JMLR: Workshop and Conference Proceedings 19 (2011) 615–634 24th Annual Conference on Learning Theory

Sequential Event Prediction with Association Rules

Cynthia Rudin RUDIN@MIT.EDU
MIT Sloan School of Management
Massachusetts Institute of Technology
77 Massachusetts Avenue Cambridge, MA 02139, USA

Benjamin Letham BLETHAM@MIT.EDU
Operations Research Center
Massachusetts Institute of Technology
77 Massachusetts Avenue Cambridge, MA 02139, USA

Ansaf Salleb-Aouissi ANSAF@CCLS.COLUMBIA.EDU
Center for Computational Learning Systems
Columbia University
475 Riverside Drive, New York, NY, 10115, USA

Eugene Kogan KOGAN.GENE@GMAIL.COM
Sourcetone
1295 5th Avenue Apt 14D, New York, NY 10029, USA

David Madigan MADIGAN@STAT.COLUMBIA.EDU
Department of Statistics
Columbia University
1255 Amsterdam Avenue, New York, NY 10027, USA

Editor: Sham Kakade, Ulrike von Luxburg

Abstract

We consider a supervised learning problem in which data are revealed sequentially and the goal is to determine what will next be revealed. In the context of this problem, algorithms based on association rules have a distinct advantage over classical statistical and machine learning methods; however, there has not previously been a theoretical foundation established for using association rules in supervised learning. We present two simple algorithms that incorporate association rules, and provide generalization guarantees on these algorithms based on algorithmic stability analysis from statistical learning theory. We include a discussion of the strict minimum support threshold often used in association rule mining, and introduce an "adjusted confidence" measure that provides a weaker minimum support condition that has advantages over the strict minimum support. The paper brings together ideas from statistical learning theory, association rule mining and Bayesian analysis.

Keywords: statistical learning theory, algorithmic stability, association rules, sequence prediction

1. Introduction

Given a "sequence database" of past event sequences to learn from, we aim to predict the next event within a current event sequence. Consider for instance, the data generated by a customer placing items into the virtual basket of an online grocery store such as NYC's

Fresh Direct, Peapod by Stop & Shop, or Roche Bros. The customer adds items one by one into the current basket, creating a sequence of events. The customer has identified him- or herself, so that all past orders are known. After each item selection, a confirmation screen contains a small list of recommendations for items that are not already in the basket. If the store can find patterns within the customer's past purchases, it may be able to accurately recommend the next item that the customer will add to the basket. There are many other domains in which a sequence of events is repeated in a somewhat similar way, and predictions need to be made before each event. Another example is to predict each next symptom of a sick patient, given the patient's past sequence of symptoms and treatments, and a database of the timeline of symptoms and treatments for other patients. In these examples, a subset of past events (for instance, a set of ingredients for a particular recipe, or a set of symptoms associated with a particular disease) can be useful in predicting the next event. We nickname the problem of predicting these sequentially revealed events based on subsets of past events "the online grocery store problem."

In order to make predictions using subsets of data, we employ *association rules* (Agrawal et al., 1993). An association rule in this setting is an implication $a \rightarrow b$ (such as *lettuce and carrots* $\rightarrow$ *tomatoes*), where a is a subset of items, and b is a single item. Association rule mining has proven successful for applications in market basket analysis (cross selling, product placement, affinity promotion, see also Kohavi et al., 2004), mining gene expression data (Jiang and Gruenwald, 2005) and weblog analysis (Huang et al., 2002). The association rule approach has the distinct advantage in being able to directly model underlying conditional probabilities $P(b|a)$ eschewing the linearity assumptions underlying many classical supervised classification, regression, and ranking methods. However, association rules are generally used as an exploratory tool, rather than a predictive tool (with some exceptions, such as the work of Liu et al., 1998; Veloso et al., 2008), and have not previously been established as a principled approach for supervised learning. Specifically, rule mining algorithms are generally used for finding patterns in databases without a hold-out test set.

