However, for real-world datasets such as the one we explore in Section 11.4.1.3, this number could be from 50 to 100 classes per customer, with tens of thousands of regular customers per store, resulting in millions of classification categories and classifiers.

In general, the metrics used to judge the performance of our list predictors per class are the standard recall, precision, accuracy, and \( f \)-measure quantities. For a set of test examples, \( \text{recall} \) is defined as the number of true positive predictions over the number of positive examples; \( \text{precision} \) is the number of true positive predictions over the total number of positive predictions; \( \text{accuracy} \) is the number of correct predictions over the total number of examples; and \( f \)-measure is the harmonic mean of \( \text{recall} \) and \( \text{precision} \):

\[
\text{f-measure} = \frac{2 \cdot \frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}}{\frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}}.
\]

In obtaining an overall measure of performance by which we measure our success in predicting shopping lists for large groups of customers, there are many considerations to take into account. Typically in a learning scenario with a large number of output classes, the above quantities can be aggregated in several ways. Micro-averaged results are obtained by aggregating the test examples from all classes together and evaluating each metric over the entire set. The alternative is to macro-average the results, in which case we evaluate each metric over each class separately, and then average the results over all classes. The first strategy tends to produce
higher results than the second. When the number of classes is large and very unbalanced, the micro-averaged results are implicitly dominated by the classes with a large number of examples, while the macro-averaged results are dominated by the smaller classes. Macro-averaging is intuitively more attractive for our purposes as it gives an idea of how we are performing for the majority of customers rather than just those with a large number of transactions.

However, the transactional nature of the purchase datasource gives additional methods to aggregate our results. One option is to aggregate all examples associated with a single customer, obtain results for the above metrics for each set, and average them. This approach lets us know how we are performing for the average customer. Although these aggregate sets are still unbalanced, given that some customers shop more than others, the average results for this approach are generally between micro- and macro-averaging. We call this customer averaging. The final type of aggregating we can do is on the transaction level. Here we aggregate all the examples from each transaction, calculate each metric, and average the results over all transactions. We call this method transaction averaging. This averaging technique is perhaps most attractive of all in light of its ability to gauge how many categories per trip predicted are bought, and how many bought per trip are predicted. However, because it breaks up example sets within classes, it is difficult to compare this approach with the other aggregation techniques.

### 11.4.1.3 List Prediction Experiments

The transactional information present in the data includes the attributes described in the previous section and lists of products purchased in each transaction. Products are arranged in a hierarchy of categories. At a fairly specific level of this hierarchy, categories resemble grocery shopping list level items. Examples of these categories include cheddar cheese, dog food, sugar, laundry detergents, red wine, heavy cream, fat-free milk, tomatoes, etc. Some 551 categories are represented in the dataset forming the set \( P \) as defined previously. Customers within our sample bought 156 distinct categories on average (with standard deviation of 59). Of these categories, we restrict the set \( P \) for each customer to include only the categories bought on greater or equal to 10 percent of their trips. This brings the average size of \( P \) for a given \( c \) to 48 with a standard deviation of 27.59.

For each transaction for the customers in the sample, examples are constructed as detailed previously. The datasets for each class ranged from 4 to 240 examples. For each class in the resulting dataset, the example sets are split into a training set composed of the first 80 percent of examples in temporal order, and a test set composed of the last 20 percent. We also show results for a top-\( n \) baseline predictor as discussed in Section 11.4.1.1, with a cutoff of ten categories. For the decision tree classifier, C4.5 was used with 25 percent pruning and default parametrization. For the linear

*<customer, product category> pair.
classification methods, the SNoW learning system was used [6]. SNoW is a general classification system incorporating several linear classifiers in a unified framework. In the experiments shown, classifiers were trained with two runs over each training set. Results averaged by transaction are shown in Table 11.1.

