Statistical Learning for Decision Making: Interpretability, Uncertainty, and Inference

by

Benjamin Letham

B.S.E., Arizona State University (2005)
M.S.E., The Johns Hopkins University (2007)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2015

Author .....
Sloan School of Management
May 12, 2015

Certified by .....
Cynthia Rudin
Associate Professor of Statistics
Thesis Supervisor

Accepted by .....
Patrick Jaillet
Dugald C. Jackson Professor, Department of Electrical Engineering and Computer Science
Co-Director, Operations Research Center

Statistical Learning for Decision Making: Interpretability, Uncertainty, and Inference

by

Benjamin Letham

Submitted to the Sloan School of Management
on May 12, 2015, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

Data and predictive modeling are an increasingly important part of decision making. Here we present advances in several areas of statistical learning that are important for gaining insight from large amounts of data, and ultimately using predictive models to make better decisions.

The first part of the thesis develops methods and theory for constructing interpretable models from association rules. Interpretability is important for decision makers to understand why a prediction is made. First we show how linear mixtures of rules can be used to make sequential predictions. Then we develop Bayesian Rule Lists, a method for learning small, ordered lists of rules. We apply Bayesian Rule Lists to a large database of patient medical histories and produce a simple, interpretable model that solves an important problem in healthcare, with little sacrifice to accuracy. Finally, we prove a uniform generalization bound for decision lists.

In the second part of the thesis we focus on decision making from sales transaction data. We develop models and inference procedures for using transaction data to estimate quantities such as willingness-to-pay and lost sales due to stock unavailability. We develop a copula estimation procedure for making optimal bundle pricing decisions. We then develop a Bayesian hierarchical model for inferring demand and substitution behaviors from transaction data with stockouts. We show how posterior sampling can be used to directly incorporate model uncertainty into the decisions that will be made using the model.

In the third part of the thesis we propose a method for aggregating relevant information from across the Internet to facilitate informed decision making. Our contributions here include an important theoretical result for Bayesian Sets, a popular method for identifying data that are similar to seed examples. We provide a generalization bound that holds for any data distribution, and moreover is independent of the dimensionality of the feature space. This result justifies the use of Bayesian Sets on high-dimensional problems, and also explains its good performance in settings where its underlying independence assumption does not hold.

Thesis Supervisor: Cynthia Rudin
Title: Associate Professor of Statistics

Acknowledgments

Thanks first to Cynthia Rudin for five years of advising. I came to MIT because I wanted to work with her, and it was as great as I hoped it would be. Thanks for giving me freedom to work on whatever research problem I was interested in and for being very supportive.

Thanks to Roy Welsch and Itai Ashlagi for being on my thesis committee, and my general exam committee before that. I've enjoyed our discussions over the years.

Thanks to the many colleagues and collaborators at MIT and elsewhere who have supported my research and given me inspiration. Thanks to Ed Browne for asking some hard questions that ended up inspiring some fun research. Thanks to all of my coauthors the last 5 years, in particular David Madigan, who played an important role in my first research project at MIT, and others since.

Thanks to friends and family. I immensely enjoyed doing research with Lydia and Portia. Thanks to my parents. Thanks to all of the friends who made us feel like family. Thanks to Matt and SalleeAnn, who helped keep me focused on the impact I can have on humanity.

And thanks most of all to Sharla, who made this thesis even more fun than the last.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Contributions

Gaining insight and meaningful predictions from large amounts of data presents a number of unique challenges and opportunities. In this thesis we present work on several facets of predictive modeling that extend beyond the accuracy of the model. These facets are important considerations when the ultimate purpose of learning the model is to make better decisions, as it often is in practice.

We first study interpretability in predictive models. Interpretability is a crucial property for a predictive model meant for decision making. Despite the wide availability of powerful learning algorithms such as AdaBoost and SVM, there are many areas of practice in which domain experts still use simple, less accurate heuristics. A notable example is in medicine, where decisions on various treatment options are often made using extremely simple scoring models, typically built from a small set of known risk factors and calibrated to a small sample of data. Doctors, and experts in many other non-mathematical domains, are often not interested in black-box models that they do not understand; they need to know *why* a particular decision is made. The ability to decompose the decision into simple components that can be understood by the domain expert can be more important than the performance gain achieved by state-of-the-art machine learning algorithms over heuristics. In Chapters 2, 3, and 4, we provide new methods and theoretical results for constructing predictive models based on association rules, which combine the power of statistical learning with the interpretability of simple rules.

In Chapter 2 we develop an empirical risk minimization framework for learning optimally weighted linear combinations of association rules, tailored for sequential prediction tasks (Letham et al, 2013b). These association rule-based models prove to be a natural approach for solving sequential prediction tasks, since they can be applied to data examples with sequentially revealed features. The natural interpretability that comes with association rules can also be important for using the model as a recommender system, for which explanations can be important: We recommend item *a because* you purchased item *b*. The learning procedure is formulated as a convex minimization problem and can be applied to large datasets. We show that our ERM-learned weighted rule models can significantly outperform a standard approach to item recommendation (collaborative filtering) on three real-world problems: email recipient recommendation, grocery item recommendation, and patient symptom prediction. Thus we see that association rules can provide interpretability without sacrificing accuracy.

Chapter 3 presents a different way of combining association rules to form a classifier, again with the goal of producing an interpretable predictive model. Here we study *decision lists*, in which rules are ordered and then chained together by *if...elseif...* statements. We define a *Bayesian association rule*, which serves as the fundamental unit for a Bayesian hierarchical model for learning short lists of rules from data called Bayesian Rule Lists (Letham et al, 2015b). These models are particularly well-suited for applications in medicine because the entire model can be expressed in a few rules and a prediction can easily be made by hand, for instance by a doctor during a visit with a patient. We used Bayesian Rule Lists to solve an important problem in medicine: using a patient's medical history to predict their 1-year stroke risk given a diagnosis of atrial fibrillation. This prediction is usually made using an extremely simple scoring system (CHADS_2), that adds up points for a total of five features. The outcome of the prediction is used to make an important decision: Should the patient be put on blood thinners or not? This decision has real consequences for the life of the patient, and as such transparency and interpretability of the model are essential. Using a large database of patient medical histories, Bayesian Rule Lists

is able to produce a simple, interpretable collection of rules that has the predictive power that comes with statistical learning but the same simplicity as the current heuristics. The rule list has much better accuracy than $CHADS_2$, as well as other interpretable approaches like decision trees. This work shows that our interpretable models can achieve accuracy comparable to the state-of-the-art opaque models, on real-sized datasets and using a reasonable amount of time for training.

Chapter 4 provides a theoretical result that connects the results of Chapters 2 and 3. Here we use the VC dimension from statistical learning theory to prove a generalization bound for decision lists. The result is a uniform generalization bound, which shows that expected performance of any decision list will not be too different from the performance on the training data (Rudin et al, 2013). Interestingly, the same bound also holds for the weighted linear rule combination models that we learned in Chapter 2. This result suggests that these two approaches for combining association rules into a predictive model should have similar generalization behavior.

In the following two chapters, Chapters 5 and 6, we turn our attention to decision making from sales transaction data. Large sales transaction datasets have the potential to be useful in operations management decisions such as pricing and inventory planning, however realizing that potential requires addressing several issues specific to inference from transaction data. In Chapter 5 we develop a method for using historical transaction data to predict the profit obtained by offering a bundle discount for a collection of items (Letham et al, 2014). This prediction is important for deciding what items to offer as a bundle, and at what discount. Predicting bundle profits requires estimating the amount that customers are willing to pay for each of the items in the bundle, as well as the correlations across items. Retail transaction data do not directly reveal how much a customer was willing to pay - they tell only whether or not the customer purchased the item at the offered price. We develop a tractable and statistically consistent procedure for inferring willingness-to-pay that uses a copula to estimate the correlations between items. The method scales to large transaction databases, and, importantly, is naturally integrated with existing models for demand estimation.

In Chapter 6 we address another way in which sales transaction data are often censored: stockouts. Estimating demand is necessary to make decisions about pricing and stocking, however in the presence of stockouts the observed sales might not reflect the actual customer demand. The sales of the stocked out item might be lower than the actual demand, and the sales of substitutable products might be higher than their actual demands. In Chapter 6 we develop a Bayesian hierarchical model for simultaneously inferring the underlying demand and customer substitution behavior from sales transaction data (Letham et al, 2015a). In doing so we address another facet of statistical learning that is important for decision making: incorporating model uncertainty into the decision making process. Our Bayesian model includes a direct measurement of its uncertainty, obtained via posterior sampling. We show how this uncertainty can be directly incorporated into the estimation of derivative quantities, such as the lost sales due to stock unavailability, thus leading to better informed decisions.

Finally, in Chapter 7 we present a new approach for a certain type of information retrieval called web-based set expansion. When making a decision it may be important to have complete information on a given topic. It is easy to find expert knowledge on the Internet on almost any topic, but obtaining a complete overview of a given topic while avoiding extraneous information is not always easy: Information can be scattered across many sources and must be aggregated to be useful. We introduce a method for intelligently growing a list of relevant items, starting from a small seed of examples (Letham et al, 2013a). This is done by finding relevant lists of information across the Internet and aggregating them to produce a single, complete list. We show that in practice our approach significantly outperforms existing solutions, and we also provide important theoretical results. The core of our method is an algorithm called Bayesian Sets, which is used to determine if scraped items are related to the seed examples. We prove a generalization bound for Bayesian Sets, showing that with high probability the score of relevant items is close to the score of the seed examples. The result is remarkable inasmuch as it does not depend on the number of features used for the scoring. The bound provides a powerful justification for using Bayesian

Sets in high-dimensional settings.

This thesis includes contributions to several areas of statistical learning that are important for decision making. Our methods for producing interpretable predictive models from association rules, particularly the Bayesian Rule Lists of Chapter 3, have the potential for significant impact in domains that have been slow to adopt modern machine learning methods due to their opacity. We hope that Bayesian Rule Lists will replace decision tree algorithms like CART as the standard approach for learning simple classification models. Our work on inference from sales transaction data lays a foundation for leveraging this ubiquitous data source for making better revenue management decisions. Through Bayesian modeling and posterior sampling we encourage incorporating model uncertainty directly into the decision making process, leading to decisions that are less sensitive to possible overfitting. Finally, our advances in set expansion provide both practical tools and theoretical underpinnings for future work in information retrieval.

Chapter 2

Weighted Association Rules and Sequential Events

Given a collection of mined association rules, how should we use these rules to build a predictive model? In this chapter, we present optimization-based algorithms for learning a weighted linear combination of association rules for sequential event prediction. Here sequential event prediction refers to a wide class of problems in which a set of initially hidden events are sequentially revealed. The goal is to use the set of revealed events, but not necessarily their order, to predict the remaining (hidden) events in the sequence. We have access to a "sequence database" of past event sequences that we can use to design the predictions. Predictions for the next event are updated each time a new event is revealed. There are many examples of sequential prediction problems. Medical conditions occur over a timeline, and the conditions that the patient has experienced in the past can be used to predict conditions that will come (McCormick et al, 2012). Music recommender systems, *e.g.* Pandora, use a set of songs for which the user has revealed his or her preference to construct a suitable playlist. The playlist is modified as new preferences are revealed. Online grocery stores such as Fresh Direct (in NYC) use the customer's current shopping cart to recommend other items. The recommendations are updated as items are added to the basket. Motivated by this application, "sequential event prediction" was formalized by Rudin et al (2011, 2013), who created a theoretical foundation along with

some simple algorithms based on association rules. The algorithms we develop in this chapter are based on the principle of empirical risk minimization (ERM). We apply our algorithms to data from three applications: an online grocery store recommender system, email recipient recommendation, and medical condition prediction.

Recommender systems are a particularly interesting example of sequential event prediction because the predictions are expected to influence the sequence (Senecal and Nantel, 2004, for example), and it may be important to take this into account. For instance, there has recently been work showing that measurements of user behavior can be used to improve search engine rankings (Agichtein et al, 2006a,b). For an online grocery store recommender system, items are added to the basket one at a time. The customer may not have an explicit preference for the order in which items are added, rather he or she may add items in whichever order is most convenient. In particular, the customer may add items in the order provided by the recommender system, which means the predictions actually alter the sequence in which events appear. Our formulation allows for models of user behavior to be incorporated while we learn the recommender system.

The same formulation used for the online grocery store recommender system can be directly applied to email recipient recommendation. Given a partial list of recipients on an email, we wish to predict the remaining recipients. An email recipient recommendation algorithm can be a very useful tool; an algorithm for this purpose was recently implemented on a very large scale by Google and is integrated into the Gmail system used by millions of people (Roth et al, 2010).

Medical condition prediction is a new yet active area of research in data mining (Davis et al, 2010; McCormick et al, 2012). Accurate predictions of subsequent patient conditions will allow for better preventative medicine, increased quality of life, and reduced healthcare costs. Rather than a sequence of single items, the data comprise a sequence of sets of conditions. Our formulation can handle sequences of sets, and we apply it to a medical dataset consisting of individual patient histories.

The sequential event prediction problems we consider here are different from time-series prediction problems, that one might handle with a Markov chain. For instance,

the online grocery store recommender system problem has no intrinsic order in which groceries should be added to the basket, and in email recipient recommendation the order of the addresses is likely of little importance. Only the *set* of past items are useful for predicting the remaining sequence. Figure 2-1 gives an illustration of this point using the online grocery store recommender system. For instance, at time $t = 2$, apples and cherries are in the basket and are together used to predict what will be added next. The fact that apples were added before cherries is not necessarily useful. In the medical condition prediction problem, collections of conditions occur at different time steps, and we use all past collections of conditions to predict the next collection. Figure 2-2 shows a sequence of these collections of conditions as they occur over time. For instance, at time $t = 1$, we use the entire collection of conditions {Hypertension, Sore throat, Gastric Ulcer} to make a prediction about the next collection. At time $t = 2$, we use the two collections {Hypertension, Sore Throat, Gastric Ulcer} and {Hypertension, Influenza} to make a prediction about the following time step. The collections of conditions occur sequentially in a certain order, however each collection is itself an unordered set of conditions. For example, it might not be sensible at $t = 3$ to say that elevated cholesterol preceded Headache. On the surface, the online grocery store recommender system and the medical condition prediction problem seem quite different, but the methodology we develop for each problem derives from a general formulation which could be adapted to a wide range of other sequential event prediction problems.

We treat each step of sequential event prediction as a supervised ranking problem. Given a set of revealed events from the current sequence, our algorithms rank all other possible events according to their likelihood of being a subsequent event in the sequence. The accuracy of our prediction is determined by how far down the list we need to look in order to find the next item(s) to be added.

Section 2.1 gives a formal introduction to sequential event prediction and provides the notation that we will use throughout the paper. Section 2.2 presents our ERM-based method for sequential event prediction. In Section 2.3 we apply the ERM-based algorithms to email recipient recommendation, in which the sequence is one of email

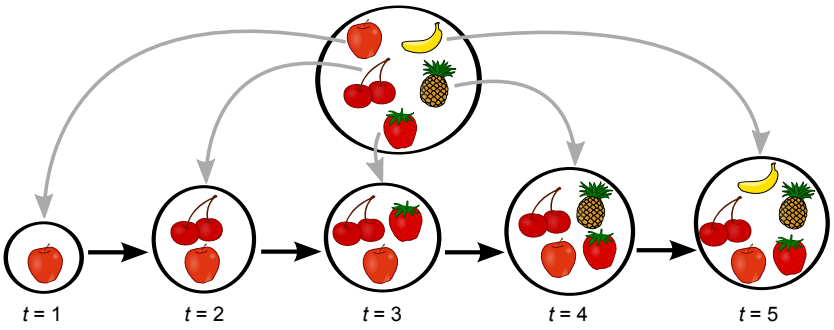

Figure 2-1: An illustration of the online grocery store recommender system, in which items from an unordered shopping list are sequentially added to the user's basket. At each time step, the set of items in the basket is used to predict future items that will be added.

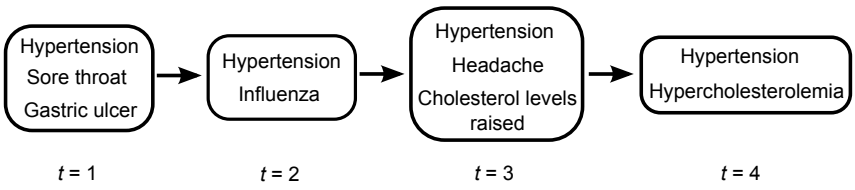

Figure 2-2: An illustration of the medical condition prediction problem, in which collections of medical conditions occur at various time steps. At each time step, we use past collections of conditions to predict conditions that will subsequently be presented.

addresses. In Section 2.4 we study patient condition prediction, and the sequences are of sets of medical conditions. The third and final application is in Section 2.5, where we apply our methods to an online grocery store recommender system. In that application we allow the recommendations to influence the order of the sequence, and provide algorithms for performing ERM. Our approach synthesizes ideas from supervised ranking in machine learning, convex optimization, and customer behavior modeling to produce flexible and powerful methods that can be used broadly for sequential event prediction problems.

2.1 Sequential Event Prediction

We begin by presenting the formal framework and notation of sequential event prediction problems, and discussing previously developed algorithms for sequential event prediction based on association rules.

We suppose that we have access to a collection of m sequences, which in our applications would be m visits from a grocery store customer, m emails, or m patient histories. The items in the sequence (*e.g.*, grocery items, email addresses, or medical conditions) come from a library of N items, $\mathcal{Z}$ being the set of these items. Sequence i in the sequence database (*e.g.*, visit i, email i, or patient i) includes a total of T_i time steps, with items (or sets of items) occurring in the sequence at time steps $t = 1, \ldots, T_i$. The item (or set of items) added to the sequence at time t is denoted $z_{i,t}$. As illustrated in Figure 2-2, each step in the sequence may be a set of items and thus we consider $z_{i,t}$ to be a set in general. In many applications, such as the online grocery store recommender system and email recipient recommendation, $z_{i,t}$ will contain only one item. The observed part of the sequence at time t is denoted $x_{i,t} = \{z_{i,j}\}_{j=1,\ldots,t}$. The full sequence, x_{i,T_i}, is denoted X_i. We denote the collection of m training sequences as X_1^m.

It is not clear how to adapt standard modeling techniques (*e.g.*, logistic regression) to sequential event prediction problems because they estimate full probabilities rather than partial probabilities. The difficulties in using regression for sequential event

prediction are discussed in detail by Rudin et al (2013), where we propose algorithms for sequential event prediction based on association rules.

Association rules have the advantage of being able to model the conditional probabilities directly. In this context, an association rule is a rule "$a \to b$," meaning that itemset a in the sequence implies item b is also in the sequence. We define the *confidence* of rule "$a \to b$" to be the proportion of training sequences with itemset a that also have item b: $\text{Conf}(a \to b) = \hat{\mathbb{P}}(b|a) = \frac{\#(a \text{ and } b)}{\#a}$. A natural strategy for using association rules for sequential event prediction is to: 0) Specify a set $\mathcal{A}$ of allowed itemsets. 1) Form all rules with left-hand side (antecedent) a an allowed itemset in the observed portion of the sequence and right-hand side b a potential future item in the sequence. 2) For each right-hand side b, find the rule with the maximum confidence. 3) Rank the right-hand sides (potential future items in the sequence) in order of descending confidence, and use this ranked list for predictions. This is the "max-confidence" algorithm, used throughout the association rule literature and applied to sequential event prediction by Rudin et al (2013).

In this work, we develop a framework for using ERM techniques in sequential event prediction. The ERM-based algorithms give increased flexibility over the max-confidence association rule algorithms by allowing the loss function to be tailored to the requirements of the specific application, and the ERM learning procedure leads to better predictions. Rather than making a prediction using the single rule with maximum confidence, we will make a prediction using an optimally weighted combination of all of the association rules.

2.2 Empirical Risk Minimization

We present a general framework for using ERM in sequential event prediction, and then show how the framework can be specified to specific applications by presenting email recipient recommendation, the online grocery store recommender system, and medical condition prediction as case studies.

The core of our ERM-based approach to sequential event prediction is a ranking

model of the relationship between items in the observed part of the sequence and potential future items. The ranking model is a scoring function $f(x_{i,t}, b)$ that, given the observed part of the sequence $x_{i,t}$, scores each item $b \in \mathcal{Z}$ according to the predicted likelihood that it is a future item in the sequence. Ideally we would like $f(x_{i,t}, b)$ to be related to $\mathbb{P}(b|x_{i,t})$, the conditional probability of item b being in the sequence given that the items in $x_{i,t}$ are in the sequence. The predictions will be made by ordering the items in descending score, so we need only that $f(x_{i,t}, b)$ is monotonically related to $\mathbb{P}(b|x_{i,t})$ in order for the predictions to be accurate. We present here two possible scoring models, which we call the *one-stage model* and the *ML-constrained model*.

2.2.1 The One-Stage Model

Our first scoring model relies on a set of real-valued variables $\{\lambda_{a,b}\}_{a,b}$ to model the influence that rule antecedent (itemset) a has on the likelihood that item b will be in the sequence, for each itemset-item pair that we are willing to consider. We let $\mathcal{A}$ be the allowed set of itemsets, and we introduce a variable $\lambda_{a,b}$ for every $a \in \mathcal{A}$ and for every $b \in \mathcal{Z}$ (that is, for every association rule). We require $\varnothing \in \mathcal{A}$ so that every (partial) sequence contains at least one itemset from $\mathcal{A}$. If itemset a and item b are likely to be present in the sequence together, $\lambda_{a,b}$ will be large and positive. Also, $\lambda_{a,b}$ can be negative in order to model negative correlations between items that are not generally found in the same sequence. The influences of the itemsets in the observed part of the sequence are combined linearly to yield the score for a given item. For example, suppose the observed sequence is $x_{i,t} = \{a_1, a_2\}$ and $\mathcal{A} = \varnothing \cup \mathcal{Z}$. In other words, $\mathcal{A}$ includes the empty itemset and all itemsets consisting of a single item. Item b is then scored as $f(\{a_1, a_2\}, b; \vec{\lambda}) = \lambda_{\varnothing,b} + \lambda_{a_1,b} + \lambda_{a_2,b}$. For a general observed sequence $x_{i,t}$, the score of item b is:

$$f(x_{i,t}, b; \vec{\lambda}) := \lambda_{\varnothing,b} + \sum_{j=1}^{t} \sum_{\substack{a \subseteq z_{i,j} \\ a \in \mathcal{A} \setminus \varnothing}} \lambda_{a,b}. \tag{2.1}$$

In applications such as medical condition prediction, events in the sequence that are far in the past may be less influential than recent events. The effect of time can be incorporated into (2.1) by inserting a weighting factor before the inner sum that is inversely proportional to the elapsed time since sequence step j. In some applications, it may be important to capture nonlinear effects of combinations of items. Feature variables for those combinations of items, such as $\lambda_{\{a \text{ and } b\},c}$, allow the model to express more complex inter-item relationships while maintaining the computational benefits of a linear model. Any antecedent discovered by the rule-mining algorithm can be used as a feature variable.

We call this the one-stage model because all of the variables $\vec{\lambda}$ are fit simultaneously in a single optimization problem. The model uses a total of $|\mathcal{A}|N$ variables: $\vec{\lambda} \in \mathbb{R}^{|\mathcal{A}|N}$. A straightforward implementation is to take $\mathcal{A}$ as itemsets of size less than or equal to 1, which is $\mathcal{A} = \varnothing \cup \mathcal{Z}$. The itemsets of size 1 give variables $\lambda_{a,b}$ $\forall a, b \in \mathcal{Z}$ that describe pairwise influences between items. The empty itemset gives rise to "base" scores $\lambda_{\varnothing,b}$ that model the likelihood of choosing item b in the absence of any information about the sequence. In this implementation, the number of variables is $|\mathcal{A}|N = N^2 + N$.

The dimensionality of the problem can be controlled by limiting the set $|\mathcal{A}|$, for instance using a maximum itemset size or a minimum support requirement, where elements of $\mathcal{A}$ must be found often enough in the dataset. Alternatively, the dimensionality of the problem could be reduced by separating items into categories and using $\lambda_{A,b}$ to model the influence of having *any* item from category A on item b. For example, a_1 and a_2 could represent individual flavors of ice cream, and A the category "ice cream." The choice of which itemsets to consider is a feature selection (or model selection) problem.

2.2.2 The ML-Constrained Model

Our second model, the ML-constrained model, reduces the dimensionality by, for every non-empty itemset a, forcing each $\lambda_{a,b}$ to be proportional to $\hat{\mathbb{P}}(b|a)$, the maximum likelihood (ML) estimate of the conditional probability of having item b in the

sequence given that itemset a is in the sequence. Specifically, we set

$$\lambda_{a,b} = \mu_a \hat{\mathbb{P}}(b|a)$$

where μ_a is a free variable that does not depend on b. $\hat{\mathbb{P}}(b|a)$ is estimated directly from the training data, prior to any optimization for model fitting, as described in Section 2.1. Then, the ML-constrained model is:

$$f_{\text{ML}}(x_{i,t}, b; \vec{\lambda}_\varnothing, \vec{\mu}) := \vec{\lambda}_{\varnothing,b} + \sum_{j=1}^{t} \sum_{\substack{a \subseteq z_{i,j} \\ a \in \mathcal{A} \setminus \varnothing}} \mu_a \hat{\mathbb{P}}(b|a), \tag{2.2}$$

To use this strategy, we first compute the ML estimates of the conditional probabilities. Then the N base scores $\lambda_{\varnothing,b}$ and the $|\mathcal{A}|$ proportionality coefficients μ_a are fit during ERM, for an optimization problem on $|\mathcal{A}| + N$ variables. Appropriate restrictions on $|\mathcal{A}|$ (for example, itemsets of size less than or equal to 1) lead to an optimization problem over $O(N)$ variables.

2.2.3 The General Loss Function

We use the training set and the ERM principle to fit vector $\vec{\lambda}$. For the sequence i at time step t, we define a set of items $L_{i,t} \subset \mathcal{Z}$ that should be ranked strictly higher than some other set of items $K_{i,t} \subset \mathcal{Z}$. For instance, $L_{i,t}$ might be the remaining items in the sequence and $K_{i,t}$ might be items not in the sequence. The value of the loss function depends on how much this is violated; specifically, we lose a point every time an item in $K_{i,t}$ is ranked above an item in $L_{i,t}$. We will subsequently explore different definitions of $L_{i,t}$ and $K_{i,t}$ appropriate for our specific applications. The most general loss function, evaluated on the training set of m sequences, is:

$$R_{\text{0-1}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i - 1} \frac{1}{T_i} \frac{1}{|K_{i,t}|} \frac{1}{|L_{i,t}|} \sum_{l \in L_{i,t}} \sum_{k \in K_{i,t}} \mathbb{1}_{[f(x_{i,t},k;\vec{\lambda}) \geq f(x_{i,t},l;\vec{\lambda})]}. \tag{2.3}$$

In our algorithms, we use the exponential loss (used in boosting), a smooth upper

bound on $R_{0\text{-}1}$. Specifically, we use that $\mathbb{1}_{[b \geq a]} \leq e^{b-a}$, and add an ℓ_2-norm regularization term:

$$R_{\exp}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i - 1} \frac{1}{T_i} \frac{1}{|K_{i,t}|} \frac{1}{|L_{i,t}|} \sum_{l \in L_{i,t}} \sum_{k \in K_{i,t}} e^{f(x_{i,t}, k; \vec{\lambda}) - f(x_{i,t}, l; \vec{\lambda})} + \beta ||\vec{\lambda}||_2^2, \tag{2.4}$$

where β is a parameter that determines the amount of regularization. Minimizing the loss function in (2.4) will produce model parameters $\vec{\lambda}$ that make accurate predictions across the sequence. Although we expressed these loss functions using f and $\vec{\lambda}$ as with the one-stage model, they apply directly to the ML-constrained model f_{ML} and its parameters $\vec{\lambda}_{\varnothing}$ and $\vec{\mu}$.

2.2.4 Scalability

In most cases, such as our email recipient recommendation and patient condition prediction examples, the loss function $R_{\exp}$ in (2.4) is convex in $\vec{\lambda}$ and thus fitting the scoring model to data requires only convex minimization in $|\mathcal{A}|N$ variables for the one-stage model, or $|\mathcal{A}| + N$ variables for the ML-constrained model. There are a number of efficient algorithms for convex minimization whose scalability has been addressed (Bertsekas, 1995). Our ERM-based algorithms inherit the scalability of whichever convex minimization algorithm is used for model fitting, subject to the dimensionality of the chosen model. Our examples show that the ERM-based algorithms can be applied to real datasets with thousands of sequences and millions of model variables. In our online grocery store recommendation example, we consider a situation where the sequence order depends directly on the recommendations. In this case the loss function is not convex, however we present algorithms based on convex programming and gradient descent. Variants of gradient descent, particularly stochastic gradient descent, are known to have excellent scalability properties in large-scale learning problems (Bottou, 2010).

2.2.5 Baseline Algorithms

In our experiments we compare the performance of our ERM-based algorithms to two baselines: the max-confidence association rule algorithm described in Section 2.1 and an item-based collaborative filtering algorithm. We use cosine similarity item-based collaborative filtering (Sarwar et al, 2001) as a baseline method. Cosine similarity is intended for a setting in which user i applies a rating $R_{i,b}$ to item b. To adapt it to sequential recommendations, we let $R_{i,b} = 1$ if sequence i contains item b, and 0 otherwise. For each item b, we construct the binary "ratings" vector $\vec{R}_b = [R_{1,b}, \ldots, R_{m,b}]$ and then compute the cosine similarity between every pair of items a and b:

$$sim(a, b) = \frac{\vec{R}_a \cdot \vec{R}_b}{||\vec{R}_a||_2 ||\vec{R}_b||_2}.$$

For each item b, we define the neighborhood of b, $\text{Nbhd}(b; k)$, as the k most similar items to b. To make a prediction from a partial sequence $x_{i,t}$, we score each item b by adding the similarities for all of the observed items in the sequence that are also in the neighborhood of b, and normalizing:

$$f_{sim}(x_{i,t}, b; k) := \frac{\sum_{a \in \bigcup_{j=1}^{t} z_{i,j} \bigcap \text{Nbhd}(b;k)} sim(a, b)}{\sum_{a \in \text{Nbhd}(b;k)} sim(a, b)}. \tag{2.5}$$

In Section 2.6, we discuss in depth why item-based collaborative filtering is not a natural fit for sequential event prediction problems. Nevertheless, since it is commonly used for similar problems, we use it as a baseline in our experiments. In our experiments, we used neighborhood sizes of 20, 40, and all items (Sarwar et al, 2001; Herlocker et al, 1999). Any ties when determining the top k most similar items were broken randomly.

2.3 Application 1: Email Recipient Recommendation

In this application we study the sequence in which recipients are added to an email. Given a partial set of recipients, the goal is to predict the remaining recipients. In this application, each item in the sequence, $z_{i,t}$, is a single email address. An email recipient recommender system knows who the sender of the email is, thus we initialize the sequence by setting $z_{i,0}$ as the address of the sender of email i. We then construct the rest of the sequence using the T_i addresses placed in the "To:" and "CC:" fields, in the order that they appear in the email.

To apply our ERM-based algorithms to this application, we must specify the sets $L_{i,t}$ and $K_{i,t}$ used in the loss function. A natural goal for this problem setting is, at each time step t, to attempt to rank all of the actual recipients that have not yet been added higher than all of the non-recipients. This goal is expressed by taking

$$L_{i,t} = \bigcup_{j=t+1}^{T_i} z_{i,j} \quad \text{and} \quad K_{i,t} = \mathcal{Z} \setminus \bigcup_{j=0}^{T_i} z_{i,j}.$$

We call this the *list loss*, as it tries to put the entire set of remaining recipients at the top of the recommendation list. For notational convenience we define $Z_i := \bigcup_{j=0}^{T_i} z_{i,j}$ to be the complete collection of email addresses in the sequence. The general loss function in (2.3) can then be rewritten

$$R^{\text{list}}_{\text{0-1}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i-1} \frac{1}{T_i(N-T_i)(T_i-t)} \sum_{l=t+1}^{T_i} \sum_{k \in \mathcal{Z} \setminus Z_i} \mathbb{1}_{[f(x_{i,t},k;\vec{\lambda}) \geq f(x_{i,t},z_{i,l};\vec{\lambda})]} \tag{2.7}$$

and (2.4) then becomes:

$$R^{\text{list}}_{\text{exp}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i-1} \frac{1}{T_i(N-T_i)(T_i-t)} \sum_{l=t+1}^{T_i} \sum_{k \in \mathcal{Z} \setminus Z_i} e^{f(x_{i,t},k;\vec{\lambda}) - f(x_{i,t},z_{i,l};\vec{\lambda})} + \beta ||\vec{\lambda}||_2^2. \tag{2.8}$$

We applied our algorithm to the Enron email dataset, a collection of about 500,000 email messages from about 150 users (http://www.cs.cmu.edu/∼enron/). We limited our experiments to the "sent" folders of the 6 users who had more than 2000 emails in their "sent" folders and only considered emails with more than 2 recipients, yielding a reduced dataset of 1845 emails with a total of 1200 unique recipients. The number of recipients per email ranged from 3 to 6.

We evaluated algorithm performance across 10 iterations, each iteration using randomly selected training and test sets of 500 emails each. For each iteration, we chose the allowed itemsets (features) $\mathcal{A}$ by applying the FP-Growth algorithm (Borgelt, 2005), a frequent itemset mining algorithm, to the training set. We mined itemsets of size up to 4, with a minimum support requirement of 3 emails. The median number of allowed itemsets across the 10 iterations was 625.5 (minimum 562, maximum 649), including the empty set. Thus the median number of variables in the one-stage model was 750,600 ($\mathcal{A}N$) and the median number of variables in the ML-constrained model was 1,825.5 ($\mathcal{A} + N$).

We used the training and test sets to evaluate the performance of the one-stage model, ML-constrained model, max-confidence association rules, and cosine similarity item-based collaborative filtering methods. For our ERM-based algorithms, we found $\vec{\lambda}$ (or, $\vec{\lambda}_\varnothing$ and $\vec{\mu}$ for the ML-constrained model) that minimized (2.8) on the training set using L-BFGS-B, the limited memory implementation of the Broyden-Fletcher-Goldfarb-Shanno algorithm (Byrd et al, 1995; Zhu et al, 1997). We set the amount of ℓ_2-norm regularization, β, using 10-fold cross validation on each training set separately with $\beta = 0, 0.001, 0.01$, and 0.1. For both the one-stage model and the ML-constrained model, for all iterations, $\beta = 0$ minimized mean error over the validation sets and was chosen. The minimum support requirement when choosing the itemsets serves as a form of regularization, which may be why ℓ_2-norm regularization was not necessary.

In Figure 2-3 we evaluated performance using the zero-one loss in (2.7). When evaluating the test error in Figure 2-3, we excluded email addresses that were not encountered in the training set because these recipients were impossible to predict and resulted in a constant error for all methods. The results show that our ERM-

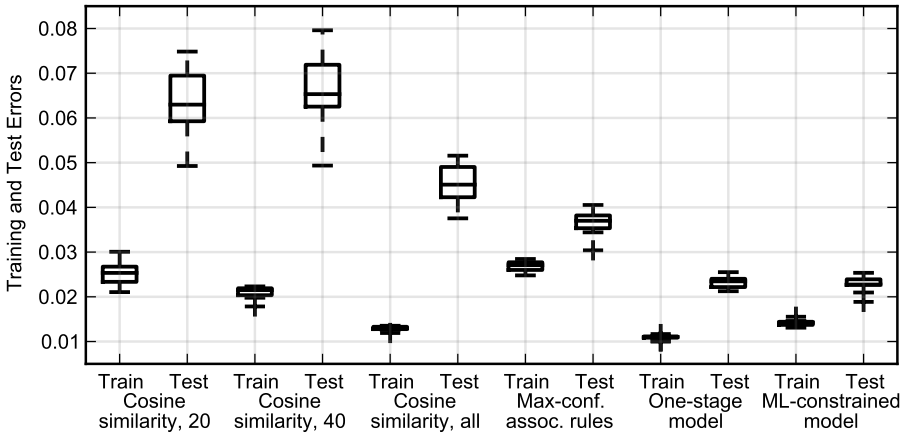

Figure 2-3: Training and test errors for email recipient recommendation.

based algorithms performed very well compared to the baseline algorithms. Cosine similarity, at all neighborhood sizes, had a tendency to overfit the data, with much higher test error than training error. The performance of the ML-constrained model was very close to that of the one-stage model, despite using many fewer variables.

We additionally evaluated performance using mean average precision. Mean average precision is a combination of precision and recall that is frequently used to evaluate ranking performance in information retrieval (Järvelin and Kekäläinen, 2000; Yue et al, 2007). The average precision of a ranked list is the average of the precision values computed at each of the relevant items. The average precision across many ranked lists is averaged to obtain the mean average precision. We measured average precision at each prediction (that is, each step in the sequence) and computed mean average precision by averaging over both time steps and sequences. We followed the procedure of McSherry and Najork (2008) to account for the presence of ties in the ranked lists. Figure 2-4 shows the mean average precision for each of the 10 iterations. Even though our methods were not optimized to maximize mean average precision, they performed well relative to both max confidence association rules and cosine similarity item-based collaborative filtering (shown in the figure only for the “all items" neighborhood, which was the best performing neighborhood size).

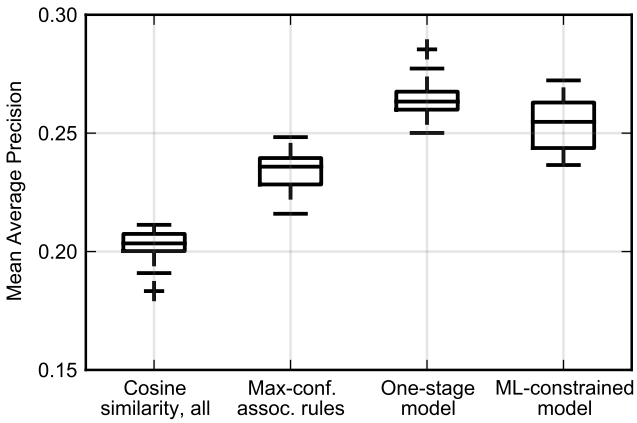

Figure 2-4: Mean average precision for email recipient recommendation. Larger numbers indicate better performance.

2.4 Application 2: Patient Condition Prediction

Here we tailor the formulation to patient condition prediction in the context of data from a large clinical trial. In this trial, patients visited the doctor periodically and reported all medical conditions for which they were taking medications. The names of the medical conditions were taken from the Medical Dictionary for Regulatory Activities (MedDRA). The dataset includes activities such as vitamin supplementation and flu shots as medical "conditions," but mainly consists of conditions that are not voluntarily chosen by the patients. We chose to predict both voluntary and involuntary conditions/activities.

In this application, each event in the sequence is a set of conditions, as opposed to the single items in the email recipient recommendation application. The patient visits are ordered in time, however, each visit itself consists of a set of symptoms which we treat as unordered. Also, the same condition can occur at multiple visits throughout the patient history, unlike email recipient recommendation, in which addresses are not repeated in a sequence. Because of this, it is important to be able to predict condition recurrence. We thus estimate $\hat{\mathbb{P}}(b|a)$ used by the max-confidence association rule algorithm and the ML-constrained model as the probability of having condition b *later in the sequence* given that a has been observed in the sequence. In this application it is not natural to make a prediction before the patient's first visit ($t = 0$), thus we

make predictions only at visits $t = 1, \ldots, T_i - 1$.

Some patients present *chronic, pre-existing* conditions that were present before their first visit and persisted after their last visit. Common chronic, pre-existing conditions include Hypertension (high blood pressure), Hypercholesterolaemia (high cholesterol), and Asthma. It is possible for a condition to be chronic, pre-existing in one patient, but not in another. For instance, some patients developed Hypertension during the study, so Hypertension was not pre-existing in those patients. We denote the set of chronic, pre-existing conditions for patient i as $C_i \subseteq \mathcal{Z}$, and place each chronic, pre-existing condition in the set of conditions for each visit: $c \in z_{i,j}$ for all $c \in C_i$, for $j = 1, \ldots, T_i$, and for all i. Chronic, pre-existing conditions were used to make predictions for subsequent conditions, but we did not attempt to predict them because predicting the recurrence of a chronic condition is trivial. We removed chronic, pre-existing conditions from the loss function by defining $\tilde{z}_{i,j} = z_{i,j} \setminus C_i$ as the set of reported conditions excluding chronic, pre-existing conditions. We then adapt the framework of (2.3) and (2.4) for training by setting $L_{i,t} = \tilde{z}_{i,t+1}$, the correct, subsequent set of non-trivial conditions, and $K_{i,t} = \mathcal{Z} \setminus z_{i,t+1}$, all other possible conditions. Then (2.3) becomes:

$$R_{0-1}^{\text{cond}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T_i - 1} \left[\frac{1}{(T_i - 1)(N - |z_{i,t+1}|)} \times \sum_{k \in \mathcal{Z} \setminus z_{i,t+1}} \sum_{l \in \tilde{z}_{i,t+1}} \frac{1}{|\tilde{z}_{i,t+1}|} \mathbb{1}_{[f(x_{i,t}, k; \vec{\lambda}) \geq f(x_{i,t}, l; \vec{\lambda})]} \right]. \tag{2.9}$$

If at a particular visit the only conditions reported were chronic, pre-existing conditions, then $\tilde{z}_{i,t+1} = \varnothing$ and the inner most sum is simply not evaluated for that i and t to avoid dividing by zero with $|\tilde{z}_{i,t+1}|$. We further write (2.4) for this application as:

$$R_{\text{exp}}^{\text{cond}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T_i - 1} \left[\frac{1}{(T_i - 1)(N - |z_{i,t+1}|)} \times \sum_{k \in \mathcal{Z} \setminus z_{i,t+1}} \sum_{l \in \tilde{z}_{i,t+1}} \frac{1}{|\tilde{z}_{i,t+1}|} e^{f(x_{i,t}, k; \vec{\lambda}) - f(x_{i,t}, l; \vec{\lambda})} \right] + \beta ||\vec{\lambda}||_2^2. \tag{2.10}$$

We used L-BFGS-B to minimize (2.10) to fit the one-stage and ML-constrained models to the medical histories of 2433 patients. Each patient made multiple visits to the doctor, at an average of 6.4 visits per patient (standard deviation, 3.0). At each visit, multiple conditions were reported, with an average of 3.2 conditions per visit (standard deviation, 2.0). We perform patient level predictions, meaning for each patient we predict the conditions that the patient will experience in the future. Conditions came from a library of 1864 possible conditions. We took $\mathcal{A}$ as all itemsets of size 1, plus the empty set. Fitting model variables required an optimization problem on 3,476,360 variables for the one-stage model ($N^2 + N$) and 3,728 variables for ML-constrained model ($2N$).

To illustrate the behavior of our models, and the differences between the one-stage model and the ML-constrained model, in Figure 2-5 we show the model influence variables corresponding to the ten most frequent conditions, fitted to a randomly selected set of 2190 ($= 0.9 \times 2433$) patients and normalized.