Our main contribution is a framework and generalization analysis, in the context of the online grocery store problem, for supervised learning algorithms based on association rules. An important part of this analysis is how a fundamental property of a rule, namely the "support," is incorporated into the generalization bounds. The "support" of an itemset for the online grocery store problem is the number of times that the itemset has appeared in the sequence database. For instance, the support of *lettuce* is the number of times lettuce has been purchased in the past. Typically in association rule mining, a strict minimum support threshold condition is placed on the support of itemsets within a rule, so that rules falling below the minimum support threshold are simply discarded. The idea of a condition on the support is not shared with other types of supervised learning algorithms, since they do not use subsets in the same way as rule mining. Thus a new aspect of generalization is explored in our framework in that it handles predictions created from subsets of data. In classical supervised learning paradigms, bounds scale only with the sample size, and a large sample is necessary for generalization. In the context of association rules, the minimum support threshold forces predictions to be made only when there are enough data. Thus, in the association rules framework, there are now two mechanisms for generalization: first a large sample, and second, a minimum support. These are separate mechanisms, in the sense that it is possible to generalize with a somewhat small sample size and a large minimum

support threshold, and it is also possible to generalize with a large sample size and no support threshold. We thus derive two types of bounds: large sample bounds, which scale with the sample size, and small sample bounds, which scale with the minimum support of rules. Using both large and small sample bounds (that is, the minimum of the two bounds) gives a complete picture. The large sample bound is of order $\mathcal{O}(\sqrt{1/m})$ as in other supervised problems (classification, ranking, regression), where m denotes the number of event sequences in the database, that is, the number of past baskets ordered by the online grocery store customer.

Our bounds are derived using a specific notion of algorithmic stability called "pointwise hypothesis stability." The original notions of algorithmic stability were invented in the 1970's and have been revitalized recently (Devroye and Wagner, 1979; Bousquet and Elisseeff, 2002), the main idea being that algorithms may be better able to generalize if they are insensitive to small changes in the training data such as the removal or change of one training example. The pointwise hypothesis stability specifically considers the average change in loss that will occur at one of the training examples if that example is removed from the training set. Our generalization analysis uses conditions on the minimum support of rules in order to bound the pointwise hypothesis stability.

There are two algorithms considered in this work. At the core of each algorithm is a method for rank-ordering association rules where the list of possible rules is generated using the customer's past purchase history and subsets of items within the current basket. These algorithms build off of the rule mining literature that has been developing since the early 1990's (Agrawal et al., 1993) by using an application-specific rule mining method as a subroutine. Both of our algorithms are simple enough that they can be understood by users, customers, patients, managers, etc; an advantage of using association rules is that they are interpretable. Rules can provide a simple reason to the customer why an item might be relevant, or identify that a key ingredient is missing from a particular recipe. One of the algorithms considered in this work uses a fixed minimum support threshold to exclude rules whose itemsets occur rarely. Then the remaining rules are ranked according to the "confidence," which for rule $a \to b$ is the empirical probability that b will be in the basket given that a is in the basket. The right-hand sides of the highest ranked rules will be recommended by the algorithm. However, the use of a strict minimum support threshold is problematic for several well-known reasons, for instance it is known that important rules ("nuggets," which are rare but strong rules) are often excluded by a minimum support threshold condition.

The other algorithm introduced in this work provides an alternative to the minimum support threshold, in that rules are ranked by an "adjusted" confidence, which is a simple Bayesian shrinkage estimator of the probability of a rule $P(b|a)$. The right-hand sides of rules with the highest adjusted confidence are recommended by the algorithm. For this algorithm, the generalization guarantee (or bias-variance tradeoff) is smoothly controlled by a parameter K, which provides only a weak (less restrictive) minimum support condition. The key benefits of an algorithm based on the adjusted confidence are that: 1) it allows the possibility of choosing very accurate (high confidence) rules that have appeared very few times in the training set (low support), and 2) given two rules with the same or similar prediction accuracy on the training set (confidence), the rule that appears more frequently

(higher support) achieves a higher adjusted confidence and is thus preferred over the other rule.