11.4.1.4 Fixing Noisy Labels

A major motivation for predicting shopping lists stems from the goal of reclaiming forgotten purchases. Of course, the data collected does not include information on the instances in which categories are forgotten. *A priori*, we would not like to make any assumptions about the instances in which forgetting has occurred. This artifact produces a dataset that has noisy labels; examples where a customer “should” have bought a particular product but forgot to do so shows up as a negative example in our data. This not only creates noise in the training set (which can be overcome to some extent by robust learning algorithms), but also reduces the reported accuracy of the results. Examples where our system predicts an item is on the list are judged as incorrect if the customer did not buy that item (even if they forgot it). However, we would hope that the algorithms we examine should be somewhat robust to label noise as long as they are not overfitting the data. To estimate this robustness and determine the value of our suggestions via reclaiming forgotten purchases, we make some assumptions about the distribution of these instances and correct noisy label values in the test data. By training on the noisy data and then evaluating on the corrected test data, we hope to see the number of true positive predictions go up without a serious increase in false negatives.

The manner in which we estimate noisy labels in the test data to correct is described as follows. First, for each class \( p \in P_c \) for a given customer, we find the mean \( \mu \) and standard deviation \( \sigma \) of the replenishment interval \( i \). Next, we identify examples for which \( i \geq \mu + c \cdot \sigma \) for different constants \( c \). For each of these examples that have negative labels, we determine whether any example within a

\[ * \text{ Note that these moments exist without specifying a distribution over the replenishment interval.} \]
window of $k$ following transactions is positive. We estimate each of these examples to be an instance of forgetting, with noisy negative labels.

To evaluate the robustness of our predictors to this noise, we flip each noisy (forgotten purchase) negative label to be positive and reevaluate each classification method on the modified test data. This technique allowed us to evaluate whether our classification methods might be robust to this noise and correctly classify these instances of forgotten purchases. Table 11.2 shows how many examples, from the set of 47,916 examples that our heuristic tags as instances of forgetting, we “reclaim” as positive predictions. These reclaimed purchases result in increased sales for the retailer.

### 11.4.2 Identifying and Predicting Behaviors

Each consumer model consists of a variety of attributes that characterize that consumer. We discuss the attributes that make up our model and describe the methods used to calculate them. These attributes range from global attributes that apply to all aspects of a customer to product-specific attributes, calculated for every product the customer buys.

#### 11.4.2.1 Basket-Size Variance

This attribute measures the variability in the total spend during a visit for a customer. It is a global measure because it spans the customer and is not calculated separately for specific products. If a person spends the same amount of money in every shopping trip, we call this person a fixed basket shopper, and this attribute will have a low value. If the customer has a lot of variance in spending from visit to visit, this attribute gets a high value. We use two variations of this score:

1. Calculate the distance of the basket-size distribution for each customer from a uniform distribution using KL-Divergence.
2. For a range of values of $X$ and $Y$, we calculate the percentage of times, $X\%$ of the customer’s baskets (in terms of spending) were within $Y$ of each other.
11.4.2.2 Pantry-Loading or Hoarding Attribute

We calculate the score for Pantry-Loading or Hoarding behavior for each customer and product. This attribute is a measure of the individual consumer’s response to a price-drop for a specific product. Specifically, we assess whether the consumer is a “negative hoarder” — taking advantage of the sale to satisfy future needs, while keeping overall consumption stable or even declining. The “positive hoarder,” by contrast, takes advantage of the sale but his net consumption increases, as do net revenues for this customer on this product. This enables promotional strategies in which discounts are presented only to a specific type of hoarder.

We calculate the hoarding score by measuring the ratio of spending on a product before and after a sale, taking into account the replenishment rate for each customer.

One way to identify negative hoarders is by comparing the total revenue for the sale and the post-sale period with that of the pre-sale period. By identifying negative hoarders as those who spend less during the sum of the sale period and the post-period than they did during the pre-sale period, we calculate the amount of revenue lost for the store.

11.4.2.3 Brand Loyalty

Brand Loyalty scores are created in a variety of ways for every person, product category pair:

1. Brand Loyalty (Person, Product Category) = (Number of brands bought by person in this category) / (Total brands available in this category).
2. Similar to 1, except the score is changed so that brands that are popular get a lower score. Brands that are not very popular get a higher score.
3. We calculate the premium that is being paid by the customer for the brand that they are loyal to. If a person is loyal to the cheapest brand, reduce their loyalty score. If they are loyal to the most expensive brand, increase the score.