The association rule confidence matrix in Figure 2-5 shows $\text{Conf}(a \to b)$ for each pair of items in row a and column b, which is equivalent to the conditional probability estimate $\hat{\mathbb{P}}(b|a)$. The high confidence values on the diagonal indicate that the conditional probability of having these conditions in the future given their past occurrence is high. In many instances, but not all, these conditions are chronic, pre-existing conditions. In addition to the high confidence values along the diagonal, the rules with Hypertension and Nutritional support on the right-hand side have higher confidences, in part because Hypertension and Nutritional support are the most common conditions. The ML-constrained influence variables, $\mu_a \hat{\mathbb{P}}(b|a)$, are obtained by weighting each row a of the association rule confidence matrix by μ_a. However, the main features of the ML-constrained model variables are different from those of the association rule confidence matrix, and in fact the ML-constrained variables are similar to those of the one-stage model, $\lambda_{a,b}$. With both models, the strength with which the recurrence of a condition is predicted (the variables on the diagonal) is greatly reduced. This is because in many instances these are chronic, pre-existing conditions, and so they are excluded from the loss function and the model has no reason to predict them. For

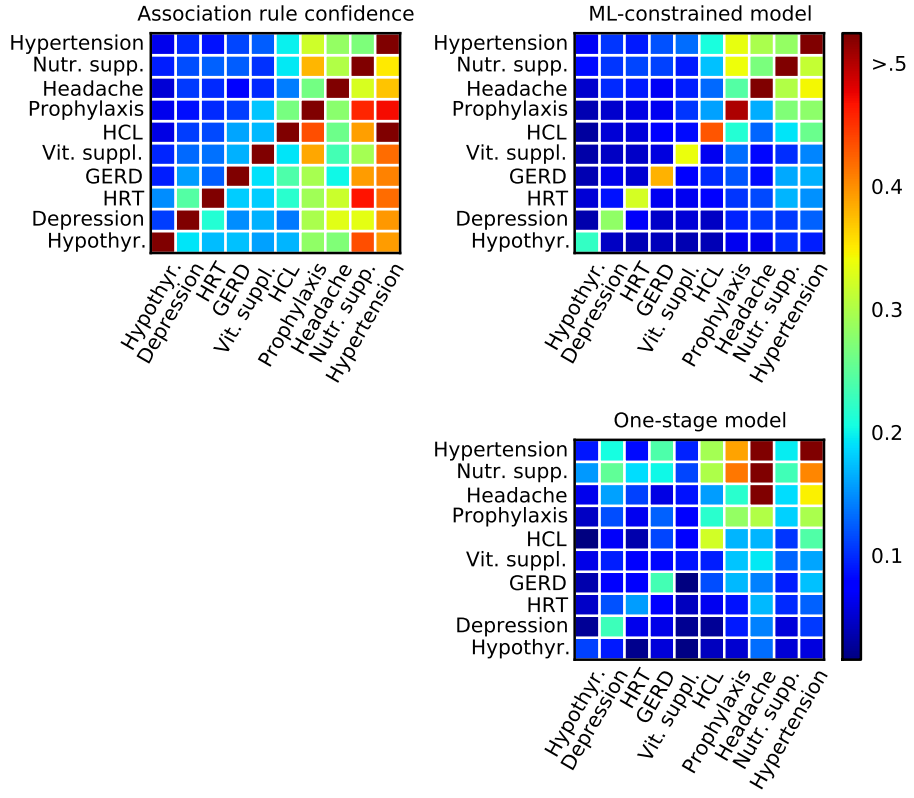

Figure 2-5: An example of fitted model variables for the ten most frequent conditions in the patient condition prediction problem, for the one-stage and ML-constrained models, together with the association rule confidence matrix. This figure illustrates the differences between the fitted variables of the two models. Row a column b is: $\text{Conf}(a \to b)$ for association rules; $\mu_a \hat{\mathbb{P}}(b|a)$ for the ML-constrained model; and $\lambda_{a,b}$ for the one-stage model. Abbreviated symptoms are Nutritional support (Nutr. supp.), Hypercholesterolaemia (HCL), Vitamin supplementation (Vit. suppl.), Gastroeophageal reflux disease (GERD), Hormone replacement therapy (HRT), and Hypothyroidism (Hypothyr.).

both models, the variables along the top row show that Hypertension most strongly predicts Hypercholesterolaemia, Prophylaxis, and Headache. Hypercholesterolaemia (high cholesterol) is correlated with obesity, as is Hypertension, so they often occur together. Prophylaxis is preventative medicine which in this context almost always means taking medications, such as aspirin, to preempt heart problems. Hypertension is a risk factor for heart problems, and so the connection with Prophylaxis is also relevant. Finally, the frequency of Headaches is also known to increase with obesity (Bigal et al, 2006). In our dataset, the reasons for Nutritional support are more varied so it is difficult to interpret the relation between Nutritional support and Prophylaxis, Headache, and Hypertension.

To evaluate the performance of the ERM-based algorithms, we performed ten iterations of random sub-sampling with training and test sets each of 500 patients. We applied the cosine similarity algorithm with varying neighborhood sizes (20, 40, and all items), the max-confidence association rule algorithm, the one-stage model, and the ML-constrained model. To set the amount of ℓ_2-norm regularization in the loss function, β, we did 10-fold cross validation on each training set separately with $\beta = 0.001, 0.005, 0.01, 0.05$, and 0.1. We then set β to the value that minimized the mean error over the validation sets. With one-stage minimization, chosen values of β ranged from 0.001 to 0.05, with $\beta = 0.005$ chosen most frequently. with ML-constrained minimization $\beta = 0.01$ was always chosen. The error on the training and test sets was evaluated using (2.9), and boxplots of the results across all 10 iterations are in Figure 2-6. When evaluating the test error in Figure 2-6, we excluded conditions that were not encountered in the training set because these conditions were impossible to predict and resulted in a constant error for all methods.

Our method, using both models, performed very well compared to the baselines, which seem to have had an overfitting problem judging from the difference between training and test results. The ML-constrained model used far fewer variables than the one-stage model (about 3000 compared to about 3.5 million) and generalized well.

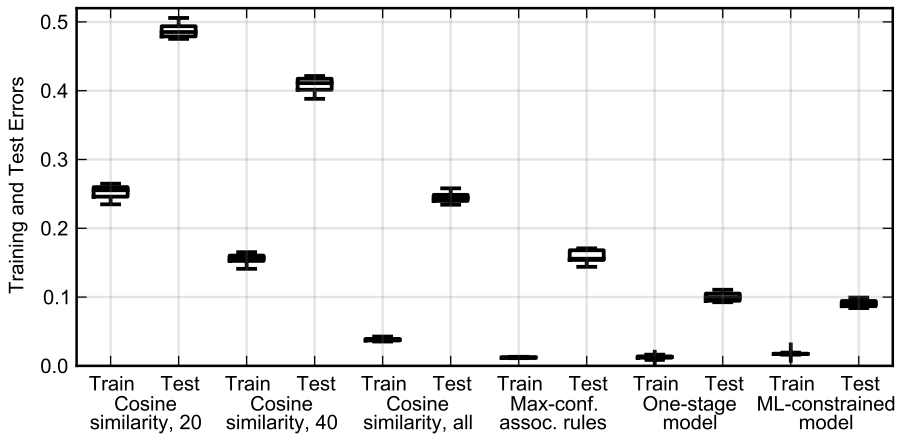

Figure 2-6: Training and test errors for patient condition prediction.

2.5 Application 3: An Online Grocery Store Recommender System

In this application a customer comes to the online grocery store with a shopping list, and sequentially adds the items from his or her shopping list into a shopping basket. The goal is to, at each time step, make predictions about the other items the customer wishes to purchase. For the purpose of this paper, the recommender system is designed to be a tool to assist the customer, *i.e.*, there is no motive to recommend higher priced items, promote sale items, etc., although these could be incorporated in an extended version of our formulation. Similar to the email recipient recommendation application, the sequence is of single, non-repeated items. The sequence is initialized as an empty basket: $z_{i,0} = \varnothing$. In this application we assume that the customer's shopping list (the collection of items that form the sequence) has no inherent order. Rather, we assume that the order in which the customer adds items to the basket depends directly on the recommendations made to the customer.

2.5.1 Fitting a Sequential Prediction Model to an Unordered Set

Although the shopping list is unordered, the predictions at each time step depend on the set of items that have already been added to the basket, and thus depend indirectly on the order in which items are added to the basket. To fit the model variables to the training data, we must impose an order for the items to be added to the basket. Here we allow the predictions to influence the ordering of the items. Specifically, we assume that the customer prefers convenience, in that the next item added to the basket is the most highly recommended item on their shopping list. For convenience, denote the contents of the basket at time t as $Z_{i,t} := \bigcup_{j=1}^{t} z_{i,j}$ and the contents of the shopping list as $Z_i := \bigcup_{j=1}^{T_i} z_{i,j}$. We then order the items according to:

$$z_{i,t+1} \in \underset{b \in Z_i \setminus Z_{i,t}}{\operatorname{argmax}} f(x_{i,t}, b; \vec{\lambda}). \tag{2.11}$$

It may be that the argmax is not unique, *i.e.*, there is a tie. Here we break ties randomly to choose the next item. The order in which items are added to the basket is a function of the model variables $\vec{\lambda}$. When fitting the model variables, we do not order the items *a priori*, rather we allow the ordering to change during fitting, together with the model variables. Our assumption in (2.11) could be replaced by an application-specific model of user behavior; (2.11) is not an accurate assumption for all applications. On the other hand, a recommender system trained using this assumption has properties that are useful in real situations, as we discuss below.

We will train the machine learning model to minimize the loss function (2.4) with respect to variables $\vec{\lambda}$, using (2.11). The qualitative effect of (2.11) is to put the items that are (conditionally) more likely to be purchased into the basket sooner, while leaving unlikely items for later. Once these items are in the basket, they will be used for making the subsequent predictions. Thus the model variables that generally play the largest role in the learning, and that are most accurately estimated, correspond to items that are more likely to be purchased.

One could imagine training the model using all permutations of each shopping list

in the training set as an alternative to (2.11). As another alternative, one could randomly permute the shopping lists and include only that ordering. Even though these approaches potentially capture some realistic situations that our ordering assumption does not, we argue that it is not a good idea to do either of these. First, the number of possible permutations on even a moderately sized training set makes it computationally intractable to train using all possible permutations. Second, if the users do adhere, even loosely, to a behavioral strategy such as our assumption in (2.11), the model would be forced to fit many permutations that would rarely occur, and would treat those rare situations as equally important to the ones more likely to occur. For example, a randomly permuted shopping list could place conditionally unlikely items at the beginning of the ordering. This could actually create bias against the correct recommendations.

2.5.2 Specifying the Loss Function

In this application we use two loss functions. First, we use the list loss that was defined in (2.7) and (2.8) for the email recipient recommendation. This loss function has the goal of placing all of the items that remain on the shopping list at the top of the recommendations. In some situations, however, we may not need the entire shopping list near the top of the recommendations, rather we might need only that one item from the shopping list is highly ranked. In this way, the loss associated with a particular basket will depend only on the highest ranked item that is still on the shopping list. We call this formulation the *item loss*. Under the item loss, a perfect prediction would have at least one item on the shopping list ranked higher than all items that are not on the shopping list:

$$\max_{b \in Z_i \setminus Z_{i,t}} f(x_{i,t}, b; \vec{\lambda}) > f(x_{i,t}, k; \vec{\lambda}), \text{ for all } k \in \mathcal{Z} \setminus Z_i \text{ and for all } i \text{ and } t.$$

This can be realized using by taking $L_{i,t} = \arg\max_{l \in Z_i \setminus Z_{i,t}} f(x_{i,t}, l; \vec{\lambda})$ and $K_{i,t} = \mathcal{Z} \setminus Z_i$. The general loss function in (2.3) then becomes

$$R^{\text{item}}_{\text{0-1}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i - 1} \frac{1}{T_i(N - T_i)} \sum_{k \in \mathcal{Z} \setminus Z_i} \mathbb{1}_{[f(x_{i,t},k;\vec{\lambda}) \geq \max_{l \in Z_i \setminus Z_{i,t}} f(x_{i,t},l;\vec{\lambda})]} \quad (2.12)$$

and (2.4) becomes

$$R^{\text{item}}_{\text{exp}}(f, X_1^m; \vec{\lambda}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T_i - 1} \frac{1}{T_i(N - T_i)} \sum_{k \in \mathcal{Z} \setminus Z_i} e^{f(x_{i,t},k;\vec{\lambda}) - \max_{l \in Z_i \setminus Z_{i,t}} f(x_{i,t},l;\vec{\lambda})} + \beta ||\vec{\lambda}||_2^2. \quad (2.13)$$

As an extreme example, suppose the recommendation list has a single item from the shopping list at the top, and the rest of the shopping list at the bottom. The list loss would be large, while the item loss would be small or zero. Qualitatively, item loss forces a form of rank diversity which we will now discuss.

At the first time step $t = 0$, there is no knowledge of the event sequence so the same recommendation list will be used for all shopping lists. Let us consider how this recommendation list might be constructed in order to achieve a low item loss for the following collection of example shopping lists:

Shopping list 1: onion, garlic, beef, peppers
Shopping list 2: onion, garlic, chicken
Shopping list 3: onion, garlic, fish
Shopping list 4: onion, lemon
Shopping list 5: flour, oil, baking powder
Shopping list 6: flour, sugar, vanilla

In these shopping lists, the three most frequent items are onion, garlic, and flour. Using item loss, we incur loss for every shopping list that does not contain the highest ranked item. A greedy strategy to minimize item loss places the most common item, onion, first on the recommendation list, thus incurring 0 loss for shopping lists 1-4. The second place in the recommendation list will not be given to the second most fre-

quent item (garlic), rather it will be given to the most frequent item among shopping lists that do not contain onion. This means the second item on the recommendation list will be flour. With onion ranked first and flour ranked second, we incur 0 loss on shopping lists 1-4, and the loss is one for each of shopping lists 5 and 6. The ranks of the remaining items do not matter for this time step, as these two ingredients have satisfied every shopping list. This greedy strategy is the same as the greedy strategy for the maximum coverage problem, in which we are given a collection of sets with some elements in common and choose k sets to cover as many elements as possible. This algorithm has been used for rank diversification (Radlinski et al, 2008). This greedy strategy would be an efficient strategy to minimize item loss if we made a prediction only at $t = 0$, however, it might not truly minimize loss, and even if it does happen to minimize loss at time $t = 0$, it might not minimize loss over all time steps. In our experiments, we found that minimizing item loss produced a diverse ranked list at each time step: It attempts to ensure that an item from each shopping list is ranked highly, as opposed to simply ranking items based on popularity.

2.5.3 ERM for the Online Grocery Store Recommender System

The model variables $\vec{\lambda}$ are chosen to minimize the loss on the training set by minimizing the list loss (2.8) or item loss (2.13). Using the assumption in (2.11), the basket at any time step $Z_{i,t}$ is itself a function of the recommender system, *i.e.*, of $\vec{\lambda}$. Small changes in $\vec{\lambda}$ can change which item has the highest score, thus changing $z_{i,t+1}$. Because the predictions at $t+1$ and all future time steps depend on $z_{i,t+1}$, this can significantly alter the value of the loss. Depending on $\vec{\lambda}$, different possibilities for $z_{i,t+1}$ could change $f(x_{i,t+1}, b; \vec{\lambda})$ by arbitrarily large amounts. This is why the loss functions in (2.8) and (2.13) are, subject to the assumption in (2.11), generally discontinuous.

The discontinuities occur at values of $\vec{\lambda}$ where there are ties in the ranked list, that is, where the model is capable of producing multiple orderings. It is when there

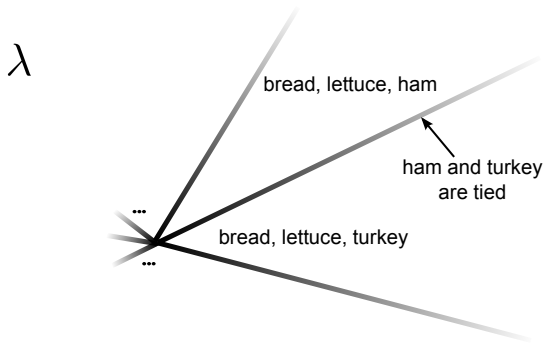

Figure 2-7: An illustration of how the model variables can be partitioned into regions that lead to different orderings of the items in each shopping basket. The borders between regions correspond to selections of model variables for which the argmax in (2.11) is not unique, *i.e.*, there is a tie. The regions are polyhedral, and the objective function is convex over each region but discontinuous at the borders.

are ties that epsilon changes in $\vec{\lambda}$ can alter $z_{i,t+1}$ and thus significantly change all future predictions. These discontinuities partition the variable space into regions that correspond to different orderings. Figure 2-7 is an illustration of how the space of $\vec{\lambda}$ is partitioned by different orderings, with ties between items on the borders. The loss function is convex over each region and discontinuities occur only at the region borders. We now show that these regions are convex, which will lead to an optimization strategy.

Proposition 1. *Let Λ_{z^*} be the set of $\vec{\lambda} \in \mathbb{R}^{|\mathcal{A}|N}$ in the one-stage model or $(\vec{\lambda}_\varnothing, \vec{\mu}) \in \mathbb{R}^{|\mathcal{A}|+N}$ in the ML-constrained model that can produce the specific ordering $\{z^*_{i,t}\}_{i,t}$ under the assumption of (2.11). Then, Λ_{z^*} is a polyhedron.*

Proof. A particular ordering z^* is produced when, for each training list i and at each time step t, the next item in the ordering $z^*_{i,t+1}$ has the highest score of all of the items remaining on the shopping list. In other words, to choose $z^*_{i,t+1}$ before $z^*_{i,k}$ for all $k > t+1$, it must be true that the score of $z^*_{i,t+1}$ is greater than or equal to the score of $z^*_{i,k}$:

$$f(x^*_{i,t}, z^*_{i,t+1}; \vec{\lambda}) \geq f(x^*_{i,t}, z^*_{i,k}; \vec{\lambda}), \forall k > t+1.$$

These constraints in fact define Λ_{z^*}:

$$\Lambda_{z^*} := \Bigg\{ \vec{\lambda} : \lambda_{\varnothing, z^*_{i,t+1}} + \sum_{j=1}^{t} \sum_{\substack{a \subseteq z^*_{i,j} \\ a \in \mathcal{A} \setminus \varnothing}} \lambda_{a, z^*_{i,t+1}} \geq \lambda_{\varnothing, z^*_{i,k}} + \sum_{j=1}^{t} \sum_{\substack{a \subseteq z^*_{i,j} \\ a \in \mathcal{A} \setminus \varnothing}} \lambda_{a, z^*_{i,k}}, \tag{2.14}$$

$$i = 1, \ldots, m,\ t = 0, \ldots, T_i - 2,\ k = t+2, \ldots, T_i \Bigg\}.$$

Thus Λ_{z^*} can be defined by a set of $\sum_{i=1}^{m} \frac{1}{2}(T_i - 1)T_i$ linear inequalities and is a polyhedron. The result holds for the ML-constrained model by replacing $\lambda_{a,b}$ with $\mu_a \hat{\mathbb{P}}(b|a)$. □

The proposition is true for each ordering z^* that can be realized, and thus the whole space can be partitioned into polyhedral regions. When the variables $\vec{\lambda}$ (or equivalently, $\vec{\lambda}_{\varnothing}$ and $\vec{\mu}$) are constrained to a particular ordering Λ_{z^*}, the list loss in (2.8) and the item loss in (2.13) are convex. We will now describe two optimization strategies for minimizing (2.8) and (2.13) subject to the assumption in (2.11). For notational convenience we will describe the algorithms in terms of $\vec{\lambda}$, but the algorithms can be directly applied to the ML-constrained model.

Convex optimization within regions, simulated annealing between regions

Because Λ_{z^*} is convex for each z^*, it is possible to find the optimal $\vec{\lambda}$ within Λ_{z^*} using convex programming. Our goal is to minimize the loss across all possible orderings z^*, so we need also to explore the space of possible orderings. Our first approach is to use simulated annealing, as detailed in Algorithm 1, to hop between the different regions, using convex optimization within each region.

Simulated annealing is an iterative procedure where $\vec{\lambda}$ is updated step by step. Steps that increase the loss are allowed with a probability that depends on a "temperature" variable. The temperature is decreased throughout the procedure so that steps that increase the loss become increasingly improbable. The algorithm begins with an initial ordering, then the minimizer within that region is found by convex optimization. Then we use a simulated annealing step to move to a neighboring or-

dering, and the process is repeated. There are many "unrealizable" orderings that can be achieved only by a trivial model in which all of the variables λ equal the same constant so that the items are tied at every prediction. Thus, randomly permuting the ordering as is usually done in simulated annealing will often yield only trivial neighbors. An alternative strategy is to choose a direction in the variable space (for example, the direction of gradient descent) and to step in that direction from the current position of $\vec{\lambda}$ until the ordering changes. This new ordering is a realizable neighbor and can be used to continue the simulated annealing. Additional neighbors can be discovered by stepping in the variable space in different directions, for instance orthogonal to the gradient. The move to the new ordering is accepted with a probability that depends on the change in loss between the optimal solutions for the two orderings, and the temperature variable. This is done for a fixed number of steps, and finally the output is the best solution that was encountered during the search.

Algorithm 1: A combination of convex optimization and simulated annealing for fitting $\vec{\lambda}$ under the assumption of (2.11).

Data: Training set X_1^m, number of simulated annealing steps T_S, annealing schedule $Temp$
Result: $\vec{\lambda}_{\text{best}}$
Begin with an initial ordering $\{z_{i,t}\}_{i,t}$
Form the constraints Λ_z associated with this ordering (Equation 2.14)
Solve the convex program $\vec{\lambda}^* \in \text{argmin}_{\vec{\lambda} \in \Lambda_z} R_{\exp}(f, X_1^m; \vec{\lambda})$ (Equation 2.8 or 2.13)
Set $\vec{\lambda}_{\text{best}} = \vec{\lambda}^*$
for $t = 1$ **to** T_S **do**
    Find a neighboring ordering $\{z'_{i,t}\}_{i,t}$
    Form the constraints $\Lambda_{z'}$ associated with the new ordering
    Solve the convex program $\vec{\lambda}'^* \in \text{argmin}_{\vec{\lambda} \in \Lambda_{z'}} R_{\exp}(f, X_1^m; \vec{\lambda})$
    Sample a number q uniformly at random from $[0, 1]$
    if $\exp((R_{\exp}(f, X_1^m; \vec{\lambda}^*) - R_{\exp}(f, X_1^m; \vec{\lambda}'^*))/Temp(t)) > q$ **then**
        Accept this move: $\vec{\lambda}^* = \vec{\lambda}'^*$
        if $R_{\exp}(f, X_1^m; \vec{\lambda}^*) < R_{\exp}(f, X_1^m; \vec{\lambda}_{\text{best}})$ **then**
            $\vec{\lambda}_{\text{best}} = \vec{\lambda}^*$

Gradient descent

When N is large, it can be expensive to solve the convex program at each step of simulated annealing in Algorithm 1, particularly using the one-stage model which requires $|\mathcal{A}|N$ variables. It may be more efficient to use an unconstrained first-order method such as pure gradient descent. It is likely that a gradient descent algorithm will cross the discontinuities, and there are no convergence guarantees. In Algorithm 2, we ensure that the gradient descent terminates by imposing a limit on the number of steps that increase the loss. We take as our result the best value that was encountered during the search.

Algorithm 2: A gradient descent algorithm to fit $\vec{\lambda}$ under assumption (2.11).

Data: Training set X_1^m, maximum number of steps that increase loss T_G, step size γ
Result: $\vec{\lambda}_{\text{best}}$
Begin with some initial $\vec{\lambda}_0$ and the associated $R_{\text{exp}}(f, X_1^m; \vec{\lambda}_0)$ (Equation 2.8 or 2.13)
Set: $\vec{\lambda}_{\text{best}} = \vec{\lambda}_0$
$t = 0$ (an index for all steps)
$l = 0$ (an index for steps that increase loss)
while $l < T_G$ **do**
  Take a step of gradient descent:
  $\vec{\lambda}_{t+1} = \vec{\lambda}_t - \gamma \bigtriangledown R_{\text{exp}}(f, X_1^m; \vec{\lambda}_t)$
  if $R_{\text{exp}}(f, X_1^m; \vec{\lambda}_{t+1}) < R_{\text{exp}}(f, X_1^m; \vec{\lambda}_{\text{best}})$ **then**
    $\vec{\lambda}_{\text{best}} = \vec{\lambda}_{t+1}$
  if $R_{\text{exp}}(f, X_1^m; \vec{\lambda}_{t+1}) > R_{\text{exp}}(f, X_1^m; \vec{\lambda}_t)$ **then**
    $l = l + 1$
  $t = t + 1$

2.5.4 Experimental Results

Our online grocery store recommender system dataset is derived from the publicly available ICCBR Computer Cooking Contest recipe book (ICCBR, 2011). The original dataset is 1490 recipes, each of which, among other things, contains a list of ingredients. We treated the ingredients in each recipe as unordered items on a shopping list. We limited our experiments to the 250 most frequently occurring ingredi-

ents. This excluded only very rare ingredients that appeared in less than 5 recipes in the dataset, for instance "alligator." We took the allowed itemsets $\mathcal{A}$ as individual items, plus the empty set. The ML-constrained model thus required an optimization problem on 500 variables ($2N$) whereas the one-stage model required solving an optimization problem on 62,500 variables ($N^2 + N$).

We fit the one-stage model and the ML-constrained model, using both list loss in (2.8) and item loss in (2.13). Training and test sets each of 100 shopping lists were selected using random sub-sampling without replacement. The models were evaluated using the zero-one loss in (2.7) or (2.12). Training and test sets were sampled independently for 20 trials to provide a distribution of training and test losses. The results for Algorithm 1 (convex programming / simulated annealing) and Algorithm 2 (gradient descent) were very similar. We found that Algorithm 2 scaled better with the dimensionality of the dataset, so we report the results of Algorithm 2 here. The amount of ℓ_2-norm regularization in the loss function, β, was set using 3-fold cross validation on each training set, separately with $\beta = 0.0001, 0.001, 0.01, 0.1, 1,$ and 10. We then set β to the value that minimized the mean error over the validation sets. With list loss and the one-stage model, chosen values of β ranged from 0.001 to 0.1, with 0.001 chosen most frequently. With list loss and the ML-constrained model, chosen values of β ranged from 0.01 to 1, with 0.1 chosen most frequently. With item loss and the one-stage model, chosen values of β ranged from 0.0001 to 0.01, with 0.001 chosen most frequently. With item loss and the ML-constrained model, chosen values of β ranged from 0.01 to 1, with 0.01 chosen most frequently. The training and test errors across the 20 trials are shown as boxplots in Figures 2-8 and 2-9 for list loss and item loss respectively. As before, the test errors in Figures 2-8 and 2-9 exclude items that were not present in the training set, as these items necessarily cannot be well predicted and provided a constant bias.

The large difference between training and test errors suggests that that there is some overfitting despite the ℓ_2-norm regularization. This is not surprising given the number of possible items (250) and the number of shopping lists used for training (100). A larger training set would lead to better generalization (and less of an ob-

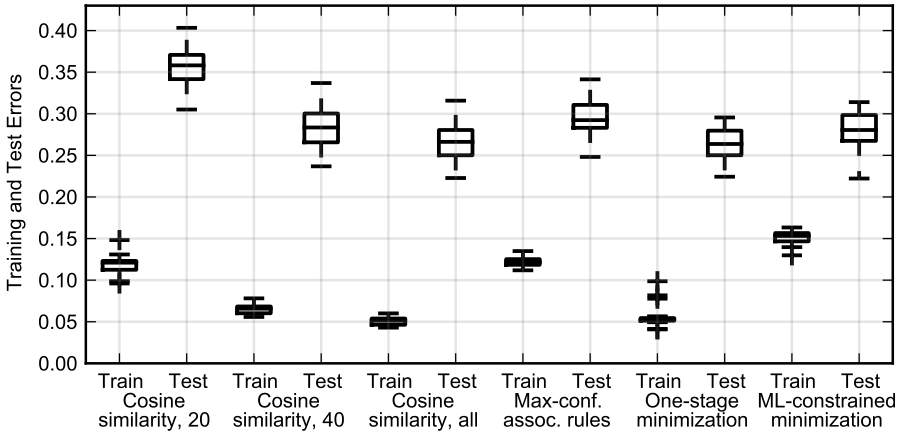

Figure 2-8: List loss training and test errors for the online grocery store recommender system.

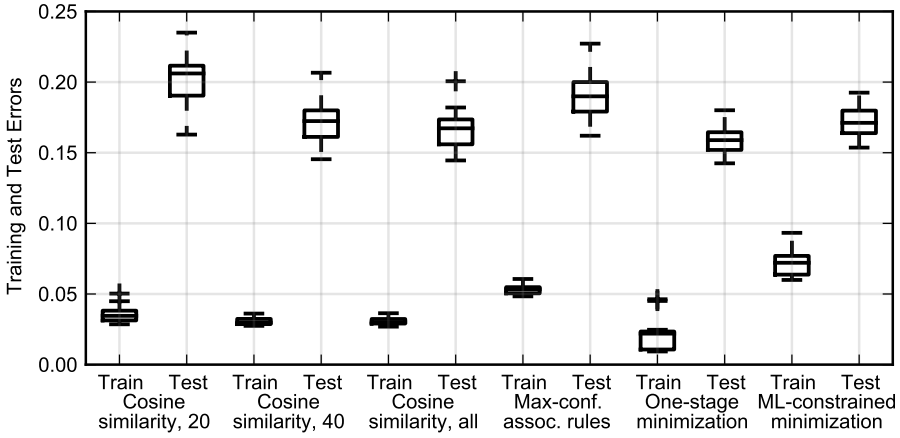

Figure 2-9: Item loss training and test errors for the online grocery store recommender system.

servable difference between the methods), although if it were desirable to fit a model individually to each customer the training data may truly be very limited. This is related to the “cold start" problem in recommender systems, when predictions need to be made when data are scarce.

For both loss functions, our method performed well compared to the cosine similarity and association rule baselines. The one-stage model performed slightly better than the ML-constrained model, although it does so at a much higher computational cost. This is consistent with the results of the other experiments in this paper, which have shown that the ML-constrained model is able to provide close to the same performance as the one-stage model.

2.6 Related Work

This work is related to previous work on recommender systems, medical condition prediction, time-series modeling and supervised ranking.

There are many different approaches to recommender systems. Adomavicius and Tuzhilin (2005) give a review of current methods. Shani et al (2005) work with sequential recommendations using Markov decision processes, which differs from our approach in that our approach does not assume the Markov property. Collaborative filtering methods have been especially common in recommender systems (see Sarwar et al, 2001, for a review). Some collaborative filtering methods rely on additional user information such as demographics and are not appropriate for our setting. Item-based collaborative filtering methods, cosine similarity in particular, are an extremely popular type of recommender system that are related to our approach as they consider only relations between various items in the sequence database (Sarwar et al, 2001; Linden et al, 2003). However, item-based collaborative filtering is generally not appropriate for these sequential prediction problems. Collaborative filtering algorithms are generally evaluated according to regression criteria (measuring accuracy in ratings) rather than ranking criteria, and is thus designed for a completely different type of learning framework. Also, when applying item-based collaborative filtering using the weighted

sum method (Section 3.2.1 in Sarwar et al, 2001), we needed to compute an inner product of the similarities with the "ratings" for all co-rated items. However, for an incomplete basket, we do not have the ratings for all co-rated items, since there is no natural way to differentiate between items that have not yet been purchased in this transaction and items that will not be purchased in this transaction, as both have a "rating" of 0 at time t. Thus, the only ratings that are available are ratings of "1" indicating that an item is in the basket. In other words, our approach is intrinsically sequential, whereas it is unnatural to force item-based collaborative filtering into a sequential framework. Additionally, cosine similarity in particular is a symmetric measure ($sim(a, b) = sim(b, a)$) and thus not related to the conditional probability of b given a. These differences help explain why in our experiments, particularly email recipient recommendation and patient condition prediction, cosine similarity item-based collaborative filtering was outperformed by our methods, both in terms of our loss function and average precision.

Medical recommender systems are discussed by Davis et al (2010). The output of their system is a ranked list of conditions that are likely to be subsequently experienced by a patient, similar to the ranked recommendation lists that we produce. Their system is based on collaborative filtering rather than bipartite ranking loss which is the core of our method. Duan et al (2011) develop a clinical recommender system which uses patient conditions to predict suitable treatment plans. Much of the work in medical data mining uses explanatory modeling (*e.g.*, finding links between conditions), which is fundamentally different from predictive modeling (Shmueli, 2010). Most work in medical condition prediction focuses on specific diseases or data sets (see Davis et al, 2010, for a literature review). Email recipient recommendation has been studied with several approaches, often incorporating the email content using language models, or finding clusters in the network of corresponding individuals (Dom et al, 2003; Pal and McCallum, 2006; Carvalho and Cohen, 2008; Roth et al, 2010).

A large body of research on time series modeling dates back at least to the 1960's and provides many approaches for sequential prediction problems. Recent applications to medicine in general and patient level prediction in particular include Enright

et al (2011), Stahl and Johansson (2009), and Hu et al (2010). Our ML-constrained model was motivated by the mixture transition distribution developed by (Berchtold and Raftery, 2002) to model high-order Markov chains. However, as we discussed earlier, typical time-series approaches focus specifically on the *order* of past events whereas in our applications the historical order seems of peripheral importance.

Our model and fitting procedure derive from previous work on supervised ranking. Many approaches to ranking have been proposed, including methods based on classification algorithms (Herbrich et al, 1999; Chapelle and Keerthi, 2010; Joachims, 2002; Freund et al, 2003; Burges et al, 2005), margin maximization (Yan and Hauptmann, 2006), order statistics (Lebanon and Lafferty, 2002; Clémençon and Vayatis, 2008), and others (Cao et al, 2007; Rudin, 2009). The loss functions that we use derive from the bipartite misranking error, and the exponential upper bound is that used in boosting. Our list loss is in fact exactly the misranking error; thus minimizing list loss corresponds to maximizing the area under the ROC curve (Freund et al, 2003). Other loss functions can be substituted as is appropriate for the problem at hand, for example our item loss is a good fit for problems where only one relevant item needs to be at the top. Minimizing misranking error does not imply optimizing other evaluation metrics, such as average precision and discounted cumulative gain as illustrated in Yue et al (2007) and Chang et al (2012). Our formulation could potentially be adapted to optimize other evaluation metrics, as is done in Yue et al (2007) and Chang et al (2012), if these metrics are the quantity of interest. The theoretical framework underpinning ranking includes work in statistics, learning theory, and computational complexity (Cohen et al, 1999; Freund et al, 2003; Clémençon et al, 2008; Cossock and Zhang, 2008; Rudin and Schapire, 2009). Our work is also related to the growing fields of preference learning and label ranking (Fürnkranz and Hüllermeier, 2003; Hüllermeier et al, 2008; Dekel et al, 2004; Shalev-Shwartz and Singer, 2006).

2.7 Conclusions

We have presented a supervised ranking framework for sequential event prediction that can be adapted to fit a wide range of applications. We proposed two ranking models, and showed how to specify our general loss function to applications in email recipient recommendation, patient condition prediction, and an online grocery store recommender system. In the online grocery store recommender system application, we allowed the predictions to alter the sequence of events resulting in a discontinuous loss function. Using the fact that the variable space can be partitioned into convex sets over which the loss function is convex, we presented two algorithms for approximately minimizing the loss. In all of our experiments, our ERM-based algorithms performed well, better than the max-confidence and cosine similarity baselines. Our ML-constrained model in particular provided good performance while keeping the dimensionality of the optimization problem small. This model combines association rules by learning a weighting for each antecedents, and weighting consequents according to their confidence.

The models built from weighted linear combinations of association rules can be quite powerful, while maintaining the interpretability inherent in having association rules as the fundamental unit of the model. We used a simple linear association rule model in a recommender system competition as part of the ECML PKDD 2013 conference. We came in third place for the offline challenge and second place in an online challenge, making predictions to real users of a baby name site (Letham, 2013). The online portion of the competition revealed another benefit for using association rules for predictions: The model can make predictions very quickly, in the milliseconds that it takes the page to load.

There are many other applications where the *set* of past events matters for predicting the future, rather than the order of past events. Our ERM-based methodology is a direct and practical approach for using association rules to solve these prediction tasks.

Chapter 3

Bayesian Association Rules and Decision Lists

An alternative approach for forming an interpretable predictive model from association rules is to construct a sparse *decision list*. A decision list is an ordered series of rules, each an *if... then...* statement, where the *if* defines a partition of a set of features and the *then* a predicted outcome. Because of this form, a decision list model naturally provides a reason for each prediction that it makes. Figure 3-1 presents an example decision list that we created using the Titanic dataset available in R. This dataset provides details about each passenger on the Titanic, including whether the passenger was an adult or child, male or female, and their class (1st, 2nd, 3rd, or crew). The goal is to predict whether the passenger survived based on his or her features. The list provides an explanation for each prediction that is made. For example, we predict that a passenger is less likely to survive than not *because* he or she was in the 3rd class. The list in Fig. 3-1 is one accurate and interpretable decision list for predicting survival on the Titanic, possibly one of many such lists. Our goal is to learn these lists from data.

Our model, called Bayesian Rule Lists (BRL), produces a posterior distribution over permutations of *if... then...* rules, starting from a large, pre-mined set of possible rules. The decision lists with high posterior probability tend to be both accurate and interpretable, where the interpretability comes from a hierarchical prior over

if male **and** adult **then** *survival probability* 21% (19% - 23%)
else if 3rd class **then** *survival probability* 44% (38% - 51%)
else if 1st class **then** *survival probability* 96% (92% - 99%)
else *survival probability* 88% (82% - 94%)

Figure 3-1: Decision list for Titanic. In parentheses is the 95% credible interval for the survival probability.

permutations of rules. The prior favors concise decision lists that have a small number of total rules, where the rules have few terms in the left-hand side.

BRL provides a new type of balance between accuracy, interpretability and computation. Consider the challenge of constructing a predictive model that discretizes the input space in the same way as decision trees (Breiman et al, 1984; Quinlan, 1993), decision lists (Rivest, 1987) or associative classifiers (Liu et al, 1998). Greedy construction methods like classification and regression trees (CART) or C5.0 are not particularly computationally demanding, but in practice the greediness heavily affects the quality of the solution, both in terms of accuracy and interpretability. At the same time, optimizing a decision tree over the full space of all possible splits is not a tractable problem. BRL strikes a balance between these extremes, in that its solutions are not constructed in a greedy way involving splitting and pruning, yet it can solve problems at the scale required to have an impact in real problems in science or society, including modern healthcare.

A major source of BRL's practical feasibility is the fact that it uses pre-mined rules, which reduces the model space to that of permutations of rules as opposed to all possible sets of splits. The complexity of the problem then depends on the number of pre-mined rules rather than on the full space of feature combinations; in a sense, this algorithm scales with the sparsity of the dataset rather than the number of features. As long as the pre-mined set of rules is sufficiently expressive, an accurate decision list can be found, and in fact the smaller model space might improve generalization (through the lens of statistical learning theory, Vapnik, 1995). An additional advantage to using pre-mined rules is that each rule is independently

both interpretable and informative about the data.

BRL's prior structure encourages decision lists that are sparse. Sparse decision lists serve not only the purpose of producing a more interpretable model, but also reduce computation, as most of the sampling iterations take place within a small set of permutations corresponding to the sparse decision lists. In practice, BRL is able to compute predictive models with accuracy comparable to state-of-the-art machine learning methods, yet maintain the same level of interpretability as medical scoring systems.

The motivation for our work lies in developing interpretable patient-level predictive models using massive observational medical data. To this end, we use BRL to construct an alternative to the CHADS_2 score of Gage et al (2001). CHADS_2 is widely-used in medical practice to predict stroke in patients with atrial fibrillation. A patient's CHADS_2 score is computed by assigning one "point" each for the presence of congestive heart failure (C), hypertension (H), age 75 years or older (A), and diabetes mellitus (D), and by assigning 2 points for history of stroke, transient ischemic attack, or thromoembolism (S_2). The CHADS_2 score considers only 5 factors, whereas the updated CHA_2DS_2-VASc score (Lip et al, 2010a) includes three additional risk factors: vascular disease (V), age 65 to 74 years old (A), and female gender (Sc). Higher scores correspond to increased risk. In the study defining the CHADS_2 score (Gage et al, 2001), the scores was calibrated with stroke risks using a database of 1,733 Medicare beneficiaries followed for, on average, about a year.

Our alternative to the CHADS_2 was constructed using 12,586 patients and 4,148 factors. Because we are using statistical learning, we are able to consider significantly more features; this constitutes over 6000 times the amount of data used for the original CHADS_2 study. In our experiments we compared the stroke prediction performance of BRL to CHADS_2 and CHA_2DS_2-VASc, as well as to a collection of state-of-the-art machine learning algorithms: C5.0 (Quinlan, 1993), CART (Breiman et al, 1984), ℓ_1-regularized logistic regression, support vector machines (Vapnik, 1995), random forests (Breiman, 2001a), and Bayesian CART (Dension et al, 1998; Chipman et al, 1998). The balance of accuracy and interpretability obtained by BRL is not easy to

obtain through other means: None of the machine learning methods we tried could obtain both the same level of accuracy and the same level of interpretability.

3.1 Bayesian Rule Lists

The setting for BRL is multi-class classification, where the set of possible labels is $1, \ldots, L$. In the case of predicting stroke risk, there are two labels: stroke or no stroke. The training data are pairs $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ are the features of observation i, and y_i are the labels, $y_i \in \{1, \ldots, L\}$. We let $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$.

In Sections 3.1.1 and 3.1.2 we provide the association rule concepts and notation upon which the method is built. Section 3.1.3 introduces BRL by outlining the generative model. Sections 3.1.4 and 3.1.5 provide detailed descriptions of the prior and likelihood, and then Sections 3.1.6 and 3.1.7 describe sampling and posterior predictive distributions.

3.1.1 Bayesian Association Rules and Bayesian Decision Lists

An association rule $a \rightarrow b$ is an implication with an antecedent a and a consequent b. For the purposes of classification, the antecedent is an assertion about the feature vector x_i that is either true or false, for example, "$x_{i,1} = 1$ and $x_{i,2} = 0$." This antecedent contains two conditions, which we call the cardinality of the antecedent. The consequent b would typically be a predicted label y. A Bayesian association rule has a multinomial distribution over labels as its consequent rather than a single label:

$$a \rightarrow y \sim \text{Multinomial}(\theta).$$

The multinomial probability is then given a prior, leading to a *prior consequent distribution*:

$$\theta|\alpha \sim \text{Dirichlet}(\alpha)$$

Given observations $(\mathbf{x}, \mathbf{y})$ classified by this rule, we let $N_{\cdot,l}$ be the number of observations with label $y_i = l$, and $N = (N_{\cdot,1}, \ldots, N_{\cdot,L})$. We then obtain a *posterior*

consequent distribution:

$$\theta|\mathbf{x}, \mathbf{y}, \alpha \sim \text{Dirichlet}(\alpha + N).$$

The core of a Bayesian decision list is an ordered antecedent list $d = (a_1, \ldots, a_m)$. Let $N_{j,l}$ be the number of observations x_i that satisfy a_j but not any of $a_1, \ldots, a_{j-1}$, and that have label $y_i = l$. This is the number of observations to be classified by antecedent a_j that have label l. Let $N_{0,l}$ be the number of observations that do not satisfy any of $a_1, \ldots, a_m$, and that have label l. Let $\mathbf{N}_j = (N_{j,1}, \ldots, N_{j,L})$ and $\mathbf{N} = (\mathbf{N}_0, \ldots, \mathbf{N}_m)$.

A Bayesian decision list $D = (d, \alpha, \mathbf{N})$ is an ordered list of antecedents together with their posterior consequent distributions. The posterior consequent distributions are obtained by excluding data that have satisfied an earlier antecedent in the list. A Bayesian decision list then takes the form

if a_1 **then** $y \sim \text{Multinomial}(\theta_1)$, $\theta_1 \sim \text{Dirichlet}(\alpha + \mathbf{N}_1)$
else if a_2 **then** $y \sim \text{Multinomial}(\theta_2)$, $\theta_2 \sim \text{Dirichlet}(\alpha + \mathbf{N_2})$
$\vdots$
else if a_m **then** $y \sim \text{Multinomial}(\theta_m)$, $\theta_m \sim \text{Dirichlet}(\alpha + \mathbf{N}_m)$
else $y \sim \text{Multinomial}(\theta_0)$, $\theta_0 \sim \text{Dirichlet}(\alpha + \mathbf{N}_0)$.

Any observations that do not satisfy any of the antecedents in d are classified using the parameter θ_0, which we call the default rule parameter.

3.1.2 Antecedent Mining

We are interested in forming Bayesian decision lists whose antecedents are a subset of a pre-selected collection of antecedents. For data with binary or categorical features this can be done using frequent itemset mining, where itemsets are used as antecedents. In our experiments, the features were binary and we used the FP-Growth algorithm (Borgelt, 2005) for antecedent mining, which finds all itemsets that satisfy constraints on minimum support and maximum cardinality. This means each

antecedent applies to a sufficiently large amount of data and does not have too many conditions. For binary or categorical features the particular choice of the itemset mining algorithm is unimportant as the output is an exhaustive list of all itemsets satisfying the constraints. Other algorithms, such as Apriori or Eclat (Agrawal and Srikant, 1994; Zaki, 2000), would return an identical set of antecedents as FP-Growth if given the same minimum support and maximum cardinality constraints. Because the goal is to obtain decision lists with few rules and few conditions per rule, we need not include any itemsets that apply only to a small number of observations or have a large number of conditions. Thus frequent itemset mining allows us to significantly reduce the size of the feature space, compared to considering all possible combinations of features.