All of the bounds are tied to the measure of quality (the evaluation metric, or loss function) used for the algorithm. We would like to directly compare the performance of algorithms for various settings of the adjusted confidence's K parameter (and for the minimum support threshold θ). It is problematic to have the loss defined using the same K value as the algorithm, in that case we would be using a different method of evaluation for each setting of K, and we would not be able to directly compare performance across different settings of K. To allow a direct comparison, we select one reference value of the adjusted confidence, called K_r (r for "reference"), and the loss depends on K_r rather than on K. The bounds are written generally in terms of K_r. The special case $K_r = 0$ is where the algorithm is evaluated with respect to the confidence measure. The small sample bound for the adjusted confidence algorithm has two terms: one that generally decreases with K (as the support increases, there is better generalization) and the other that decreases as K gets closer to K_r (better generalization as the algorithm is closer to the way it is being measured). These two terms are thus agreeing if $K_r > K$ and competing if $K_r < K$. In practice, the choice of K can be determined in several ways: K can be manually determined (for instance by the customer), it can be set using side information by "empirical Bayes" as considered by McCormick et al. (2011), or it can be set via cross-validation on an extra hold-out set.

Section 2 describes the max confidence, min support algorithm that has the hard support threshold, and the adjusted confidence algorithm that has the soft threshold. Section 3 provides the generalization analysis. Section 4 provides experimental results. Section 5 contains a discussion and summary of relevant literature. The longer version of the work (Rudin et al., 2011) contains proofs, strengthened versions of some of the results presented here, analogous results for binary classification, and a discussion regarding the suitability of other methods, specifically regression, for solving the sequential prediction problem.

2. Derivation of Algorithms

We assume an interface similar to that of Fresh Direct, where users add items one by one into the basket. After each selection, a confirmation screen contains a handful of recommendations for items that are not already in the customer's basket. The customer's past orders are known. The set of items is $\mathcal{X}$, for instance $\mathcal{X}$={apples, bananas, pears, etc}; $\mathcal{X}$ is the set of possible events. The customer has a past history of orders S which is a collection of m baskets, $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$; S is the sequence database. The customer's current basket is usually denoted by $B \subset \mathcal{X}$; B is the current sequence. An algorithm uses B and S to find rules $a \rightarrow b$, where a is in the basket and b is not in the basket. For instance, if *salsa* and *guacamole* are in the basket B and also if *salsa, guacamole* and *tortilla chips* were often purchased together in S, then the rule (*salsa* and *guacamole*) $\rightarrow$ *tortilla chips* might be used to recommend *tortilla chips*.

The support of a, written $\text{Sup}(a)$ or $\#a$, is the number of times in the past the customer has ordered itemset a, $\text{Sup}(a) := \#a := \sum_{i=1}^{m} \mathbf{1}_{[a \subseteq z_i]}$. If $a = \varnothing$, meaning a contains no items, then $\#a := \sum_i 1 = m$. The confidence of a rule $a \rightarrow b$ is $\text{Conf}(a \rightarrow b) := f_{S,0}(a, b) := \frac{\#(a \cup b)}{\#a}$, the fraction of times b is purchased given that a is purchased. It is an estimate

of the conditional probability of b given a. Ultimately an algorithm should order rules by conditional probability; however, the rules that possess the highest confidence values often have a left-hand side with small support, and their confidence values do not yield good estimates for the true conditional probabilities. We introduce the "adjusted" confidence as a remedy for this problem: the *adjusted confidence* for rule $a \to b$ is:

$$f_{S,K}(a,b) \;:=\; \frac{\#(a \cup b)}{\#a + K}.$$

The adjusted confidence for $K = 0$ is equivalent to the confidence. The adjusted confidence is a particular Bayesian estimate of the confidence. Specifically, assuming a beta prior distribution for the confidence, the posterior mean is given by:

$$\hat{p} = \frac{L + \#(a \cup b)}{L + K + \#a},$$

where L and K denote the parameters of the beta prior distribution. The beta distribution is the "conjugate" prior distribution for a binomial likelihood. For the adjusted confidence we choose $L = 0$. This choice yields the benefits of the lower bounds derived in this section, and the stability properties described later. The prior for the adjusted confidence tends to bias rules towards the *bottom* of the ranked list. Any rule achieving a high adjusted confidence must overcome this bias.

Other choices for L and K are meaningful. A *collaborative filtering prior* would have $L/(L+K)$ represent the probability of purchasing item b given that item a was purchased, calculated over a subset of other customers. A *revenue management prior* would have L and K be based on the item price, favoring expensive items.