11.4.2.4 Individualized Product Substitutions

We calculate product substitution categories at multiple levels: store-level substitute groups and customer level. For example, Coke and Diet Coke may be substitutes at the store level but for a given customer, Coke and Diet Coke may NOT be substitutes.

For every customer and every pair of items in the store — say i and j — we calculate \( P(i) \), \( P(j) \), \( P(i, j) \), where \( P(X) \) is the probability that the customer will buy product \( X \) and \( P(X, Y) \) is the joint probability of buying both \( X \) and \( Y \). Let
$C(i, j) = 0$ if $i$ and $j$ are in different categories, 1 if they are in the same category. The substitution score for a customer and products $i$ and $j$ is high if $P(i, j) < P(i) \times P(j)$ and $C(i, j) = 1$.

### 11.4.2.5 Behavioral Categories

We create several behavioral categories that each consist of a set of products that are bought by customers showing a very specific behavior. Examples of our behavioral categories include “TV dinner buyers,” “vegetarians,” “organic buyers,” “smoking quitters,” and “foodies.” Due to limited space, we do not give an exhaustive list of categories that we construct.

Each of these categories consists of a set of products that fall into these categories. We calculate the Symmetric Ratio Spend Score (SRSS) for each customer $C$ and each category $T$. $\text{SRSS}(C, T) = (C1/C2)^I$, where $I = 1$ if $C1 > C2$ or $-1$ otherwise, and $C1 = (\text{Money spent on products from category } T \text{ by customer } C) / (\text{Total spend of customer } C)$, and $C2 = \text{Average}(C1)$ for all customers who buy at least one product from category $T$.

We also calculate a Symmetric Ratio Quantity Score where, instead of using spending on a category, we use the number of products bought in that category.

The intuition behind using these scores is to capture how much more a particular person spends on a particular category compared to other customers who buy something from that category. A simple ratio score would be unbounded, ranging from 0 to infinity, and not be symmetric. The space from 0 to 1 is taken up by customers who spend less than the average, whereas 1 to infinity is taken by those who spend more than the average. By using the Symmetric Ratio Score, we use the sign to denote more or less than the average, and the actual number denotes the magnitude.

### 11.4.2.6 Price Sensitivity

We measure how sensitive shoppers are to prices for every product. When the data is sparse, we aggregate to the product category level. In addition, we enable comparisons across shoppers by calculating the percentile price they typically pay for a particular product. Knowledge of price sensitivity helps retailers target promotions to those who need the additional inducement to trigger a purchase.

The price sensitivities are calculated for each customer with respect to each product and using shrinkage-like techniques to smooth these estimates. Given customer $C$, product $P$, calculate pairs $(R_i, P(R_i))$ where $R$ is the set of all unique prices for product $P$ during all of customer $C$ visits, and $P(R_i) = \text{Number of times customer } C \text{ visited the store and bought product } P \text{ at price } R_i / \text{Number of times customer } C \text{ visited the store and price of product } P \text{ was } R_i$. 
This gives us a price sensitivity distribution over all price points of a product for a particular customer. We also get a single score for the person, product pair by taking pairs \( (R_i, P(R_i)) \), and calculating a least squares fit to get a linear equation relating \( R_i \) and \( P(R_i) \). The slope of that line is the price sensitivity and \( R^2 \) is the confidence. These individual price sensitivities are aggregated and used to calculate price sensitivities at subcategory and category levels.

11.4.2.7 Price Efficiency

We calculate several measures of price efficiency for each customer. The idea behind these attributes is to capture how “savvy” the customer is in getting better prices than the rest of the population. The opportunistic index measures the average difference between the price the shopper paid and the most common price of a product (the mode of the daily price over two years). A person who shops on sales a lot will have a highly negative opportunistic index. However, his score would increase for getting a lower price than the mode price even if the product is not on sale. This will include permanent price drops, coupons, etc. The coupon index is more difficult to beat. It is the difference between the price the shopper paid and the price paid by most people that day (the mode of the day). This is a measure of how much of an individual price a shopper gets. It is not enough to shop on sales, the customer has to get a lower price than others are paying that day. This is useful for analyzing individual promotions such as coupons (unless they are very popular coupons that most people use during a day). The sales ratio is the ratio of number of products bought during sales to total number of products. It is useful for analyzing effects of advertised sales. We have a sale sensitivity index that is the percentage of change in quantity bought during the sale as opposed to normally. The index is computed for individual customers as well as individual products and product categories.