The frequent itemset mining that we do in our experiments produces only antecedents with sets of features, such as "diabetes and heart disease." Other techniques could be used for mining antecedents with negation, such as "not diabetes" (Wu et al, 2004). For data with continuous features, a variety of procedures exist for antecedent mining (Fayyad and Irani, 1993; Dougherty et al, 1995; Srikant and Agrawal, 1996). Alternatively, one can create categorical features using interpretable thresholds (e.g, ages 40-49, 50-59, etc.) or interpretable quantiles (e.g., quartiles) - we took this approach in our experiments.

We let $\mathcal{A}$ represent the complete, pre-mined collection of antecedents, and suppose that $\mathcal{A}$ contains $|\mathcal{A}|$ antecedents with up to C conditions in each antecedent.

3.1.3 Generative Model

We now sketch the generative model for the labels $\mathbf{y}$ from the observations $\mathbf{x}$ and antecedents $\mathcal{A}$. Define $a_{<j}$ as the antecedents before j in the rule list if there are any, *e.g.* $a_{<3} = \{a_1, a_2\}$. Similarly, let c_j be the cardinality of antecedent a_j, and $c_{<j}$ the cardinalities of the antecedents before j in the rule list. The generative model is then:

– Sample a decision list length $m \sim p(m|\lambda)$.

– Sample the default rule parameter $\theta_0 \sim \text{Dirichlet}(\alpha)$.

– For decision list rule $j = 1, \ldots, m$:

Sample the cardinality of antecedent a_j in d as $c_j \sim p(c_j | c_{<j}, \mathcal{A}, \eta)$.

Sample a_j of cardinality c_j from $p(a_j | a_{<j}, c_j, \mathcal{A})$.

Sample rule consequent parameter $\theta_j \sim \text{Dirichlet}(\alpha)$.

– For observation $i = 1, \ldots, n$:

Find the antecedent a_j in d that is the first that applies to x_i.

If no antecedents in d apply, set $j = 0$.

Sample $y_i \sim \text{Multinomial}(\theta_j)$.

Our goal is to sample from the posterior distribution over antecedent lists:

$$p(d | \mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta) \propto p(\mathbf{y} | \mathbf{x}, d, \alpha) p(d | \mathcal{A}, \lambda, \eta).$$

Given d, we can compute the posterior consequent distributions required to construct a Bayesian decision list as in Section 3.1.1. Three prior hyperparameters must be specified by the user: α, λ, and η. We will see in Sections 3.1.4 and 3.1.5 that these hyperparameters have natural interpretations that suggest the values to which they should be set.

3.1.4 The Hierarchical Prior for Antecedent Lists

Suppose the list of antecedents d has length m and antecedent cardinalities $c_1, \ldots, c_m$. The prior probability of d is defined hierarchically as

$$p(d | \mathcal{A}, \lambda, \eta) = p(m | \mathcal{A}, \lambda) \prod_{j=1}^{m} p(c_j | c_{<j}, \mathcal{A}, \eta) p(a_j | a_{<j}, c_j, \mathcal{A}). \tag{3.1}$$

We take the distributions for list length m and antecedent cardinality c_j to be Poisson with parameters λ and η respectively, with proper truncation to account for the finite number of antecedents in $\mathcal{A}$. Specifically, the distribution of m is Poisson truncated

at the total number of pre-selected antecedents:

$$p(m|\mathcal{A},\lambda) = \frac{(\lambda^m/m!)}{\sum_{j=0}^{|\mathcal{A}|}(\lambda^j/j!)}, \quad m = 0,\ldots,|\mathcal{A}|.$$

This truncated Poisson is a proper prior, and is a natural choice because of its simple parameterization. Specifically, this prior has the desirable property that when $|\mathcal{A}|$ is large compared to the desired size of the decision list, as will generally be the case when seeking an interpretable decision list, the prior expected decision list length $\mathbb{E}[m|\mathcal{A},\lambda]$ is approximately equal to λ. The prior hyperparameter λ can then be set to the prior belief of the list length required to model the data. A Poisson distribution is used in a similar way in the hierarchical prior of Wu et al (2007).

The distribution of c_j must be truncated at zero and at the maximum antecedent cardinality C. Additionally, any cardinalities that have been exhausted by point j in the decision list sampling must be excluded. Let $R_j(c_1,\ldots,c_j,\mathcal{A})$ be the set of antecedent cardinalities that are available after drawing antecedent j. For example, if $\mathcal{A}$ contains antecedents of size 1, 2, and 4, then we begin with $R_0(\mathcal{A}) = \{1,2,4\}$. If $\mathcal{A}$ contains only 2 rules of size 4 and $c_1 = c_2 = 4$, then $R_2(c_1,c_2,\mathcal{A}) = \{1,2\}$ as antecedents of size 4 have been exhausted. We now take $p(c_j|c_{<j},\mathcal{A},\eta)$ as Poisson truncated to remove values for which no rules are available with that cardinality:

$$p(c_j|c_{<j},\mathcal{A},\eta) = \frac{(\eta^{c_j}/c_j!)}{\sum_{k\in R_{j-1}(c_{<j},\mathcal{A})}(\eta^k/k!)}, \quad c_j \in R_{j-1}(c_{<j},\mathcal{A}).$$

If the number of rules of different sizes is large compared to λ, and η is small compared to C, the prior expected average antecedent cardinality is close to η. Thus η can be set to the prior belief of the antecedent cardinality required to model the data.

Once the antecedent cardinality c_j has been selected, the antecedent a_j must be sampled from all available antecedents in $\mathcal{A}$ of size c_j. Here, we use a uniform distribution over antecedents in $\mathcal{A}$ of size c_j, excluding those in $a_{<j}$:

$$p(a_j|a_{<j},c_j,\mathcal{A}) \propto 1, \quad a_j \in \{a \in \mathcal{A} \setminus a_{<j} : |a| = c_j\}. \tag{3.2}$$

It is straightforward to sample an ordered antecedent list d from the prior by following the generative model, using the provided distributions.

3.1.5 The Likelihood Function

The likelihood function follows from the generative model. Let $\theta = (\theta_0, \ldots, \theta_m)$ be the consequent parameters corresponding to each antecedent in d, together with the default rule parameter θ_0. Then, the likelihood is the product of the multinomial probability mass functions for the observed label counts at each rule:

$$p(\mathbf{y}|\mathbf{x}, d, \theta) = \prod_{j:\sum_l N_{j,l}>0} \text{Multinomial}(\mathbf{N}_j|\theta_j),$$

with

$$\theta_j \sim \text{Dirichlet}(\alpha).$$

We can marginalize over θ_j in each multinomial distribution in the above product, obtaining, through the standard derivation of the Dirichlet-multinomial distribution,

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}, d, \alpha) &= \prod_{j=0}^{m} \frac{\Gamma(\sum_{l=1}^{L} \alpha_l)}{\Gamma(\sum_{l=1}^{L} N_{j,l} + \alpha_l)} \times \prod_{l=1}^{L} \frac{\Gamma(N_{j,l} + \alpha_l)}{\Gamma(\alpha_l)} \\ &\propto \prod_{j=0}^{m} \frac{\prod_{l=1}^{L} \Gamma(N_{j,l} + \alpha_l)}{\Gamma(\sum_{l=1}^{L} N_{j,l} + \alpha_l)}. \end{aligned}$$

The prior hyperparameter α has the usual Bayesian interpretation of pseudo-counts. In our experiments, we set $\alpha_l = 1$ for all l, producing a uniform prior. Other approaches for setting prior hyperparameters such as empirical Bayes are also applicable.

3.1.6 Markov Chain Monte Carlo Sampling

We do Metropolis-Hastings sampling of d, generating the proposed d^* from the current d^t using one of three options: 1) Move an antecedent in d^t to a different position in the list. 2) Add an antecedent from $\mathcal{A}$ that is not currently in d^t into the list. 3)

Remove an antecedent from d^t. Which antecedents to adjust and their new positions are chosen uniformly at random at each step. The option to move, add, or remove is also chosen uniformly. The probabilities for the proposal distribution $Q(d^*|d^t)$ depend on the size of the antecedent list, the number of pre-mined antecedents, and whether the proposal is a move, addition, or removal. For the uniform distribution that we used, the proposal probabilities for a d^* produced by one of the three proposal types is:

$$Q(d^*|d^t, \mathcal{A}) = \begin{cases} \frac{1}{(|d^t|)(|d^t|-1)} & \text{if move proposal,} \\ \frac{1}{(|\mathcal{A}|-|d^t|)(|d^t|+1)} & \text{if add proposal,} \\ \frac{1}{|d^t|} & \text{if remove proposal.} \end{cases}$$

To explain these probabilities, if there is a move proposal, we consider the number of possible antecedents to move and the number of possible positions for it; if there is an add proposal, we consider the number of possible antecedents to add to the list and the number of positions to place a new antecedent; for remove proposals we consider the number of possible antecedents to remove. This sampling algorithm is related to those used for Bayesian Decision Tree models (Chipman et al, 2002, 1998; Wu et al, 2007) and to methods for exploring tree spaces (Madigan et al, 2011).

For every MCMC run, we ran 3 chains, each initialized independently from a random sample from the prior. We discarded the first half of simulations as burn-in, and then assessed chain convergence using the Gelman-Rubin convergence diagnostic applied to the log posterior density (Gelman and Rubin, 1992). We considered chains to have converged when the diagnostic $\hat{R} < 1.05$.

3.1.7 The Posterior Predictive Distribution and Point Estimates

Given the posterior $p(d|\mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta)$, we consider estimating the label $\tilde{y}$ of a new observation $\tilde{x}$ using either a point estimate (a single Bayesian decision list) or the posterior predictive distribution. Given a point estimate of the antecedent list d, we

have that

$$p(\tilde{y} = l|\tilde{x}, d, \mathbf{x}, \mathbf{y}, \alpha) = \int_{\theta} \theta_l p(\theta|\tilde{x}, d, \mathbf{x}, \mathbf{y}, \alpha) d\theta$$
$$= \mathbb{E}[\theta_l|\tilde{x}, d, \mathbf{x}, \mathbf{y}, \alpha].$$

Let $j(d, \tilde{x})$ be the index of the first antecedent in d that applies to $\tilde{x}$. The posterior consequent distribution is

$$\theta|\tilde{x}, d, \mathbf{x}, \mathbf{y}, \alpha \sim \text{Dirichlet}\left(\alpha + \mathbf{N}_{j(d,\tilde{x})}\right). \tag{3.3}$$

Thus,

$$p(\tilde{y} = l|\tilde{x}, d, \mathbf{x}, \mathbf{y}, \alpha) = \frac{\alpha_l + N_{j(d,\tilde{x}),l}}{\sum_{k=1}^{L}\left(\alpha_k + N_{j(d,\tilde{x}),k}\right)}.$$

Additionally, (3.3) allows for the estimation of 95% credible intervals using the Dirichlet distribution function.

The posterior mean is often a good choice for a point estimate, but the interpretation of "mean" here is not clear since the posterior is a distribution over antecedent lists. We thus look for an antecedent list whose statistics are similar to the posterior mean statistics. Specifically, we are interested in finding a point estimate $\hat{d}$ whose length m and whose average antecedent cardinality $\bar{c} = \frac{1}{m}\sum_{j=1}^{m} c_j$ are close to the posterior mean list length and average cardinality. Let $\bar{m}$ be the posterior mean decision list length and $\bar{\bar{c}}$ the posterior mean average antecedent cardinality, as estimated from the MCMC samples. Then, we choose our point estimate $\hat{d}$ as the list with the highest posterior probability among all samples with $m \in \{\lfloor \bar{m} \rfloor, \lceil \bar{m} \rceil\}$ and $\bar{c} \in [\lfloor \bar{\bar{c}} \rfloor, \lceil \bar{\bar{c}} \rceil]$. We call this point estimate *BRL-point*.

Another possible point estimate is the decision list with the highest posterior probability - the maximum *a posteriori* estimate. Given two list lengths there are many more possible lists of the longer length than of the shorter length, so prior probabilities in (3.1) are generally higher for shorter lists. The maximum *a posteriori* estimate might yield a list that is much shorter than the posterior mean decision list length, so we prefer the BRL-point.

In addition to point estimates, we can use the entire posterior $p(d|\mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta)$ to estimate y. The posterior predictive distribution for y is

$$\begin{aligned} p(y = l|x, \mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta) &= \sum_{d \in \mathbf{D}} p(y = l|x, d, \mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha) p(d|\mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta) \\ &= \sum_{d \in \mathbf{D}} \frac{\alpha_l + N_{j(d,x),l}}{\sum_{k=1}^{L} \left(\alpha_k + N_{j(d,x),k}\right)} p(d|\mathbf{x}, \mathbf{y}, \mathcal{A}, \alpha, \lambda, \eta) \end{aligned}$$

where $\mathbf{D}$ is the set of all ordered subsets of $\mathcal{A}$. The posterior samples obtained by MCMC simulation, after burn-in, can be used to approximate this sum. We call the classifier that uses the full collection of posterior samples *BRL-post*. Using the entire posterior distribution to make a prediction means the classifier is no longer interpretable. One could, however, use the posterior predictive distribution to classify, and then provide several point estimates from the posterior to the user as example explanations for the prediction.

3.2 Simulation Studies

We use simulation studies and a deterministic dataset to show that when data are generated by a decision list model, the BRL method is able to recover the true decision list.

3.2.1 Simulated Data Sets

Given observations with arbitrary features and a collection of rules on those features, we can construct a binary matrix where the rows represent observations and the columns represent rules, and the entry is 1 if the rule applies to that observation and 0 otherwise. We need only simulate this binary matrix to represent the observations without losing generality. For our simulations, we generated independent binary rule sets with 100 rules by setting each feature value to 1 independently with probability 1/2.

We generated a random decision list of size 5 by selecting 5 rules at random, and

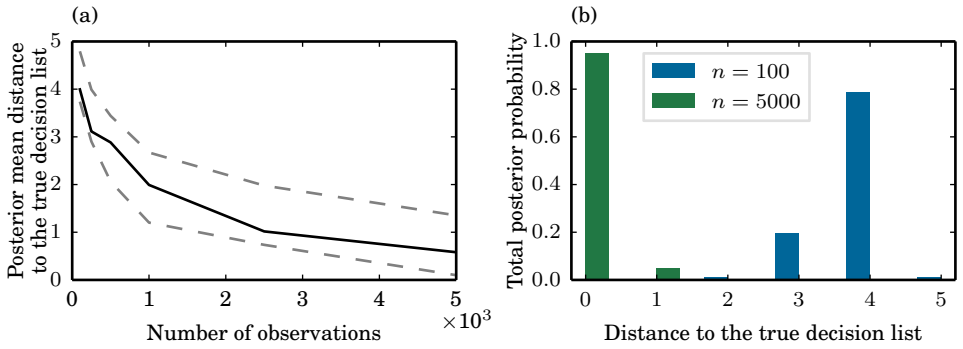

Figure 3-2: (a) Average Levenshtein distance from posterior samples to the true decision list, for differing numbers of observations. The black solid line indicates the median value across the 100 simulated datasets of each size, and the gray dashed lines indicate the first and third quartiles. (b) The proportion of posterior samples with the specified distance to the true decision list, for a randomly selected simulation with $n = 100$ observations and a randomly selected simulation with $n = 5000$.

adding the default rule. Each rule in the decision list was assigned a consequent distribution over labels using a random draw from the Beta$(1/2, 1/2)$ distribution, which ensures that the rules are informative about labels. Labels were then assigned to each observation using the decision list: For each observation, the label was taken as a draw from the label distribution corresponding to the first rule that applied to that observation.

For each number of observations $N \in \{100, 250, 500, 1000, 2500, 5000\}$, we generated 100 independent data sets $(\mathbf{x}, \mathbf{y})$, for a total of 600 simulated datasets. We did MCMC sampling with three chains as described in Section 3.1 for each dataset. For all datasets, 20,000 samples were sufficient for the chains to converge.

To appropriately visualize the posterior distribution, we binned the posterior antecedent lists according to their distance from the true antecedent list, using the Levenshtein string edit distance (Levenshtein, 1966) to measure the distance between two antecedent lists. This metric measures the minimum number of antecedent substitutions, additions, or removals to transform one decision list into the other. The results of the simulations are given in Fig. 3-2. Figure 3-2(a) shows that as the number of observations increases, the posterior mass concentrates on the true decision list. Figure 3-2(b) illustrates this concentration with two choices of the distribution

of posterior distances to the true decision list, for n small and for n large.

3.2.2 A Deterministic Problem

We fit BRL to the Tic-Tac-Toe Endgame dataset from the UCI Machine Learning Repository (Bache and Lichman, 2013) of benchmark datasets. The Tic-Tac-Toe Endgame dataset provides all possible end board configurations for the game Tic-Tac-Toe, with the task of determining if player "X" won or not. The dataset is deterministic, with exactly 8 ways that player "X" can win, each one of the 8 ways to get 3 "X"'s in a row on a 3x3 grid. We split the dataset into 5 folds and did cross-validation to estimate test accuracy. For each fold of cross-validation, we fit BRL with prior hyperparameters $\lambda = 8$ and $\eta = 3$, and the point estimate decision list contained the 8 ways to win and thus achieved perfect accuracy. In Table 3.1, we compare accuracy on the test set with C5.0, CART, ℓ_1-regularized logistic regression (ℓ_1-LR), RBF kernel support vector machines (SVM), random forests (RF), and Bayesian CART (BCART). For SVM, we used the LIBSVM (Chang and Lin, 2011) implementation with a radial basis function kernel. We selected the slack parameter C_{SVM} and the kernel parameter γ using a grid search over the ranges $C_{\text{SVM}} \in \{2^{-2}, 2^0, \ldots, 2^6\}$ and $\gamma \in \{2^{-6}, 2^{-4}, \ldots, 2^2\}$. We chose the set of parameters with the best 3-fold cross-validation performance using LIBSVM's built-in cross-validation routine. For C5.0 we used the R library "C50" with default settings. For CART we used the R library "rpart" with default parameters and pruned using the complexity parameter that minimized cross-validation error. For logistic regression we used the LIBLINEAR (Fan et al, 2008) implementation of logistic regression with ℓ_1 regularization. We selected the regularization parameter C_{LR} from $\{2^{-6}, 2^{-4}, \ldots, 2^6\}$ as that with the best 3-fold cross-validation performance, using LIBLINEAR's built-in cross-validation routine. Random forests was done using the R library "randomForest." The optimal value for the parameter "mtry" was found using "tuneRF," with its default 50 trees. The optimal "mtry" was then used to fit a random forests model with 500 trees, the library default. Bayesian CART was run using the R library "tgp," function "bcart" with default settings. None of these other methods were able to achieve perfect accu-

Table 3.1: Mean classification accuracy in the top row, with standard deviation in the second row, for machine learning algorithms using 5 folds of cross-validation on the Tic-Tac-Toe Endgame dataset.

	BRL	C5.0	CART	ℓ_1-LR	SVM	RF	BCART
Mean accuracy	1.00	0.94	0.90	0.98	0.99	0.99	0.71
Standard deviation	0.00	0.01	0.04	0.01	0.01	0.01	0.04

racy. Decision trees in particular are capable of providing a perfect classifier for this problem, but the greedy learning done by C5.0 and CART did not find the perfect classifier.

3.3 Stroke Prediction

We used Bayesian Rule Lists to derive a stroke prediction model using the MarketScan Medicaid Multi-State Database (MDCD). MDCD contains administrative claims data for 11.1 million Medicaid enrollees from multiple states. This database forms part of the suite of databases from the Innovation in Medical Evidence Development and Surveillance (IMEDS, `http://imeds.reaganudall.org/`) program that have been mapped to a common data model (Stang et al, 2010).

We extracted every patient in the MDCD database with a diagnosis of atrial fibrillation, one year of observation time prior to the diagnosis, and one year of observation time following the diagnosis (n=12,586). Of these, 1,786 (14%) had a stroke within a year of the atrial fibrillation diagnosis.

As candidate predictors, we considered all drugs and all conditions. Specifically, for every drug and condition, we created a binary predictor variable indicating the presence or absence of the drug or condition in the full longitudinal record prior to the atrial fibrillation diagnosis. These totaled 4,146 unique medications and conditions. We included features for age and gender. Specifically, we used the natural values of 50, 60, 70, and 80 years of age as split points, and for each split point introduced a pair of binary variables indicating if age was less than or greater than the split point. Considering both patients and features, here we apply our method to a dataset that is over 6000 times larger than that originally used to develop the CHADS_2 score (which

had n=1,733 and considered 5 features).

We did five folds of cross-validation. For each fold, we pre-mined the collection of possible antecedents using frequent itemset mining with a minimum support threshold of 10% and a maximum cardinality of 2. The total number of antecedents used ranged from 2162 to 2240 across the folds. We set the antecedent list prior hyperparameters λ and η to 3 and 1 respectively, to obtain a Bayesian decision list of similar complexity to the $CHADS_2$ score. For each fold, we evaluated the performance of the BRL point estimate by constructing a receiver operating characteristic (ROC) curve and measuring area under the curve (AUC) for each fold.

In Fig. 3-3 we show the BRL point estimate recovered from one of the folds. The list indicates that past history of stroke reveals a lot about the vulnerability toward future stroke. In particular, the first half of the decision list focuses on a history of stroke, in order of severity. Hemiplegia, the paralysis of an entire side of the body, is often a result of a severe stroke or brain injury. Cerebrovascular disorder indicates a prior stroke, and transient ischaemic attacks are generally referred to as "mini-strokes." The second half of the decision list includes age factors and vascular disease, which are known risk factors and are included in the CHA_2DS_2-VASc score. The BRL-point lists that we obtained in the 5 folds of cross-validation were all of length 7, a similar complexity to the $CHADS_2$ and CHA_2DS_2-VASc scores which use 5 and 8 features respectively.

We found that there was significant overlap in the antecedents in the point estimates across the five folds. This suggests that the model may be more stable in practice than decision trees, which are notorious for producing entirely different models after small changes to the training set (Breiman, 1996a,b).

In Fig. 3-4 we give ROC curves for all 5 folds for BRL-point, $CHADS_2$, and CHA_2DS_2-VASc, and in Table 3.2 we report mean AUC across the folds. These results show that with complexity and interpretability similar to $CHADS_2$, the BRL point estimate decision lists performed significantly better at stroke prediction than both $CHADS_2$ and CHA_2DS_2-VASc. Interestingly, we also found that $CHADS_2$ outperformed CHA_2DS_2-VASc despite CHA_2DS_2-VASc being an extension of $CHADS_2$.

if hemiplegia **and** age>60 **then** *stroke risk* 58.9% (53.8% - 63.8%)
else if cerebrovascular disorder **then** *stroke risk* 47.8% (44.8% - 50.7%)
else if transient ischaemic attack **then** *stroke risk* 23.8% (19.5% - 28.4%)
else if occlusion and stenosis of carotid artery without infarction **then** *stroke risk* 15.8% (12.2% - 19.6%)
else if altered state of consciousness **and** age>60 **then** *stroke risk* 16.0% (12.2% - 20.2%)
else if age≤70 **then** *stroke risk* 4.6% (3.9% - 5.4%)
else *stroke risk* 8.7% (7.9% - 9.6%)

Figure 3-3: Decision list for determining 1-year stroke risk following diagnosis of atrial fibrillation from patient medical history. The risk given is the mean of the posterior consequent distribution, and in parentheses is the 95% credible interval.

This is likely because the model for the CHA_2DS_2-VASc score, in which risk factors are added linearly, is a poor model of actual stroke risk. For instance, the stroke risks estimated by CHA_2DS_2-VASc are not a monotonic function of score. Within the original CHA_2DS_2-VASc calibration study, (Lip et al, 2010b) estimate a stroke risk of 9.6% with a CHA_2DS_2-VASc score of 7, and a 6.7% risk with a score of 8. The indication that more stroke risk factors can correspond to a lower stroke risk suggests that the CHA_2DS_2-VASc model may be misspecified, and highlights the difficulty in constructing these interpretable models manually.

The results in Table 3.2 give the AUC for BRL, $CHADS_2$, CHA_2DS_2-VASc, along with the same collection of machine learning algorithms used in Section 3.2.2. The decision tree algorithms CART and C5.0, the only other interpretable classifiers, were outperformed even by $CHADS_2$. The BRL-point performance was comparable to that of SVM, and not substantially worse than ℓ_1 logistic regression and random forests. Using the full posterior, BRL-post matched random forests for the best performing method.

All of the methods were applied to the data on the same, single Amazon Web Services virtual core with a processor speed of approximately 2.5Ghz and 4GB of memory. Bayesian CART was unable to fit the data since it ran out of memory, and so it is not included in Table 3.2.

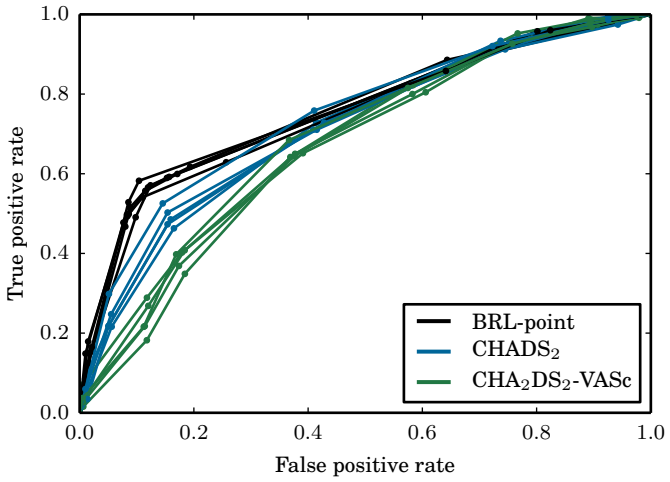

Figure 3-4: ROC curves for stroke prediction on the MDCD database for each of 5 folds of cross-validation, for the BRL point estimate, $CHADS_2$, and CHA_2DS_2-VASc.

Table 3.2: Mean, and in parentheses standard deviation, of AUC and training time across 5 folds of cross-validation for stroke prediction. Note that the $CHADS_2$ and CHA_2DS_2-VASc models are fixed, so no training time is reported.

	AUC	Training time (mins)
BRL-point	0.756 (0.007)	21.48 (6.78)
$CHADS_2$	0.721 (0.014)	no training
CHA_2DS_2-VASc	0.677 (0.007)	no training
CART	0.704 (0.010)	12.62 (0.09)
C5.0	0.704 (0.011)	2.56 (0.27)
ℓ_1 logistic regression	0.767 (0.010)	0.05 (0.00)
SVM	0.753 (0.014)	302.89 (8.28)
Random forests	0.774 (0.013)	698.56 (59.66)
BRL-post	0.775 (0.015)	21.48 (6.78)

The BRL MCMC chains were simulated until convergence, which required 50,000 iterations for 4 of the 5 folds, and 100,000 for the fifth. The three chains for each fold were simulated in serial, and the total CPU time required per fold is given in Table 3.2, together with the CPU times required for training the comparison algorithms on the same processor. Table 3.2 shows that the BRL MCMC simulation was more than ten times faster than training SVM, and more than thirty times faster than training random forests, using standard implementations of these methods.

3.3.1 Additional Experiments

We further investigated the properties and performance of the BRL by applying it to two subsets of the data, female patients only and male patients only. The female dataset contained 8368 observations, and the number of pre-mined antecedents in each of 5 folds ranged from 1982 to 2197. The male dataset contained 4218 observations, and the number of pre-mined antecedents in each of 5 folds ranged from 1629 to 1709. BRL MCMC simulations and comparison algorithm training were done on the same processor as the full experiment. The AUC and training time across five folds for each of the datasets is given in Table 3.3.

The BRL point estimate again outperformed the other interpretable models, namely $CHADS_2$, CHA_2DS_2-VASc, CART, and C5.0. Furthermore, the BRL-post performance matched that of random forests for the best performing method. As before, BRL MCMC simulation required significantly less time than SVM or random forests training.

3.4 Related Work and Discussion

Most widely used medical scoring systems are designed to be interpretable, but are not necessarily optimized for accuracy, and generally are derived from few factors. The Thrombolysis In Myocardial Infarction (TIMI) Score (Antman et al, 2000), Apache II score for infant mortality in the ICU (Knaus et al, 1985), the CURB-65 score for predicting mortality in community-acquired pneumonia (Lim et al, 2003), and the

Table 3.3: Mean, and in parentheses standard deviation, of AUC and training time (mins) across 5 folds of cross-validation for stroke prediction

	Female patients		Male patients	
	AUC	Training time	AUC	Training time
BRL-point	0.747 (0.028)	9.12 (4.70)	0.738 (0.027)	6.25 (3.70)
$CHADS_2$	0.717 (0.018)	no training	0.730 (0.035)	no training
CHA_2DS_2-VASc	0.671 (0.021)	no training	0.701 (0.030)	no training
CART	0.704 (0.024)	7.41 (0.14)	0.581 (0.111)	2.69 (0.04)
C5.0	0.707 (0.023)	1.30 (0.09)	0.539 (0.086)	0.55 (0.01)
ℓ_1 LR	0.755 (0.025)	0.04 (0.00)	0.739 (0.036)	0.01 (0.00)
SVM	0.739 (0.021)	56.00 (0.73)	0.753 (0.035)	11.05 (0.18)
Random forests	0.764 (0.022)	389.28 (33.07)	0.773 (0.029)	116.98 (12.12)
BRL-post	0.765 (0.025)	9.12 (4.70)	0.778 (0.018)	6.25 (3.70)

$CHADS_2$ score (Gage et al, 2001) are examples of interpretable predictive models that are very widely used. Each of these scoring systems involves very few calculations, and could be computed by hand during a doctor's visit. In the construction of each of these models, heuristics were used to design the features and coefficients for the model; none of these models was fully learned from data.

In contrast with these hand-designed interpretable medical scoring systems, recent advances in the collection and storing of medical data present unprecedented opportunities to develop powerful models that can predict a wide variety of outcomes (Shmueli, 2010). The front-end user interface of medical risk assessment tools are increasingly available online (e.g., `http://www.r-calc.com`). At the end of the assessment, a patient may be told he or she has a high risk for a particular outcome but without understanding why the predicted risk is high, particularly if many pieces of information were used to make the prediction.

In general, humans can handle only a handful of cognitive entities at once (Miller, 1956; Jennings et al, 1982). It has long since been hypothesized that simple models predict well, both in the machine learning literature (Holte, 1993), and in the psychology literature (Dawes, 1979). The related concepts of explanation and comprehensibility in statistical modeling have been explored in many past works (Bratko, 1997; Madigan et al, 1997; Giraud-Carrier, 1998; Rüping, 2006; Nowozin et al, 2007; Huysmans et al, 2011; Vellido et al, 2012; Freitas, 2014, for example).

Decision lists have the same form as models used in the expert systems literature from the 1970's and 1980's (Leondes, 2002), which were among the first successful types of artificial intelligence. The knowledge base of an expert system is composed of natural language statements that are *if... then...* rules. Decision lists are a type of associative classifier, meaning that the list is formed from association rules. In the past, associative classifiers have been constructed from heuristic greedy sorting mechanisms (Rivest, 1987; Liu et al, 1998; Li et al, 2001; Yin and Han, 2003; Marchand and Sokolova, 2005; Yi and Hüllermeier, 2005; Rudin et al, 2013). Some of these sorting mechanisms work provably well in special cases, for instance when the decision problem is easy and the classes are easy to separate, but are not optimized to handle more general problems. Sometimes associative classifiers are formed by averaging several rules together, but the resulting classifier is not generally interpretable (Friedman and Popescu, 2008; Meinshausen, 2010).

Decision trees are closely related to decision lists, and are in some sense equivalent: any decision tree can be expressed as a decision list, and any decision list is a one-sided decision tree. Decision trees are almost always constructed greedily from the top down, and then pruned heuristically upwards and cross-validated to ensure accuracy. Because the trees are not fully optimized, if the top of the decision tree happened to have been chosen badly at the start of the procedure, it could cause problems with both accuracy and interpretability. Bayesian decision trees (Chipman et al, 1998; Dension et al, 1998; Chipman et al, 2002) use Markov chain Monte Carlo (MCMC) to sample from a posterior distribution over trees. Since they were first proposed, several improvements and extensions have been made in both sampling methods and model structure (Wu et al, 2007; Chipman et al, 2010; Taddy et al, 2011). The space of decision lists using pre-mined rules is significantly smaller than the space of decision trees, making it substantially easier to obtain MCMC convergence, and to avoid the pitfalls of local optima. Moreover, rule mining allows for the rules to be individually powerful. Constructing a single decision tree is extremely fast, but sampling over the space of decision trees is extremely difficult (unless one is satisfied with local maxima). To contrast this with our approach: the rule mining step is extremely fast,

yet sampling over the space of decision lists is very practical.

Interpretable models are generally not unique (stable), in the sense that there many be many equally good models, and it is not clear in advance which one will be returned by the algorithm. For most problems, the space of high quality predictive models is fairly large (called the "Rashomon Effect" Breiman, 2001b), so we cannot expect uniqueness. In practice, as we showed, the rule lists across test folds were very similar, but if one desires stability to small perturbations in the data generally, we recommend using the full posterior rather than a point estimate.

This work is related to the Hierarchical Association Rule Model (HARM), a Bayesian model that uses rules (McCormick et al, 2012). HARM estimates the conditional probabilities of each rule jointly in a conservative way. Each rule acts as a separate predictive model, so HARM does not explicitly aim to learn an ordering of rules. The work of Wang and Rudin (2015) provides an extension to BRL whereby the probabilities for the rules are monotonically decreasing down the list. Another possible extension is to give preference to particular antecedents, such as known risk factors. This sort of preference could be expressed in the antecedent prior distribution in (3.2).

3.5 Conclusion

We are working under the hypothesis that many real datasets permit predictive models that can be surprisingly small. This was hypothesized over a decade ago (Holte, 1993), however, we now are starting to have the computational tools to truly test this hypothesis. The BRL method introduced in this work aims to hit the "sweet spot" between predictive accuracy, interpretability, and tractability.

Interpretable models have the benefits of being both concise and convincing. A small set of trustworthy rules can be the key to communicating with domain experts and to allow machine learning algorithms to be more widely implemented and trusted. In practice, a preliminary interpretable model can help domain experts to troubleshoot the inner workings of a complex model, in order to make it more ac-

curate and tailored to the domain. We demonstrated that interpretable models lend themselves to the domain of predictive medicine, and there is a much wider variety of domains in science, engineering, and industry, where these models would be a natural choice.

Chapter 4

Statistical Learning Theory and Association Rules

A strength of modern machine learning techniques is that they come with theoretical guarantees, often obtained through the framework of statistical learning theory. We now provide a theoretical guarantee of generalization for decision lists used for binary classification, as was done in the previous chapter.

In the binary classification problem, each data example $x \in \mathcal{X}$ receives a single label y that is one of two possible labels $\{+1, -1\}$. Suppose that we sample labeled examples $z = (x, y)$. Each labeled example z is chosen randomly (iid) from a fixed but unknown probability distribution $\mathcal{D}$ over examples and labels. Given a training set S of m labeled examples, we wish to construct a classifier that can assign the correct label to new, unlabeled examples.

Suppose we have a collection of rule antecedents $\mathcal{A}$. Each rule antecedent a is an assertion about the feature vector x that is either true or false. We do not need to concern ourselves with the nature of the feature space $\mathcal{X}$ as the classifier will only interact with the features through the rule antecedents in $\mathcal{A}$. We begin by defining a scoring function $g : \mathcal{A} \times \{-1, 1\} \to \mathbb{R}$ that assigns score $g(a, y)$ to the rule $a \to y$. The set of antecedents $\mathcal{A}$ can be any collection so long as every $x \in \mathcal{X}$ satisfies at least one $a \in \mathcal{A}$. This condition can be satisfied by including in $\mathcal{A}$ a "default" antecedent that is by definition satisfied by every feature vector. We define a *valid* scoring

function as one that produces no ties: $\forall a \in \mathcal{A}$, $g(a,1) \neq g(a,-1)$ and $\forall a_1, a_2 \in \mathcal{A}$, $\max_{y\in\{-1,1\}} g(a_1,y) \neq \max_{y\in\{-1,1\}} g(a_2,y)$. The validity requirement will be discussed later. Define G to be the class of all valid scoring functions. We now define a class of decision functions that use a valid scoring function $g \in G$ to provide a label to example x, $f_g : \mathcal{X} \to \{-1,1\}$. The decision function assigns the label corresponding to the highest scoring rule whose antecedent is satisfied by x. We denote the event that a is satisfied by x as $a(x) = 1$. Then, the decision function is

$$f_g(x) = \operatorname*{argmax}_{y\in\{-1,1\}} \max_{a\in\mathcal{A}:a(x)=1} g(a,y). \tag{4.1}$$

We call this classifier a "max-score association rule classifier" because it uses the association rule with the maximum score to perform the classification.

Decision lists are max-score association rule classifiers: g orders the association rules, and we make a classification with the first association rule in the list whose antecedent is satisfied by x. A short decision list is obtained by including a default rule a_0 such that $a_0(x) = 1$ for all x, and allowing $g(a_0, y)$ to be higher than any rules that should not be on the list.

Let $\mathcal{F}_{\text{maxscore}}$ be the class of all max-score association rule classifiers: $\mathcal{F}_{\text{maxscore}} := \{f_g : g \in G\}$. We will calculate the VC dimension of the class $\mathcal{F}_{\text{maxscore}}$. The VC dimension is defined as the size of the largest set of examples to which arbitrary labels can be assigned using some $f_g \in \mathcal{F}_{\text{maxscore}}$, a process known as "shattering."

The argmax in (4.1) is unique because g is valid, thus there are no ties. If ties are allowed but broken randomly, arbitrary labels can be realized with some probability, for example by taking $g(a,y) = 0$ for all a and y. In this case the VC dimension can be considered to be infinite, which motivates our definition of a valid scoring function. This problem actually happens with any classification problem where function $f(x) = 0\ \forall x$ is within the hypothesis space, thereby allowing all points to sit on the decision boundary. Our definition of validity is equivalent to one in which ties are allowed but are broken deterministically using a pre-determined ordering on the rules. In practice, ties are generally broken in a deterministic way by the computer, so the inclusion of

the function $f = 0$ is not problematic.

The true risk of the max-score association rule classifier is the expected misclassification error, which we denote as

$$R(f_g) := \mathbb{E}_{(x,y)\sim\mathcal{D}} \mathbb{1}_{[f_g(x)\neq y]}.$$

The empirical risk is the average misclassification error over a training set of m examples:

$$\hat{R}(f_g) := \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{[f_g(x_i)\neq y_i]}.$$

The main result of this chapter is the following theorem, which indicates that generalization depends on the number of association rules.

Theorem 1. *The VC dimension h of the set of max-score classifiers is bounded by the number of antecedents:*

$$h \leq |\mathcal{A}|.$$

If $\mathcal{X} \subseteq 2^{\mathcal{Z}}$ for some finite set $\mathcal{Z}$ and $\mathcal{A}$ is a collection of unique itemsets $a \subseteq \mathcal{Z}$ then

$$h = |\mathcal{A}|.$$

Proof. First we show that $h \leq |\mathcal{A}|$. To do this, we must show that for any collection of examples $x_1, \ldots, x_N$, $N > |\mathcal{A}|$, there exists a corresponding set of labels $y_1, \ldots, y_N$ that cannot be realized by any max-score association rule classifier. For each x_i, we introduce a binary vector $\bar{x}_i$ of length $|\mathcal{A}|$, where each element corresponds to an $a \in \mathcal{A}$. The element of $\bar{x}_i$ corresponding to a is 1 if a is satisfied by x_i and 0 otherwise. Each vector $\bar{x}_i$ is an element of $\mathbb{R}^{|\mathcal{A}|}$, so the collection of vectors $\bar{x}_1, \ldots, \bar{x}_N$ must be linearly dependent if $N > |\mathcal{A}|$. By linear dependence and the fact that every $\bar{x}_i$ is non-zero and non-negative, there must exist coefficients c_i and disjoint, non-empty sets M_0 and M_1 such that $M_0 \cup M_1 = \{1, \ldots, N\}$ and

$$\sum_{i\in M_0} c_i \bar{x}_i = \sum_{i\in M_1} c_i \bar{x}_i,\ c_i > 0. \tag{4.2}$$

Define $\mathcal{A}_0 = \{a \in \mathcal{A} : a(x_i) = 1 \text{ for some } i \in M_0\}$ and similarly $\mathcal{A}_1 = \{a \in \mathcal{A} : a(x_i) \text{ for some } i \in M_1\}$. If $a(x_i) = 1$ for some a and some $i \in M_0$, then the corresponding element of $\bar{x}_i$ will be 1 and the same element in the left part of (4.2) will be strictly positive. Then, (4.2) implies that $a(x_j) = 1$ for some $j \in M_1$. Thus, $\mathcal{A}_0 \subseteq \mathcal{A}_1$. The reverse argument shows $\mathcal{A}_1 \subseteq \mathcal{A}_0$, so $\mathcal{A}_0 = \mathcal{A}_1$. However, for any valid g there exists a satisfied rule with maximum score, whose antecedent is

$$\begin{aligned} a^* &= \arg\max_{a \in \mathcal{A}_0} \max_{y \in \{-1,1\}} g(a, y) \\ &= \arg\max_{a \in \mathcal{A}_1} \max_{y \in \{-1,1\}} g(a, y). \end{aligned}$$

Any x_i that satisfies a^* will get label $y^* = \arg\max_{y \in \{-1,1\}} g(a^*, y)$. Thus for at least one $i \in M_0$ and at least one $j \in M_1$, $f_g(x_i) = y^* = f_g(x_j)$. Set $y_i = -1$ for all $i \in M_0$ and $y_j = 1$ for all $j \in M_1$ and this set of labels cannot be realized, which shows that $h \leq |\mathcal{A}|$.

For the second part of the theorem, we now show that this upper bound can be achieved by providing a set of $|\mathcal{A}|$ examples and finding elements of $\mathcal{F}_{\text{maxscore}}$ that can assign them arbitrary labels. We list the elements of $\mathcal{A}$ as $a_1, \ldots, a_{|\mathcal{A}|}$ - for this part of the proof, each of these is an itemset (a subset of a ground set $\mathcal{Z}$). We construct our set of examples by taking $x_i = a_i$, for $i = 1, \ldots, |\mathcal{A}|$. Thus each example is one of the itemsets from $\mathcal{A}$. Some of the itemsets in $\mathcal{A}$ might be larger than others, and in fact some itemsets might contain others. We will place the elements of $\mathcal{A}$ in order of increasing size. The possible sizes of itemsets in $\mathcal{A}$ are $0, \ldots, |\mathcal{Z}|$. We arrange the elements of $\mathcal{A}$ into sets based on their sizes: $S_k = \{i : |a_i| = l_k\}$, $k = 0, 1, \ldots, |\mathcal{Z}|$. We are now ready to construct a classifier f_g that can produce arbitrary labels $\{y_i\}_i$ for these examples $\{x_i\}$.