A rule cannot have a high adjusted confidence unless it has both a large enough confidence and a large enough support on the left side. To see this, take $f_{S,K}(a,b)$ large, meaning for some η, we have $f_{S,K}(a,b) > \eta$, implying:

$$\mathrm{Conf}(a \to b) = f_{S,0}(a,b) > \eta \frac{\#a + K}{\#a} \geq \eta,$$

$$\mathrm{Sup}(a) = \#a \geq (\#a + K)\left[\frac{\#(a \cup b)}{\#a + K}\right] > (\#a + K)\eta, \text{ implying } \mathrm{Sup}(a) = \#a > \frac{\eta K}{1 - \eta}. \quad (1)$$

And further, expression (1) implies:

$$\mathrm{Sup}(a \cup b) = \#(a \cup b) > \eta(\#a + K) > \eta K/(1 - \eta).$$

Thus, rules attaining high values of adjusted confidence have a lower bound on confidence, and a lower bound on support of both the right- and left-hand sides, which means a better estimate of the conditional probability. As K increases, rules with low support are heavily penalized with respect to the adjusted confidence, so they tend not to be at the top of the list. On the other hand, such rules might be chosen when all other rules have low confidence. That is the main advantage of having no *firm* minimum support cutoff: "nuggets" that have fairly low support may filter to the top.

We now formally state the recommendation algorithms. Both algorithms use a subroutine for mining association rules to generate a set of candidate rules. One of the simplest

Input: $(S, B, \mathcal{X})$, that is, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, set of items $\mathcal{X}$

Output: Set of all rules $\{a_j \to b_j\}_j$ where b_j is a single item that is not in the basket B, and where a_j is either a subset of items in the basket B, or else it is the empty set. Also the left-hand side a_j must be allowed (meaning it is in A). That is, output rules $\{a_j \to b_j\}_j$ such that $b_j \in \mathcal{X} \backslash B$ and $a_j \subseteq B \subset \mathcal{X}$ with $a_j \in A$, or $a_j = \varnothing$.

Algorithm 1: *Subroutine GenRules.*

Input: $(\theta, \mathcal{X}, S, B,$ *GenRules*, $c)$, that is, minimum threshold parameter θ, set of items $\mathcal{X}$, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, *GenRules* generates candidate rules $\textit{GenRules}(S, B, \mathcal{X}) = \{a_j \to b_j\}_j$, number of recommendations $c \geq 1$

Output: Recommendation List, which is a subset of c items in $\mathcal{X}$

Algorithm:

1) Apply $\textit{GenRules}(S, B, \mathcal{X})$ to get rules $\{a_j \to b_j\}_j$ where a_j is in the basket B and b_j is not.

2) Compute score for each rule $a_j \to b_j$ as $\bar{f}_{S,\theta}(a_j, b_j) = f_{S,0}(a_j, b_j) = \frac{\#(a_j \cup b_j)}{\#a_j}$ when support $\#a_j \geq \theta$, and $\bar{f}_{S,\theta}(a_j, b_j) = 0$ otherwise.

3) Reorder rules by decreasing score.

4) Find the top c rules with distinct right-hand sides, and let Recommendation List be the right-hand sides of these rules.

Algorithm 2: Max Confidence, Min Support Algorithm.

such rule mining algorithms is *GenRules*, provided as Algorithm 1, which in practice should be made specific by using a rule mining algorithm that retrieves a set of rules tailored to the application. *GenRules* can be replaced by any algorithm for mining association rules; there is a vast literature on such algorithms since the field of association rule mining evolved on their development, e.g. Apriori (Agrawal et al., 1993). *GenRules* requires a set A which is the set of allowed left-hand sides of rules.

2.1. Max Confidence, Min Support Algorithm

The max confidence, min support algorithm, shown as Algorithm 2, is based on the idea of eliminating rules whose itemsets occur rarely, which is commonly done in the rule-mining literature. For this algorithm, the rules are ranked by confidence, and rules that do not achieve a predetermined fixed minimum support threshold are completely omitted. The algorithm recommends the right-hand sides from the top ranked rules. Specifically, if c items are to be recommended to the user, the algorithm picks the top ranked c distinct items.