11.4.3 Consumer Interactions

A main part of our work following up on creating individualized consumer models has focused on how to interact with shoppers during their visits. One aspect worth discussing in this regard includes supporting varying degrees of privacy — from biometric check-in through anonymous shopping. Biometric or card-based identification allows us to make full use of individual models, while the anonymous shopping mode still allows us to create an incremental profile that enables a limited degree of personalization. This incremental profile could be created through clustering of anonymous shopper behaviors, or a nearest-neighbor technique with existing identified individual models.

In our system, as the shopper enters the store, we present a likely shopping list, followed by aisle-specific recommendations as shown in Figure 11.2. The aisle-specific recommendations presented for a given user are accompanied by specifically
targeted personalized promotions. Section 11.5 describes a systematic method to create these promotions based on the goals of the retailer and manufacturer.

11.5 Intelligent Promotion Planning

Creating profiles of entities (customers, products, stores, etc.) has been tackled by many data mining studies. What has been traditionally missing from this research is a framework to manage these models at a high level, in a goal-directed manner. Creating individual models of customers that can predict consumer behavior is useful, but it also has enormous implications on how retailers do business and interact with manufacturers and customers. In addition to using newspapers, in-store displays, and end-caps to highlight their products and run promotions, retailers can influence individuals in a vastly different way using individual consumer models. These capabilities create the need for systems that can take high-level business goals and apply them at an individual customer level. As more and more corporations start using data mining in their everyday business, just predicting an event (or purchase) is not enough — what is also needed is a method to operationalize this capability.

Planning promotions based on individual models using the simulation and optimization techniques that follow allows retailers to evaluate the costs and benefits
of such promotions with much greater efficiency. In addition, it allows manufacturers to pay for delivered business results in terms of new customers, enhanced loyalty, and incremental revenue and lift. Any retailer utilizing the system then has a distinct advantage in bidding for promotional dollars from a manufacturer. This section describes the process of planning promotions in an environment where individual consumer models are available.

As mentioned previously, creating consumer models is of value because it enables specific promotional strategies. These strategies include presenting a shopping list, including items likely to be forgotten, moving fixed basket size shoppers to higher-margin products, offering variable basket shoppers discounts on larger pack sizes, withholding discounts from “pantry loaders,” extending the brand for the brand loyal by offering related branded products, etc. In each case, when a promotion is delivered to a particular customer based on some aspect of their individual model, there are a number of ways that it could have been generated.

One possible way would be to manually examine each customer’s profile and predicted shopping list for the next trip, and create a set of promotions for each individually that serves the purposes of the store or product manufacturer. The obvious disadvantage of this approach is that it cannot scale well to thousands of customers across millions of transactions. An alternative is to manually create static rules defining if-then scenarios for giving particular promotions based on properties of an individual’s behavioral model; for example, if customer $a$ is predicted to buy category $c$ and has brand loyalty $x$, then give an $n\%$ discount on product $p$. While this type of rule creation is arguably more scalable than a totally manual technique, promotions could be targeted much more dynamically to serve changing goals.

We propose that the task of creating personalized promotions based on individual models is no simpler than the creation of the individual models themselves. It requires facilities to view and reason about the goals, parameters, and results of a promotion for a single product, while simulating these promotions for each customer using their personal model. To this end, we have created an Individualized Promotion Planning system. This system allows a user to modify the purposes of each promotion using a set of high-level goals, which are mapped to the practical parameters of a sale to produce general rules of the type mentioned above. By simulating the effects of a promotion on each customer targeted with respect to specific retailer or manufacturer goals, this system allows a completely new type of pricing model for trade promotions, bringing the pay-for-performance philosophy to a domain that has traditionally been administered on a very crude basis.