If $\varnothing \in \mathcal{A}$, then there will be some i such that $x_i = \varnothing$. For this i, we set $g(a_i, y_i) = c_0$, any positive number, and $g(a_i, -y_i) = 0$, thereby ensuring that that example will be correctly labeled. Then, for all $i \in S_1$, we set $g(a_i, y_i) = c_1$, any positive number, and $g(a_i, -y_i) = 0$. Thus, for the corresponding x_i, $f_g(x_i) = y_i$. Similarly, for all $i \in S_2$, we set $g(a_i, y_i) = c_2$, $c_2 > c_1$, and $g(a_i, -y_i) = 0$. For any $i \in S_2$, it may be

that there exists some $j \in S_1$ such that $a_j \subset x_i$. However, because $c_2 > c_1$, the rule with the maximum score will be "$a_i \to y_i$" and x_i is labeled as desired. In general, for any $i \in S_k$, we set $g(a_i, y_i) = c_k$, where $c_{k-1} < c_k < c_{k+1}$ and $g(a_i, -y_i) = 0$ to get $f_g(x_i) = y_i$. Because this set of $|\mathcal{A}|$ examples can be arbitrarily labeled using elements of $\mathcal{F}_{\text{maxscore}}$, we have $h \geq |\mathcal{A}|$, which combined with the previous result shows that in this case $h = |\mathcal{A}|$. □

From this theorem, classical results such as those of Vapnik (1999, Equations 20 and 21) can be directly applied to obtain a generalization bound:

Corollary 1. *(Uniform Generalization Bound for Classification)*
With probability at least $1-\delta$ *the following holds simultaneously for all* $f_g \in \mathcal{F}_{\text{maxscore}}$*:*

$$R(f_g) \leq \hat{R}(f_g) + \frac{\epsilon}{2}\left(1 + \sqrt{1 + \frac{4\hat{R}(f_g)}{\epsilon}}\right), \text{ where } \epsilon = 4\frac{|\mathcal{A}|\left(\ln\frac{2m}{|\mathcal{A}|} + 1\right) - \ln\delta}{m}.$$

The result of Theorem 1, and the corresponding generalization bound, holds generally, regardless of the details of the scoring function. The ranking could be done using a simple confidence-based algorithm, such as by Rudin et al (2011, 2013), or it could be done using more sophisticated methods such as in Chapter 3.

We can also use a standard argument involving Hoeffding's inequality and the union bound over elements of $\mathcal{F}_{\text{maxscore}}$ to obtain that with probability at least $1-\delta$, the following holds for all $f_g \in \mathcal{F}_{\text{maxscore}}$:

$$R(f_g) \leq \hat{R}(f_g) + \sqrt{\frac{1}{2m}\left(\ln(2|\mathcal{F}_{\text{maxscore}}|) + \ln\frac{1}{\delta}\right)}.$$

The value of $|\mathcal{F}_{\text{maxscore}}|$ is at most $2^{|\mathcal{A}|}$. This is because there are $|\mathcal{A}|$ ways to determine $\max_{a\in\mathcal{A}, a\subseteq x} g(a, y)$, and there are 2 ways to determine the $\arg\max$ over y. The bound then depends on $\sqrt{|\mathcal{A}|}$, but not $\log|\mathcal{A}|$.

Theorem 1 provides the VC dimension for the decision lists that we learned in Chapter 3. For a linear classifier, the VC dimension equals the number of variables in the model. Thus the result $h = |\mathcal{A}|$ is exactly the VC dimension of a classifier created

using a linear combination of association rules, as was done in Chapter 2. Although these were two very different approaches to using association rules to make predictions, we have the same prediction guarantees. We then expect similar generalization behavior for the two different approaches.

Chapter 5

Decision Making from Sales Transaction Data: Bundle Pricing

Sales transaction data are immediately available for most retailers, but it is not always straightforward to use these data to make better decisions. In this chapter we look at how sales transaction data can be used for optimal bundle pricing.

Item bundles, when a collection of items are sold together at a discount, are used across many industries, especially in retail. Both theoretical and empirical work has shown that introducing an appropriately priced bundle can significantly increase profits, with low risk to the retailer (Eppen et al, 1991). Even if a bundle has not been previously offered, useful information about how to price the bundle can be obtained from the sales history of the individual items included in the bundle. Choosing the optimal bundle price relies critically on a knowledge of the price consumers are willing to pay for each item in the bundle, called their valuations, as well as the interplay between the valuations of items in the bundle. A retailer generally does not know the full, joint distribution of valuations. However, the retailer likely does have historical sales transaction data for the individual items. We introduce a procedure for learning the joint distribution of valuations from individual item sales transaction data, thus allowing for optimal bundle pricing.

The economics literature on bundling has extensively examined the economic efficiency of bundling and how bundling can be used for price discrimination (Adams

and Yellen, 1976; Schmalensee, 1982; McAfee et al, 1989). These foundational studies have been extended in many directions. Several papers have focused on analytical solutions for the optimal bundle price and other quantities of interest (Venkatesh and Kamakura, 2003; McCardle et al, 2007; Eckalbar, 2010). These analytical results were obtained for the special case of uniformly distributed valuations, with the distributions for items in the bundle either independent or perfectly correlated. Schmalensee (1984) obtained some analytical results and insights by assuming the joint distribution to be bivariate normal. Other results have been obtained for a finite collection of deterministic valuations (Hanson and Martin, 1990).

A number of useful insights can be gained from these simplified models (see, for example, Stremersch and Tellis, 2002). However, our main interest is in learning the consumer response to bundling from data. When working with data, such strong assumptions about the joint distribution, particularly independence, are no longer appropriate. Jedidi et al (2003) eschew independence assumptions and use methodology based in utility theory to measure valuations. Their measurement procedure requires offering the bundle at various prices to elicit the demand function for the bundle. Based on their empirical results, they report that "models that assume statistical independence are likely to be misspecified." Venkatesh and Mahajan (1993) also study bundle pricing without distributional assumptions for valuations, by mailing out questionnaires that directly asked consumers for their valuations. Conjoint analysis has also been used to estimate the valuation distribution from questionnaire data in the context of bundling (Goldberg et al, 1984; Wuebeker and Mahajan, 1999).

Our contribution in this chapter is an inference procedure for predicting the expected change in profits when a bundle is offered at a particular price. The procedure is developed for sales transaction data, and does not require collecting sales data for the bundle *a priori*, nor does it require direct elicitation of valuations via questionnaires. The procedure is based on inference of a copula model over latent consumer valuations, which allows for arbitrary marginal distributions and does not assume independence. Because the valuations are unobserved, the likelihood function involves integrating over the latent valuations, and standard formulas for copula fitting can-

not be directly applied. We show how these computationally intractable integrals can be transformed into distribution function evaluations, thus allowing for efficient estimation. Our simulation studies and data experiments suggest that the inference procedure allows for data-based bundling decisions which can help retailers increase profits.

5.1 Copula Inference and Bundle Pricing

We suppose that a collection of n items have been selected as a candidate bundle, and our goal is to determine the optimal price and its associated profit if the bundle were to be introduced[1]. We consider the situation where the items have not previously been offered as a bundle, but historical sales transaction data are available for the individual items.

The transaction data that we consider consist of two components: purchase data $\boldsymbol{y}^t$ and price data $\boldsymbol{x}^t$. Specifically, we let $\boldsymbol{y}^t = [y_1^t, \ldots, y_n^t]$ denote the sales data for transaction t, with $y_i^t = 1$ if item i was purchased in transaction t, and 0 otherwise. We assume that the price of each item at the time of each transaction is known, and denote the price of item i at the time of transaction t as x_i^t. Let T denote the total number of transactions.

5.1.1 Valuations and Consumer Rationality

We suppose that each consumer has a valuation for each item, with v_i^t representing the (unobserved) valuation for item i by the consumer in transaction t. As is done throughout the bundling literature and much of the economics literature, we assume that consumers are rational. Specifically, we model consumers as having infinite budget, and as purchasing the assortment of items that maximizes the total difference

[1]The type of bundle that we consider here is called *mixed bundling*, in which consumers are offered both the bundle and the individual items, with the bundle discounted relative to the sum of the item prices.

between their valuation and the price:

$$\boldsymbol{y}^t \in \underset{\boldsymbol{y}\in\{0,1\}^n}{\operatorname{argmax}} \sum_{i=1}^{n} (v_i^t - x_i^t) y_i. \tag{5.1}$$

The rationality assumption implies that $y_i^t = 1$ if and only if $v_i^t > x_i^t$. [2]

The rationality assumption provides a model for the relationship between valuations v_i^t and transaction data y_i^t and x_i^t. Using this model, we now derive likelihood formulas for inferring a joint distribution of valuations from sales transaction data. Then we show how the valuation distribution can be used to find the optimal bundle price.

5.1.2 Joint Distribution Models and Copula Inference

The most straightforward approach to model a joint distribution is to assume independence. This type of joint model allows for arbitrary margins, however independence is a potentially unreasonable assumption, especially because correlations are quite important for bundling, as we show in Section 5.2. Modeling the joint distribution as a multivariate normal allows for correlations via a covariance matrix, however it requires the margins to be normally distributed, which can also be a strong assumption when learning from data. Here we model the joint distribution using a copula model, which is a class of joint distributions that allows for both correlation structures and arbitrary margins. Copula models are widely used in statistics and finance, and are becoming increasingly utilized for machine learning due to their flexibility and computational properties (see, for example, Elidan, 2010, 2013).

We assume consumers are homogeneous, and model the consumer valuations $\boldsymbol{v}^t$ as independent draws from a joint distribution with distribution function $F(v_1, \ldots, v_n)$. Our goal is to infer this joint distribution. Let $F_i(v_i)$ be the marginal distribution function for item i. Then, a copula $\mathbb{C}(\cdot)$ for $F(\cdot)$ is a distribution function over $[0,1]^n$

[2] We model v_i^t as a continuous random variable, and thus do not need to devote attention to the case $v_i^t = x_i^t$.

with uniform margins such that

$$F(v_1, \ldots, v_n) = \mathbb{C}(F_1(v_1), \ldots, F_n(v_n)).$$

The copula combines the margins in such a way as to return the joint distribution. A copula allows for the correlation structure to be modeled separately from the marginal distributions, in a specific way which we show below. The field of copula modeling is based on a representation theorem by Sklar (1973) which shows that every distribution has a copula, and if the margins are continuous, the copula is unique. The copula representation for a joint distribution has a number of interesting properties that are helpful for efficient inference - see Trivedi and Zimmer (2005) for a more detailed exposition.

Our approach to estimating $F(\cdot)$ will be to choose parametric forms for the margins $F_i(\cdot)$ and the copula $\mathbb{C}(\cdot)$, and then find the parameters for which $\mathbb{C}(F_1(v_1), \ldots, F_n(v_n))$ is closest to $F(v_1, \ldots, v_n)$, in a likelihood sense. Specifically, suppose each margin is a distribution function with parameters $\boldsymbol{\theta}_i$, and the copula distribution belongs to a family with parameters $\boldsymbol{\phi}$. We denote the parameterized margins as $F_i(v_i; \boldsymbol{\theta}_i)$ and the parameterized joint distribution as $F(\boldsymbol{v}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{C}(F_1(v_1; \boldsymbol{\theta}_1), \ldots, F_n(v_n; \boldsymbol{\theta}_n); \boldsymbol{\phi})$. We are interested in the maximum likelihood problem

$$\left(\hat{\boldsymbol{\theta}}_{\mathrm{ML}}, \hat{\boldsymbol{\phi}}_{\mathrm{ML}}\right) \in \underset{\boldsymbol{\theta}, \boldsymbol{\phi}}{\operatorname{argmax}} \ \ell(\boldsymbol{\theta}, \boldsymbol{\phi}),$$

where $\ell(\boldsymbol{\theta}, \boldsymbol{\phi})$ is the appropriate log-likelihood function. The main advantage in using a copula model is that the parameters can be separated into those that are specific to one margin ($\boldsymbol{\theta}_i$) and those that are common to all margins ($\boldsymbol{\phi}$). Using a procedure called inference functions for margins (IFM) (Joe and Xu, 1996), the optimization can be performed in two steps. First each margin is fit independently, and then the margin estimates are used to fit the correlation structure:

$$\hat{\boldsymbol{\theta}}_i \in \underset{\boldsymbol{\theta}_i}{\operatorname{argmax}} \ \ell_i(\boldsymbol{\theta}_i), \quad i = 1, \ldots, n \tag{5.2}$$

$$\hat{\boldsymbol{\phi}} \in \underset{\boldsymbol{\phi}}{\operatorname{argmax}}\ \ell(\hat{\boldsymbol{\theta}}, \boldsymbol{\phi}). \tag{5.3}$$

This gives computational tractability by significantly reducing the dimensionality of the optimization problem that must be solved. In general, IFM does not yield exactly the maximum likelihood estimate: $(\hat{\boldsymbol{\theta}}_{\mathrm{ML}}, \hat{\boldsymbol{\phi}}_{\mathrm{ML}}) \neq (\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}})$. However, the IFM estimates $(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}})$, like the maximum likelihood estimates, are statistically consistent and asymptotically normal (Joe and Xu, 1996; Xu, 1996).

The inference problem that we face here differs from a typical copula modeling problem because the distribution of interest is that over valuations, which are unobserved, latent variables. In the next two sections, we use the rationality assumption of (5.1) to derive tractable likelihood formulas to be used in (5.2) and (5.3).

5.1.3 Margin Likelihood and Demand Models

We first consider the margin maximum likelihood problem in (5.2). Let $p_i(x_i^t)$ be the probability of purchase for item i at price x_i^t, that is, the demand model for item i. The following proposition shows an equivalence between the marginal valuation distribution function and demand models.

Proposition 2. *The demand function and the inverse marginal valuation distribution function are identical, i.e.,*

$$p_i(x_i^t) = 1 - F_i(x_i^t; \boldsymbol{\theta}_i).$$

Proof. By the rationality assumption of (5.1), item i is purchased if and only if $v_i^t > x_i^t$:

$$p_i(x_i^t) = \mathbb{P}(v_i^t > x_i^t) = 1 - F_i(x_i^t; \boldsymbol{\theta}_i).$$

□

We thus choose the following likelihood model for the observed purchase data:

$$y_i^t \sim \text{Bernoulli}(1 - F_i(x_i^t; \boldsymbol{\theta}_i)).$$

Given data $\{x_i^t, y_i^t\}_{t=1}^T$, the log-likelihood function for each margin is:

$$\ell_i(\boldsymbol{\theta}_i) = \sum_{t=1}^{T} \big(y_i^t \log(1 - F_i(x_i^t; \boldsymbol{\theta}_i)) + (1 - y_i^t) \log(F_i(x_i^t; \boldsymbol{\theta}_i)) \big). \tag{5.4}$$

If $F_i(\cdot; \boldsymbol{\theta}_i)$ is linear in $\boldsymbol{\theta}_i$, for example when using a linear demand model, then the maximum likelihood problem is a concave maximization. For general demand models, a local maximum can easily be found using standard optimization techniques. In Section 5.1.7 we discuss some possible choices for the family of $F_i(\cdot; \boldsymbol{\theta}_i)$.

5.1.4 Copula Inference over Latent Variables

Once the margin parameters $\hat{\boldsymbol{\theta}}_i$ have been estimated by maximizing (5.4), these estimates are used, together with the data, to obtain an estimate of the copula parameters $\boldsymbol{\phi}$. We now derive an expression for the log-likelihood of $\boldsymbol{\phi}$.

$$\begin{aligned} \ell(\hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) &= \sum_{t=1}^{T} \log p(\boldsymbol{y}^t | \boldsymbol{x}^t, \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) \\ &= \sum_{t=1}^{T} \log \int p(\boldsymbol{y}^t | \boldsymbol{v}^t, \boldsymbol{x}^t, \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) p(\boldsymbol{v}^t | \boldsymbol{x}^t, \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) d\boldsymbol{v}^t. \end{aligned} \tag{5.5}$$

Given $\boldsymbol{v}^t$ and $\boldsymbol{x}^t$, $\boldsymbol{y}^t$ is deterministic, with $y_i^t = 1$ if $v_i^t > x_i^t$ and 0 otherwise. Thus the integral over $\boldsymbol{v}^t$ can be limited to all $\boldsymbol{v}^t$ that are consistent with $\boldsymbol{y}^t$ and $\boldsymbol{x}^t$, meaning the integral is over $v_i^t > x_i^t$ for i such that $y_i^t = 1$, and over $v_i^t \leq x_i^t$ for i such that $y_i^t = 0$. We then define the lower and upper limits of integration as,

$$v_i^{t,\ell} = \begin{cases} -\infty & \text{if } y_i^t = 0, \\ x_i^t & \text{if } y_i^t = 1, \end{cases} \quad \text{and} \quad v_i^{t,u} = \begin{cases} x_i^t & \text{if } y_i^t = 0, \\ +\infty & \text{if } y_i^t = 1. \end{cases}$$

The quantity $p(\boldsymbol{v}^t|\boldsymbol{x}^t, \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) = p(\boldsymbol{v}^t|\hat{\boldsymbol{\theta}}, \boldsymbol{\phi})$ is exactly the copula density function, which we denote as $f(\cdot; \hat{\boldsymbol{\theta}}, \boldsymbol{\phi})$. Continuing the likelihood expression from (5.5), we have,

$$\ell(\hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) = \sum_{t=1}^{T} \log \int_{v_n^{t,\ell}}^{v_n^{t,u}} \dots \int_{v_1^{t,\ell}}^{v_1^{t,u}} f(v_1^t, \dots, v_n^t; \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) dv_1^t \dots dv_n^t. \tag{5.6}$$

The integral in (5.6) renders the likelihood formula intractable. To allow for efficient inference, we will use the following formula for a rectangular integral of a probability density function. This formula is critical to the scalability of our inference procedure as it allows us to replace the multidimensional integral in (5.6) with distribution function evaluations.

Lemma 1. *Let $f(\cdot)$ be a joint probability density function over continuous random variables $z_1, \dots, z_n$ with the corresponding joint distribution function $F(\cdot)$. Then,*

$$\int_{z_n^\ell}^{z_n^u} \dots \int_{z_1^\ell}^{z_1^u} f(z_1, \dots, z_n) dz_1 \dots dz_n = \sum_{k=0}^{n} (-1)^k \sum_{\substack{I \subseteq \{1,\dots,n\} \\ |I|=k}} F(\tilde{\boldsymbol{z}}(I)),$$

where

$$\tilde{z}_i(I) = \begin{cases} z_i^\ell & \text{if } i \in I, \\ z_i^u & \text{otherwise.} \end{cases}$$

Proof. Define the probability events $A_i = \{z_i \leq z_i^\ell\}$ for each i. Let $B = \cap_{i=1}^n \{z_i \leq z_i^u\}$. Then,

$$\begin{aligned}
\int_{z_n^\ell}^{z_n^u} \dots \int_{z_1^\ell}^{z_1^u} f(z_1, \dots, z_n) dz_1 \dots dz_n &= \mathbb{P}\left(B \cap (\cap_{i=1}^n A_i^\mathsf{c})\right) \\
&= \mathbb{P}\left(B \cap (\cup_{i=1}^n A_i)^\mathsf{c}\right) \\
&= \mathbb{P}(B) - \mathbb{P}\left(B \cap (\cup_{i=1}^n A_i)\right) \\
&= \mathbb{P}(B) - \mathbb{P}\left(\cup_{i=1}^n \left(B \cap A_i\right)\right) \\
&= \mathbb{P}(B) - \sum_{k=1}^{n} (-1)^{k-1} \sum_{\substack{I \subseteq \{1,\dots,n\} \\ |I|=k}} \mathbb{P}(B \cap A_I)
\end{aligned}$$

by the inclusion-exclusion formula, with $A_I = \cap_{i \in I} A_i$. We then substitute $\mathbb{P}(B) =$

$F(z_1^u, \ldots, z_n^u)$ and $\mathbb{P}(B \cap A_I) = F(\tilde{\boldsymbol{z}}(I))$ as defined above to obtain the statement of the lemma. □

With Lemma 1, we are now equipped to evaluate the log-likelihood expression in (5.6):

$$\ell(\hat{\boldsymbol{\theta}}, \boldsymbol{\phi}) = \sum_{t=1}^{T} \log \sum_{k=0}^{n} (-1)^k \sum_{\substack{I \subseteq \{1,\ldots,n\} \\ |I|=k}} F(\tilde{\boldsymbol{v}}^t(I); \hat{\boldsymbol{\theta}}, \boldsymbol{\phi}), \tag{5.7}$$

where as before

$$\tilde{v}_i^t(I) = \begin{cases} v_i^{t,\ell} & \text{if } i \in I, \\ v_i^{t,u} & \text{otherwise.} \end{cases}$$

For the most simple case of two items in a bundle, the inner expression in (5.7) evaluates to

$$\sum_{k=0}^{2} (-1)^k \sum_{\substack{I \subseteq \{1,2\} \\ |I|=k}} F(\tilde{v}_1^t(I), \tilde{v}_2^t(I)) = \begin{cases} F(x_1^t, x_2^t) & \text{if } y = (0,0), \\ F_1(x_1^t) - F(x_1^t, x_2^t) & \text{if } y = (0,1), \\ F_2(x_2^t) - F(x_1^t, x_2^t) & \text{if } y = (1,0), \\ 1 - F_1(x_1^t) - F_2(x_2^t) + F(x_1^t, x_2^t) & \text{if } y = (1,1). \end{cases}$$

5.1.5 Consistency and Scalability

Combining (5.4) and (5.7) yields the complete inference procedure, which we give in the following proposition.

Proposition 3. *The inference procedure*

$$\hat{\boldsymbol{\theta}}_i \in \operatorname*{argmax}_{\boldsymbol{\theta}_i} \sum_{t=1}^{T} \left(y_i^t \log(1 - F_i(x_i^t; \boldsymbol{\theta}_i)) + (1 - y_i^t) \log(F_i(x_i^t; \boldsymbol{\theta}_i)) \right)$$

$$\hat{\boldsymbol{\phi}} \in \operatorname*{argmax}_{\boldsymbol{\phi}} \sum_{t=1}^{T} \log \sum_{k=0}^{n} (-1)^k \sum_{\substack{I \subseteq \{1,\ldots,n\} \\ |I|=k}} F(\tilde{\boldsymbol{v}}^t(I); \hat{\boldsymbol{\theta}}, \boldsymbol{\phi})$$

is statistically consistent.

Because the inference is exactly the IFM procedure, it follows from Joe and Xu (1996) that it is statistically consistent.

The computation is exponential in the size of the bundle n, however in retail practice bundle offers generally do not contain a large number of items. Importantly, the computation is linear in the number of transactions T, which allows inference to be performed even on very large transaction databases. The main computational step is evaluating the copula distribution function in (5.7). For many copula models, such as the Gaussian copula which we describe in Section 5.1.7, efficient techniques are available for distribution function evaluation.

5.1.6 Computing the Optimal Bundle Price

Given the joint valuation distribution, the expected profit per consumer as a function of item and bundle prices can be computed. For notational convenience, here we give the result for $n = 2$. Consumers are rational, in that they choose the option (item 1 only, item 2 only, bundle, or no purchase) that maximizes their surplus $v_i - x_i$. For this result, we assume that the valuation for the bundle is the sum of the component valuations $v_B = v_1 + v_2$, although this could easily be relaxed to other bundle valuation models such as those in Venkatesh and Kamakura (2003). Note that inferring the joint valuation distribution does not require any assumption on how valuations combine, rather this assumption is only used to compute the expected profit of bundling. We denote the cost of item i as c_i and assume that the bundle cost is the sum of the component costs.

Proposition 4. *For joint valuation density function $f(\cdot)$ and joint valuation distribution function $F(\cdot)$, the expected profit per consumer obtained when items 1, 2, and the bundle are priced at x_1, x_2, and x_B respectively is*

$$\begin{aligned}\mathbb{E}\left[\textit{profit}\right] = &(x_1 - c_1)(F_2(x_B - x_1) - F(x_1, x_B - x_1)) \\ &+ (x_2 - c_2)(F_1(x_B - x_2) - F(x_B - x_2, x_2)) \\ &+ (x_B - c_1 - c_2)\Big(1 - F_1(x_B - x_2) - F_2(x_B - x_1)\end{aligned}$$

$$+ F(x_B - x_2, x_B - x_1) - \int_{x_B - x_2}^{x_1} \int_{x_B - x_1}^{x_B - v_1} f(v_1, v_2) dv_2 dv_1 \Bigg).$$

Proof. The profit can be decomposed into that obtained from each of the purchase options.

$$\begin{aligned}
\mathbb{E}\left[\text{profit}\right] = {} & (x_1 - c_1)\mathbb{P}(\text{Purchase item 1 only}) \\
& + (x_2 - c_2)\mathbb{P}(\text{Purchase item 2 only}) \\
& + (x_B - c_1 - c_2)\mathbb{P}(\text{Purchase the bundle}).
\end{aligned}$$

The options no purchase, purchasing item 1 only, purchasing item 2 only, and purchasing the bundle give the consumer surplus 0, $v_1 - x_1$, $v_2 - x_2$, and $v_1 + v_2 - x_B$ respectively. Let us consider the consumers that purchase only item 1. By the rationality assumption, $v_1 - x_1 \geq 0$, $v_1 - x_1 \geq v_2 - x_2$, and $v_1 - x_1 \geq v_1 + v_2 - x_B$. Thus,

$$\begin{aligned}
\mathbb{P}(\text{Purchase item 1 only}) &= \mathbb{P}\left(\{v_1 \geq x_1\} \cap \{v_2 \leq x_B - x_1\}\right) \\
&= F_2(x_B - x_1) - F(x_1, x_B - x_1),
\end{aligned}$$

by Lemma 1. A similar derivation applies to item 2. For the bundle,

$$\begin{aligned}
&\mathbb{P}(\text{Purchase the bundle}) \\
&= \mathbb{P}\left(\{v_1 \geq x_B - x_2\} \cap \{v_2 \geq x_B - x_1\} \cap \{v_1 + v_2 \geq x_B\}\right) \\
&= \mathbb{P}\left(\{v_1 \geq x_B - x_2\} \cap \{v_2 \geq x_B - x_1\}\right) \\
&\quad - \mathbb{P}\left(\{v_1 \geq x_B - x_2\} \cap \{v_2 \geq x_B - x_1\} \cap \{v_1 + v_2 \leq x_B\}\right) \\
&= 1 - F_1(x_B - x_2) - F_2(x_B - x_1) + F(x_B - x_2, x_B - x_1) \\
&\quad - \int_{x_B - x_2}^{x_1} \int_{x_B - x_1}^{x_B - v_1} f(v_1, v_2) dv_2 dv_1,
\end{aligned}$$

using Lemma 1. □

Similar results, albeit notationally complex, can be obtained for $n > 2$. The

inference procedure from Proposition 3 is used to estimate the valuation distribution function, which allows the expression in Proposition 4 to be evaluated. Maximizing the expected profit with respect to x_B yields the optimal bundle price, or maximizing over x_B and the item prices simultaneously yields a complete pricing strategy. The formula in Proposition 4 is not concave in general, but a local maximum can be found using standard numerical optimization techniques.

5.1.7 Distributional Assumptions

The likelihood formulas in (5.4) and (5.7) hold for arbitrary margins $F_i(\cdot;\boldsymbol{\theta}_i)$ and an arbitrary copula model $\mathbb{C}(\cdot;\boldsymbol{\phi})$. To apply these formulas to data requires choosing the distributional form of the margins and the copula family.

The connection between marginal valuation distributions and demand models given in Proposition 2 shows that the margin distribution can naturally be selected by choosing an appropriate demand model. Many retailers already use demand models for sales forecasting, and these existing models could be directly converted to marginal valuation distributions. For example, two common choices for demand models are the linear demand model and the normal-cdf demand model. The linear demand model is

$$p(x_i;\beta_i,\eta_i) = \min(1,\max(0,\beta_i - \eta_i x_i)),$$

and the corresponding valuation distribution is uniform:

$$v_i \sim \text{Unif}\left(\frac{\beta_i - 1}{\eta_i}, \frac{\beta_i}{\eta_i}\right).$$

When the demand model is the normal distribution function

$$p(x_i;\mu_i,\sigma_i^2) = 1 - \Phi(x_i;\mu_i,\sigma_i^2),$$

the corresponding marginal valuation distribution is the normal distribution:

$$v_i \sim \mathcal{N}(\mu_i,\sigma_i^2).$$

Additional covariates like competitors' prices or the prices of substitutable and complimentary products are sometimes used in demand modeling, for instance in a choice model. Seasonality effects are also often handled using covariates. Models with covariates can also be transformed into valuation distributions using Proposition 2.

There is a large selection of copula models, which differ primarily in the types of correlation they can express. One of the most popular copula models, and that which we use in our simulations and data experiments here, is the Gaussian copula:

$$\mathbb{C}(u_1, \ldots, u_n; \boldsymbol{\phi}) = \Phi(\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_n); \boldsymbol{\phi}),$$

where $\Phi(\cdot; \boldsymbol{\phi})$ is the multivariate normal with correlation matrix $\boldsymbol{\phi}$, and $\Phi(\cdot)$ the standard normal. The Gaussian copula is in essence an extension of the multivariate normal distribution, in that it extends the multivariate normal correlation structure to arbitrary margins, as opposed to constraining the margins to be normally distributed. If a correlation matrix structure is not appropriate to model the dependencies in a particular application, then alternative copula models are available - see Trivedi and Zimmer (2005).

5.2 Simulation Studies

We demonstrate the inference procedure using a series of simulation studies. We first use simulations to show empirically how the estimated parameters converge to their true values as T grows. We then use a simulated dataset to illustrate the importance of including correlations in the model.

We generated purchase data for a pair of items using uniform marginal valuation distributions and a Gaussian copula, which for two items is characterized by the correlation coefficient ϕ. The correlation coefficient ϕ was taken to be each of $\{-0.9, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 0.9\}$ and the number of transactions T was taken from $\{100, 250, 500, 750, 1000, 1500, 2000\}$. For each combination of ϕ and T, 500 datasets were generated, for a total of 31,500 simulated datasets. For each dataset,

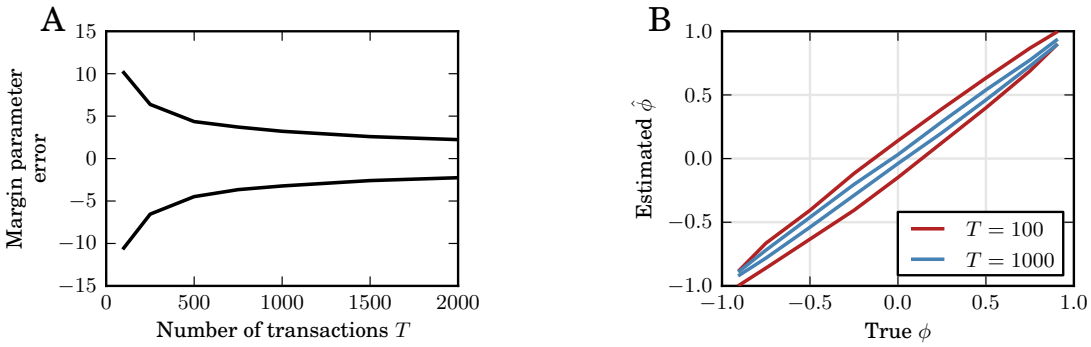

Figure 5-1: Convergence of both (A) margin parameters and (B) the correlation coefficient to their true values as the number of simulated transactions T is increased. In (A), the lines indicate the first and third quartiles of the margin parameter errors across all simulations with the same number of transactions T. In (B), each pair of lines shows the first and third quartiles of the estimated correlation coefficient $\hat{\phi}$ across all simulations with the corresponding values of ϕ and T.

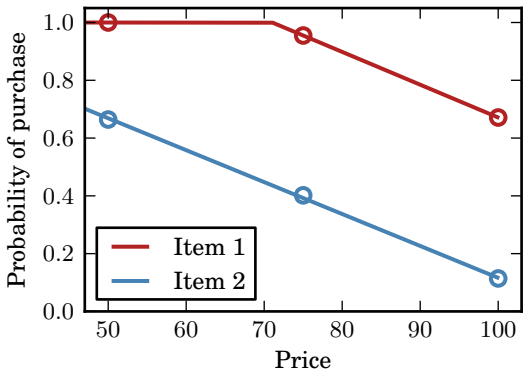

Figure 5-2: Demand models for each of the two items for one of the simulated datasets. The circles give the empirical purchase probabilities measured from the data, and the lines show the fitted margin distribution function.

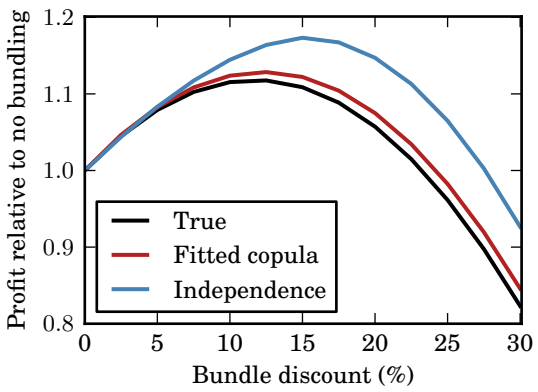

Figure 5-3: Change in relative profits from introducing the bundle at a particular discount relative to the sum of item prices, as estimated from the true distribution, the fitted copula model, and a distribution using the fit margins but assuming independence.

the margin parameters $v_{\min}$ and $v_{\max}$ for each of the two uniform valuation distributions were chosen independently at random, to allow the simulations to capture a large range of margin distributions. The parameter $v_{\min}$ was chosen from a uniform distribution over $[-25, 75]$ and $v_{\max}$ chosen from a uniform distribution over $[100, 200]$. For all simulations, the transactions were spread uniformly across three price points, with the prices of the two items taken to be 100 for one third of transactions, 75 for one third, and 50 for the remaining third. In each simulation, the copula defined by the combination of the correlation coefficient and the margin parameters was used to generate T sets of valuations for the items. These valuations were combined with the prices under the rationality assumption of (5.1) to produce binary purchase data.

We applied the inference procedure in Proposition 3 to the transaction data, with the goal of recovering the true, generating copula model. Figure 5-1 shows that as the number of transactions grows, both the margin estimates and the correlation coefficient estimates converge to their true values. This holds for the full range of possible values of the correlation coefficient. In these simulations, only a few thousand samples were required to recover the true distribution with high accuracy, suggesting that these techniques are not limited to retailers with very large datasets.

To further illustrate the simulation results, we selected at random a simulated dataset with $T = 2000$ transactions and $\phi = 0.5$. We show in Figure 5-2 the fit-

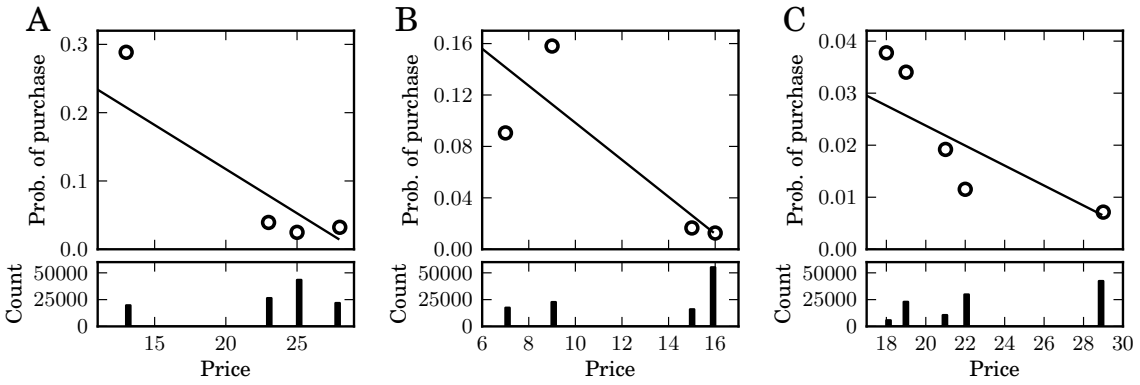

Figure 5-4: Fitted marginal distributions for items (A) 38, (B) 14, and (C) 08 from the Ta-Feng retail transaction dataset. The underset histogram shows the number of transactions for which the item was offered at each price. For each price at which the item was offered, the circles indicate the purchase probability at that price as measured from the data. The line gives the model fit.

ted margins for this particular simulated dataset. The estimated correlation coefficient, found by maximizing (5.7), was 0.48. To illustrate the potential profitability of bundling, in Figure 5-3 we held the item prices at 100 and set the cost per item to the retailer to a 50% markup, meaning, sales price 50% higher than the retailer's cost. We show for a range of bundle discounts the profit relative to the profit obtained in the absence of a bundle discount. The estimated distribution is very close to the true distribution, and both reveal that offering a bundle discount of about 12% will increase profits by about 10%. Using the same estimated margins but assuming independence to obtain a joint distribution yields very different results. This example highlights the importance of accounting for correlations in valuations when estimating the response to bundle discounts.

5.3 Data Experiments

We provide further evaluation and illustration of the inference procedure by applying it to actual retail transaction data. We use the publicly available Ta-Feng dataset, which contains four months of transaction level data from a Taiwanese warehouse club, totaling about 120,000 transactions and 24,000 items (Hsu et al, 2004). Each entry in the Ta-Feng dataset corresponds to the sale of a single item within a transaction.

To form the complete transaction of (potentially) multiple items, we grouped all sales that occurred on the same day with the same user ID. For simplicity, we assumed that for each day there was a single price for each item. If there were multiple prices at which an item was sold on a given day, we took that day's price as the median of the observed prices. If an item was not sold on a particular day, then we took that day's price as the price of the preceding day. To further smooth the prices, we allowed only prices that covered at least 5% of transactions, and any price that did not meet that support threshold was rounded to the nearest price that did. After removing items that did not have at least three prices in the data, we selected the three items with the highest support, which were (EAN-13) 4714981010038, 4711271000014, and 4710583996008. We considered the four possible bundles that could be obtained from these three items (three pairs and one bundle of three). Throughout this section we refer to the three items as item 38, item 14, and item 08. Note that in these experiments the inference procedure scales to a much larger dataset than those used in the simulation studies.

As in Section 5.2, we model the joint valuation distribution using linear demand models (uniform marginal valuation distributions) and a Gaussian copula. In Figure 5-4 we show the demand models fit by maximizing (5.4) for each item. The off-diagonal elements of the correlation matrix ϕ corresponding to pairs 38-14, 38-08, and 14-08 were jointly estimated as 0.085, 0.133, and 0.172 respectively.

We evaluated the predictive performance of the copula model using 10-fold cross validation, by fitting the model to 9 folds of the data and then evaluating the (predictive) log-likelihood on the remaining fold. This was done separately for each pair of items (38-14, 38-08, and 14-08) and for the collection of all three items (38-14-08), and the results are compared to the model using the same fitted margins but assuming independence. Figure 5-5 shows that for all 10 folds and for all bundles, the copula model had higher predictive likelihoods than the corresponding independence model.

To illustrate the results, we report the relative expected profit under various bundle scenarios in Figure 5-6. For these results we took the item prices as the mode of the price distribution in the data, and since the item costs are unknown, we set

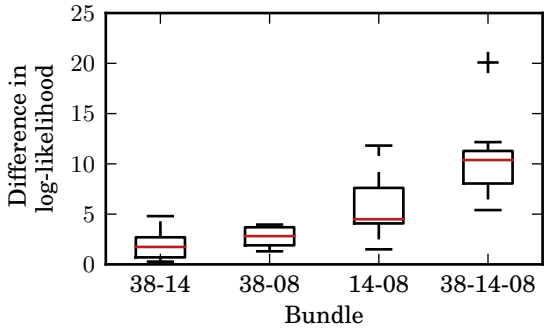

Figure 5-5: Copula predictive log-likelihood minus the independence model log-likelihood, across 10 folds of cross-validation for each of the four bundles.

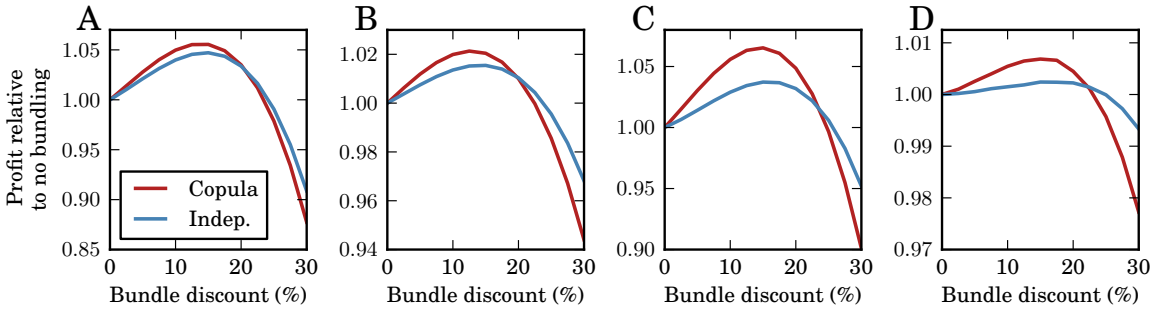

Figure 5-6: Change in relative profits by introducing bundles (A) 38-14, (B) 38-08, (C) 14-08, and (D) 38-14-08 as a function of the level of bundle discount, estimated from the Ta-Feng dataset. In red is the prediction obtained from the fitted copula model, and in blue is the prediction obtained using the same fitted margins, but assuming independence.

them to a 35% markup. In a similar way as Figure 5-3, Figure 5-6 shows that introducing a discounted bundle can increase profits, and that assuming independence can lead to very different predictions. This further highlights the importance of including correlations in the valuation distribution model.

5.4 Discussion and Conclusions

We used copula modeling in the context of an important business analytics problem, and in the process have developed new methodological results on learning a copula distribution over latent variables. Our work provides foundational results for inferring consumer valuations. The ability to predict the effect of introducing a bundle at a particular price using only historical sales data is a major advancement in data-driven

pricing, and the copula model at the core of the inference here is flexible enough to be useful in real applications. Because the copula allows for arbitrary margins, if a retailer has already developed demand models for a particular item, the demand model can be used directly to obtain the marginal valuation distribution. The likelihood formulas that we derived in this paper provide a theoretically and computationally sound framework for copula learning over latent valuations.

Chapter 6

Decision Making from Sales Transaction Data: Stockouts and Demand Estimation

An important common challenge facing retailers is to understand customer preferences in the presence of stockouts. When an item is out of stock, some customers will leave, while others will substitute a different product. From the transaction data collected by retailers, it is challenging to determine exactly what the customer's original intent was, or, because of no-purchase arrivals, even how many customers there actually were.

The task that we consider here is to infer both the arrival rate, including those that left without a purchase, and the substitution model from sales transaction and stock level data. These quantities are a necessary input for inventory management and assortment planning problems. In this paper we apply the model and inference procedure to bakery data to estimate lost sales due to stock unavailability. We will see that for some items there are substantial lost sales, while for others, due to substitution, there are not. Knowing which items are being substituted and which are not will help the retailer to better focus resources.

There are several contributions made by our model. First, we allow the model for the arrival rate to be nonhomogeneous in time. For example, in our experiments

with bakery data we treat each day as a time period and model the arrival rate with a function that peaks at the busiest time for the bakery and then tapers off towards the end of the day. Nonhomogeneous arrival rates are likely to be present in many retail settings where stockouts are common. For example, in our experiments we use transaction data from a bakery, where many of the items are intended to stockout every day as they must be baked fresh the next morning. As we will see in Section 6.4, the daily arrival rate at the bakery is far from constant. As another example, Johnson et al (2014) describe a relatively new industry of retailers that operate flash sales in which the most popular items quickly stockout. Using data from one of these retailers they show that the purchase rate has a peak near the start of the sale and then decreases.

The second major contribution is that our model can incorporate practically any choice model, including nonparametric models. The third is that the model allows for multiple customer segments, each with their own substitution models. We show how this can be used to borrow strength across data from multiple stores. Finally, our inference is fully Bayesian. In many cases the model parameters are not of interest *per se*, but are to be used for making predictions and decisions. In Section 6.0.2 we discuss how Bayesian inference provides a natural framework for incorporating the uncertainty in the inference into the decisions that are based on the inference.

In this paper we describe the model and the Bayesian inference procedure. We then use a series of simulations to illustrate the inference, and to show that we can recover the true, generating values. Finally, we demonstrate how the model can be fit to real transaction data obtained from a local bakery. We use the results to estimate the bakery's lost sales due to stock unavailability.

6.0.1 Prior Work

The primary work on estimating demand and choice from sales transaction data with stockouts was done by Vulcano et al (2012). They model customer arrivals using a homogeneous Poisson process within each time period, meaning the arrival rate is constant throughout each time period. Customers then choose an item, or an unob-

served no-purchase, according to the multinomial logit (MNL) choice model. They show that when the no-purchase customers are not observed, the MNL choice model parameters are not all identifiable. Rather, the retailer must conjecture the proportion of arrivals that do not purchase anything even when all items are in stock. They derive an EM algorithm to solve the corresponding maximum likelihood problem.

Our model uses a nonhomogeneous Poisson process for customer arrivals that allows the arrival rate to vary throughout each time period. The nonhomogeneity will prove important when we work with real data in Section 6.4, which are nonhomogeneous throughout the day. Our model also does not require using the MNL model and can be used with models that are entirely identifiable, thus no longer requiring the retailer to know beforehand the unobserved proportion of no-purchases. The exogenous model that we describe in Section 6.1.3 is one such model that we use. Finally, we take a Bayesian approach to inference which comes with advantages over maximum likelihood estimation in using the model to make predictions, as we describe in Section 6.0.2.

Anupindi et al (1998) also present a method for estimating demand and choice probabilities from transaction data with stockouts. Customer arrivals are modeled with a homogeneous Poisson process and purchase probabilities are modeled explicitly for each stock combination, as opposed to using a choice model. They find the maximum likelihood estimates for the arrival rate and purchase probabilities. Their model does not scale well to a large number of items as the likelihood expression includes all stock combinations found in the data.

Vulcano and van Ryzin (2014) extend the work of Vulcano et al (2012) to incorporate nonparametric choice models, for which maximum likelihood estimation becomes a large-scale concave program that must be solved via a mixed integer program subproblem. Our model naturally incorporates nonparametric models from a pre-specified subset of relevant types. Their approach generates relevant types, but requires a constant arrival rate over time periods and involves a computationally intensive optimization.

There is work on estimating demand and choice in settings different from that

which we consider here, such as discrete time (Talluri and van Ryzin, 2001; Vulcano et al, 2010), panel or aggregate sales data (Campoa et al, 2003; Kalyanam et al, 2007; Musalem et al, 2010), negligible no purchases (Kök and Fisher, 2007), and online learning with simultaneous ordering decisions (Jain et al, 2015). Jain et al (2015) provide an excellent review of the various threads of research in demand and choice estimation.

6.0.2 The Bayesian Approach

Suppose we have data $\boldsymbol{t}$ and latent model parameters $\boldsymbol{z}$. A common estimation approach is the maximum likelihood estimate: $\boldsymbol{z}^* \in \arg\max_{\boldsymbol{z}} p(\boldsymbol{t} \mid \boldsymbol{z})$. Suppose now that there is another quantity Q that we wish to predict, which depends on the model parameters according to the probability model $p(Q \mid \boldsymbol{z})$. For instance, the lost sales due to stockouts is one such quantity that we estimate. Using the maximum likelihood estimate, the estimate of Q given $\boldsymbol{t}$ is $Q \sim p(Q \mid \boldsymbol{z}^*)$, from which samples can be drawn with Monte Carlo sampling.

In Bayesian inference, the objective is not a single point estimate, rather it is to draw samples from the posterior distribution $p(\boldsymbol{z} \mid \boldsymbol{t})$. Given these samples, we can estimate the actual posterior distribution of Q:

$$p(Q \mid \boldsymbol{t}) = \int p(Q \mid \boldsymbol{z}) p(\boldsymbol{z} \mid \boldsymbol{t}) d\boldsymbol{z}.$$

The posterior distribution of Q incorporates all of the uncertainty in $\boldsymbol{z}$ directly into the estimate of Q. Suppose that there is a range of values of $\boldsymbol{z}$ with similar likelihood to $\boldsymbol{z}^*$, but that produce very different values of Q. The uncertainty in $\boldsymbol{z}$ that remains after observing $\boldsymbol{t}$ will be translated to the corresponding uncertainty in Q.

6.1 A Generative Model for Transaction Data with Stockouts

We begin by introducing the notation that we use to describe the observed data. We then introduce the nonhomogeneous model for customer arrivals, followed by a discussion of various possible choice models. Section 6.1.4 discusses how multiple customer segments are modeled. We then in Section 6.1.5 introduce the likelihood model: the probabilistic model for how the data are generated. Finally, Section 6.1.6 discusses the prior distributions, at which point the model is ready for inference.

6.1.1 The Data

We suppose that we have data from a collection of stores $\sigma = 1, \ldots, S$. For each store, data come from a number of time periods $l = 1, \ldots, L^\sigma$, throughout each of which time varies from 0 to T. For example, in our experiments a time period was one day. We consider a collection of items $i = 1, \ldots, n$. We suppose that we have two types of data: purchase times and stock levels. We denote the number of purchases of item i in time period l at store σ as $m_i^{\sigma,l}$. Then, we let $\boldsymbol{t}_i^{\sigma,l} = \left\{t_{i,1}^{\sigma,l}, \ldots, t_{i,m_i^{\sigma,l}}^{\sigma,l}\right\}$ be the observed purchase times of item i in time period l at store σ. For notational convenience, we let $\boldsymbol{t}^{\sigma,l} = \left\{\boldsymbol{t}_i^{\sigma,l}\right\}_{i=1}^{n}$ be the collection of all purchase times for that store and time period, and let $\boldsymbol{t} = \left\{\boldsymbol{t}^{\sigma,l}\right\}_{\substack{l=1,\ldots,L^\sigma \\ \sigma=1,\ldots,S}}$ be the complete set of arrival time data.

In addition to purchase times, we suppose that we know the stock levels. We denote the known initial stock level as $N_i^{\sigma,l}$ and assume that stocks are not replenished throughout the time period. That is, $m_i^{\sigma,l} \leq N_i^{\sigma,l}$ and equality implies a stockout. As before, we let $\boldsymbol{N}^{\sigma,l}$ and $\boldsymbol{N}$ represent respectively the collection of initial stock data for store σ and time period l, and for all stores and all time periods.

Given $\boldsymbol{t}_i^{\sigma,l}$ and $N_i^{\sigma,l}$, we can compute a stock indicator as a function of time. We

define this indicator function as

$$s_i(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}) = \begin{cases} 0 & \text{if item } i \text{ is out of stock at time } t \\ 1 & \text{if item } i \text{ is in stock at time } t. \end{cases}$$

6.1.2 Modeling Customer Arrivals

We model the times of customer arrivals using a nonhomogeneous Poisson process (NHPP). An NHPP is a generalization of the Poisson process that allows for the intensity to be described by a function $\lambda(t) \geq 0$ as opposed to being constant. We assume that the intensity function has been parameterized, with parameters $\boldsymbol{\eta}^\sigma$ potentially different for each store σ. For example, if we set $\lambda(t \mid \boldsymbol{\eta}^\sigma) = \eta_1^\sigma$ we obtain a homogeneous Poisson process of rate η_1^σ. As another example, we can produce an intensity function that rises to a peak and then decays by letting

$$\lambda(t \mid \boldsymbol{\eta}^\sigma) = \eta_1^\sigma \frac{\left(\frac{\eta_2^\sigma}{\eta_3^\sigma}\right)\left(\frac{t}{\eta_3^\sigma}\right)^{\eta_2^\sigma - 1}}{\left(1 + \left(\frac{t}{\eta_3^\sigma}\right)^{\eta_2^\sigma}\right)^2}, \tag{6.1}$$

which is the derivative of the Hill equation (Goutelle et al, 2008). This is the parameterization that we use in our bakery data experiments.

The modeler chooses a parameterization for the rate function that is appropriate for their data source, but does not choose the actual values of $\boldsymbol{\eta}^\sigma$. The posterior distribution of $\boldsymbol{\eta}^\sigma$ will be inferred. To do this we use the conditional density function for NHPP arrivals, which we provide now.

Lemma 2. *Consider arrival times $t_1, t_2, \ldots$ generated by an NHPP with intensity function $\lambda(t \mid \boldsymbol{\eta}^\sigma)$. Then,*

$$p(t_j \mid t_{j-1}, \boldsymbol{\eta}^\sigma) = \exp(-\Lambda(t_{j-1}, t_j \mid \boldsymbol{\eta}^\sigma))\lambda(t_j \mid \boldsymbol{\eta}^\sigma),$$

where $\Lambda(t_{j-1}, t_j \mid \boldsymbol{\eta}^\sigma) = \int_{t_{j-1}}^{t_j} \lambda(t \mid \boldsymbol{\eta}^\sigma) dt$.

Proof. The NHPP can be defined by its counting process:

$$\mathbb{P}(m \text{ arrivals in the interval } (\tau_1, \tau_2]) = \frac{(\Lambda(\tau_1, \tau_2))^m \exp(-\Lambda(\tau_1, \tau_2))}{m!},$$

where

$$\Lambda(\tau_1, \tau_2) = \int_{\tau_1}^{\tau_2} \lambda(u) \mathrm{d}u.$$

Let random variables $S_1, S_2, \ldots$ be the arrival process for the NHPP. Consider a pair of times t_j and t_{j-1}, with $t_j > t_{j-1}$. The conditional distribution function for the arrival times is

$$\begin{aligned} F_{S_j}(t_j \mid S_{j-1} = t_{j-1}) &= 1 - \mathbb{P}(S_j > t_j \mid S_{j-1} = t_{j-1}) \\ &= 1 - \mathbb{P}\left(\text{no arrivals in the interval } (t_{j-1}, t_j]\right) \\ &= 1 - \exp(-\Lambda(t_{j-1}, t_j)). \end{aligned} \tag{6.2}$$

Differentiating (6.2) yields the corresponding density function. □

We now provide the density function for a collection of arrivals, which will be important for the likelihood function.

Lemma 3. *For* $t_1, \ldots, t_m \sim NHPP(\lambda(t), T)$,

$$p(t_1, \ldots, t_m) = \exp(-\Lambda(0, T)) \prod_{j=1}^{m} \lambda(t_j).$$

Proof. Let random variables $S_1, S_2, \ldots$ be the NHPP arrival process.

$$\begin{aligned} p(t_1, \ldots, t_m) &= f_{S_1}(t_1) \left(\prod_{j=2}^{m} f_{S_j}(t_j \mid S_{j-1} = t_{j-1}) \right) \mathbb{P}(S_{m+1} > T \mid S_m = t_m) \\ &= \left(\prod_{j=2}^{m} \lambda(t_j) \exp\left(-\Lambda(t_{j-1}, t_j)\right) \right) \left(\lambda(t_1) \exp(-\Lambda(0, t_1))\right) \exp(-\Lambda(t_m, T)) \\ &= \left(\prod_{j=1}^{m} \lambda(t_j) \right) \exp\left(- \left(\Lambda(t_1) + \sum_{j=2}^{m} \Lambda(t_{j-1}, t_j) + \Lambda(t_m, T) \right) \right) \end{aligned}$$

$$= \exp(-\Lambda(0,T)) \prod_{j=1}^{m} \lambda(t_j).$$

□

We let $\boldsymbol{\eta} = \{\boldsymbol{\eta}^\sigma\}_{\sigma=1}^{S}$ represent the complete collection of rate function parameters to be inferred.

6.1.3 Models for Substitution Behavior

We have modeled customers arriving according to an NHPP described by parameters $\boldsymbol{\eta}$. In the next piece of the model, each of those customers will either purchase an item or will choose the "no-purchase" option. If they purchase an item and which item they purchase will depend on the stock availability as well as some choice model parameters which we will describe below. We define $f_i(s(t), \boldsymbol{\phi}^k, \tau^k)$ to be the probability that a customer purchases product i given the current stock $s(t)$ and choice model parameters $\boldsymbol{\phi}^k$ and τ^k. The index k indicates the parameters for a particular customer segment, which we will discuss in Section 6.1.4. The modeler is free to choose whatever form for the choice function f_i he or she finds to be most appropriate. Posterior distributions for the parameters $\boldsymbol{\phi}^k$ and τ^k are then inferred. We now discuss how several common choice models fit into this framework, and we use these choice models in our simulation and data experiments.

Choice with no substitution

Here we let the parameters $\phi_1^k, \ldots, \phi_n^k$ specify a preference distribution over products, that is, $\phi_i^k \geq 0$ and $\sum_{i=1}^{n} \phi_i^k = 1$. Each customer selects a product according to that distribution. If they select a product that is out of stock then there is no substitution, they leave as a no-purchase:

$$f_i^{\text{ns}}(s(t), \boldsymbol{\phi}^k) = s_i(t)\phi_i^k.$$

The distribution $\boldsymbol{\phi}^k$ describes exactly the primary demand, and the parameter τ^k is not used.

Multinomial logit choice

The MNL is a popular choice model that derives from a random utility model. As in the previous model, $\boldsymbol{\phi}^k$ specifies a preference distribution over products. When an item goes out of stock, substitution takes place by transferring purchase probability to the other items proportionally to their original probability, including to the no-purchase option. In order to have positive probability of customers substituting to the no-purchase option, a proportion of arrivals must be no-purchases even when all items are in stock. We let $\tau^k/(1+\tau^k)$ be the no-purchase probability when all items are in stock, and obtain the MNL choice probabilities by normalizing with the preference vector $\boldsymbol{\phi}^k$ accordingly:

$$f_i^{\text{mnl}}(s(t), \boldsymbol{\phi}^k) = \frac{s_i(t)\phi_i^k}{\tau^k + \sum_{v=1}^n s_v(t)\phi_v^k}.$$

Vulcano et al (2012) show that the MNL model parameter τ^k is not identifiable when the arrival function is also unknown, and so they assume it to be a known, fixed parameter.

Single-substitution exogenous model

The exogenous model overcomes some shortcomings of the MNL choice model, and allows for the no-purchase option to be chosen only if there is a stock unavailability. According to the exogenous proportional substitution model (Kök and Fisher, 2007), a customer samples a first choice from the preference distribution $\boldsymbol{\phi}^k$. If that item is available, he or she purchases the item. If the first choice is not available, with probability $1-\tau^k$ the customer leaves as no-purchase. With the remaining τ^k probability, the customer picks a second choice according to a preference vector that has been re-weighted to exclude the first choice. Specifically, if the first choice was j then the probability of choosing i as the second choice is $\phi_i^k / \sum_{v \neq j} \phi_v^k$. If the second choice is

in stock it is purchased, otherwise the customer leaves as no-purchase. The formula for the purchase probability follows directly:

$$f_i^{\text{exo}}(s(t), \boldsymbol{\phi}^k, \tau^k) = s_i(t)\phi_i^k + s_i(t)\tau^k \sum_{j=1}^{n}(1 - s_j(t))\phi_j^k \frac{\phi_i^k}{\sum_{v \neq j} \phi_v^k}. \tag{6.3}$$

For this model, posterior distributions for both $\boldsymbol{\phi}^k$ and τ^k are inferred.

Nonparametric choice model

Nonparametric models often offer a lucid description of substitution behavior. Rather than being a probability vector as in the parametric models, here the parameter $\boldsymbol{\phi}^k$ is an ordered subset of the items $\{1, \ldots, n\}$. Customers purchase ϕ_1^k if it is in stock. If not, they purchase ϕ_2^k if it is in stock. If not, they continue substituting down $\boldsymbol{\phi}^k$ until they reach the first item that is available. If none of the items in $\boldsymbol{\phi}^k$ are available, they leave as a no-purchase. The purchase probability for this model is then

$$f_i^{\text{np}}(s(t), \boldsymbol{\phi}^k) = \begin{cases} 1 & \text{if } i = \min\{j \in \{1, \ldots, |\boldsymbol{\phi}^k|\} : s_{\phi_j^k}(t) = 1\} \\ 0 & \text{otherwise.} \end{cases} \tag{6.4}$$

Because this model requires all customers to behave exactly the same, it is most useful when customers are modeled as coming from a number of different segments k, each with its own preference ranking $\boldsymbol{\phi}^k$. This is precisely what we do in our model, as we describe in the next section. For the nonparametric model the rank orders for each segment $\boldsymbol{\phi}^k$ are fixed and it is the distribution of customers across segments that is inferred.

6.1.4 Segments and Mixtures of Choice Models

We model customers as each coming from one of K segments $k = 1, \ldots, K$, each with its own choice model parameters $\boldsymbol{\phi}^k$ and τ^k. Let $\boldsymbol{\theta}^\sigma$ be the customer segment distribution for store σ, with θ_k^σ the probability that an arrival at store σ belongs to segment k, $\theta_k^\sigma \geq 0$, and $\sum_{k=1}^{K} \theta_k^\sigma = 1$. As with other variables, we denote the

collection of segment distributions across all stores as $\boldsymbol{\theta}$. Similarly, we denote the collections of choice model parameters across all segments as $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$.

For the nonparametric choice model, each of these segments would have a different rank ordering of items and multiple segments are required in order to have a diverse set of preferences. For the MNL and exogenous choice models, customer segments can be used to borrow strength across multiple stores. All stores share the same underlying segment parameters $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$, but each store's arrivals are represented by a different mixing of these segments, $\boldsymbol{\theta}^\sigma$. This model allows us to use data from all of the stores for inferring the choice model parameters, while still allowing stores to differ from each other by having a different mixture of segments.

6.1.5 The Likelihood Model

We now describe the underlying model for how customer segments, choice models, stock levels, and the arrival function all interact to create transaction data. Consider store σ and time period l. Customers arrive according to the NHPP for this store. Let $\tilde{t}_1^{\sigma,l}, \ldots, \tilde{t}_{\tilde{m}^{\sigma,l}}^{\sigma,l}$ represent all of the arrival times; these are unobserved, as they may include no-purchases. Each arrival has probability θ_k^σ of belonging to segment k. They then purchase an item or leave as no-purchase according to the choice model f_i. If the j'th arrival purchases an item then we observe that purchase at time $\tilde{t}_j^{\sigma,l}$; if they leave as no-purchase we do not observe that arrival at all. The generative model for the observed data $\boldsymbol{t}$ is thus:

- For store $\sigma = 1, \ldots, S$:
  - For time period $l = 1, \ldots, L^\sigma$:
    - Sample customer arrival times $\tilde{t}_1^{\sigma,l}, \ldots, \tilde{t}_{\tilde{m}^{\sigma,l}}^{\sigma,l} \sim \text{NHPP}(\lambda(t \mid \boldsymbol{\eta}^\sigma), T)$.
    - For customer arrival $j = 1, \ldots, \tilde{m}^{\sigma,l}$:
      - Sample this customer's segment as $k \sim \text{Multinomial}(\boldsymbol{\theta}^\sigma)$.
      - Customer purchases item i, with probability $f_i(s(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k)$, or the no-purchase option with probability $1 - \sum_{i=1}^n f_i(s(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k)$.

· If item i purchased, add the time to $\boldsymbol{t}_i^{\sigma,l}$.

We now provide the likelihood function corresponding to this generative model.

Theorem 2. *The log-likelihood function of* $\boldsymbol{t}$ *is:*

$$\log p(\boldsymbol{t} \mid \boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) = \sum_{\sigma=1}^{S} \sum_{l=1}^{L^\sigma} \sum_{i=1}^{n} \left(\sum_{j=1}^{m_i^{\sigma,l}} \log \left(\tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l}) \right) - \tilde{\Lambda}_i^{\sigma,l}(0,T) \right),$$

where

$$\tilde{\lambda}_i^{\sigma,l}(t) = \lambda(t \mid \boldsymbol{\eta}^\sigma) \sum_{k=1}^{K} \theta_k^\sigma f_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k) \quad \textit{and} \quad \tilde{\Lambda}_i^{\sigma,l}(0,T) = \int_0^T \tilde{\lambda}_i^{\sigma,l}(t) dt.$$

The result is actually that which would be obtained if we treated the purchases for each item as independent NHPPs with rate $\tilde{\lambda}_i^{\sigma,l}(t)$, which is the purchase rate for item i incorporating stock availability and customer choice. In reality, however, they are not independent NHPPs inasmuch as they depend on each other via the stock function $s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l})$. The key element of the proof is that while the purchase processes depend on each other, they do not depend on the no-purchase arrivals.

We now provide an important lemma, and follow its proof with the proof of Theorem 2.

Lemma 4.

$$\Lambda(0,T \mid \boldsymbol{\eta}^\sigma) = \sum_{i=0}^{n} \tilde{\Lambda}_i^{\sigma,l}(0,T).$$

Proof.

$$\begin{aligned}
\Lambda(0,T \mid \boldsymbol{\eta}^\sigma) &= \int_0^T \lambda(t \mid \boldsymbol{\eta}^\sigma) dt \\
&= \int_0^T \sum_{i=0}^{n} \lambda(t \mid \boldsymbol{\eta}^\sigma) \pi_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^\sigma) dt \\
&= \int_0^T \sum_{i=0}^{n} \tilde{\lambda}_i^{\sigma,l}(t) dt \\
&= \sum_{i=0}^{n} \int_0^T \tilde{\lambda}_i^{\sigma,l}(t) dt
\end{aligned}$$

$$= \sum_{i=0}^{n} \tilde{\Lambda}_i^{\sigma,l}(0,T),$$

where the second line uses $\sum_{i=1}^{n} \pi_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^{\sigma}) = 1$. □

Proof of Theorem 2. We consider the density function for the complete arrivals $\tilde{\boldsymbol{t}}^{\sigma,l}$, which include both the observed arrivals $\boldsymbol{t}^{\sigma,l}$ as well as the unobserved arrivals that left as no-purchase, which we here denote $\boldsymbol{t}_0^{\sigma,l} = \left\{\boldsymbol{t}_{0,j}^{\sigma,l}\right\}_{j=1}^{m_0^{\sigma,l}}$. Let $f_0(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k) = 1 - \sum_{i=1}^{n} f_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k)$ be the probability that a customer of segment k chooses the no-purchase option. Also, let $\pi_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^{\sigma}) = \sum_{k=1}^{K} \theta_k^{\sigma} f_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k)$ be the probability that a randomly chosen arrival purchases product i, or the no-purchase $i = 0$. Finally, we set $\tilde{I}_j^{\sigma,l}$ equal to i if the customer at time $\tilde{t}_j^{\sigma,l}$ purchased item i, or 0 if this customer left as no-purchase. For store σ and time period l,

$$\begin{aligned}
&p(\boldsymbol{t}_0^{\sigma,l}, \boldsymbol{t}^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}, \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) \\
&\quad = \mathbb{P}\left(\text{no arrivals in } \left(\tilde{t}_{\tilde{m}^{\sigma,l}}^{\sigma,l}, T\right] \mid \tilde{\boldsymbol{t}}^{\boldsymbol{\sigma},\boldsymbol{l}}, \boldsymbol{\eta}^{\sigma}\right) p(\tilde{t}_1^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}) p(\tilde{I}_1^{\sigma,l} \mid \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}) \\
&\qquad \times \prod_{j=2}^{\tilde{m}^{\sigma,l}} p(\tilde{t}_j^{\sigma,l} \mid \tilde{t}_1^{\sigma,l}, \ldots, \tilde{t}_{j-1}^{\sigma,l}, \boldsymbol{\eta}^{\sigma}) p(\tilde{I}_j^{\sigma,l} \mid \tilde{t}_1^{\sigma,l}, \ldots, \tilde{t}_{j-1}^{\sigma,l}, \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}) \\
&\quad = \exp(-\Lambda(\tilde{t}_{\tilde{m}^{\sigma,l}}, T \mid \boldsymbol{\eta}^{\sigma})) \lambda(\tilde{t}_1^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}) \exp(-\Lambda(0, \tilde{t}_1^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma})) \\
&\qquad \times \pi_{\tilde{I}_1^{\sigma,l}}(s(\tilde{t}_1^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^{\sigma}) \\
&\qquad \times \prod_{j=2}^{\tilde{m}^{\sigma,l}} \lambda(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}) \exp(-\Lambda(\tilde{t}_{j-1}^{\sigma,l}, \tilde{t}_j^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma})) \pi_{\tilde{I}_j^{\sigma,l}}(s(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^{\sigma}) \\
&\quad = \exp(-\Lambda(0, T \mid \boldsymbol{\eta}^{\sigma})) \prod_{i=0}^{n} \prod_{j:\tilde{I}_j^{\sigma,l}=i} \lambda(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}) \pi_i(s(\tilde{t}_j^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}^{\sigma}) \\
&\quad = \exp(-\Lambda(0, T \mid \boldsymbol{\eta}^{\sigma})) \prod_{i=0}^{n} \prod_{j=1}^{m_i^{\sigma,l}} \lambda(t_{i,j}^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}) \pi_i(s(t_{i,j}^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{\theta}) \\
&\quad = \left(\exp(-\tilde{\Lambda}_0^{\sigma,l}(0,T)) \prod_{j=1}^{m_0^{\sigma,l}} \tilde{\lambda}_0^{\sigma,l}(t_{0,j}^{\sigma,l})\right) \left(\prod_{i=1}^{n} \exp(-\tilde{\Lambda}_i^{\sigma,l}(0,T)) \prod_{j=1}^{m_i^{\sigma,l}} \tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l})\right).
\end{aligned}$$

The second equality uses Lemma 2, and the final uses Lemma 4. We have then that

$$\begin{aligned}
p(\boldsymbol{t}^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}, \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) &= \int p(\boldsymbol{t}_0^{\sigma,l}, \boldsymbol{t}^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}, \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) d\boldsymbol{t}_0^{\sigma,l} \\
&= \left(\int \exp(-\tilde{\Lambda}_0^{\sigma,l}(0,T)) \prod_{j=1}^{m_0^{\sigma,l}} \tilde{\lambda}_0^{\sigma,l}(t_{0,j}^{\sigma,l}) d\boldsymbol{t}_0^{\sigma,l} \right) \\
&\quad \times \left(\prod_{i=1}^{n} \exp(-\tilde{\Lambda}_i^{\sigma,l}(0,T)) \prod_{j=1}^{m_i^{\sigma,l}} \tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l}) \right) \\
&= \prod_{i=1}^{n} \exp(-\tilde{\Lambda}_i^{\sigma,l}(0,T)) \prod_{j=1}^{m_i^{\sigma,l}} \tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l}),
\end{aligned}$$

as the quantity being integrated is exactly the density function for $m_0^{\sigma,l}$ arrivals from an NHPP with rate $\tilde{\lambda}_0^{\sigma,l}(t)$ over interval $[0,T]$, as seen in Lemma 3. Thus,

$$\begin{aligned}
\log p(\boldsymbol{t} \mid \boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) &= \sum_{\sigma=1}^{S} \sum_{l=1}^{L^{\sigma}} \log p(\boldsymbol{t}^{\sigma,l} \mid \boldsymbol{\eta}^{\sigma}, \boldsymbol{\theta}^{\sigma}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) \\
&= \sum_{\sigma=1}^{S} \sum_{l=1}^{L^{\sigma}} \sum_{i=1}^{n} \left(\sum_{j=1}^{m_i^{\sigma,l}} \log \left(\tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l}) \right) - \tilde{\Lambda}_i^{\sigma,l}(0,T) \right).
\end{aligned}$$

□

The quantity $\tilde{\Lambda}_i^{\sigma,l}(0,T)$ can be expressed analytically in terms of $\Lambda(0,T \mid \boldsymbol{\eta}^{\sigma})$ and thus computed efficiently. This is done by looking at each of the time intervals where the stock $s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l})$ is constant. Let the sequence of times $q_1^{\sigma,l}, \ldots, q_{Q^{\sigma,l}}^{\sigma,l}$ demarcate the intervals of constant stock. That is, $[0,T] = \bigcup_{r=1}^{Q^{\sigma,l}-1} [q_r^{\sigma,l}, q_{r+1}^{\sigma,l}]$ and $s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l})$ is constant for $t \in [q_r^{\sigma,l}, q_{r+1}^{\sigma,l})$ for $r = 1, \ldots, Q^{\sigma,l} - 1$. Then,

$$\begin{aligned}
\tilde{\Lambda}_i^{\sigma,l}(0,T) &= \int_0^T \tilde{\lambda}_i^{\sigma,l}(t) dt \\
&= \int_0^T \lambda(t \mid \boldsymbol{\eta}^{\sigma}) \sum_{k=1}^{K} \theta_k^{\sigma} f_i(s(t \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k) dt \\
&= \sum_{r=1}^{Q^{\sigma,l}-1} \left(\int_{q_r^{\sigma,l}}^{q_{r+1}^{\sigma,l}} \lambda(t \mid \boldsymbol{\eta}^{\sigma}) \sum_{k=1}^{K} \theta_k^{\sigma} f_i(s(q_r^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k) dt \right)
\end{aligned}$$

$$= \sum_{r=1}^{Q^{\sigma,l}-1} \left(\sum_{k=1}^{K} \theta_k^\sigma f_i(s(q_r^{\sigma,l} \mid \boldsymbol{t}^{\sigma,l}, \boldsymbol{N}^{\sigma,l}), \boldsymbol{\phi}^k, \tau^k) \right) \Lambda(q_r^{\sigma,l}, q_{r+1}^{\sigma,l} \mid \boldsymbol{\eta}^\sigma).$$

With this formula, the likelihood function can be computed for any parameterization $\lambda(t \mid \boldsymbol{\eta}^\sigma)$ desired so long as it is integrable.

6.1.6 Prior Distributions and the Log-Posterior

To do Bayesian inference we must specify a prior distribution for each of the latent variables: $\boldsymbol{\eta}$, $\boldsymbol{\theta}$, and $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$ as required by the choice model. The variables $\boldsymbol{\theta}$, $\boldsymbol{\phi}$, and $\boldsymbol{\tau}$ are all probability vectors, so the natural choice is to assign them a Dirichlet or Beta prior:

$$\begin{aligned} \boldsymbol{\theta} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) \\ \boldsymbol{\phi}^k &\sim \text{Dirichlet}(\boldsymbol{\beta}), \quad k = 1, \ldots, K \\ \tau^k &\sim \text{Beta}(\boldsymbol{\gamma}), \quad k = 1, \ldots, K. \end{aligned}$$

Here $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ are prior hyperparameters chosen by the modeler. If there is actually some expert knowledge about the choice models and segment distributions then it can be encoded in these hyperparameters. Otherwise, a natural choice is to use a uniform prior distribution by setting each of these hyperparameters to be a vector of ones. In our experiments, we used uniform priors. Similarly, for $\boldsymbol{\eta}$, a natural choice for the prior distribution is a uniform distribution for each element:

$$\eta_v^\sigma \sim \text{Uniform}(\boldsymbol{\delta}^v), \quad v = 1, \ldots, |\eta^\sigma|, \quad \sigma = 1, \ldots, S.$$

In our experiments we chose the interval $\boldsymbol{\delta}^v$ large enough to not be restrictive. For the Hill rate that we use in our data experiments, $|\eta^\sigma| = 3$.

We can then compute the prior probability as

$$p(\boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}) = p(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) \left(\prod_{k=1}^{K} p(\boldsymbol{\phi}^k \mid \boldsymbol{\beta}) p(\tau^k \mid \boldsymbol{\gamma}) \right) \left(\prod_{\sigma=1}^{S} \prod_{v=1}^{|\eta^\sigma|} p(\eta_v^\sigma \mid \boldsymbol{\delta}^v) \right)$$

$$\propto \left(\prod_{k=1}^{K} \left((\theta_k)^{\alpha_k - 1} \left(\tau^k\right)^{\gamma_1 - 1} \left(1 - \tau^k\right)^{\gamma_2 - 1} \prod_{i=1}^{n} \left(\phi_i^k\right)^{\beta_i - 1} \right) \right)$$
$$\times \left(\prod_{\sigma=1}^{S} \prod_{v=1}^{|\eta^\sigma|} \mathbf{1}_{\{\eta_v^\sigma \in [\delta_1^v, \delta_2^v]\}} \right). \tag{6.5}$$

Bayes' theorem yields:

$$\log p(\boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau} \mid \boldsymbol{t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{N}, T)$$
$$\propto \log p(\boldsymbol{t} \mid \boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau}, \boldsymbol{N}, T) + \log p(\boldsymbol{\eta}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\tau} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}), \tag{6.6}$$

and these two quantities are available in Theorem 2 and in (6.5). With this result we are now equipped to do posterior inference.

6.2 Stochastic Gradient MCMC Inference

We use Markov chain Monte Carlo (MCMC) techniques to simulate posterior samples, specifically the stochastic gradient Riemannian Langevin dynamics (SGRLD) algorithm of Patterson and Teh (2013). This algorithm uses a stochastic gradient that does not require the full likelihood function to be evaluated in every MCMC iteration, thus allowing posterior inference to be done even on very large transaction databases. Also, the SGRLD algorithm is well suited for variables on the probability simplex, as are $\boldsymbol{\theta}$, $\boldsymbol{\phi}^k$, and τ^k. Metropolis-Hastings sampling is difficult in this setting because it requires evaluating the full likelihood as well as dealing with the simplex constraints in the proposal distribution.

6.2.1 The Expanded-Mean Parameterization

We first transform each of the probability variables using the expanded-mean parameterization (Patterson and Teh, 2013). The latent variable $\boldsymbol{\theta}$ has as constraints $\theta_k \geq 0$ and $\sum_{k=1}^{K} \theta_k = 1$. Take $\tilde{\boldsymbol{\theta}}$ a random variable with support on $\mathbb{R}_+^K$. We give $\tilde{\boldsymbol{\theta}}$ a prior distribution consisting of a product of Gamma$(\alpha_k, 1)$ distributions:

$p(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{\alpha}) \propto \prod_{k=1}^{K} \tilde{\theta}_k^{\alpha_k - 1} \exp(-\tilde{\theta}_k)$. The posterior sampling is done over variables $\tilde{\boldsymbol{\theta}}$ by mirroring any negative proposal values about 0. We then compute $\theta_k = \tilde{\theta}_k / \sum_{r=1}^{K} \tilde{\theta}_r$. This parameterization is equivalent to sampling on $\boldsymbol{\theta}$ with a Dirichlet(α) prior, but does not require the probability simplex constraint. The same transformation is done to $\boldsymbol{\phi}^k$ and τ^k.

6.2.2 Riemannian Langevin Dynamics

Let $\boldsymbol{z} = \{\boldsymbol{\eta}, \tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\phi}}, \tilde{\boldsymbol{\tau}}\}$ represent the complete collection of transformed latent variables whose posterior we are inferring. From state $\boldsymbol{z}_w$ on MCMC iteration w, the next iteration moves to the state $\boldsymbol{z}_{w+1}$ according to

$$\boldsymbol{z}_{w+1} = \boldsymbol{z}_w + \frac{\epsilon_w}{2} \left(\operatorname{diag}(\boldsymbol{z}_w) \nabla \log p(\boldsymbol{z}_w \mid \boldsymbol{t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{N}, T) + \mathbf{1} \right) + \operatorname{diag}(\boldsymbol{z}_w)^{\frac{1}{2}} \psi,$$

where $\psi \sim \mathcal{N}(0, \epsilon_w I)$. The iteration performs a gradient step plus normally distributed noise, using the natural gradient of the log posterior, which is the manifold direction of steepest descent using the metric $G(\boldsymbol{z}) = \operatorname{diag}(\boldsymbol{z})^{-1}$. From (6.6),

$$\nabla \log p(\boldsymbol{z}_w \mid \boldsymbol{t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{N}, T) = \nabla \log p(\boldsymbol{t} \mid \boldsymbol{z}_w, \boldsymbol{N}, T) + \nabla \log p(\boldsymbol{z}_w \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}).$$

We use a stochastic gradient approximation for the likelihood gradient. On MCMC iteration w, rather than use all L^σ time periods to compute the gradient we use a uniformly sampled collection of time periods $\mathcal{L}_w^\sigma$. The gradient approximation is then

$$\nabla \log p(\boldsymbol{t} \mid \boldsymbol{z}_w, \boldsymbol{N}, T) \approx \sum_{\sigma=1}^{S} \frac{L^\sigma}{|\mathcal{L}_w^\sigma|} \sum_{l \in \mathcal{L}_w^\sigma} \sum_{i=1}^{n} \nabla \left(\sum_{j=1}^{m_i^{\sigma,l}} \log \left(\tilde{\lambda}_i^{\sigma,l}(t_{i,j}^{\sigma,l}) \right) - \tilde{\Lambda}_i^{\sigma,l}(0, T) \right).$$

The iterations will converge to the posterior samples if the step size schedule is chosen such that $\sum_{w=1}^{\infty} \epsilon_w = \infty$ and $\sum_{w=1}^{\infty} \epsilon_w^2 < \infty$ (Welling and Teh, 2011). In our simulations and experiments we used three time periods for the stochastic gradient approximations. We followed Patterson and Teh (2013) and took $\epsilon_w = a((1 + q/b)^{-c})$ with step size parameters chosen using cross-validation to minimize out-of-sample

perplexity. We drew 10,000 samples from each of three chains initialized at a local maximum *a posteriori* solution found from a random sample from the prior. We verified convergence using the Gelman-Rubin diagnostic after discarding the first half of the samples as burn-in (Gelman and Rubin, 1992), and then merged samples from all three chains to estimate the posterior.

6.3 Simulation Study

We use a collection of simulations to illustrate and analyze the model and the inference procedure. The simulation results show that, for a variety of rate functions and choice models, the posterior samples concentrate around the true, generating values. Furthermore, the posterior becomes more concentrated around the true values as the amount of data increases.

6.3.1 Homogeneous Rate and Exogenous Choice

The first set of simulations used the homogeneous rate function $\lambda(t \mid \boldsymbol{\eta}^\sigma) = \eta_1^\sigma$ and the exogenous choice model given in (6.3). We set the number of segments $K = 2$, the number of items $n = 3$, and set the choice model parameters to $\tau^1 = \tau^2 = 0.75$, $\boldsymbol{\phi}^1 = [0.75, 0.2, 0.05]$, and $\boldsymbol{\phi}^2 = [0.33, 0.33, 0.34]$. We simulated data from three stores $S = 3$, for each of which the segment distribution $\boldsymbol{\theta}^\sigma$ was chosen independently at random from a uniform Dirichlet distribution and the arrival rate η_1^σ was chosen independently at random from a uniform distribution on $[2, 4]$. For each store, we simulated 25 time periods, each of length $T = 1000$ and with the initial stock for each item chosen uniformly between 0 and 500, independently at random for each item, time period, and store. Purchase data were then generated according to the generative model in Section 6.1.5. This simulation was repeated 10 times, each with different random initializations of $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$. Inference was done with the prior hyperparameter for η_1^σ, $\boldsymbol{\delta}^1$, set to $[2, 4]$.

To illustrate the result of the inference, Figure 6-1 shows the posterior density for $\boldsymbol{\theta}$ for one of the simulations, as estimated by MCMC sampling. The figure shows that

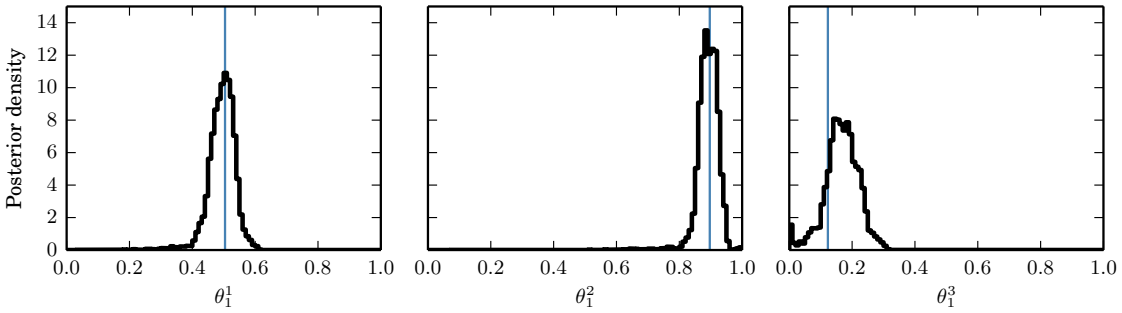

Figure 6-1: Normalized histograms of posterior samples of θ_1^σ for each of the three stores used in the simulation. The vertical line indicates the true value.

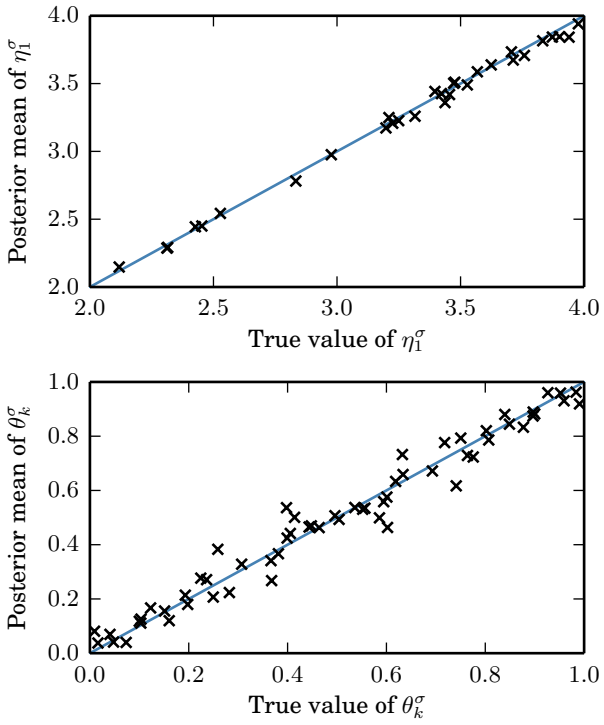

Figure 6-2: Markers in the top panel show, for each randomly chosen value of η_1^σ used in the set of simulations (3 stores $\times$ 10 simulations), the corresponding estimate of the posterior mean. The bottom panel shows the same result for each value of θ_k^σ used (3 stores $\times$ 2 segments $\times$ 10 simulations).

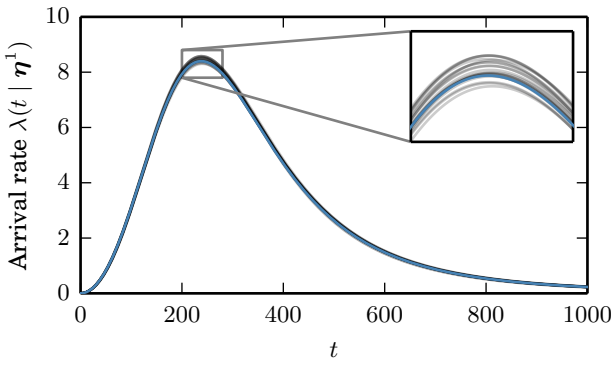

Figure 6-3: Each gray line is the rate function evaluated using a $\boldsymbol{\eta}^1$ randomly sampled from the posterior, with a total of 20 such samples. The blue line is the true rate function for this simulation.

the posterior samples are concentrated around the true values. Figure 6-2 shows the posterior means estimated from the MCMC samples across all of the 10 repeats of the simulation, showing that across the full range of parameter values used in these simulations the posterior mean was close to the true value.

6.3.2 Hill Rate and Exogenous Choice

In a second set of simulations, we used the same design as the first set but replaced the homogeneous arrival rate with the Hill arrival rate, given in (6.1). We did only one simulation, with the rate function parameters $\boldsymbol{\eta}^\sigma = [3000, 3, 300]$ to obtain a mean rate similar to that of the simulations in the previous section. In the inference, we used prior hyperparameters $\boldsymbol{\delta}^1 = [2000, 4000]$, $\boldsymbol{\delta}^2 = [2, 4]$, and $\boldsymbol{\delta}^3 = [200, 400]$.

Figure 6-3 shows posterior samples of the rate function $\lambda(t \mid \boldsymbol{\eta}^1)$. The posterior estimates of the rate function closely match the rate function used to generate the data.

6.3.3 Hill Rate and Nonparametric Choice

In the final set of simulations we use the Hill rate function with the nonparametric choice function from (6.4), with 3 items. We used all sets of preference rankings of size 1 and 2, which for 3 items requires a total of 9 segments. We simulated data for

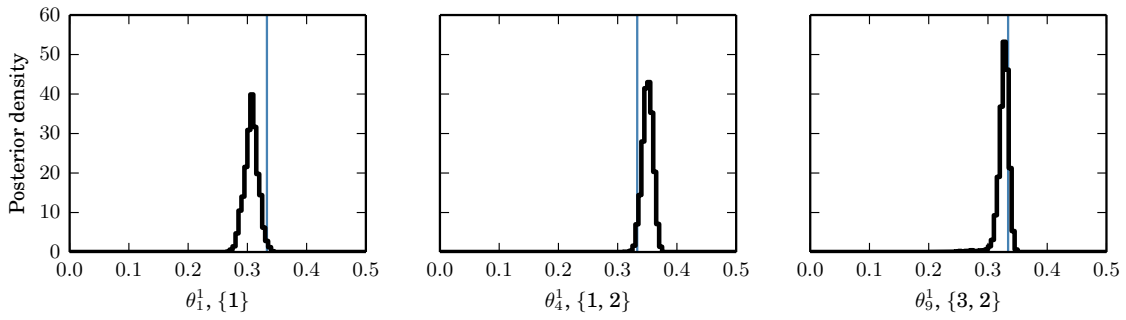

Figure 6-4: Posterior density for the non-zero segment proportions from a simulation with nonparametric choice. The corresponding ordering $\boldsymbol{\phi}^k$ is given below each panel.

a single store, with the segment proportion θ_k^1 set to 0.33 for preference rankings $\{1\}$, $\{1, 2\}$, and $\{3, 2\}$. The segment proportions for the remaining 6 preference rankings were set to zero. With this simulation we also study the effect of the number of time periods used in the inference, L^1. L^1 was taken from $\{5, 10, 25, 50, 100\}$, and for each of these values 10 simulations were done.