The system also offers the advantage of building and simulating sets of rules collectively and evaluating their interactions rather than manually in isolation. Below we describe the operations of the Individual Promotion Planning system from goal selection, to optimization, simulation, and pricing. We believe that reasoning and optimizing trends on an aggregate level extrapolated from data in multitudes of complex individualized profiles can be done in many data mining domains dealing with transactional data.
11.5.1 Goal Selection

Promotions are presented to consumers for the retailer or product manufacturers to achieve certain goals. The scope of the goals available becomes much wider when individualized consumer models are present. For example, a manufacturer may care about many things beyond basic revenue levels, such as the brand loyalty of their consumers, or their market share. The four main goal areas we believe our individual consumer models may affect are:

1. **Brand goals.** These subgoals represent results reflecting the brand loyalty and number of consumers who buy a particular brand: *brand switches, brand extensions, new trials,* and *average loyalty level.*
2. **Revenue goals.** These subgoals represent revenue results for the product being promoted: *short-term revenue, long-term revenue, revenue trends.*
3. **Lift goals.** These subgoals represent results for relative volume increases over current levels: *short-term lift, long-term lift,* and *brand lift.*
4. **Market share goals.** These subgoals represent results for market share changes for a brand: *product market share.*

Our system allows the user to weight each of these high-level goal groups relative to each other and also assign weightings to each individual goal within a group. These weightings can be used in an optimization procedure to map goal weights to the appropriate promotion parameters that maximize the results for each goal.

11.5.2 Promotion Parameters

In a system designed to deliver promotions based on individual models of consumers, the parameters of these promotions include many more factors than the discount and duration of the sale. In principle, using information from individual models could allow us to assign personalized prices [3] and to show promotions to different consumers for different periods of time. However, in the initial system we have developed, we restrict the discount and duration of a promotion to a set amount across all consumers, in an effort to provide more compatibility with existing systems. The parameters available in the current system for the promotional rules are as follows:

1. **Discount:** the price reduction percentage applied to a product from its baseline price.
2. **Duration:** the period of time in days that a promotion will be shown to the target consumers.
3. **Minloy,maxloy:** the minimum and maximum loyalty scores for the consumers in the target group (between 0 – 100).
4. **Minhoard,maxhoard:** the minimum and maximum hoarding scores for the consumers in the target group (between −100 and 100).
5. Minsensitivity, maxsensitivity: the minimum and maximum price sensitivity scores for the consumers (between 0 and 100).
6. Mintrial, maxtrial: the minimum and maximum new trial rate scores for the consumers (between 0 and 100) (Figure 11.3.)

For a given customer $c$, the promotional rules will then have the following form: *if c has scores within ranges 3,4,5,6, then reduce price by discount for duration days.*

### 11.5.3 Simulation

The first main advantage of our system is its ability to show the user simulations of promotional results directly related to the goals of Section 11.5.1 by creating promotional rules based on the parameters described above and applying these rules iteratively to each customer. We then apply heuristic measures to gauge the results related to each goal defined above. Many other sets of heuristics or learned models could be created to explain the results. For each heuristic $h_i$ below, we sum $h_i$ over all customers to produce the final simulated result. Also, $sens(x)$ refers to the price sensitivity (Section 11.4.2) of the given customer for the given product at price $x$.

![Figure 11.3  Screen shot from the Promotion Planning prototype.](image-url)
base refers to the base price of the given product, while discount refers to the discount variable introduced in Section 11.5.2.