Figure 6-4 shows the posterior densities for the non-zero segment proportions θ_k^1, for one of the simulations with $L^1 = 25$. The posterior densities for the other six segment proportions were all concentrated near zero. Figure 6-5 describes how the posterior depends on the number of time periods. The top panel shows that the posterior mean tends closer to the true value as more data are made available. The bottom panel shows the actual concentration of the posterior, where the interquartile range of the posterior decreases with the number of time periods. Because we use a stochastic gradient approximation, using more time periods came at no additional computational cost: We used 3 time periods for each gradient approximation regardless of the available number.

6.4 Data Experiments

We now provide the results of the model applied to real transaction data. We used the real data to evaluate the predictive power of the model, and to compute a posterior distribution of lost sales due to stockouts.

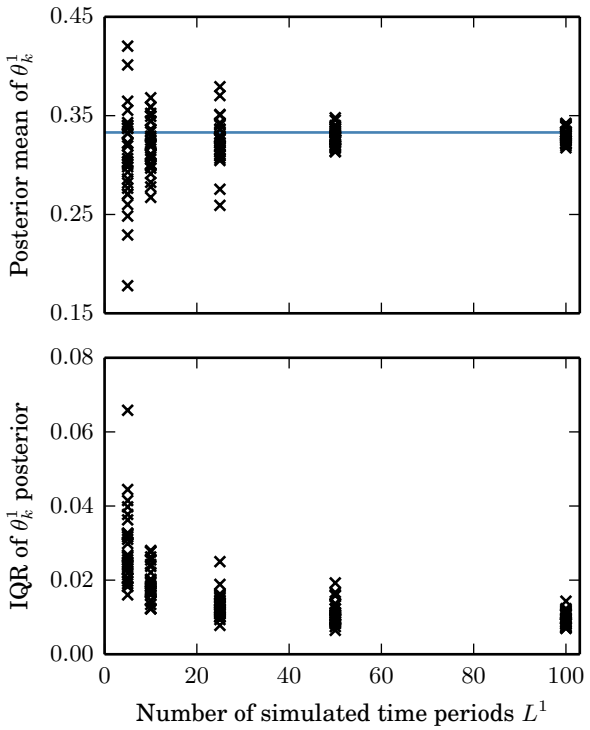

Figure 6-5: Each marker corresponds to the posterior distribution for θ_k^1 from a simulation with the corresponding number of time periods, across the 3 values of k where the true value equaled 0.33. The top panel shows the posterior mean for each of the simulations across the different number of time periods. The bottom panel shows the interquartile range (IQR) of the posterior.

We obtained one semester of sales data from the bakery at 100 Main Marketplace, a cafe located at MIT. The data were for a collection of breakfast pastries (bagel, scone, and croissant) and for a collection of cookies (oatmeal, double chocolate, and chocolate chip). The data set included all purchase times for 151 days; we treated each day as a time period. For the breakfast pastries the time period was from 7:00 a.m. to 2:00 p.m., and for the cookies the time period was from 11:00 a.m. to 7:00 p.m. The breakfast pastries comprised a total of 3869 purchases, and the cookies comprised 4084 purchases. Stock data were not available, only purchase times, so for the purpose of these experiments we set the initial stock for each time period equal to the number of purchases for the time period - thus every item was treated as stocked out after its last recorded purchase. This is a reasonable assumption given that these are perishable baked goods, whose stock levels are designed to stock out by the end of the day.

The empirical purchase rates for the two sets of items, shown in Figures 6-6 and 6-9, were markedly nonhomogeneous, so we used the Hill rate function from (6.1). For all of the data experiments we took the rate prior hyperparameters to be $\boldsymbol{\delta}^1 = [0, 200]$, $\boldsymbol{\delta}^2 = [1, 10]$, and $\boldsymbol{\delta}^3 = [0, 1000]$, which we found to be a large enough range so as to be unrestrictive. For each set of items we fit the model using both the exogenous and nonparametric choice models.

6.4.1 Inferring Demand for Breakfast Pastries

We began by fitting the breakfast pastry data using the nonparametric choice model. Figure 6-6 shows the actual purchase times in the data set across all three items, along with 20 random posterior samples from the model's predicted average purchase rate over all time periods, which equals

$$\frac{1}{151} \sum_{l=1}^{151} \sum_{i=1}^{3} \tilde{\lambda}_i^{1,l}. \tag{6.7}$$

The purchase rate shows a significant morning rush, as is expected for these types of items at a bakery.

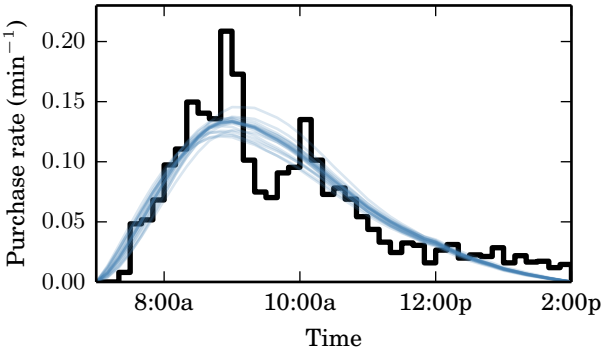

Figure 6-6: In black is a normalized histogram of the purchase times for the breakfast pastries, across all 151 days. Each blue line is a posterior sample for the model fit of this quantity, given in (6.7).

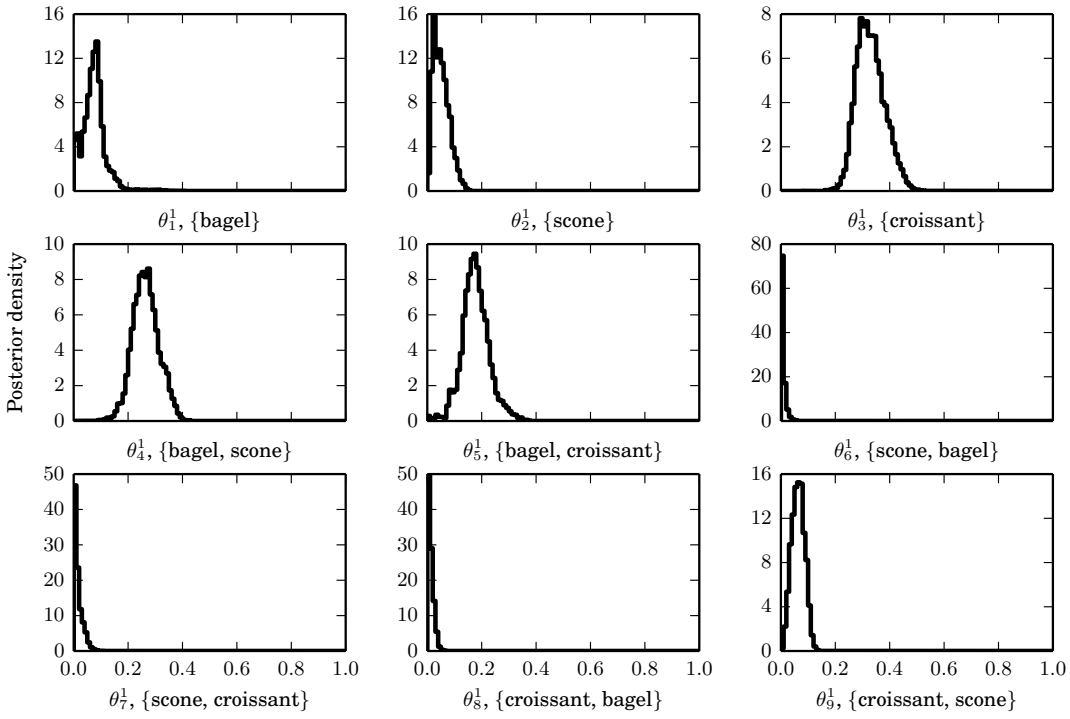

Figure 6-7: Normalized histograms of posterior samples for each segment proportion, for the breakfast pastries with the nonparametric choice model. The corresponding ordered list for each segment is indicated.

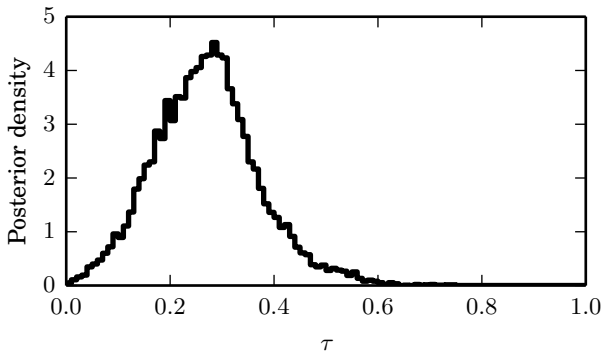

Figure 6-8: Normalized histogram of posterior samples of the exogenous choice model substitution rate, for the breakfast pastry data.

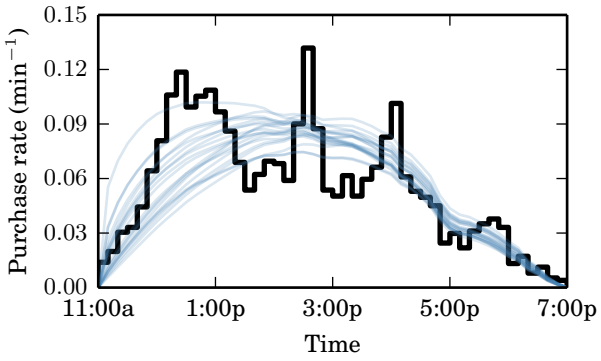

Figure 6-9: A normalized histogram of purchase times for the cookies, across time periods, along with posterior samples for the model's corresponding predicted purchase rate.

Figure 6-7 shows the posterior densities for the segment probabilities $\boldsymbol{\theta}$. The densities indicate that customers whose first choice is bagel are generally willing to substitute, those whose first choice is croissant less so, and customers seeking a scone are generally unwilling to substitute.

The model was also fit using the exogenous choice model, with $K = 1$ customer segment. The posterior density for the substitution rate τ^1 is given in Figure 6-8.

6.4.2 Inferring Demand for Cookies

We then fit the model to the cookie dataset using the nonparametric choice model. The empirical average purchase rate is given in Figure 6-9, along with 20 posterior

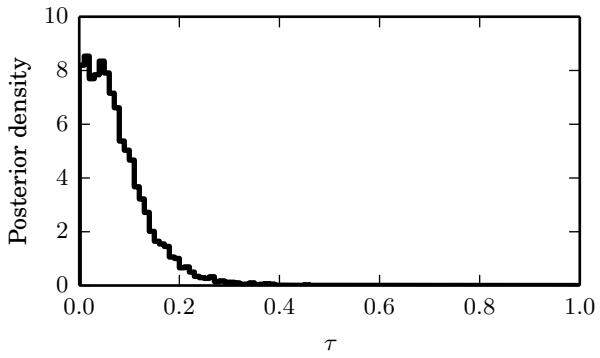

Figure 6-10: Normalized histogram of posterior samples of the exogenous choice model substitution rate, for the cookie data.

samples for the model's predicted average purchase rate from (6.7). The purchase rate shows a lunch time rush, followed by a sustained afternoon rate that finally tapers off in the evening. There are also significant rushes during the periods between afternoon classes. The Hill rate function that we use is not able to capture these afternoon peaks, however the model can incorporate any integrable rate function. Given a rate function that can produce three peaks, the inference would proceed in the same way.

The uncertainty in the posterior is clear from the variance in the samples in Figure 6-9. This uncertainty is the motivation for using the full posterior in making predictions, as described in Section 6.0.2.

The model was also fit using the exogenous choice model, and the density for $\boldsymbol{\tau}$ is given in Figure 6-10.

6.4.3 An Evaluation of Predictive Performance

Now that we have fit the model to data, it is important to establish that it has predictive power. We evaluated the predictive power of the model by predicting out-of-sample purchase counts during periods of varying stock availability. We took 80% of the time periods (120 time periods) as training data and did posterior inference. The remaining 31 time periods were held out as test data. We considered each possible level of stock unavailability, *i.e.*, $s = [1, 0, 0]$, $s = [0, 1, 0]$, etc. For each stock level, we found all of the time intervals in the test periods with that stock. The prediction

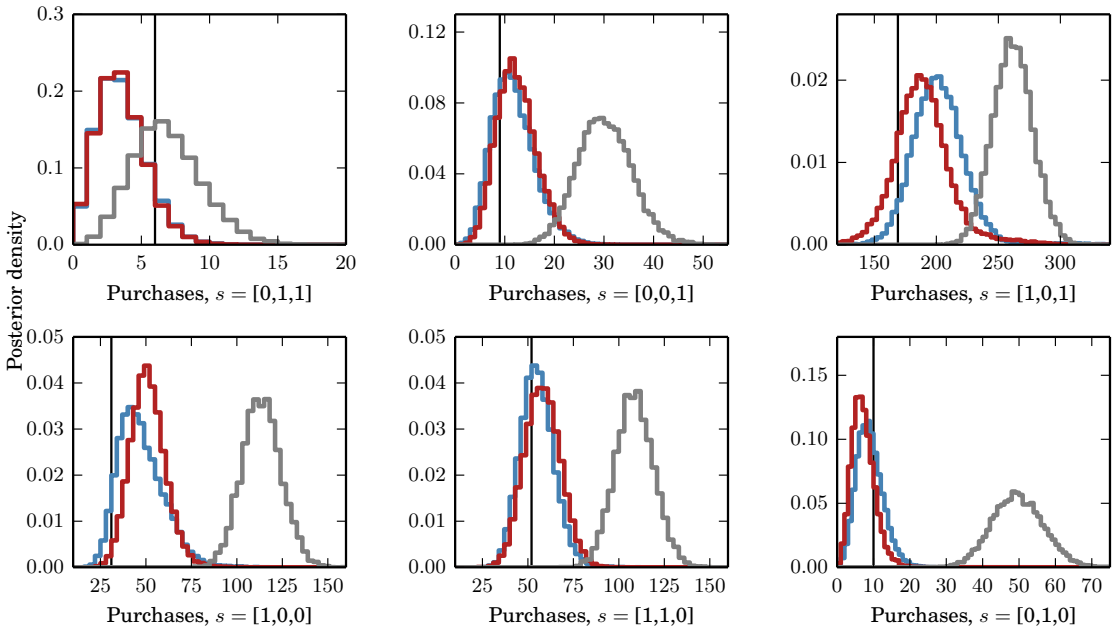

Figure 6-11: Posterior densities for the number of purchases during test set intervals with the indicated stock availability for items [bagel, scone, croissant]. The density in blue is for the nonparametric choice, red is for the exogenous choice, and gray is for a homogeneous arrival rate with MNL choice. The vertical line indicates the true value.

task was, given only the time intervals and the corresponding stock level, to predict the total number of purchases that took place during those time intervals in the test periods. The actual number of purchases is known and thus predictive performance can be evaluated.

This is a meaningful prediction task because good performance requires being able to accurately model exactly the two main components of our model: the arrival rate as a function of time, and how the actual purchases then depend on the stock. We did this task using the nonparametric and exogenous choice models as done in Sections 6.4.1 and 6.4.2. We also did the prediction task using the maximum likelihood model with a homogeneous arrival rate and the MNL choice model, which is similar to the model used by Vulcano et al (2012). For the MNL model we set $\tau^1 = 0.1$, as it cannot be inferred.

Posterior densities for the predicted counts for the breakfast pastries are given in Figure 6-11. These were obtained as described in Section 6.0.2. Despite their very

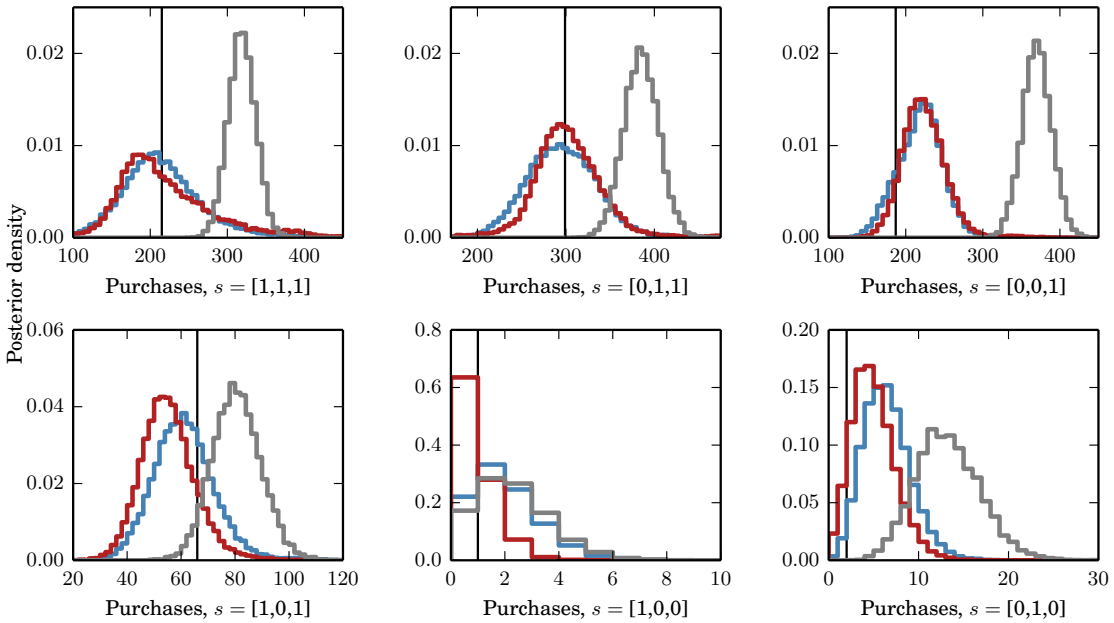

Figure 6-12: Posterior densities for the number of purchases during test set intervals with the indicated stock availability for cookies [oatmeal, double chocolate, chocolate chip]. The density in blue is for the nonparametric choice, red is for the exogenous choice, and gray is for a homogeneous arrival rate with MNL choice. The vertical line indicates the true value.

different natures, the predictions made by the exogenous and nonparametric models are quite similar, and are both consistent with the true values for all stock levels. The model with a homogeneous arrival rate and MNL choice is unable to accurately predict the purchase rates, most likely because of the poor model for the arrival rate. Figure 6-12 shows the same results for the cookies data.

6.4.4 Lost Sales Due to Stockouts

Once the model parameters have been inferred, we can estimate what the sales would have been had there not been any stockouts. We estimated posterior densities for the number of purchases of each item across 151 time periods, with full stock. In Figures 6-13 and 6-14 we compare those densities to the actual number of purchases in the data, for the cookies and breakfast pastry data respectively.

For each of the three cookies, the actual number of purchases was significantly less

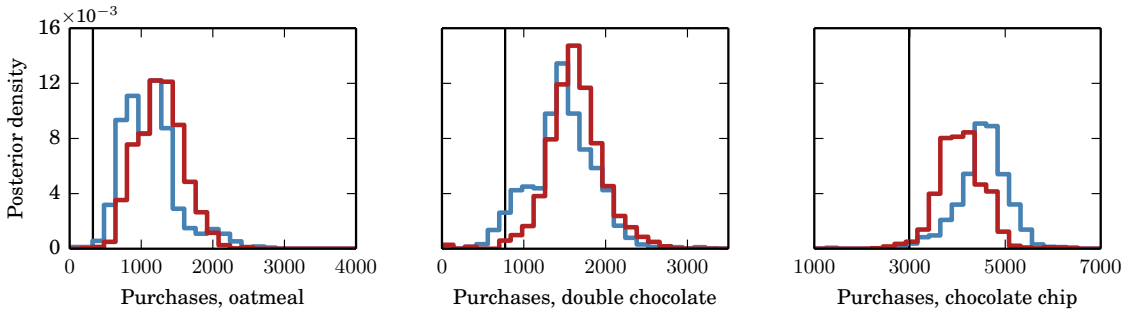

Figure 6-13: For the cookie data, posterior densities for the number of purchases during all periods, if there had been no stockouts. The blue density is the result with the nonparametric choice model, and the red with the exogenous. The vertical line indicates the number of purchases in the data.

than the posterior density for purchases with full stock, indicating that there were substantial lost sales due to stock unavailability. With the nonparametric model, the difference between the full-stock posterior mean and the actual number of purchases was 791 oatmeal cookies, 707 double chocolate cookies, and 1535 chocolate chip cookies.

Figure 6-14 shows the results for the breakfast pastries. Here the results do not support substantial lost sales due to stockouts. For the nonparametric model, the 95% credible interval for the full-stock number of bagel purchases is $1945 - 2951$, which contains the actual value of 2126 and so is not indicative of lost sales. The number of scone purchases also lies within the full-stock 95% credible interval. Only for croissants does the actual number of purchases fall outside the 95% credible interval, with a difference of 531 croissants between the full-stock posterior mean and the observed purchases.

Figures 6-8 and 6-10 give some insight into the different impact of stockouts on sales for the two sets of items. These figures show the posterior densities for the exogenous model substitution rate τ^1, for the breakfast pastries and cookies respectively. The posterior mean of τ^1 for the breakfast pastries was 0.27, whereas for the cookies it was 0.08. These results indicate that customers are much less willing to substitute cookies, hence the lost sales.

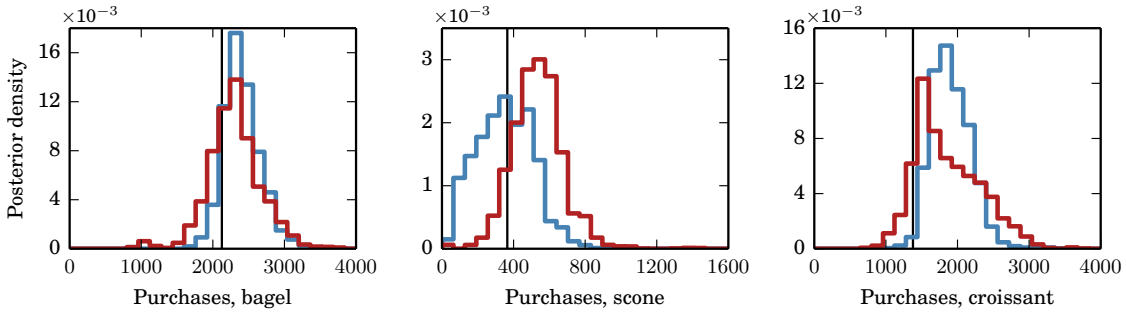

Figure 6-14: For the breakfast pastry data, posterior densities for the number of purchases during all periods, if there had been no stockouts. The blue density is the result with the nonparametric choice model, and the red with the exogenous. The vertical line indicates the number of purchases in the data.

6.5 Discussion

We have developed a Bayesian model for inferring primary demand and consumer choice in the presence of stockouts. The model can incorporate a realistic model of the customer arrival rate, and is flexible enough to handle a variety of different choice models. Our model is closely related to models like latent Dirichlet allocation, used in the machine learning community for topic modeling (Blei et al, 2003). Variants of topic models are regularly applied to very large text corpora, with a large body of research on how to effectively infer these models. That research was the source of the stochastic gradient MCMC algorithm that we used, which allows inference from even very large transaction databases. In our data experiments, sampling took just a few minutes on a standard laptop computer.

The simulation study showed that when data are actually generated from the model, we are able to recover the true, generating values. They further showed that the posterior bias and variance decrease as more data are made available, an improvement without any additional computational cost due to the stochastic gradient. We applied the model and inference to real sales transaction data from a local bakery. The daily purchase rate in the data was clearly nonhomogeneous, with a rush of purchases. The rush of purchases illustrates the importance of modeling nonhomogeneous arrival rates in many retail settings. In a prediction task that required accurate modeling

of both the arrival rate and the choice model, we showed that the model was able to make accurate predictions and significantly outperformed the baseline approach.

Finally, we showed how the model can be used to estimate a specific quantity of interest: lost sales due to stockouts. For bagels and scones there was no indication of lost sales due to stockouts, whereas for cookies the posterior provided evidence of substantial lost sales. The model and inference procedure we have developed provide a new level of power and flexibility that will aid decision makers in using transaction data to make smarter decisions.

Chapter 7

Bayesian Sets and Information Retrieval

Informed decision making requires information. There are many cases where important information is freely available on the Internet, but is fragmented over many different sites. Consider, for example, the task of open-ended list aggregation, inspired by the collective intelligence problem of finding all planned events in a city. There are many online “experts" that list Boston events, such as Boston.com or Yelp, however these lists are incomplete. As an example of the difficulties caused by information fragmentation, traffic in parts of greater Boston can be particularly bad when there is a large public event such as a street festival or fundraising walk. Even though these events are planned well in advance, the lack of a central list of events makes it hard to avoid traffic jams, and the number of online sources makes it difficult to compile a complete list manually.

As the amount of information on the Internet continues to grow, it becomes increasingly important to be able to compile information automatically in a fairly complete way, for any given domain. The development of general methods that automatically aggregate this kind of collective knowledge is a vital area of current research, with the potential to positively impact the spread of useful information to users across the Internet.

Our contribution in this chapter is a real system for growing lists of relevant items

from a small “seed" of examples by aggregating information across many internet experts. We provide an objective evaluation of our method to show that it performs well on a wide variety of list growing tasks, and significantly outperforms existing methods. We provide some theoretical motivation by giving bounds for the Bayesian Sets method used within our algorithm. None of the components of our method are particularly complicated; the value of our work lies in combining these simple ingredients in the right way to solve a real problem.

There are two existing, publicly available methods for growing a list of items related to a user-specified seed. The first was introduced ten years ago on a large scale by Google Sets, which is accessible via Google Spreadsheet. The second is a more recent online system called Boo!Wa! (http://boowa.com), which is similar in concept to Google Sets. In our experiments, we found that Boo!Wa! is a substantial advance above Google Sets, and the algorithm introduced here is a similarly sized leap in technology above Boo!Wa!. In a set of 50 experiments shown in Section 7.4, the lower 25th percentile of our performance was better than the median performance of both Google Sets and Boo!Wa! in Precision@10 and average precision. More generally, our work builds on “search” and other work in information retrieval. Search engines locate documents containing relevant information, but to produce a list one would generally need to look through the webpages and aggregate the information manually. We build on the speed of search, but do the aggregation automatically and in a much more complete way than a single search.

7.1 Algorithm for Retrieval and Aggregation

Algorithm 3 gives an outline of the list growing algorithm, which we now discuss in detail.

Source discovery: We begin by using the seed items to locate sites on the Internet that serve as expert sources for other relevant items. We use a combinatorial search strategy that relies on the assumption that a site containing at least two of the seed items likely contains other items of interest. Specifically, for every pair of seed items,

Algorithm 3: Outline of the list growing algorithm

Input: A list of seed items
Output: A ranked list of new items related to the seed items
for as many iterations as desired **do**
  for each pair of seed items **do**
    Source discovery: Find all sites containing both items
    for each source site **do**
      List extraction: Find all items on the site that are represented similarly to the seed items
    end for
  end for
  for each discovered item **do**
    Feature space: Using a search with the item as the query, construct a binary feature vector of domains where the item is found
    Ranking: Score the item according to the seed using Bayesian Sets
  end for
  Implicit feedback: Add the highest-ranked non-seed item to the seed
end for

we search for all websites that contain both of the items; this step takes advantage of the speed of "search."

In some cases, seed items may appear together in several contexts. For example, suppose one were to grow a list of Category 5 Atlantic hurricanes with "Hurricane Katrina" and "Hurricane Emily" in the seed. In addition to being Category 5 hurricanes, both of these hurricanes were in the 2005 hurricane season, and are found together on lists of 2005 Atlantic hurricanes. A search for sites containing both of these seed items would then recover both relevant sources on Category 5 hurricanes, as well as irrelevant sources on 2005 hurricanes. When more than two seed items are available, the context can be constrained by requiring source sites to contain all seed items, as opposed to just pairs. While this can potentially reduce the number of incorrect items, it could also miss relevant items that are found only on fragmented lists with little overlap. With our pairwise search strategy, our primary goal is complete coverage of relevant items, and we use the later ranking step to push the incorrect items to the bottom of the list. In Section 7.4.2 we explore experimentally how additional seed items constrain the context when ranking, and experimental results in Section 7.4.3 provide additional motivation for source discovery with pairs.

This step requires submitting the query *"term1" "term2"* to a search engine. In our experiments we used Google as the search engine, but any index would suffice. We retrieved the top 100 results.

List extraction: The output of the combinatorial search is a collection of source sites, each of which contains at least two seed items. We then extract all of the new items from each of these sites. Here our strategy relies on the assumption that human experts organize information on the Internet using HTML tags. For each site found with the combinatorial search, we look for HTML tags around the seed items. We then find the largest set of HTML tags that are common to both seed items (for this site) and extract all items on the page that use the same HTML tags. In some situations, this strategy can result in noisy lists because it allows any HTML tags, including generic ones like <b> and <a>. An alternative strategy is to limit item extraction to list-specific HTML tags like <li>, and we explore this and related strategies experimentally in Section 7.4.2. As before, our goal at this step is complete coverage of relevant items, so we allow all HTML tags and use the later ranking step to ensure that the noise is pushed to the bottom of the list.

The following lines of HTML illustrate the extraction process:

```
<h2><b><a href="example1.com"> Boston Harborfest</a></b></h2>
<b><a href="example2.com"> Jimmy fund scooper bowl </a></b>
<b><a href ="example3.com"> the Boston Arts Festival 2012</a></b>
<h3><b><a href="example4.com">Boston bacon takedown</a></b></h3>
<a href="example5.com"> Just a url </a>
```

For each of the two seed items used to discover this source, we search the HTML for the pattern:

<largest set of tags>(≤ 5 words) seed item (≤ 5 words)<matching end tags>.

In the above example, if the first seed item is "Boston arts festival," then it matches the pattern with the HTML tags: <b><a>. If the second seed item is "Boston harborfest," it matches the pattern with HTML tags: <h2><b><a>. We then find

the largest set of HTML tags that are common to both seed items, for this site. In this example, "Boston arts festival" does not have the <h2> tag, so the largest set of common tags is: <b><a>. If there are no HTML tags common to both seed items, we discard the site. Otherwise, we extract all items on the page that use the same HTML tags. In this example, we extract everything with both a <b> and an <a> tag, which means "Jimmy fund scooper bowl" and "Boston bacon takedown," but not "Just a url."

In our experiments, to avoid search spam sites with extremely long lists of unrelated keywords, we reject sources that return more than 300 items. We additionally applied a basic filter rejecting items of more than 60 characters or items consisting of only numbers and punctuation. No other processing was done.

Feature Space: At this point the algorithm has discovered a collection of lists, each from a different source. We now combine these lists so that the most relevant information is at the top of the final, merged list. To determine which of the discovered items are relevant, we construct a feature space in which to compare them to the seed items. Specifically, for each discovered item x, we construct a binary feature vector where each feature j corresponds to an internet domain (like boston.com or mit.edu), and $x_j = 1$ if item x can be found on internet domain j. This set of internet domains is found using a search engine with the item as the query.

The assumption behind this strategy is that related items should be found on a set of mainly overlapping domains, so we determine relevance by looking for items that cluster well with the seed items in the feature space. Constructing this feature space requires an additional search query for each discovered item. An alternative strategy that does not require additional search queries is to construct the feature space using only the source sites, that is, sites containing at least two seed items. This strategy, however, does not provide a way to distinguish between items that are found often in general, both with or without seed items, and items that are found often specifically with seed items. We compare these two approaches empirically in Section 7.4.2.

We do separate Google searches for each item we have extracted to find the set of webpages containing it. We use quotes around the query term, discard results

when Google's spelling correction system modifies the query, and retrieve the top 300 search results.

Ranking: The Bayesian Sets algorithm (Ghahramani and Heller, 2005) ranks items according to the likelihood that they form a cluster with the seed, based on a probabilistic model for the feature space. Specifically, we suppose that each feature (in general, x_j) is a Bernoulli random variable with probability θ_j of success: $x_j \sim \text{Bern}(\theta_j)$. Following the typical Bayesian practice, we assign a Beta prior to the probability of success: $\theta_j \sim \text{Beta}(\alpha_j, \beta_j)$. Bayesian Sets assigns a score $f(x)$ to each item x by comparing the likelihood that x and the seed $S = \{x^1, \ldots, x^m\}$ were generated by the same distribution to the likelihood they are independent:

$$f(x) := \log \frac{p(x, S)}{p(x)p(S)}. \tag{7.1}$$

Suppose there are N features: $x \in \{0, 1\}^N$. Because of the Bernoulli-Beta conjugacy, Ghahramani and Heller (2005) show that (7.1) has an analytical form under the assumption of independent features. However, the score given in Ghahramani and Heller (2005) can be arbitrarily large as m (the number of seed examples) increases. We prefer a normalized score because it leads to guarantees that the results are stable, or not too sensitive to any one seed item, as we show in Section 7.2. We use the following scoring function which differs from that in Ghahramani and Heller (2005) only by constant factors and normalization:

$$f_S(x) := \frac{1}{Z(m)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j}, \tag{7.2}$$

where

$$Z(m) := N \log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)$$

and $\gamma_{\min} := \min_j \min\{\alpha_j, \beta_j\}$ is the weakest prior hyperparameter. The normalization ensures that $f_S(x) \in [0, 1]$, as we now show.

Lemma 5. $0 \leq f_S(x) \leq 1$.

Proof. It is easy to see that $f_S(x) \geq 0$. To see that $f_S(x) \leq 1$,

$$\begin{aligned}
&\max_{S,x} f_S(x) \\
&\quad = \frac{1}{Z(m)} \max_{S,x} \sum_{j=1}^{N} \left(x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j} \right) \\
&\quad \leq \frac{1}{Z(m)} \sum_{j=1}^{N} \max_{x_j, x_j^1, \ldots, x_j^m} \left(x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j} \right) \\
&\quad = \frac{1}{Z(m)} \sum_{j=1}^{N} \max \left\{ \max_{x_j^1, \ldots, x_j^m} \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j}, \max_{x_j^1, \ldots, x_j^m} \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j} \right\} \\
&\quad = \frac{1}{Z(m)} \sum_{j=1}^{N} \max \left\{ \log \frac{\alpha_j + m}{\alpha_j}, \log \frac{\beta_j + m}{\beta_j} \right\} \\
&\quad = \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\min\{\alpha_j, \beta_j\} + m}{\min\{\alpha_j, \beta_j\}} \\
&\quad \leq \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\gamma_{\min} + m}{\gamma_{\min}} \\
&\quad = 1.
\end{aligned}$$

□

Given the seed and the prior, (7.2) is linear in x, and can be formulated as a single matrix multiplication. When items are scored and then ranked using Bayesian Sets, the items that were most likely to have been generated by the same distribution as the seed items are put high on the list.

As is typically the case in Bayesian analysis, there are several options for selecting the prior hyperparameters α_j and β_j, including the non-informative prior $\alpha_j = \beta_j = 1$. Heller and Ghahramani (2006) recommend using the empirical distribution. Given n items to score $x^{(1)}, \ldots, x^{(n)}$, we let

$$\alpha_j = \kappa_1 \left(\frac{1}{n} \sum_{i=1}^{n} x_j^{(i)} \right), \qquad \beta_j = \kappa_2 \left(1 - \frac{1}{n} \sum_{i=1}^{n} x_j^{(i)} \right). \tag{7.3}$$

The first term in the sum in (7.2) corresponds to the amount of score obtained by

x for the co-occurrence of feature j with the seed, and the second term corresponds to the amount of score obtained for the non-occurrence of feature j with the seed. When $\alpha_j = \beta_j$, the amount of score obtained when x_j and the seed both occur is equivalent to the amount of score obtained when x_j and the seed both do not occur. Increasing β_j relative to α_j gives higher emphasis to co-occurring features. This is useful when the feature vectors are very sparse, as they are here; thus we take $\kappa_2 > \kappa_1$. In all of our experiments we took $\kappa_2 = 5$ and $\kappa_1 = 2$, similarly to that done in Heller and Ghahramani (2006).

There are a number of alternative ranking algorithms that could be considered within this same framework, including non-probabilistic metrics. Stand-alone comparisons of Bayesian Sets and several alternatives done in Heller and Ghahramani (2006) provide empirical motivation for its use. The generalization bounds provided in Section 7.2 provide theoretical motivation for using Bayesian Sets in high-dimensional settings with correlated features.

Feedback: Once the lists have been combined, we continue the discovery process by expanding the seed. A natural, unsupervised way of expanding the seed is to add the highest ranked non-seed item into the seed. Though not done here, one could also use a domain expert or even crowdsourcing to quickly scan the top ranked items and manually expand the seed from the discovered items. Then the process starts again; we do a combinatorial search for websites containing all pairs with the new seed item(s), extract possible new items from the websites, etc. We continue this process for as many iterations as we desire. In practice, there is no need to repeat searches for pairs of seed items that have already been explored in previous iterations. Also, we track which sites have been visited during source discovery and do not revisit these sites in later iterations. To avoid filling the seed with duplicate items like "Boston arts festival" and "The boston arts festival 2012," in our implicit feedback we do not add items to the seed if they are a sub- or super-string of a current seed item.

All of the components of our approach scale well to work fast on large problems: Item discovery is done with a regular expression, ranking is done with a single matrix multiplication, and all of the remaining steps require simple web queries. We used

Google as the search engine for our experiments, however, Google creates an artificial restriction on the number of queries one can make per minute. This, and the speed of downloading webpages for item extraction, are the only two slow steps in our method - with the webpages already downloaded, the whole process took on average 1.9 seconds in our experiments in Section 7.4.1 (on a laptop with a 2.6GHz i5 processor). Both issues would be fixed if we had our own web index and search engine, for instance, if we had direct access to Google's resources. Google or another search engine could implement this method and it would work in real time, and would give a substantial advantage over the state-of-the-art. In the meantime, companies or researchers wanting to curate master lists on specific topics can implement our method using one of the available search engines, as we do in our experiments in Section 7.4.

7.2 Generalization Bounds for Bayesian Sets

The derivation for Bayesian Sets assumes independent features. In this application, features are internet domains, which are almost certainly correlated. Because Bayesian sets is the core of our method, we motivate its use in this application by showing that even in the presence of arbitrary dependence among features, prediction ability can be guaranteed as the sample size increases. We consider an arbitrary distribution from which the seed S is drawn, and prove that as long as there are a sufficient number of items, x will on expectation score highly if it is from the same distribution as the seed S. Specifically, we provide a lower bound for $\mathbb{E}_x\left[f_S(x)\right]$ that shows that the expected score of x is close to the score of S with high probability. We provide two results that use different proof techniques. This is a case where statistical learning theory provides theoretical backing for a Bayesian method.

Theorem 3. *Suppose $x^1, \ldots, x^m$ are sampled independently from the same distribution $\mathcal{D}$ over $\{0,1\}^N$. For all $m \geq 2$, with probability at least $1-\delta$ on the draw of the*

training set $S = \{x^1, \ldots, x^m\}$,

$$\mathbb{E}_x\left[f_S(x)\right] \geq \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \sqrt{\frac{1}{2m}\log\left(\frac{2N}{\delta}\right)}.$$

Proof. The proof is an application of Hoeffding's inequality and the union bound. For convenience, denote the seed sample average as $\mu_j := \frac{1}{m}\sum_{s=1}^{m} x_j^s$, and the probability that $x_j = 1$ as $p_j := \mathbb{E}_x[x_j]$. Then,

$$\begin{aligned}
&\frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \mathbb{E}_x\left[f_S(x)\right] \\
&\quad= \frac{1}{N\log\left(\frac{\gamma_{\min}+m}{\gamma_{\min}}\right)}\sum_{j=1}^{N}\left((\mu_j - p_j)\log\frac{\alpha_j + m\mu_j}{\alpha_j} + (p_j - \mu_j)\log\frac{\beta_j + m(1-\mu_j)}{\beta_j}\right) \\
&\quad\leq \frac{1}{N}\sum_{j=1}^{N}|\mu_j - p_j|. \qquad (7.4)
\end{aligned}$$

For any particular feature j, Hoeffding's inequality (Hoeffding, 1963) bounds the difference between the empirical average and the expected value:

$$\mathbb{P}(|\mu_j - p_j| > \epsilon) \leq 2\exp\left(-2m\epsilon^2\right). \qquad (7.5)$$

We then apply the union bound to bound the average over features:

$$\begin{aligned}
\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N}|\mu_j - p_j| > \epsilon\right) &\leq \mathbb{P}\left(\bigcup_{j=1}^{N}\{|\mu_j - p_j| > \epsilon\}\right) \\
&\leq \sum_{j=1}^{N}\mathbb{P}\left(|\mu_j - p_j| > \epsilon\right) \\
&\leq 2N\exp\left(-2m\epsilon^2\right). \qquad (7.6)
\end{aligned}$$

Thus,

$$\mathbb{P}\left(\frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \mathbb{E}_x\left[f_S(x)\right] > \epsilon\right) \leq 2N\exp\left(-2m\epsilon^2\right), \qquad (7.7)$$

and the theorem follows directly. □

Theorem 4 provides an additional result that has a weaker dependence on δ, but has no dependence on the number of features N, which in this application is the number of internet domains and is thus extremely large. Theorem 4 also gives insight into the dependence of generalization on the problem parameters.

Theorem 4. *Suppose $x^1, \ldots, x^m$ are sampled independently from the same distribution $\mathcal{D}$. Define p_j to be the probability that feature j takes value 1. Let $p_{\min} = \min_j \min\{p_j, 1 - p_j\}$ be the probability of the rarest feature. For all $p_{\min} > 0$, $\gamma_{\min} > 0$ and $m \geq 2$, with probability at least $1 - \delta$ on the draw of the training set $S = \{x^1, \ldots, x^m\}$,*

$$\mathbb{E}_{x \sim \mathcal{D}}\left[f_S(x)\right] \geq \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \sqrt{\frac{1}{2m\delta} + \frac{6}{g(m)\delta} + O\left(\frac{1}{m^2 \log m}\right)},$$

where

$$g(m) := (\gamma_{\min} + (m-1)p_{\min}) \log\left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}}\right)$$

The proof of Theorem 4 involves showing that Bayesian Sets is a "stable" algorithm, in the sense of "pointwise hypothesis stability" (Bousquet and Elisseeff, 2002). We show that the Bayesian Sets score is not too sensitive to perturbations in the seed set. Specifically, when an item is removed from the seed, the average change in score is bounded by a quantity that decays as $\frac{1}{m \log m}$. This stability allows us to apply a generalization bound from Bousquet and Elisseeff (2002). The proof of pointwise hypothesis stability is given in the next section.

The two quantities with the most direct influence on the bound are $\gamma_{\min}$ and $p_{\min}$. We show in the next section that for $p_{\min}$ small relative to $\gamma_{\min}$, the bound improves as $\gamma_{\min}$ increases (a stronger prior). This suggests that a strong prior improves stability when learning data with rare features. As $p_{\min}$ decreases, the bound becomes looser, suggesting that datasets with rare features will be harder to learn and will be more prone to errors.

The fact that the bound in Theorem 4 is independent of N provides motivation for using Bayesian Sets on very large scale problems, even when the feature independence

assumption does not hold. It indicates that Bayesian Set's performance may not degrade when faced with high dimensional data.

The gap between the expected score of x and the empirical score of the seed goes to zero as $\frac{1}{\sqrt{m}}$. Thus when the seed is sufficiently large, regardless of the distribution over relevant items, we can be assured that the relevant items generally have high scores.

7.3 Algorithmic Stability and Bayesian Sets

We now present the proof of Theorem 1. The result uses the algorithmic stability bounds of Bousquet and Elisseeff (2002), specifically the bound for pointwise hypothesis stability. We begin by defining an appropriate loss function. Suppose x and S were drawn from the same distribution $\mathcal{D}$. Then, we wish for $f_S(x)$ to be as large as possible. Because $f_S(x) \in [0, 1]$, an appropriate metric for the loss in using f_S to score x is:

$$\ell(f_S, x) = 1 - f_S(x). \tag{7.8}$$

Further, $\ell(f_S, x) \in [0, 1]$.

For algorithmic stability analysis, we will consider how the algorithm's performance changes when an element is removed from the training set. To analyze this, we define a modified training set in which the i'th element has been removed: $S^{\setminus i} := \{x^1, \ldots, x^{i-1}, x^{i+1}, \ldots, x^m\}$. We then define the score of x according to the modified training set:

$$f_{S^{\setminus i}}(x) = \frac{1}{Z(m-1)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s \neq i} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + (m-1) - \sum_{s \neq i} x_j^s}{\beta_j}, \tag{7.9}$$

where

$$Z(m-1) = N \log \left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}} \right). \tag{7.10}$$

We further define the loss using the modified training set:

$$\ell(f_{S^{\setminus i}}, x) = 1 - f_{S^{\setminus i}}(x). \tag{7.11}$$

The general idea of algorithmic stability is that if the results of an algorithm do not depend too heavily on any one element of the training set, the algorithm will be able to generalize. One way to quantify the dependence of an algorithm on the training set is to examine how the results change when the training set is perturbed, for example by removing an element from the training set. The following definition of pointwise hypothesis stability, taken from Bousquet and Elisseeff (2002), states that an algorithm has pointwise hypothesis stability if, on expectation, the results of the algorithm do not change too much when an element of the training set is removed.

Definition 1 (Bousquet and Elisseeff, 2002)**.** *An algorithm has pointwise hypothesis stability η with respect to the loss function ℓ if the following holds*

$$\forall i \in \{1, \ldots, m\}, \quad \mathbb{E}_S \left[|\ell(f_S, x^i) - \ell(f_{S^{\setminus i}}, x^i)| \right] \leq \eta. \tag{7.12}$$

The algorithm is said to be stable if η scales with $\frac{1}{m}$.

In our theorem, we suppose that all of the data belong to the same class of "relevant" items. The framework of Bousquet and Elisseeff (2002) can easily be adapted to the single-class setting, for example by framing it as a regression problem where all of the data points have the identical "true" output value 1. The following theorem comes from Bousquet and Elisseeff (2002), with the notation adapted to our setting.

Theorem 5 (Bousquet and Elisseeff, 2002)**.** *If an algorithm has pointwise hypothesis stability η with respect to a loss function ℓ such that $0 \leq \ell(\cdot, \cdot) \leq 1$, we have with probability at least $1 - \delta$,*

$$\mathbb{E}_x \left[\ell(f_S, x) \right] \leq \frac{1}{m} \sum_{i=1}^{m} \ell(f_S, x^i) + \sqrt{\frac{1 + 12m\eta}{2m\delta}}. \tag{7.13}$$

We now show that Bayesian Sets satisfies the conditions of Definition 1, and

determine the corresponding η. The proof of Theorem 4 comes from inserting our findings for η into Theorem 5. We begin with a lemma providing a bound on the central moments of a Binomial random variable.

Lemma 6. *Let $t \sim$ Binomial(m,p) and let $\mu_k = \mathbb{E}\left[(t - \mathbb{E}[t])^k\right]$ be the k^{th} central moment. For integer $k \geq 1$, μ_{2k} and μ_{2k+1} are $O\left(m^k\right)$.*

Proof. We will use induction. For $k = 1$, the central moments are well known (*e.g.*, Johnson et al, 2005): $\mu_2 = mp(1-p)$ and $\mu_3 = mp(1-p)(1-2p)$, which are both $O(m)$. We rely on the following recursion formula (Johnson et al, 2005; Romanovsky, 1923):

$$\mu_{s+1} = p(1-p)\left(\frac{d\mu_s}{dp} + ms\mu_{s-1}\right). \tag{7.14}$$

Because μ_2 and μ_3 are polynomials in p, their derivatives will also be polynomials in p. This recursion makes it clear that for all s, μ_s is a polynomial in p whose coefficients include terms involving m.

For the inductive step, suppose that the result holds for $k = s$. That is, μ_{2s} and μ_{2s+1} are $O(m^s)$. Then, by (7.14),

$$\mu_{2(s+1)} = p(1-p)\left(\frac{d\mu_{2s+1}}{dp} + (2s+1)m\mu_{2s}\right). \tag{7.15}$$

Differentiating μ_{2s+1} with respect to p yields a term that is $O(m^s)$. The term $(2s+1)m\mu_{2s}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)}$ is $O(m^{s+1})$. Also,

$$\mu_{2(s+1)+1} = p(1-p)\left(\frac{d\mu_{2(s+1)}}{dp} + 2(s+1)m\mu_{2s+1}\right). \tag{7.16}$$

Here $\frac{d\mu_{2(s+1)}}{dp}$ is $O(m^{s+1})$ and $2(s+1)m\mu_{2s+1}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)+1}$ is $O(m^{s+1})$.

This shows that if the result holds for $k = s$ then it must also hold for $k = s+1$ which completes the proof. □

The next lemma provides a stable, $O\left(\frac{1}{m}\right)$, bound on the expected value of an important function of a binomial random variable.

Lemma 7. *For $t \sim$ Binomial(m, p) and $\alpha > 0$,*

$$\mathbb{E}\left[\frac{1}{\alpha + t}\right] = \frac{1}{\alpha + mp} + O\left(\frac{1}{m^2}\right). \tag{7.17}$$

Proof. We expand $\frac{1}{\alpha+t}$ at $t = mp$:

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{\alpha + t}\right] &= \mathbb{E}\left[\sum_{i=0}^{\infty}(-1)^i \frac{(t - mp)^i}{(\alpha + mp)^{i+1}}\right] \\
&= \sum_{i=0}^{\infty}(-1)^i \frac{\mathbb{E}\left[(t - mp)^i\right]}{(\alpha + mp)^{i+1}} \\
&= \frac{1}{\alpha + mp} + \sum_{i=2}^{\infty}(-1)^i \frac{\mu_i}{(\alpha + mp)^{i+1}}
\end{aligned} \tag{7.18}$$

where μ_i is the i^{th} central moment and we recognize that $\mu_1 = 0$. By Lemma 6,

$$\frac{\mu_i}{(\alpha + mp)^{i+1}} = \frac{O\left(m^{\lfloor \frac{i}{2} \rfloor}\right)}{O\left(m^{i+1}\right)} = O\left(m^{\lfloor \frac{i}{2} \rfloor - i - 1}\right). \tag{7.19}$$

The alternating sum in (7.18) can be split into two sums:

$$\sum_{i=2}^{\infty}(-1)^i \frac{\mu_i}{(\alpha + mp)^{i+1}} = \sum_{i=2}^{\infty} O\left(m^{\lfloor \frac{i}{2} \rfloor - i - 1}\right) = \sum_{i=2}^{\infty} O\left(\frac{1}{m^i}\right) + \sum_{i=3}^{\infty} O\left(\frac{1}{m^i}\right). \tag{7.20}$$

These are, for m large enough, bounded by a geometric series that converges to $O\left(\frac{1}{m^2}\right)$. □

The following three lemmas provide results that will be useful for proving the main lemma, Lemma 11.

Lemma 8. *For all $\alpha > 0$,*

$$g(\alpha, m) := \frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{\log\left(\frac{\alpha+m-1}{\alpha}\right)} \tag{7.21}$$

is monotonically non-decreasing in α for any fixed $m \geq 2$.

Proof. Define $a = \frac{m-1}{\alpha}$ and $b = \frac{m}{m-1}$. Observe that $a \geq 0$ and $b \geq 1$, and that for

fixed m, a is inversely proportional to α. We reparameterize (7.21) to

$$g(a,b) := \frac{\log(ab+1)}{\log(a+1)}. \tag{7.22}$$

To prove the lemma, it is sufficient to show that $g(a,b)$ is monotonically non-increasing in a for any fixed $b \geq 1$. Well,

$$\frac{\partial g(a,b)}{\partial a} = \frac{\frac{b}{ab+1}\log(a+1) - \frac{1}{a+1}\log(ab+1)}{(\log(a+1))^2},$$

so $\frac{\partial g(a,b)}{\partial a} \leq 0$ if and only if

$$h(a,b) := (ab+1)\log(ab+1) - b(a+1)\log(a+1) \geq 0. \tag{7.23}$$

$h(a,1) = (a+1)\log(a+1) - (a+1)\log(a+1) = 0$, and,

$$\begin{aligned}\frac{\partial h(a,b)}{\partial b} &= a\log(ab+1) + a - (a+1)\log(a+1) \\ &= a(\log(ab+1) - \log(a+1)) + (a - \log(a+1)) \\ &\geq 0 \quad \forall a \geq 0,\end{aligned}$$

because $b \geq 1$ and $a \geq \log(1+a)\ \forall a \geq 0$. This shows that (7.23) holds $\forall a \geq 0, b \geq 1$, which proves the lemma. □

Lemma 9. *For any $m \geq 2$, $t \in [0, m-1]$, $\alpha > 0$, and $\gamma_{\min} \in (0, \alpha]$,*

$$\frac{1}{Z(m)}\log\frac{\alpha+t+1}{\alpha} \geq \frac{1}{Z(m-1)}\log\frac{\alpha+t}{\alpha}. \tag{7.24}$$

Proof. Denote

$$g(t;m,\alpha) := \frac{1}{Z(m)}\log\frac{\alpha+t+1}{\alpha} - \frac{1}{Z(m-1)}\log\frac{\alpha+t}{\alpha}. \tag{7.25}$$

By Lemma 8 and $\gamma_{\min} \leq \alpha$, for any $\alpha > 0$ and for any $m \geq 2$,

$$\frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{\log\left(\frac{\alpha+m-1}{\alpha}\right)} \geq \frac{\log\left(\frac{\gamma_{\min}+m}{\gamma_{\min}}\right)}{\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)} = \frac{Z(m)}{Z(m-1)}.$$

Thus,

$$\frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{Z(m)} \geq \frac{\log\left(\frac{\alpha+m-1}{\alpha}\right)}{Z(m-1)}, \tag{7.26}$$

which shows

$$g(m-1;m,\alpha) = \frac{1}{Z(m)}\log\frac{\alpha+m}{\alpha} - \frac{1}{Z(m-1)}\log\frac{\alpha+m-1}{\alpha} \geq 0. \tag{7.27}$$

Furthermore, because $Z(m) > Z(m-1)$,

$$\frac{\partial g(t;m,\alpha)}{\partial t} = \frac{1}{Z(m)}\frac{1}{\alpha+t+1} - \frac{1}{Z(m-1)}\frac{1}{\alpha+t} < 0, \tag{7.28}$$

for all $t \geq 0$. Equations 7.27 and 7.28 together show that $g(t;m,\alpha) \geq 0$ for all $t \in [0, m-1], m \geq 2$, proving the lemma. □

Lemma 10. *For any $m \geq 2$, $t \in [0, m-1]$, $\beta > 0$, and $\gamma_{\min} \in (0, \beta]$,*

$$\frac{1}{Z(m)}\log\frac{\beta+m-t}{\beta} \geq \frac{1}{Z(m-1)}\log\frac{\beta+m-1-t}{\beta}. \tag{7.29}$$

Proof. Let $\tilde{t} = m-t-1$. Then, $\tilde{t} \in [0, m-1]$ and by Lemma 9, replacing α with β,

$$\frac{1}{Z(m)}\log\frac{\beta+\tilde{t}+1}{\beta} \geq \frac{1}{Z(m-1)}\log\frac{\beta+\tilde{t}}{\beta}. \tag{7.30}$$

□

The next lemma is the key lemma that shows Bayesian Sets satisfies pointwise hypothesis stability, allowing us to apply Theorem 5.

Lemma 11. *The Bayesian Sets algorithm satisfies the conditions for pointwise hy-*

pothesis stability with

$$\eta = \frac{1}{\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)(\gamma_{\min}+(m-1)p_{\min})} + O\left(\frac{1}{m^2\log m}\right). \tag{7.31}$$

Proof.

$$\begin{aligned}
&\mathbb{E}_S|\ell(f_S,x^i)-\ell(f_{S^{\setminus i}},x^i)| \\
&= \mathbb{E}_S\left|f_{S^{\setminus i}}(x^i)-f_S(x^i)\right| \\
&= \mathbb{E}_S\left|\frac{1}{Z(m-1)}\sum_{j=1}^N\left[x_j^i\log\frac{\alpha_j+\sum_{s\neq i}x_j^s}{\alpha_j}+(1-x_j^i)\log\frac{\beta_j+(m-1)-\sum_{s\neq i}x_j^s}{\beta_j}\right]\right. \\
&\quad \left.-\frac{1}{Z(m)}\sum_{j=1}^N\left[x_j^i\log\frac{\alpha_j+\sum_{s=1}^m x_j^s}{\alpha_j}+(1-x_j^i)\log\frac{\beta_j+m-\sum_{s=1}^m x_j^s}{\beta_j}\right]\right| \\
&\leq \mathbb{E}_S\sum_{j=1}^N x_j^i\left|\frac{1}{Z(m-1)}\log\frac{\alpha_j+\sum_{s\neq i}x_j^s}{\alpha_j}-\frac{1}{Z(m)}\log\frac{\alpha_j+\sum_{s=1}^m x_j^s}{\alpha_j}\right| \\
&\quad +(1-x_j^i)\left|\frac{1}{Z(m-1)}\log\frac{\beta_j+(m-1)-\sum_{s\neq i}x_j^s}{\beta_j}-\frac{1}{Z(m)}\log\frac{\beta_j+m-\sum_{s=1}^m x_j^s}{\beta_j}\right| \qquad (7.32) \\
&:= \mathbb{E}_S\sum_{j=1}^N x_j^i\mathrm{term}_j^1+(1-x_j^i)\mathrm{term}_j^2 \qquad (7.33) \\
&= \sum_{j=1}^N\mathbb{E}_{x_j^1,\ldots,x_j^m}\left[x_j^i\mathrm{term}_j^1+(1-x_j^i)\mathrm{term}_j^2\right] \\
&= \sum_{j=1}^N\mathbb{E}_{x_j^i}\left[\mathbb{E}_{x_j^{s\neq i}|x_j^i}\left[x_j^i\mathrm{term}_j^1\right]\right]+\mathbb{E}_{x_j^i}\left[\mathbb{E}_{x_j^{s\neq i}|x_j^i}\left[(1-x_j^i)\mathrm{term}_j^2\right]\right] \\
&= \sum_{j=1}^N\mathbb{E}_{x_j^i}\left[x_j^i\mathbb{E}_{x_j^{s\neq i}|x_j^i}\left[\mathrm{term}_j^1\right]\right]+\mathbb{E}_{x_j^i}\left[(1-x_j^i)\mathbb{E}_{x_j^{s\neq i}|x_j^i}\left[\mathrm{term}_j^2\right]\right] \\
&= \sum_{j=1}^N\mathbb{E}_{x_j^{s\neq i}}\left[\mathrm{term}_j^1|x_j^i=1\right]\mathbb{P}\left(x_j^i=1\right)+\mathbb{E}_{x_j^{s\neq i}}\left[\mathrm{term}_j^2|x_j^i=0\right]\mathbb{P}\left(x_j^i=0\right) \\
&\leq \sum_{j=1}^N\max\left\{\mathbb{E}_{x_j^{s\neq i}}\left[\mathrm{term}_j^1|x_j^i=1\right],\mathbb{E}_{x_j^{s\neq i}}\left[\mathrm{term}_j^2|x_j^i=0\right]\right\}, \qquad (7.34)
\end{aligned}$$

where (7.32) uses the triangle inequality, and in (7.33) we define term_j^1 and term_j^2 for

notational convenience. Now consider each term in (7.34) separately,

$$\begin{aligned}
&\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^1 | x_j^i = 1\right] \\
&\quad = \mathbb{E}_{x_j^{s\neq i}}\left|\frac{1}{Z(m-1)}\log\frac{\alpha_j + \sum_{s\neq i} x_j^s}{\alpha_j} - \frac{1}{Z(m)}\log\frac{\alpha_j + \sum_{s\neq i} x_j^s + 1}{\alpha_j}\right| \\
&\quad = \mathbb{E}_{x_j^{s\neq i}}\left[\frac{1}{Z(m)}\log\frac{\alpha_j + \sum_{s\neq i} x_j^s + 1}{\alpha_j} - \frac{1}{Z(m-1)}\log\frac{\alpha_j + \sum_{s\neq i} x_j^s}{\alpha_j}\right], \quad (7.35)
\end{aligned}$$

where we have shown in Lemma 9 that this quantity is non-negative. Because $\{x^s\}$ are independent, $\{x_j^s\}$ are independent for fixed j. We can consider $\{x_j^s\}_{s\neq i}$ to be a collection of $m-1$ independent Bernoulli random variables with probability of success $p_j = \mathbb{P}_{x\sim\mathcal{D}}(x_j = 1)$, the marginal distribution. Let $t = \sum_{s\neq i} x_j^s$, then $t \sim \text{Binomial}(m-1, p_j)$. Continuing (7.35),

$$\begin{aligned}
\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^1 | x_j^i = 1\right] &= \mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\frac{1}{Z(m)}\log\frac{\alpha_j + t + 1}{\alpha_j} - \frac{1}{Z(m-1)}\log\frac{\alpha_j + t}{\alpha_j}\right] \\
&\leq \frac{1}{Z(m-1)}\mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\log\frac{\alpha_j + t + 1}{\alpha_j + t}\right] \\
&= \frac{1}{Z(m-1)}\mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\log\left(1 + \frac{1}{\alpha_j + t}\right)\right] \\
&\leq \frac{1}{Z(m-1)}\log\left(1 + \mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\frac{1}{\alpha_j + t}\right]\right) \\
&= \frac{1}{Z(m-1)}\log\left(1 + \frac{1}{\alpha_j + (m-1)p_j} + O\left(\frac{1}{m^2}\right)\right). \quad (7.36)
\end{aligned}$$

The second line uses $Z(m) \geq Z(m-1)$, the fourth line uses Jensen's inequality, and the fifth line uses Lemma 7. Now we turn to the other term.

$$\begin{aligned}
&\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^2 | x_j^i = 0\right] \\
&\quad = \mathbb{E}_{x_j^{s\neq i}}\left|\frac{1}{Z(m-1)}\log\frac{\beta_j + (m-1) - \sum_{s\neq i} x_j^s}{\beta_j} - \frac{1}{Z(m)}\log\frac{\beta_j + m - \sum_{s\neq i} x_j^s}{\beta_j}\right| \\
&\quad = \mathbb{E}_{x_j^{s\neq i}}\left[\frac{1}{Z(m)}\log\frac{\beta_j + m - \sum_{s\neq i} x_j^s}{\beta_j} - \frac{1}{Z(m-1)}\log\frac{\beta_j + (m-1) - \sum_{s\neq i} x_j^s}{\beta_j}\right]. \quad (7.37)
\end{aligned}$$

We have shown in Lemma 10 that this quantity is non-negative. Let $q_j = 1 - p_j$. Let

$t = m - 1 - \sum_{s \neq i} x_j^s$, then $t \sim \text{Binomial}(m-1, q_j)$. Continuing (7.37):

$$\begin{aligned}\mathbb{E}_{x_j^{s \neq i}}\left[\text{term}_j^2 | x_j^i = 0\right] &\leq \frac{1}{Z(m-1)} \mathbb{E}_{t \sim \text{Bin}(m-1, q_j)}\left[\log \frac{\beta_j + t + 1}{\beta_j + t}\right] \\ &\leq \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\beta_j + (m-1)q_j} + O\left(\frac{1}{m^2}\right)\right), \end{aligned} \tag{7.38}$$

where the steps are as with (7.36). We now take (7.36) and (7.38) and use them to continue (7.34):

$$\begin{aligned}&\mathbb{E}_S |\ell(f_S, x^i) - \ell(f_{S^{\setminus i}}, x^i)| \\ &\quad \leq \sum_{j=1}^{N} \max\left\{ \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\alpha_j + (m-1)p_j} + O\left(\frac{1}{m^2}\right)\right), \right. \\ &\qquad\qquad \left. \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\beta_j + (m-1)q_j} + O\left(\frac{1}{m^2}\right)\right)\right\} \\ &\quad \leq \sum_{j=1}^{N} \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\min\{\alpha_j, \beta_j\} + (m-1)\min\{p_j, q_j\}} + O\left(\frac{1}{m^2}\right)\right) \\ &\quad \leq \frac{N}{Z(m-1)} \log\left(1 + \frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right)\right) \\ &\quad := \eta. \end{aligned} \tag{7.39}$$

Using the Taylor expansion of $\log(1+x)$,

$$\begin{aligned}\eta &= \frac{N}{Z(m-1)} \left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right) \right. \\ &\qquad \left. -\frac{1}{2}\left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right)\right)^2 \right) \\ &= \frac{N}{Z(m-1)} \left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right) \right) \\ &= \frac{1}{\log\left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}}\right)(\gamma_{\min} + (m-1)p_{\min})} + O\left(\frac{1}{m^2 \log m}\right). \end{aligned} \tag{7.40}$$

□

The proof of Theorem 4 is now a straightforward application of Theorem 5 using the result of Lemma 11.

Proof of Theorem 4. By Lemma 11, we can apply Theorem 5 to see that with probability at least $1-\delta$ on the draw of S,

$$\begin{aligned}
\mathbb{E}_x\left[\ell(f_S,x)\right] &\leq \frac{1}{m}\sum_{i=1}^{m}\ell(f_S,x^i)+\sqrt{\frac{1+12m\eta}{2m\delta}}\\
\mathbb{E}_x\left[1-f_S(x)\right] &\leq \frac{1}{m}\sum_{s=1}^{m}(1-f_S(x^s))+\sqrt{\frac{1+12m\eta}{2m\delta}}\\
\mathbb{E}_x\left[f_S(x)\right] &\geq \frac{1}{m}\sum_{s=1}^{m}f_S(x^s)-\sqrt{\frac{1+12m\eta}{2m\delta}}\\
&= \frac{1}{m}\sum_{s=1}^{m}f_S(x^s)-\sqrt{\frac{1}{2m\delta}+\frac{6}{g(m)\delta}+O\left(\frac{1}{\delta m^2\log m}\right)},
\end{aligned}$$

where

$$g(m) := (\gamma_{\min}+(m-1)p_{\min})\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right).$$

□

7.3.1 The Effect of the Prior on Generalization.

The prior influences the generalization bound via the quantity

$$h(\gamma_{\min},m,p_{\min}) := \log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)(\gamma_{\min}+(m-1)p_{\min}). \tag{7.41}$$

As this quantity increases, the bound becomes tighter. We can thus study the influence of the prior on generalization by studying the behavior of this quantity as $\gamma_{\min}$ varies. The second term, $(\gamma_{\min}+(m-1)p_{\min})$, is similar to many results from Bayesian analysis in which the prior plays the same role as additional data. This term is *increasing* with $\gamma_{\min}$, meaning it yields a tighter bound with a stronger prior. The first term, $\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)$, is inherited from the normalization $Z(m)$. This term is *decreasing* with $\gamma_{\min}$, that is, it gives a tighter bound with a weaker prior. The overall effect of $\gamma_{\min}$ on generalization depends on how these two terms balance each other, which in turn depends primarily on $p_{\min}$.

Exact analysis of the behavior of $h(\gamma_{\min},m,p_{\min})$ as a function of $\gamma_{\min}$ does not

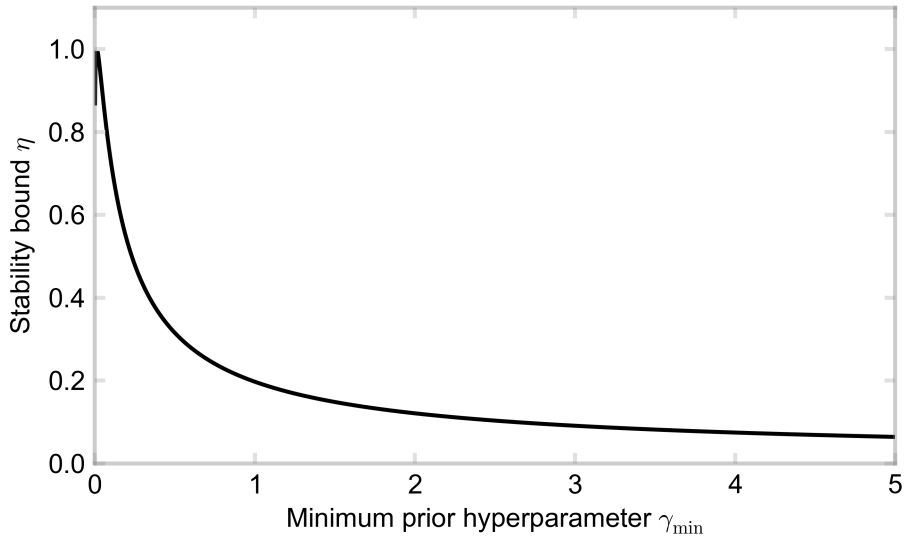

Figure 7-1: The stability bound η as a function of the prior $\gamma_{\min}$, for fixed $m = 100$ and $p_{\min} = 0.001$. For $\gamma_{\min}$ large enough relative to $p_{\min}$, stronger priors yield tighter bounds.

yield interpretable results, however we gain some insight by considering the case where $\gamma_{\min}$ scales with m: $\gamma_{\min} := \tilde{\gamma}(m-1)$. Then we can consider (7.41) as a function of $\tilde{\gamma}$ and $p_{\min}$ alone:

$$h(\tilde{\gamma}, p_{\min}) := \log\left(\frac{\tilde{\gamma}+1}{\tilde{\gamma}}\right)(\tilde{\gamma}+p_{\min})\,. \tag{7.42}$$

The bound becomes tighter as $\tilde{\gamma}$ increases, as long as we have $\frac{\partial h(\tilde{\gamma},p_{\min})}{\partial \tilde{\gamma}} > 0$. This is the case when

$$p_{\min} < \tilde{\gamma}(\tilde{\gamma}+1)\log\left(\frac{\tilde{\gamma}+1}{\tilde{\gamma}}\right) - \tilde{\gamma}. \tag{7.43}$$

The quantity on the right-hand side is increasing with $\tilde{\gamma}$. Thus, for $p_{\min}$ small enough relative to $\tilde{\gamma}$, stronger priors lead to a tighter bound. To illustrate this behavior, in Figure S1 we plot the stability bound η (excluding $O\left(\frac{1}{m^2 \log m}\right)$ terms) as a function of $\gamma_{\min}$, for $m = 100$ and $p_{\min} = 0.001$. For $\gamma_{\min}$ larger than about 0.01, the bound tightens as the prior is increased.

7.3.2 Bayesian Sets and Uniform Stability.

In addition to pointwise hypothesis stability, Bousquet and Elisseeff (2002) define a stronger notion of stability called "uniform stability."

Definition 2 (Bousquet and Elisseeff, 2002)**.** *An algorithm has uniform stability κ with respect to the loss function ℓ if the following holds*

$$\forall S, \quad \forall i \in \{1, \ldots, m\}, \quad ||\ell(f_S, \cdot) - \ell(f_{S^{\setminus i}}, \cdot)||_\infty \leq \kappa. \tag{7.44}$$

The algorithm is said to be stable if κ scales with $\frac{1}{m}$.

Uniform stability requires a $O\left(\frac{1}{m}\right)$ bound for all training sets, rather than the average training set as with pointwise hypothesis stability. The bound must also hold for all possible test points, rather than testing on the perturbed point. Uniform stability is actually a very strong condition that is difficult to meet, since if (7.44) can be violated by any possible combination of training set and test point, then uniform stability does not hold. Bayesian Sets does not have this form of stability, as we now show with an example.

Choose the training set of m data points to satisfy:

$$\begin{aligned} x_j^i &= 0 \quad \forall j, \quad i = 1, \ldots, m-1 \\ x_j^m &= 1 \quad \forall j, \end{aligned}$$

and as a test point x, take $x_j = 1 \; \forall j$. Let x^m be the point removed from the training set. Then,

$$\begin{aligned} \kappa &= |\ell(f_S, x) - \ell(f_{S^{\setminus m}}, x)| \\ &= |f_{S^{\setminus m}}(x) - f_S(x)| \\ &= \left| \frac{1}{Z(m-1)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s - x_j^m}{\alpha_j} - \frac{1}{Z(m)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} \right| \\ &= \left| \frac{1}{Z(m-1)} \sum_{j=1}^{N} \log \frac{\alpha_j}{\alpha_j} - \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\alpha_j + 1}{\alpha_j} \right| \end{aligned}$$

$$= \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\alpha_j + 1}{\alpha_j}$$
$$\geq \frac{\log \frac{\max_j \alpha_j + 1}{\max_j \alpha_j}}{\log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)}, \tag{7.45}$$

which scales with m as $\frac{1}{\log m}$, not the $\frac{1}{m}$ required for stability.

7.4 Experiments

We demonstrate and evaluate the algorithm with two sets of experiments. In the first set of experiments, we provide an objective comparison between our method, Google Sets, and Boo!Wa! using a randomly selected collection of list growing problems for which there exist gold standard lists. The true value of our work lies in the ability to construct lists for which there are not gold standards, so in a second set of experiments we demonstrate the algorithm's performance on more realistic, open-ended list growing problems. For all experiments, the steps and parameter settings of the algorithm were exactly the same and completely unsupervised other than specifying two seed items. Boo!Wa! and Google Sets are online tools and we used them as provided by http://boowa.com and Google Spreadsheet, respectively. The results for Boo!Wa! and Google Sets used in Section 7.4.1 were retrieved from their respective online tools on November 14, 2012, and those used in Section 7.4.3 were retrieved on December 13, 2012. Results for our algorithm, which depend on Google search results and the content of the source webpages, were obtained in the period March 14-21, 2013 for the Wikipedia gold-standard experiments in Sections 7.4.1, and in the period May 28-30, 2012 for the open-ended experiments in Section 7.4.3.

7.4.1 Wikipedia Gold Standard Lists

Many of the experiments in past work on set completion, such as those used to develop the technology behind Boo!Wa! of Wang and Cohen (2008), involve manually

constructed gold standards on arbitrarily selected topics. Manually constructed lists are inherently subjective, and experiments on a small set of arbitrarily selected topics do not demonstrate that the method will perform well in general. We thus use Wikipedia lists on randomly selected topics as gold standards, which is the same experimental design used by Sarmento et al (2007) and Pantel et al (2009).

The "List of ..." articles on Wikipedia form a large corpus of potential gold standard lists that cover a wide variety of topics. We limited our experiments to the "featured lists," which are a collection of over 2,000 Wikipedia lists selected by Wikipedia editors due to their high quality. We required the lists used in our experiments to have at least 20 items, and excluded any lists of numbers (such as dates or sports scores). We created a random sample of list growing problems by randomly selecting 50 Wikipedia lists that met the above requirements. The selected lists covered a wide range of topics, including, for example, "storms in the 2005 Atlantic hurricane season," "current sovereign monarchs," "tallest buildings in New Orleans," "X-Men video games," and "Pittsburgh Steelers first-round draft picks." We treated the Wikipedia list as the gold standard for the associated list growing problem.

For each of the 50 list growing problems, we randomly selected two list items from the gold standard to form a seed. We used the seed as an input to our algorithm, and ran one iteration. We used the same seed as an input to Google Sets and Boo!Wa!. We compared the lists returned by our method, Google Sets, and Boo!Wa! to the gold standard list by computing two ranking measures: Precision@10 and average precision. Precision@10 measures the fraction of the top 10 items in the list that are found on the gold standard list:

$$\text{Precision@10} = \frac{1}{10} \sum_{i=1}^{10} \mathbb{1}_{[\text{item } i \text{ in ranked list is correct}]}. \tag{7.46}$$

Precision@10 is an intuitive measure that corresponds directly to the number of accurate results at the top of the list. Average precision is a commonly used measure that combines precision and recall, to ensure that lists are both accurate and complete. The average precision of a ranked list is defined as the average of the precision recall

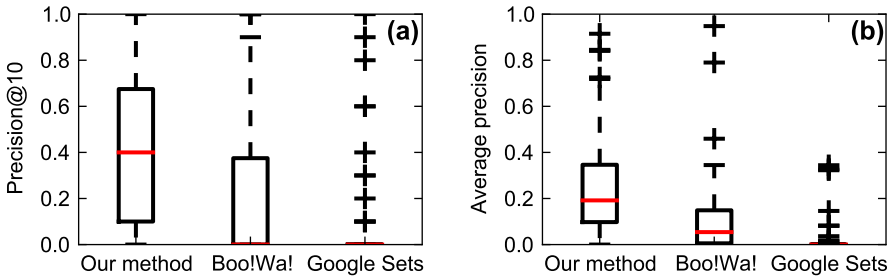

Figure 7-2: **(a)** Precision@10 and **(b)** average precision across all 50 list growing problems sampled from Wikipedia. The median is indicated in red.

curve. Before computing these measures, the seed items were removed from each list, as it is trivial to return the seed.

In Figure 7-2 we show boxplots of the results across all 50 gold standard experiments. For both Google Sets and Boo!Wa!, the median precision at 10 was 0, indicating no correct results in the top 10. Our method performed significantly better ($p < 0.001$, signed-rank test), with median precision of 0.4, indicating 4 correct results in the top 10. Our method returned at least one relevant result in the top 10 for 80% of the experiments, whereas Google Sets and Boo!Wa! returned at least one relevant result in the top 10 for only 24% and 34% of experiments, respectively. Our method performed well also in terms of recall, as measured in average precision, with a median average precision of 0.19, compared to 0 for Google Sets and 0.05 for Boo!Wa!. Boo!Wa! is a significant improvement over Google Sets, and our method is a large improvement over Boo!Wa!.

There are some flaws with using Wikipedia lists as gold standards in these experiments. First, the gold standards are available online and could potentially be pulled directly without requiring any aggregation of experts across different sites. However, all three methods had access to the gold standards and the experiments did not favor any particular method, thus the comparison is informative. A more interesting experiment is one that necessitates aggregation of experts across different sites; these experiments are given in Section 7.4.3. Second, these results are only accurate insofar as the Wikipedia gold standard lists are complete. We limited our experiments to "featured lists" to have the best possible gold standards. A truly objective compar-

ison of methods requires both randomly selected list problems and gold standards, and the Wikipedia lists, while imperfect, provide a useful evaluation.

7.4.2 Experimental Analysis of Algorithm Steps

We performed several experiments modifying individual steps of the algorithm to explore the effect of design choices on performance, and to gather further insight into how each step contributes to the performance gain seen in Section 7.4.1 relative to the baseline methods. We use the Wikipedia gold standard lists to explore experimentally the impact of which HTML tags are used for item extraction, the size of the seed when scoring, and the set of domains used in constructing the feature space.

In the item extraction step of our algorithm, we find the largest collection of HTML tags common to both seed items and extract all other items on the page that use that same collection of HTML tags. An alternative choice would be to look for a specific type of HTML tag, for example, list tags <li>, which could possibly reduce the number of incorrect items extracted. In Figure 7-3(a) we repeated the Wikipedia gold standard experiments from Section 7.4.1, with a modified item extraction step in which we searched only for a specific type of tag: list tags <li> in one experiment, and hyperlink tags <a> in a second. Restricting to list tags significantly reduced average precision, while restricting to hyperlink tags produced results comparable to those obtained using all tags. Figure 7-3(b) provides insight into this difference by showing the proportion of all of the correct items extracted in the Wikipedia gold standard experiments that were extracted using a particular HTML tag, for the six most common tags. An item may be extracted using multiple HTML tags, either in a collection of tags or by discovering the same item on multiple pages, thus these proportions do not sum to 1. The value of 0.21 for <li> indicates that when extraction was limited to only <li> tags, we only obtained 21% of the correct items that were obtained using all tags, which resulted in the significant performance drop seen in Figure 7-3(a). Limiting to <a> tags recovered 81% of the correct items, which was enough to yield average precision comparable to that obtained using all tags. These results suggest that item extraction could be limited to link extraction,

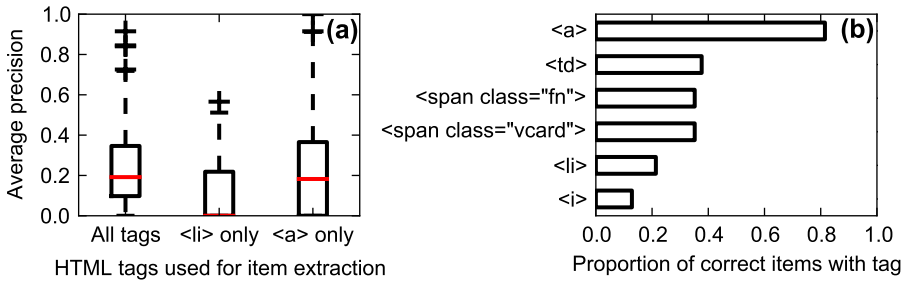

Figure 7-3: **(a)** Average precision across the Wikipedia gold standard problems when extracting items using all tags (original implementation), `<li>` tags only, and `<a>` tags only. **(b)** The proportion of correct items extracted during the Wikipedia gold standard experiments that were found using a specific tag, for the six most commonly found tags.

a problem for which many efficient software packages are widely available, without much loss.

In the gold standard experiments in Section 7.4.1 we used a seed of 2 items, the smallest possible seed size. When seed items are related in multiple contexts, as discussed for the case of Atlantic hurricanes, two seed items may not be enough to produce an accurate ranked list. In Figure 7-4(a), for each Wikipedia gold standard experiment we randomly selected additional seed items from the gold standard and used the larger seed to compute a new ranked list. Increasing the seed size, which further constrained the context of the relation between the seed items, produced a modest increase in performance. These results indicate that for many of the lists used in these experiments, two items were sufficient to specify the context. However, there is some gain to be had with a larger seed, and in general it is best for the user to specify as large a seed as possible.

We construct the binary feature space for each item using the domains of all of the sites where the item can be found. An alternative approach is to restrict the search to sites containing at least two seed items, that is, the sites found during the source discovery step. In Figure 7-4(b) we repeated the Wikipedia gold standard experiments using this feature space strategy, and found that it significantly reduced average precision. In fact, Boo!Wa! uses a strategy similar to this one to construct a feature space, as we discuss in Section 7.5.

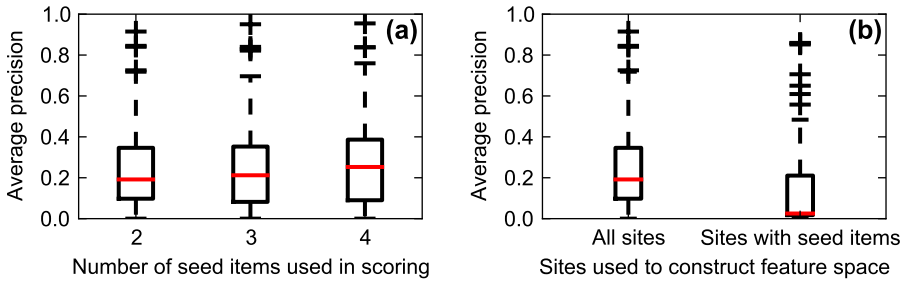

Figure 7-4: Average precision across the Wikipedia gold standard problems when **(a)** expanding the number of seed items used in scoring, and **(b)** restricting the feature space construction to sites containing at least two seed items, that is, sites found in source discovery.

7.4.3 Open-Ended Experiments

In this set of experiments we demonstrate our method's performance on more realistic, open-ended list growing problems. For these problems gold standard lists are not available, and it is essential for the algorithm to aggregate results across many experts. We focus on four open-ended list growing problems: Boston events, Jewish foods, smartphone apps, and U.S. politicians. These are the sort of problems that our algorithm was designed to solve, and it performs very well, especially compared to the baselines.

Boston events

In this experiment, the seed items were two Boston events: "Boston arts festival" and "Boston harborfest." We ran the algorithm for 5 iterations, yielding 3,090 items. Table 7.1 shows the top 50 ranked items, together with the domain of the source site where they were discovered. There is no gold standard list to compare to directly, but the results are overwhelmingly actual Boston events. The events were aggregated across a variety of expert sources, including event sites, blogs, travel guides, and hotel pages. Table 7.2 shows the full set of results returned from Google Sets with the same two events as the seed, and the top 25 results returned by Boo!Wa!. Not only is the Google Sets list very short, but it does not contain any actual Boston events. Boo!Wa! was able to return some Boston events, but with a substantial amount of noise.

Item	Source
[0]Boston arts festival	(original seed)
[3]Cambridge river festival	bizbash.com
[0]Boston harborfest	(original seed)
harborfest	
[1]Boston chowderfest	celebrateboston.com
[4]Berklee beantown jazz festival	pbase.com
the berklee beantown jazz festival,	
berklee bean town jazz festival	
[2]Chinatown main street festival	blog.charlesgaterealty.com
www.chinatownmainstreet.org	
4th of july boston pops concert & fireworks display	travel2boston.us
boston 4th of july fireworks & concert	
Boston common frog pond	bostonmamas.com
ice skating on boston common frog pond	
First night boston	what-is-there-to-do.com
Boston dragon boat festival	pbase.com
hong kong dragon boat festival of boston	
dragon boat festival of boston	
Boston tea party re enactment	ef.com
Christopher columbus waterfront park	bostonmamas.com
Jimmy fund scooper bowl	bizbash.com
Opening our doors day	ef.com
Oktoberfest harvard square & harpoon brewery	sheratonbostonhotel.com
August moon festival	ef.com
Annual boston wine festival	worldtravelguide.net
Cambridge carnival	soulofamerica.com
Regattabar	berklee.edu
Arts on the arcade	berklee.edu
Franklin park zoo	hotels-rates.com
Faneuil hall annual tree lighting ceremony	ef.com
Annual oktoberfest and honk festival	ef.com
honk! festival	
Boston jazz week	telegraph.co.uk
Boston ballet	celebrateboston.com
Fourth of july reading of the declaration of independence	ef.com
Isabella stewart gardner museum	hotels-rates.com
Revere beach sand sculpting festival	bizbash.com
Shakespeare on the common	boston-discovery-guide.com
Boston bacon takedown	[...]bostonevents.blogspot.com
Jazz at the fort	berklee.edu
Cambridge dance party	[...]thrillsboston.blogspot.com
Boston celtic music festival	ef.com
Taste of the south end	bizbash.com
Greenway open market	travel2boston.us
Boston winter jubilee	ef.com
Urban ag fair	bostonmamas.com
Figment boston	festivaltrek.com
Boston kite festival	bostoneventsinsider.com
Chefs in shorts	bizbash.com
Old south meeting house	hotels-rates.com

Table 7.1: Items and the domain of their source sites from the top of the ranked list for the Boston events experiment. Superscript numbers indicate the iteration at which the item was added to the seed via implicit feedback. "[...]" indicates the URL was truncated to fit in the figure. To improve readability, duplicate items were grouped and placed in italics.

Jewish foods

In this experiment, the seed items were two Jewish foods: "Challah" and "Knishes."

Although there are lists of foods that are typically found in Jewish cuisine, there

Google Sets	**Boo!Wa!**
Boston arts festival	*Boston arts festival*
Boston harborfest	*Boston harborfest*
Whats going this month	Boston Fall Foliage
Interview with ann scott	Boston Wine Expo
Studio view with dannyo	Boston Flower and Garden Show
Tony savarino	Boston Vegetarian Food Festival
Artwalk 2011	The Boston Wine Expo
Greater boston convention visitors bureau	The Jazz Festival
	First Night Celebration
Cambridge chamber of commerce	Thumbboston-vegetarian-food-festival
	Boston College Eagles
Boston tours	Boston Vacation Rentals
3 county fairground	Best Boston Restaurants
Boston massacre	Boston Red Sox
	Boston Bruins
	Attractions in Boston:North End
	Attractions in Boston:Freedom Trail
	Attractions in Boston:Museum of Science
	Attractions in Boston:Prudential Center
	Attractions in Boston:New England Aquarium
	Attractions in Boston: Boston Public Garden
	Attractions in Boston:St. Patrick's Cathedral
	Attractions in Boston:South Beach - Ocean Drive
	Vacation-rentals-boston
	Boston-restaurants
	Parking-in-boston
	Shopping-boston

Table 7.2: Complete Google Sets results and top 25 Boo!Wa! results for the Boston events experiment (seed italicized). Google Sets and our implementation of our method return results all lower case, and in these tables we have capitalized the first letter for aesthetics. Boo!Wa! returns capitalized results, and we use here the capitalization that was returned.

is variety across lists and no authoritative definition of what is or is not a Jewish food. We completed 5 iterations of the algorithm, yielding 8,748 items. Table 7.3 shows the top 50 ranked items, together with their source domains. Almost all of the items are closely related to Jewish cuisine. The items on our list came from a wide variety of expert sources that include blogs, informational sites, bakery sites, recipe sites, dictionaries, and restaurant menus. In fact, the top 100 most highly ranked items came from a total of 52 unique sites. This diversity in source sites shows that the relevant items are found in many small lists, which provides motivation for using pairs of seed items for source discovery, as opposed to requiring all seed items to be on every source. In Table 7.4, we show the complete set of results returned from Google Sets for the same seed of Jewish foods, and the top 25 results returned by Boo!Wa!. Although the Google Sets results are foods, they are not closely related to Jewish cuisine. Boo!Wa! was able to return some Jewish foods, but also a lot of irrelevant results like “Shop Online," “Lipkin’s Bakery," and “Apple."