11.5.3.1 Brand Heuristics

\[
\begin{align*}
    b_{\text{switch}} &= \begin{cases} 
    \text{result\_prob} & \text{if } \text{numvisits} \geq \text{loy}_{\text{obs}} \cdot \text{conv\_rate} \\
    0 & \text{else}
    \end{cases} \\
    b_{\text{extensions}} &= \begin{cases} 
    \text{result\_prob} & \text{if } \text{numvisits} \geq 0 \\
    (1 - \text{loy}_{\text{dis}}) \cdot \text{conv\_rate} & \text{else}
    \end{cases} \\
    b_{\text{neutrals}} &= \begin{cases} 
    \text{result\_prob} & \text{if } \text{numvisits} \geq 0 \\
    (\text{loy}_{\text{obs}} + \text{neutral}) \cdot \text{conv\_rate} & \text{else}
    \end{cases} \\
    b_{\text{loyalty\_level}} &= \text{sens(base – discount)} \\
    && \cdot \text{loyalty\_change} \cdot \text{num\_visits} \\
    \text{result\_prob} &= \text{sens(base – discount)}
\end{align*}
\]

In all of the above, we use the average replenishment rate per customer repl_rate (as detailed in [7] to calculate the quantity) numvisits as repl_corrected. The conversion rate conv_rate is a constant estimated as an average of the number of visits by the customer required to obtain a positive result for each heuristic. The constant loyalty_change refers to the average loyalty level change estimated per promotion computed per customer or product.

11.5.3.2 Revenue Heuristics

The revenue heuristics encode the relative increase or decrease in revenues in the short term (promotion duration) and long term (four replenishment rates after promotion):

\[
\begin{align*}
    b_{\text{short\_term}} &= (\text{base – discount}) \cdot \text{sens(discount)} \cdot \text{num\_visits} \\
    b_{\text{long\_term}} &= \begin{cases} 
    \text{boarding\_score} & \text{if } \text{sens(base – discount)} > .5 \\
    0 & \text{else}
    \end{cases}
\end{align*}
\]
The brand revenue heuristic is evaluated by summing either the short- or long-term revenue heuristics over all products in the brand.

11.5.3.3 Lift Heuristics

\[
b_{\text{short\_term}} = (\text{sens(base - discount) - sens(base)}) \cdot \text{num\_custs}
\]

\[
b_{\text{long\_term}} = (\text{sens(discount \_ price) - sens(base)}) \cdot \frac{\text{boarding\_score}}{\text{base}}
\]

The brand lift heuristic is evaluated by summing either the short- or long-term lift heuristics over all products in the brand.

11.5.3.4 Market Share Heuristics

\[
b_{\text{market\_share}} = b_{\text{switches}} + \frac{b_{\text{extensions}}}{\text{avg\_ext}}
\]

Here, \(\text{avg\_ext}\) is the average number of extensions over all brands in the category. \(b_{\text{market\_share}}\) is then the estimated increase/decrease in market share in terms of loyal customers over the next promotion period.

11.5.4 Optimization

In addition to simulation, our prototype offers the user a flexible optimization tool. In the optimization procedure, the user first enters relative weightings for each goal result defined in Section 11.5.1. The optimizer then attempts to set parameter values that optimize a closed-form approximation of each result heuristic described above in Section 11.5.3. Several types of constraints can be placed on each parameter variable within the framework.

More concretely, let \(x_1 \ldots x_n\) be the parameter variables described in Section 11.5.1. We define a hierarchical multi-objective optimization problem of the following form:

\[
\text{argmax}\ f(x_1 \ldots n) = \text{argmax}_{x_1 \ldots x_n} \begin{bmatrix} f_{\text{brand}}(x_1 \ldots x_n) \\ f_{\text{revenue}}(x_1 \ldots x_n) \\ f_{\text{lift}}(x_1 \ldots x_n) \\ f_{\text{market\_share}}(x_1 \ldots x_n) \end{bmatrix}
\]

\[\text{wrt } C = \{c_1 \ldots c_q\}\]
where the set $C$ of constraints on $x_1 \ldots x_n$ are given by the user. We reformulate $f(x_1 \ldots x_n)$ as a weighted sum:

$$f(x_1 \ldots x_n) = g_1 \cdot f_{brand}(x_1 \ldots x_n) + g_2 \cdot f_{revenue}(x_1 \ldots x_n)$$

$$+ g_3 \cdot f_{lift}(x_1 \ldots x_n) + g_4 \cdot f_{msbnde}(x_1 \ldots x_n)$$

Each term in this sum is itself expressed as a weighted sum, yielding a single objective function. The subobjectives are as follows:

$$f_{brand}(x_1 \ldots x_n) = b_1 \cdot switches(x_1 \ldots x_n) +$$

$$b_2 \cdot extensions(x_1 \ldots x_n) + b_3 \cdot neutrals(x_1 \ldots x_n)$$

$$b_4 \cdot loylevel(x_1 \ldots x_n)$$

$$f_{revenue}(x_1 \ldots x_n) = r_1 \cdot shortrev(x_1 \ldots x_n) +$$

$$r_2 \cdot longrev(x_1 \ldots x_n) + r_3 \cdot brandrev(x_1 \ldots x_n)$$

$$f_{lift}(x_1 \ldots x_n) = l_1 \cdot shortlift(x_1 \ldots x_n) +$$

$$l_2 \cdot longlift(x_1 \ldots x_n) + l_3 \cdot brandlift(x_1 \ldots x_n)$$

$$f_{msbnde}(x_1 \ldots x_n) = m_1 \cdot prodshare(x_1 \ldots x_n)$$

Each term in the subobjectives is a closed-form approximation of the result of its associated heuristic, which in general may be nonlinear. For example, the $switches$ term of $f_{brand}$ is defined as follows:

$$switches(x_1 \ldots x_n) =$$

$$\left( \frac{\text{duration}}{\text{avg repl}} - \frac{\text{conv rate} \cdot (\text{minloy+maxloy})}{2} \right) \cdot \text{custs(minloy,maxloy)}$$

$$\frac{\text{duration}^2}{(\text{duration})^2}$$

where $\text{avg repl}$ is the average replenishment rate of customers who buy the target category, and $\text{conv rate}$ is the number of promotion instances needed to switch a customer with 100 percent loyalty to a competing brand (estimated from training
transactions). We estimate the \textit{custs} quantity by assuming the number of customers is distributed normally with respect to brand loyalty, and using the normal cumulative distribution function:

$$
custs(minloy, maxloy) = \frac{1}{\sigma \sqrt{2\pi}} \int_{minloy/\sigma}^{maxloy/\sigma} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt
$$

The set of constraints \( C \) can contain equality/inequality constraints over any of the input variables \( x_1 \ldots x_n \) to express rules such as \textit{promotions on product p may never exceed 50 days in duration}.

To solve this nonlinear optimization task, we employed a sequential quadratic programming procedure. A line search is used to guide the SQP in a manner similar to [10, 13], with the solution implemented in Matlab.

### 11.6 Conclusion

This chapter described a real-world data mining project using customer purchase data from a retailer to create individual consumer models to detect and predict the shopping behaviors of customers. These consumer models enable the retailer to provide customers with individual and personalized interactions as they navigate through the retail store by making very fine, accurate predictions about a particular individual customer in a given shopping trip. The shopping list prediction system results in a direct increase of revenues by reminding customers of purchases they would otherwise forget. The shopping list also helps the system narrow down the list of products for which to give personalized promotions and discounts. We described a set of attributes that constitute the individual consumer model. These attributes are used to present personalized promotions to each customer based on their historical shopping data. We also described the Intelligent Promotion Planning system, which functions as a “mediator” between the high-level business goals of a retailer or product manufacturer and the individual promotions offered to customers. It not only offers an optimization capability, where optimal parameters of the promotion are chosen based on high-level goals set by the retailer, but also offers a simulation environment where many what-if scenarios can be considered and the results of potential promotions evaluated before implementation.

We believe that utilizing transaction data, learning individual behavioral models of customers using data mining techniques, and then optimizing higher-level objective functions based on these individual models is applicable in a variety of areas.

We soon expect to implement the Individual Consumer Modeling system in a brick-and-mortar store, allowing us to evaluate the accuracy of the goal heuristics we have developed. At this point they can become the basis for a bootstrapped model
that we can update continuously through active and online learning methods. Additionally, the setup of a larger-scale optimization problem representing each consumer’s promotional variables (discount, duration, loyalty, etc.) directly such as in [3] should provide even greater efficiency in terms of maximizing goal results and correct pricing.

References