Smartphone apps

In this experiment, we began with two popular smartphone apps as the seed items: “Word lens” and “Aroundme.” We ran the algorithm for 5 iterations, throughout which 7,630 items were extracted. Table 7.5 shows the top 50 most highly ranked items, together with the source domain where they were discovered. Not only are the results almost exclusively apps, but they come from a wide variety of sources including personal sites, review sites, blogs, and news sites. In Table 7.6, we show the lists returned by Google Sets and Boo!Wa! for the same seed, which are also predominantly apps.

U.S. politicians

In this experiment, we began with two prominent U.S. politicians as the seed items: “Barack obama” and “Scott brown.” We ran the algorithm for 5 iterations, yielding 8,384 items. Table 7.7 shows the top 50 most highly ranked items, together with the source site where they were discovered. All of the items in our list are names

Item	Source
[0]Challah	(original seed)
braided challah	
[3]Potato latkes	jewishveg.com
latkes; sweet potato latkes; potato latke	
[1]Blintzes	jewfaq.org
cheese blintzes; blintz	
[0]Knishes	(original seed)
potato knishes; knish	
[2]Noodle kugel	pinterest.com
noodle kugel recipe; kugel; sweet noodle kugel	
[4]Tzimmes	jewfaq.org
carrot tzimmes	
Matzo balls	jewishveg.com
matzo ball soup; matzo; matzoh balls	
Potato kugel	challahconnection.com
Passover recipes	lynnescountrykitchen.net
hanukkah recipes	
Gefilte fish	jewfaq.org
Honey cake	kveller.com
Soups, kugels & liver	allfreshkosher.com
Charoset	jewishveg.com
haroset	
Hamantaschen	butterfloureggs.com
Matzo meal	glattmart.net
Rugelach	pinterest.com
rugelach recipe	
Matzo brei	ilovekatzs.com
Cholent	jewfaq.org
Sufganiyot	kosheronabudget.com
Potato pancakes	jewishveg.com
Noodle pudding	epicurious.com
Kreplach	allmenus.com
Barley soup	ecampus.com
Mushroom barley	zagat.com
mushroom barley soup	
Chopped liver	ryedeli.com
Garlic mashed potatoes	tovascatering.com
Caponata	lynnescountrykitchen.net
Compote	kveller.com
Farfel & mushrooms	hungariankosher.com
farfel	
Kasha varnishkes	jinsider.com

Table 7.3: Items and their source domains from the top of the ranked list for the Jewish foods experiment.

of politicians or politically influential individuals. In Table 7.8, we show the results returned from Google Sets and Boo!Wa! for the same seed. Google Sets managed to return only a few people related to politics. Boo!Wa! performed better than Google Sets, but the list still contains some noise, like “U.S. Senate 2014," “President 2016," and “Secret-service."

Google Sets	**Boo!Wa!**
Knishes	*Knishes*
Challah	*Challah*
Crackers	Applestrudel
Dinner rolls	Holishkes
Focaccie	Blintzes
Pains sucres	Gefilte
Pains plats	Apple
Biscotti integral de algarroba	Kasha
Souffle de zanahorias	Soup
Tarta de esparragos	Knishes.pdf
Leftover meat casserole	Knishes recipe PDF
Pan de canela	Shop Online
Focaccia	Hamantashen
Sweet hobz	Kamish Bread
Pranzu rolls	Apple Strudel
Focacce	Location
Chicken quesadillas	Danishes
Baked chicken chimichangas	Lipkin's Bakery
Honey mustard salad dressing	Flax Seed Bread
Dixxijiet hobz	Babka
Roast partridge	Pumpernickel Loaf
Fanny farmer brownies	Schnitzel
Pan pratos	Latke
Pan doce	Cole slaw
Cea rolls	Chopped Liver
Flat paes	Mini Potato Knish
Hobz dixxijiet	Oven Roasted Chicken

Table 7.4: Complete Google Sets results and top 25 Boo!Wa! results for the Jewish foods experiment (seed italicized).

7.5 Related Work

There is a substantial body of work in areas or tasks related to the one which we have presented, which we can only briefly review here. There are a number of papers on various aspects of "set expansion," often for completing lists of entities from structured lists, like those extracted from Wikipedia (Sarmento et al, 2007), using rules from natural language processing or topic models (Tran et al, 2010; Sadamitsu et al, 2011), or from opinion corpora (Zhang and Liu, 2011). The task we explore here is *web-based set expansion* and methods developed for other set expansion tasks are not directly applicable. See, for example, Jindal and Roth (2011), for a review of different set expansion problems.

There is good deal of work in the machine learning community on aggregating ranked lists (*e.g.*, Dwork et al, 2001). These are lists that are typically already cleaned, fixed in scope, and ranked by individual experts, unlike our case. There is also a body of work on aggregated search (Lalmas, 2011; Renda and Straccia, 2003;

Item	Source
[0]Word lens	(original seed)
[2]Read it later *read later*	iapps.scenebeta.com
[0]Aroundme	(original seed)
[3]Instapaper *instapaper app*	time.com
[4]Evernote *evernote app*	crosswa.lk
[1]Flipboard	crosswa.lk
Dolphin browser	1mobile.com
Skitch	worldwidelearn.com
Facebook messenger	crosswa.lk
Zite	adriandavis.com
Tweetbot	duckduckgo.com
Google currents	secure.crosswa.lk
Springpad	time.com
Imessage	iphoneae.com
Retina display	twicpic.blogspot.com
Ibooks	crosswa.lk
Dropbox *dropbox (app); dropbox app*	mobileappreviews.craveonline.com
Marco arment	wired.com
Doubletwist	appolicious.com
Google latitude	iapps.scenebeta.com
Gowalla	mobileappreviews.craveonline.com
Skype for ipad	secure.crosswa.lk
Hulu plus	appadvice.com
Icloud	thetechcheck.com
Qik video *qik*	1mobile.com
Find my friends	oradba.ch
Skydrive	crosswa.lk
Google shopper	mobileappreviews.craveonline.com
Swype	techcrunch.com
Pulse news reader	techcrunch.com
Spotify	crosswa.lk
Readability	tips.flipboard.com
Apple app store	socialmediaclub.org
Tweetdeck	iapps.scenebeta.com
Angry birds space	appys.com
Smartwatch	theverge.com
Vlingo	mobileappreviews.craveonline.com
Rdio	techcrunch.com
Google goggles	sofialys.com
Xmarks	40tech.com
Ios 6	zomobo.net
Ibooks author	duckduckgo.com
Google drive	geekandgirliestuff.blogspot.com
Facetime	bgpublishers.com.au

Table 7.5: Items and their source domains from the top of the ranked list for the smartphone apps experiment.

Hsu and Taksa, 2005; Beg and Ahmad, 2003), which typically uses a text query to aggregate results from multiple search engines, or of multiple formats or domains (e.g. image and news), and returns links to the full source. Our goal is not to rank URLs but to scrape out and rank information gleaned from them. There are many resources for performing a search or query by example. They often involve using

Google Sets	**Boo!Wa!**
Word lens	*Word lens*
Aroundme	*Aroundme*
Lifestyle	Plants v. Zombies
View in itunes	Amazon
Itunes	Bloom
Jcpenney weekly deals	AIM
Coolibah digital scrapbooking	Plants vs. Zombies
Epicurious recipes shopping list	Layar
170000 recipes bigoven	Bjrk: Biophilia
Cf iviewer	Wikipedia Mobile
Txtcrypt	Web Source Viewer
Speak4it	WhatTheFont
Off remote free	The Weather Channel
Catholic calendar	EDITION29 STRUCTURES
Gucci	Dexigner
Board	Google
Ziprealty real estate	Zipcar
Allsaints spitalfields	Thrutu
Lancome make up	Google Earth
Pottery barn catalog viewer	Four-Square
Amazon mobile	Wikipedia
Gravity clock	Facebook
Dace	Kindle
Zara	Skype
Style com	Mint
Iridiumhd	Wi-Fi Finder App
Ebanner lite	Green Gas Saver

Table 7.6: Complete Google Sets results and top 25 Boo!Wa! results for the smartphone apps experiment (seed italicized).

a single example of a full document or image in order to retrieve more documents, structures within documents, or images (Chang and Lui, 2001; Liu et al, 2003; Wang and Lochovsky, 2003; Zhai and Liu, 2005).

Methods such as that of Gupta and Sarawagi (2009) and Pantel et al (2009) learn semantic classes, which could be used to grow a list, but require preprocessing which crawls the web and creates an index of HTML lists in an unsupervised manner. Kozareva et al (2008) present a method for using a semantic class name and a seed of example instances to discover other instances from the same class on the web, using search queries. They limit the search to instances that match a very specific pattern of words ("*class name* such as *seed item* and *"), thus requiring the semantic class to have enough instances and web coverage that all instances match the pattern somewhere on the Internet. We found that this was not the case for more realistic open-ended problems, like Boston events and the others in Section 4.3. Paşca (2007a,b) also discovers semantic class attributes and instances, but using web query logs rather

Item	Source
[0]Barack obama	(original seed)
obama	
[0]Scott brown	(original seed)
[1]John kerry	publicpolicypolling.com
[3]Barney frank	masslive.com
[4]John mccain	publicpolicypolling.com
mccain	
[2]Nancy pelosi	theladypatriot.com
pelosi	
Mitch mcconnell	publicpolicypolling.com
Joe lieberman	publicpolicypolling.com
Mike huckabee	publicpolicypolling.com
Mitt romney	masslive.com
Bill clinton	mediaite.com
John boehner	audio.wrko.com
boehner	
Hillary clinton	blogs.wsj.com
Jon kyl	tpmdc.talkingpointsmemo.com
Joe biden	publicpolicypolling.com
Rudy giuliani	publicpolicypolling.com
Harry reid	theladypatriot.com
Olympia snowe	publicpolicypolling.com
Lindsey graham	politico.com
Newt gingrich	masspoliticsprofs.com
Jim demint	theladypatriot.com
Arlen specter	theladypatriot.com
Dick cheney	blogs.wsj.com
George w bush	wellgroomedmanscape.com
george w. bush	
Eric holder	disruptthenarrative.com
Dennis kucinich	publicpolicypolling.com
Timothy geithner	tpmdc.talkingpointsmemo.com
Barbara boxer	publicpolicypolling.com
Tom coburn	itmakessenseblog.com
Orrin hatch	publicpolicypolling.com
Michael bloomberg	masspoliticsprofs.com
Elena kagan	audio.wrko.com
Maxine waters	polination.wordpress.com
Al sharpton	porkbarrel.tv
Rick santorum	audio.wrko.com
Ted kennedy	newomenforchange.org
Janet napolitano	disruptthenarrative.com
Jeff sessions	tpmdc.talkingpointsmemo.com
Jon huntsman	publicpolicypolling.com
Michele bachmann	publicpolicypolling.com
Al gore	publicpolicypolling.com
Rick perry	publicpolicypolling.com
Eric cantor	publicpolicypolling.com
Ben nelson	publicpolicypolling.com
Karl rove	politico.com

Table 7.7: Items and their source domains from the top of the ranked list for the U.S. politicians experiment.

than actual internet sites.

Systems for learning categories and relations of entities on the web, like the Never-Ending Language Learner (NELL) system (Carlson et al, 2010a,b; Verma and Hruschka, 2012), or KnowItAll (Etzioni et al, 2005) can be used to construct lists but

Google Sets	**Boo!Wa!**
Barack obama	*Barack obama*
Scott brown	*Scott brown*
Our picks movies	William Galvin
Sex	Secret-service
Department of justice	Sheldon Whitehouse
Viral video	Debbie Stabenow
Africa	Dennis Kucinich
One persons trash	Susana Martinez
Donald trump	Stephen Colbert
New mom confessions	Martin O'Malley
Nonfiction	Claire McCaskill
Libya	U.S. Senate 2012
Sarah palin	Brian Schweitzer
Mtv	Michele Bachmann
Alan greenspan	Condoleezza Rice
Great recession	U.S. Senate 2014
Life stories	Lisa Murkowski
Jon hamm	Lindsey Graham
Islam	Maria Cantwell
The killing	Jeanne Shaheen
American idol	South Carolina
Middle east	North Carolina
Celebrity	Terry Branstad
Tea parties	President 2016
Budget showdown	Tommy Thompson
	Brian Sandoval
	Offshore drilling

Table 7.8: Complete Google Sets results and top 25 Boo!Wa! results for the U.S. politicians experiment (seed italicized).

require extensive preprocessing. We do not preprocess, instead we perform information extraction online, deterministically, and virtually instantaneously given access to a search engine. There is no restriction to HTML list structures or need for more time consuming learning methods (Freitag, 1998; Soderland et al, 1999). We also do not require human-labeled web pages like wrapper induction methods (Kushmerick, 1997).

The Set Expander for Any Language (SEAL) of Wang and Cohen (2007, 2008), implemented in Boo!Wa!, at first appears similar to our work but differs in significant ways. Wang and Cohen (2008) describe four strategies for source discovery, of which "unsupervised increasing seed size (ISS)" is most similar to ours. Unsupervised ISS begins with two seed items and iteratively expands the seed in the same way as our implicit feedback, by adding the most highly ranked non-seed item to the seed at each iteration. Within each iteration, unsupervised ISS uses only a subset of the seed items to try to further expand the set. Specifically, it uses the most recently added

seed item together with three additional randomly-selected seed items, and searches for source sites containing all four of these items. Our source discovery differs in two major ways. First, our combinatorial search strategy uses all seed items in every iteration, rather than a randomly-selected subset of four seed items. Second, we use only pairs of seed items to find source sites, rather than requiring the source sites to contain four seed items. With this strategy we find all of the sites discovered by ISS, as well as additional sites that have less than four seed items. Once the source sites have been found, SEAL extracts new items by learning a character-based wrapper that finds patterns of characters that are common to the seed items. This is similar in concept to the way that we extract new items, although SEAL allows arbitrary patterns of characters whereas we look specifically for patterns in the HTML tree structure. Possibly the most significant differences between SEAL and our approach lie in ranking the extracted items. When the initial number of seed items is small, as in the list growing problems that we considered here, Wang and Cohen (2008) recommend a ranking algorithm that uses a random walk over a graph that contains nodes for the extracted items, the wrappers learned for each source site, and the source sites themselves. Wang and Cohen (2008) also considered using Bayesian Sets to rank, and in fact recommended its use when the number of initial seed items was large. However, the way in which SEAL constructs the feature space to be used by Bayesian Sets is critically different. SEAL uses two sets of features: the sources sites on which the extracted item was found during list extraction, and the wrappers that extracted it. We use the complete set of domains (not sites) where the extracted item can be found, and do not limit ourselves to the domains of source sites. Using all domains as features rather than just those containing seed items is very important for reducing the rank of items that happened to show up on the same site as a few seed items but in general are found on very different types of domains, as shown in our experiments in Section 7.4.2. Finding this full set of domains requires an additional set of queries, one for each item to be ranked, however these types of queries can be done efficiently when one has access to a web index. These differences between our method and Boo!Wa! translate into the order of magnitude improvement in the

quality of the returned lists shown throughout Section 7.4.

7.6 Conclusions

The next generation of search engines should not simply retrieve URLs, but should aim at retrieving information. We designed a system that leads into this next generation, leveraging information from across the Internet to grow an authoritative list on almost any topic.

The gold standard experiments showed that our method performs well on a wide range of list growing problems, and provided insight into the effect of design choices on performance. There are several conclusions that can be drawn from the empirical results. First, we showed how increasing the number of seed items can improve performance by constraining the relationship between seed items, suggesting that users should be encouraged to provide as many seed items as possible. Second, even when a large seed is available, our results in Section 4.3 demonstrate the importance of using small groups of seed items for source site discovery (we used pairs). There we showed that in real problems, relevant items must be aggregated from many websites and are often only found together with a small collection of other relevant items.

Of all of the design choices, we found that the construction of the feature space for ranking discovered items had the largest impact on performance. Items that are likely to be correct, and should thus be highly ranked, are those that are found frequently on websites where seed items are found and, equally importantly, are not found frequently where seed items are not found. A feature space that considers only the sites on which seed items are found is not able to distinguish between items that are highly correlated with the seed and items that are just generally common. Our solution was to construct the feature space using an individual search query for each discovered item, allowing us to verify that the item was not frequently found without seed items. This led to substantially improved results compared to a feature space using only sites containing seed items, though at a cost of more search queries.

This feature space construction is a major source of improvement, but can be time

consuming given the restrictions that Google and other search engines place on the number of queries per minute. Without this restriction, our results can be obtained in real-time on almost any computer. One major challenge that needs to be overcome to have a real-time implementation for public use is either to embed code like ours within a search engine infrastructure, or to find ways to use fewer search queries, chosen in an intelligent way, to construct a similar feature space that incorporates information about sites without seed items. Another challenge not handled here is to build in knowledge of language. Our results are not English-specific, but with some knowledge of natural language, higher quality results could potentially be obtained.

The Wikipedia gold-standard experiments provided a framework for quantifying performance on a range of list topics, but the open-ended experiments showed, qualitatively, the true strength of the developed method. For real problems for which complete lists were not available online, we found that the algorithm produced meaningful lists, with information extracted from a wide variety of sources. Moreover, the lists compared favorably with those from existing related technology.

In addition to these empirical results, we presented a theoretical bound that justifies the use of Bayesian Sets in a setting where its feature independence assumptions are not met. This bound will help to motivate its continued use in set expansion problems.

The list growing algorithm we presented was implemented on a laptop, with minimal heuristics and hand-tuning, and no language-specific processing or handling of special cases. Yet, the results are good enough to be directly useful to users in many cases. These encouraging results are an indication of the power of high-quality algorithms to gather crowdsourced information.

Bibliography

Adams WJ, Yellen JL (1976) Commodity bundling and the burden of monopoly. The Quarterly Journal of Economics 90(3):475–498

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6):734–749

Agichtein E, Brill E, Dumais S (2006a) Improving web search ranking by incorporating user behavior information. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, pp 19–26

Agichtein E, Brill E, Dumais S, Ragno R (2006b) Learning user interaction models for predicting web search result preferences. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, pp 3–10

Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Databases, VLDB '94, pp 487–499

Antman EM, Cohen M, Bernink PJ, McCabe CH, Horacek T, Papuchis G, Mautner B, Corbalan R, Radley D, Braunwald E (2000) The TIMI risk score for unstable angina/non-ST elevation MI: a method for prognostication and therapeutic decision making. The Journal of the American Medical Association 284(7):835–842

Anupindi R, Dada M, Gupta S (1998) Estimation of consumer demand with stock-out based substitution: An application to vending machine products. Marketing Science 17(4):406–423

Bache K, Lichman M (2013) UCI machine learning repository. `http://archive.ics.uci.edu/ml`

Beg MMS, Ahmad N (2003) Soft computing techniques for rank aggregation on the world wide web. World Wide Web 6(1):5–22

Berchtold A, Raftery AE (2002) The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. Statistical Science 17(3):328–356

Bertsekas DP (1995) Nonlinear Programming. Athena Scientific, Belmont, Massachusetts

Bigal ME, Liberman JN, Lipton RB (2006) Obesity and migraine. Neurology 66(4):545–550

Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. Journal of Machine Learning Research 3:993–1022

Borgelt C (2005) An implementation of the FP-growth algorithm. In: Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, OSDM '05, pp 1–5

Bottou L (2010) Large-scale machine learning with stochastic gradient descent. In: Proceedings of the 19th International Conference on Computational Statistics, COMPSTAT '10, pp 177–187

Bousquet O, Elisseeff A (2002) Stability and generalization. Journal of Machine Learning Research 2:499–526

Bratko I (1997) Machine learning: between accuracy and interpretability. In: Della Riccia G, Lenz HJ, Kruse R (eds) Learning, Networks and Statistics, International Centre for Mechanical Sciences, vol 382, Springer Vienna, pp 163–177

Breiman L (1996a) Bagging predictors. Machine Learning 24:123–140

Breiman L (1996b) Heuristics of instability and stabilization in model selection. Annals of Statistics 24:2350–2383

Breiman L (2001a) Random forests. Machine Learning 45:5–32

Breiman L (2001b) Statistical modeling: the two cultures. Statistical Science 16(3):199–231

Breiman L, Friedman JH, Olshen RA, Stone CJ (1984) Classification and Regression Trees. Wadsworth

Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. In: Proceedings of the 22nd International Conference on Machine Learning, ICML '05, pp 89–96

Byrd RH, Lu P, Nocedal J, Zhu C (1995) A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing 16(5):1190–1208

Campoa K, Gijsbrechtsb E, Nisol P (2003) The impact of retailer stockouts on whether, how much, and what to buy. International Journal of Research in Marketing 20:273–286

Cao Z, Qin T, Liu TY, Tsai MF, Li H (2007) Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning, ICML '07, pp 129–136

Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka ER, Mitchell TM (2010a) Toward an architecture for never-ending language learning. In: Proceedings of the 24th Conference on Artificial Intelligence, AAAI '10

Carlson A, Betteridge J, Wang RC, Hruschka ER, Mitchell TM (2010b) Coupled semi-supervised learning for information extraction. In: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining, WSDM '10, pp 101–110

Carvalho VR, Cohen WW (2008) Ranking users for intelligent message addressing. In: Proceedings of the 30th European conference on IR Research, ECIR '08, pp 321–333

Chang A, Rudin C, Cavaretta M, Thomas R, Chou G (2012) How to reverse-engineer quality rankings. Machine Learning 88(3):369–398

Chang CC, Lin CJ (2011) LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2:27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

Chang CH, Lui SC (2001) IEPAD: Information extraction based on pattern discovery. In: Proceedings of the 10th International Conference on World Wide Web, WWW '01, pp 681–688

Chapelle O, Keerthi SS (2010) Efficient algorithms for ranking with SVMs. Information Retrieval 13(3):201–215

Chipman HA, George EI, McCulloch RE (1998) Bayesian CART model search. Journal of the American Statistical Association 93(443):935–948

Chipman HA, George EI, McCulloch RE (2002) Bayesian treed models. Machine Learning 48(1/3):299–320

Chipman HA, George EI, McCulloch RE (2010) BART: Bayesian additive regression trees. Annals of Applied Statistics 4(1):266–298

Clémençon S, Vayatis N (2008) Empirical performance maximization for linear rank statistics. In: Advances in Neural Information Processing Systems 22, pp 305–312

Clémençon S, Lugosi G, Vayatis N (2008) Ranking and empirical minimization of U-statistics. Annals of Statistics 36(2):844–874

Cohen WW, Schapire RE, Singer Y (1999) Learning to order things. Journal of Artificial Intelligence 10:243–270

Cossock D, Zhang T (2008) Statistical analysis of Bayes optimal subset ranking. IEEE Transactions on Information Theory 54(11):5140–5154

Davis DA, Chawla NV, Christakis NA, Barabasi AL (2010) Time to CARE: A collaborative engine for practical disease prediction. Data Mining and Knowledge Discovery 20:388–415

Dawes RM (1979) The robust beauty of improper linear models in decision making. American Psychologist 34(7):571–582

Dekel O, Manning CD, Singer Y (2004) Log-linear models for label ranking. In: Advances in Neural Information Processing Systems 16, pp 497–504

Dension D, Mallick B, Smith A (1998) A Bayesian CART algorithm. Biometrika 85(2):363–377

Dom B, Eiron I, Cozzi A, Zhang Y (2003) Graph-based ranking algorithms for e-mail expertise analysis. In: Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, DMKD '03, pp 42–48

Dougherty J, Kohavi R, Sahami M (1995) Supervised and unsupervised discretization of continuous features. In: Proceedings of the 12th International Conference on Machine Learning, ICML '95, pp 194–202

Duan L, Street W, Xu E (2011) Healthcare information systems: data mining methods in the creation of a clinical recommender system. Enterprise Information Systems 5(2):169–181

Dwork C, Kumar R, Naor M, Sivakumar D (2001) Rank aggregation methods for the web. In: Proceedings of the 10th International Conference on World Wide Web, WWW '01, pp 613–622

Eckalbar JC (2010) Closed-form solutions to bundling problems. Journal of Economics and Management Strategy 19(2):513–544

Elidan G (2010) Copula Bayesian networks. In: Advances in Neural Information Processing Systems 23, NIPS '10, pp 559–567

Elidan G (2013) Copulas in machine learning. In: Copulae in Mathematical and Quantitative Finance, Lecture Notes in Statistics, pp 39–60

Enright C, Madden M, Madden N, Laffey J (2011) Clinical time series data analysis using mathematical models and DBNs. In: Peleg M, Lavrac N, Combi C (eds) Artificial Intelligence in Medicine, Lecture Notes in Computer Science, vol 6747, Springer Berlin / Heidelberg, pp 159–168

Eppen GD, Hanson WA, Kipp MR (1991) Bundling - new products, new markets, low risk. Sloan Management Review 32(4):7–14

Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A (2005) Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligence 165(1):91–134

Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ (2008) LIBLINEAR: a library for large linear classification. Journal of Machine Learning Research 9:1871–1874

Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 1993 International Joint Conference on Artificial Intelligence, IJCAI '93, vol 2, pp 1022–1027

Freitag D (1998) Information extraction from HTML: application of a general machine learning approach. In: Proceedings of the 15th National Conference on Artificial Intelligence, AAAI '98, pp 517–523

Freitas AA (2014) Comprehensible classification models: a position paper. ACM SIGKDD Explorations Newsletter 15(1):1–10

Freund Y, Iyer R, Schapire RE, Singer Y (2003) An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research 4:933–969

Friedman JH, Popescu BE (2008) Predictive learning via rule ensembles. Annals of Applied Statistics 2(3):916–954

Fürnkranz J, Hüllermeier E (2003) Pairwise preference learning and ranking. In: Proceedings of the 14th European Conference on Machine Learning, ECML '03, pp 145–156

Gage BF, Waterman AD, Shannon W, Boechler M, Rich MW, Radford MJ (2001) Validation of clinical classification schemes for predicting stroke. Journal of the American Medical Association 285(22):2864–2870

Gelman A, Rubin DB (1992) Inference from iterative simulation using multiple sequences. Statistical Science 7:457–511

Ghahramani Z, Heller KA (2005) Bayesian sets. In: Advances in Neural Information Processing Systems 18, NIPS '05, pp 435–442

Giraud-Carrier C (1998) Beyond predictive accuracy: what? In: Proceedings of the ECML-98 Workshop on Upgrading Learning to Meta-Level: Model Selection and Data Transformation, pp 78–85

Goldberg SM, Green PE, Wind Y (1984) Conjoint analysis of price premiums for hotel amenities. Journal of Business 57:S111–S132

Goutelle S, Maurin M, Rougier F, Barbaut X, Bourguignon L, Ducher M, Maire P (2008) The Hill equation: a review of its capabilities in pharmacological modelling. Fundamental & Clinical Pharmacology 22:633–648

Gupta R, Sarawagi S (2009) Answering table augmentation queries from unstructured lists on the web. Proceedings of the VLDB Endowment 2:289–300

Hanson W, Martin RK (1990) Optimal bundle pricing. Management Science 36(2):155–174

Heller KA, Ghahramani Z (2006) A simple Bayesian framework for content-based image retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR '06, pp 2110–2117

Herbrich R, Graepel T, Obermayer K (1999) Support vector learning for ordinal regression. In: Proceedings of the 9th International Conference on Artificial Neural Networks, ICANN '99, pp 97–102

Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '99, pp 230–237

Hoeffding W (1963) Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association 58(301):13–30

Holte RC (1993) Very simple classification rules perform well on most commonly used datasets. Machine Learning 11(1):63–91

Hsu CN, Chung HH, Huang HS (2004) Mining skewed and sparse transaction data for personalized shopping recommendation. Machine Learning 57(1-2):35–59

Hsu DF, Taksa I (2005) Comparing rank and score combination methods for data fusion in information retrieval. Information Retrieval 8(3):449–480

Hu Q, Huang X, Melek W, Kurian C (2010) A time series based method for analyzing and predicting personalized medical data. In: Yao Y, Sun R, Poggio T, Liu J, Zhong N, Huang J (eds) Brain Informatics, Lecture Notes in Computer Science, vol 6334, Springer Berlin / Heidelberg, pp 288–298

Hüllermeier E, Fürnkranz J, Cheng W, Brinker K (2008) Label ranking by learning pairwise preferences. Artificial Intelligence 172(16–17):1897–1916

Huysmans J, Dejaeger K, Mues C, Vanthienen J, Baesens B (2011) An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decision Support Systems 51(1):141–154

ICCBR (2011) International Conference on Case-Based Reasoning (ICCBR) Computer cooking contest recipe book. URL `http://liris.cnrs.fr/ccc/ccc2011/`

Jain A, Rudi N, Wang T (2015) Demand estimation and ordering under censoring: Stock-out timing is (almost) all you need. Operations Research 63(1):134–150

Järvelin K, Kekäläinen J (2000) IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '00, pp 41–48

Jedidi K, Jagpal S, Manchanda P (2003) Measuring heterogeneous reservation prices for product bundles. Marketing Science 22(1):107–130

Jennings DL, Amabile TM, Ross L (1982) Informal covariation assessments: data-based versus theory-based judgements. In: Kahneman D, Slovic P, Tversky A (eds) Judgment Under Uncertainty: Heuristics and Biases,, Cambridge Press, Cambridge, MA, pp 211–230

Jindal P, Roth D (2011) Learning from negative examples in set-expansion. In: Proceedings of the 2011 11th IEEE International Conference on Data Mining, ICDM '11, pp 1110–1115

Joachims T (2002) Optimizing search engines using clickthrough data. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02, pp 133–142

Joe H, Xu JJ (1996) The estimation method of inference functions for margins for multivariate models. Tech. Rep. 166, Department of Statistics, University of British Columbia

Johnson K, Lee BHA, Simchi-Levi D (2014) Analytics for an online retailer: Demand forecasting and price optimization, working paper

Johnson NL, Kemp AW, Kotz S (2005) Univariate Discrete Distributions. John Wiley & Sons

Kalyanam K, Borle S, Boatwright P (2007) Deconstructing each item's category contribution. Marketing Science 26(3):327–341

Knaus WA, Draper EA, Wagner DP, Zimmerman JE (1985) APACHE II: a severity of disease classification system. Critical Care Medicine 13:818–829

Kök AG, Fisher ML (2007) Demand estimation and assortment optimization under substitution: Methodology and application. Operations Research 55(6):1001–1021

Kozareva Z, Riloff E, Hovy E (2008) Semantic class learning from the web with hyponym pattern linkage graphs. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL '08, pp 1048–1056

Kushmerick N (1997) Wrapper induction for information extraction. PhD thesis, University of Washington

Lalmas M (2011) Aggregated search. In: Melucci M, Baeza-Yates R (eds) Advanced Topics on Information Retrieval, Springer

Lebanon G, Lafferty J (2002) Cranking: Combining rankings using conditional probability models on permutations. In: Proceedings of the 19th International Conference on Machine Learning, ICML '02, pp 363–370

Leondes CT (2002) Expert systems: the technology of knowledge management and decision making for the 21st century. Academic Press

Letham B (2013) Similarity-weighted association rules for a name recommender system. In: Proceedings of the 15th ECML PKDD Discovery Challenge, pp 73–80

Letham B, Heller K, Rudin C (2013a) Growing a list. Data Mining and Knowledge Discovery 27:372–395

Letham B, Rudin C, Madigan D (2013b) Sequential event prediction. Machine Learning 93:357–380

Letham B, Sun W, Sheopuri A (2014) Latent variable copula inference for bundle pricing from retail transaction data. In: Proceedings of the 31st International Conference on Machine Learning, ICML'14

Letham B, Letham LM, Rudin C (2015a) Bayesian inference of arrival rate and substitution behavior from sales transaction data with stockouts `arXiv:1502.04243[stat.AP]`

Letham B, Rudin C, McCormick TH, Madigan D (2015b) Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. Annals of Applied Statistics to appear

Levenshtein VI (1966) Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10(8):707–710

Li W, Han J, Pei J (2001) CMAR: accurate and efficient classification based on multiple class-association rules. IEEE International Conference on Data Mining pp 369–376

Lim W, van der Eerden M, Laing R, Boersma W, Karalus N, Town G, Lewis S, Macfarlane J (2003) Defining community acquired pneumonia severity on presentation to hospital: an international derivation and validation study. Thorax 58(5):377–382

Linden G, Smith B, York J (2003) Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1):76–80

Lip G, Nieuwlaat R, Pisters R, Lane D, Crijns H (2010a) Refining clinical risk stratification for predicting stroke and thromboembolism in atrial fibrillation using a novel risk factor-based approach: the Euro heart survey on atrial fibrillation. Chest 137:263–272

Lip GY, Frison L, Halperin JL, Lane DA (2010b) Identifying patients at high risk for stroke despite anticoagulation: a comparison of contemporary stroke risk stratification schemes in an anticoagulated atrial fibrillation cohort. Stroke 41:2731–2738

Liu B, Hsu W, Ma Y (1998) Integrating classification and association rule mining. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, KDD '98, pp 80–96

Liu B, Grossman R, Zhai Y (2003) Mining data records in web pages. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, pp 601–606

Madigan D, Mosurski K, Almond R (1997) Explanation in belief networks. Journal of Computational and Graphical Statistics 6:160–181

Madigan D, Mittal S, Roberts F (2011) Efficient sequential decision making algorithms for container inspection operations. Naval Research Logistics 58:637–654

Marchand M, Sokolova M (2005) Learning with decision lists of data-dependent features. Journal of Machine Learning Research 6:427–451

McAfee PR, McMillan J, Whinston MD (1989) Multiproduct monopoly, commodity bundling, and correlation of values. The Quarterly Journal of Economics 104(2):371–383

McCardle KF, Rajaram K, Tang CS (2007) Bundling retail products: models and analysis. European Journal of Operational Research 177:1197–1217

McCormick TH, Rudin C, Madigan D (2012) Bayesian hierarchical rule modeling for predicting medical conditions. Annals of Applied Statistics 6(2):652–668

McSherry F, Najork M (2008) Computing information retrieval performance measures efficiently in the presence of tied scores. In: Proceedings of the 30th European conference on IR Research, ECIR '08, pp 414–421

Meinshausen N (2010) Node harvest. Annals of Applied Statistics 4(4):2049–2072

Miller GA (1956) The magical number seven, plus or minus two: some limits to our capacity for processing information. The Psychological Review 63(2):81–97

Musalem A, Olivares M, Bradlow ET, Terwiesch C, Corsten D (2010) Structural estimation of the effect of out-of-stocks. Management Science 56(7):1180–1197

Nowozin S, Tsuda K, Uno T, Kudo T, Bakir G (2007) Weighted substructure mining for image analysis. In: Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR '07

Paşca M (2007a) Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In: Proceedings of the 16th International Conference on World Wide Web, WWW '07, pp 101–110

Paşca M (2007b) Weakly-supervised discovery of named entities using web search queries. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management, CIKM '07, pp 683–690

Pal C, McCallum A (2006) Cc prediction with graphical models. In: Proceedings of the 3rd Conference on Email and Anti-Spam, CEAS '06

Pantel P, Crestan E, Borkovsky A, Popescu AM, Vyas V (2009) Web-scale distributional similarity and entity set expansion. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP '09, pp 938–947

Patterson S, Teh YW (2013) Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In: Advances in Neural Information Processing Systems 26, NIPS '13, pp 3102–3110

Quinlan JR (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann

Radlinski F, Kleinberg R, Joachims T (2008) Learning diverse rankings with multi-armed bandits. In: Proceedings of the 25th International Conference on Machine Learning, ICML '08, pp 784–791

Renda ME, Straccia U (2003) Web metasearch: rank vs. score based rank aggregation methods. In: Proceedings of the 2003 ACM Symposium on Applied Computing, SAC '03, pp 841–846

Rivest RL (1987) Learning decision lists. Machine Learning 2(3):229–246

Romanovsky V (1923) Note on the moments of a binomial $(p + q)^n$ about its mean. Biometrika 15:410–412

Roth M, Ben-David A, Deutscher D, Flysher G, Horn I, Leichtberg A, Leiser N, Matias Y, Merom R (2010) Suggesting friends using the implicit social graph. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10, pp 233–242

Rudin C (2009) The P-norm Push: A simple convex ranking algorithm that concentrates at the top of the list. Journal of Machine Learning Research 10:2233–2271

Rudin C, Schapire RE (2009) Margin-based ranking and an equivalence between AdaBoost and RankBoost. Journal of Machine Learning Research 10:2193–2232

Rudin C, Letham B, Salleb-Aouissi A, Kogan E, Madigan D (2011) Sequential event prediction with association rules. In: Proceedings of the 24th Annual Conference on Learning Theory, COLT '11, pp 615–634

Rudin C, Letham B, Madigan D (2013) Learning theory analysis for association rules and sequential event prediction. Journal of Machine Learning Research 14:3384–3436

Rüping S (2006) Learning interpretable models. PhD thesis, Universität Dortmund

Sadamitsu K, Saito K, Imamura K, Kikui G (2011) Entity set expansion using topic information. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL '11, vol 2, pp 726–731

Sarmento L, Jijkoun V, de Rijke M, Oliveira E (2007) "More like these" : growing entity classes from seeds. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management, CIKM '07, pp 959–962

Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, WWW '01, pp 285–295

Schmalensee R (1982) Commodity bundling by single-product monopolies. Journal of Law and Economics 25(1):67–71

Schmalensee R (1984) Gaussian demand and commodity bundling. Journal of Business 57(1):S211–S230

Senecal S, Nantel J (2004) The influence of online product recommendations on consumers' online choices. Journal of Retailing 80:159–169

Shalev-Shwartz S, Singer Y (2006) Efficient learning of label ranking by soft projections onto polyhedra. Journal of Machine Learning Research 7:1567–1599

Shani G, Heckerman D, Brafman RI (2005) An MDP-based recommender system. Journal of Machine Learning Research 6:1265–1295

Shmueli G (2010) To explain or to predict? Statistical Science 25(3):289–310

Sklar A (1973) Random variables, joint distributions, and copulas. Kybernetica 9(6):449–460

Soderland S, Cardie C, Mooney R (1999) Learning information extraction rules for semi-structured and free text. Machine Learning 34(1-3):233–272

Srikant R, Agrawal R (1996) Mining quantitative association rules in large relational tables. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, SIGMOD '96, pp 1–12

Stahl F, Johansson R (2009) Diabetes mellitus modeling and short-term prediction based on blood glucose measurements. Mathematical Biosciences 217(2):101–117

Stang P, Ryan P, Racoosin J, Overhage J, Hartzema A, Reich C, Welebob E, Scarnecchia T, Woodcock J (2010) Advancing the science for active surveillance: rationale and design for the observational medical outcomes partnership. Annals of Internal Medicine 153:600–606

Stremersch S, Tellis GJ (2002) Strategic bundling of products and prices: a new synthesis for marketing. Journal of Marketing 66(1):55–72

Taddy MA, Gramacy RB, Polson NG (2011) Dynamic trees for learning and design. Journal of the American Statistical Association 106(493):109–123

Talluri K, van Ryzin G (2001) Revenue management under a general discrete choice model of consumer behavior. Management Science 50(1):15–33

Tran MV, Nguyen TT, Nguyen TS, Le HQ (2010) Automatic named entity set expansion using semantic rules and wrappers for unary relations. In: Proceedings of the 2010 International Conference on Asian Language Processing, IALP '10, pp 170–173

Trivedi PK, Zimmer DM (2005) Copula modeling: an introduction for practitioners. Foundations and Trends in Econometrics 1(1):1–111

Vapnik VN (1995) The Nature of Statistical Learning Theory. Springer-Verlag, New York

Vapnik VN (1999) An overview of statistical learning theory. IEEE Transactions on Neural Networks 10(5):988–999

Vellido A, Martín-Guerrero JD, Lisboa PJ (2012) Making machine learning models interpretable. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning

Venkatesh R, Kamakura W (2003) Optimal bundling and pricing under a monopoly: constrasting complements and substitutes from independently valued products. Journal of Business 76(2):211–231

Venkatesh R, Mahajan V (1993) A probabilistic approach to pricing a bundle of products or services. Journal of Marketing Research 30:494–508

Verma S, Hruschka ER (2012) Coupled Bayesian sets algorithm for semi-supervised learning and information extraction. In: Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD '12, pp 307–322

Vulcano G, van Ryzin G (2014) A market discovery algorithm to estimate a general class of nonparametric choice models. Management Science 61(2):281–300

Vulcano G, van Ryzin G, Chaar W (2010) Choice-based revenue management: An empirical study of estimation and optimization. Manufacturing & Service Operations Management 12(3):371–392

Vulcano G, van Ryzin G, Ratliff R (2012) Estimating primary demand for substitutable products from sales transaction data. Operations Research 60(2):313–334

Wang F, Rudin C (2015) Falling rule lists. In: Proceedings of the 18th International Conference on Artificial Intelligence and Statistics, AISTATS '15, pp 1013–1022

Wang J, Lochovsky FH (2003) Data extraction and label assignment for web databases. In: Proceedings of the 12th International Conference on World Wide Web, WWW '03, pp 187–196

Wang RC, Cohen WW (2007) Language-independent set expansion of named entities using the web. In: Proceedings of the 2007 7th IEEE International Conference on Data Mining, ICDM '07, pp 342–350

Wang RC, Cohen WW (2008) Iterative set expansion of named entities using the web. In: Proceedings of the 2008 8th IEEE International Conference on Data Mining, ICDM '08, pp 1091–1096

Welling M, Teh YW (2011) Bayesian learning via stochastic gradient Langevin dynamics. In: Proceedings of the 28th International Conference on Machine Learning, ICML'11

Wu X, Zhang C, Zhang S (2004) Efficient mining of both positive and negative association rules. ACM Transactions on Information Systems 22(3):381–405

Wu Y, Tjelmeland H, West M (2007) Bayesian CART: prior specification and posterior simulation. Journal of Computational and Graphical Statistics 16(1):44–66

Wuebeker G, Mahajan V (1999) A conjoint analysis-based procedure to measure reservation price and to optimally price product bundles. In: Fuerderer R, Hermann A, Wuebeker G (eds) Optimal Bundling: Marketing Strategies for Improving Economic Performance, Springer-Verlag, pp 157–174

Xu JJ (1996) Statistical modelling and inference for multivariate and longitudinal discrete response. PhD thesis, Department of Statistics, University of British Columbia

Yan R, Hauptmann AG (2006) Efficient margin-based rank learning algorithms for information retrieval. In: Proceedings of the 5th International Conference on Image and Video Retrieval, CIVR '06, pp 113–122

Yi Y, Hüllermeier E (2005) Learning complexity-bounded rule-based classifiers by combining association analysis and genetic algorithms. In: Proceedings of the Joint 4th International Conference in Fuzzy Logic and Technology, EUSFLAT '05, pp 47–52

Yin X, Han J (2003) Cpar: classification based on predictive association rules. In: Proceedings of the 2003 SIAM International Conference on Data Mining, ICDM '03, pp 331–335

Yue Y, Finley T, Radlinski F, Joachims T (2007) A support vector method for optimizing average precision. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, pp 271–278

Zaki MJ (2000) Scalable algorithms for association mining. IEEE Transactions on Knowledge and Data Engineering 12(3):372–390

Zhai Y, Liu B (2005) Web data extraction based on partial tree alignment. In: Proceedings of the 14th International Conference on World Wide Web, WWW '05, pp 76–85

Zhang L, Liu B (2011) Entity set expansion in opinion documents. In: Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia, HT '11, pp 281–290

Zhu C, Byrd RH, Lu P, Nocedal J (1997) Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on Mathematical Software 23(4):550–560