It is common that the minimum support threshold is imposed on the right and left side $\text{Sup}(a \cup b) \geq \theta$; however, as long as $\text{Sup}(a)$ is large, we can get a reasonable estimate of $P(b|a)$. In that sense, it is sufficient (and less restrictive) to impose the minimum support

Input: $(K, \mathcal{X}, S, B,$ *GenRules*, $c)$, that is, parameter K, set of items $\mathcal{X}$, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, *GenRules* generates candidate rules $GenRules(S, B, \mathcal{X}) = \{a_j \to b_j\}_j$, number of recommendations $c \geq 1$
Output: Recommendation List, which is a subset of c items in $\mathcal{X}$
Algorithm:
1) Apply $GenRules(S, B, \mathcal{X})$ to get rules $\{a_j \to b_j\}_j$ where a_j is in the basket B and b_j is not.
2) Compute adjusted confidence of each rule $a_j \to b_j$ as $f_{S,K}(a_j, b_j) = \frac{\#(a_j \cup b_j)}{\#a_j + K}$.
3) Reorder rules by decreasing adjusted confidence.
4) Find the top c rules with distinct right-hand sides, and let Recommendation List be the right-hand sides of these rules.

Algorithm 3: Adjusted Confidence Algorithm.

threshold on the left side: $\text{Sup}(a) \geq \theta$. In this work, we only have a required minimum support on the left side. As a technical note, we might worry about the minimum support threshold being so high that there are no rules that meet the threshold. This is actually not a major concern because of the minimum support being imposed only on the left-hand side: as long as $m \geq \theta$, all rules $\varnothing \to b$ meet the minimum support threshold. The thresholded confidence is denoted by $\bar{f}_{S,\theta}$:

$$\bar{f}_{S,\theta}(a, b) := f_{S,0}(a, b) \text{ if } \#a \geq \theta, \text{and } \bar{f}_{S,\theta}(a, b) := 0 \text{ otherwise.}$$

2.2. Adjusted Confidence Algorithm

The adjusted confidence algorithm is shown as Algorithm 3. A chosen value of K is used to compute the adjusted confidence for each rule, and rules are then ranked according to adjusted confidence.

The definition of the adjusted confidence makes an implicit assumption that the order in which items were placed into previous baskets is irrelevant. It is easy to include a dependence on the order by defining a "directed" version of the adjusted confidence, and calculations can be adapted accordingly. The numerator of the adjusted confidence becomes the number of past orders where a is placed in the basket *before* b.

$$f_{S,K}^{(\text{directed})}(a, b) \quad = \quad \frac{\#\{(a \cup b) : b \text{ follows } a\}}{\#a + K}.$$

3. Generalization and Stability

Our main calculations show that each algorithm's empirical error does not dramatically change by altering one of the training examples. These calculations will be used within algorithmic stability analysis (Rogers and Wagner, 1978; Devroye and Wagner, 1979; Bousquet and Elisseeff, 2002). Stability bounds depend on how the space of functions is searched by the algorithm (rather than the size of the function space). There are many different ways to measure the stability of an algorithm; the bounds presented here use pointwise hypothesis stability so that the bounds scale correctly with the number of training examples m. For

simplicity, the algorithm recommends only one item, $c = 1$. Section 3.2 provides bounds for the large sample asymptotic regime where neither the minimum support threshold θ nor the choice of K matters. Then we consider the new small m regime in Section 3.3, starting with a bound that formally shows that minimum support thresholds lead to better generalization. From there, we present a small sample bound for the adjusted confidence.

If the top recommendation has a higher adjusted confidence than the next item added, the algorithm incurs an error. (Even if that item is added later on, the algorithm incurs an error at this timestep.) To measure the size of that error, we can use a 0-1 loss, indicating whether or not our algorithm gave the highest adjusted confidence to the next item added. However, the 0-1 loss does not capture how close our algorithm was to correctly predicting the next item, though this information might be useful in determining how well the algorithm will generalize. We approximate the 0-1 loss using a modified loss that decays linearly near the discontinuity. This modified loss allows us to consider differences in adjusted confidence, not just whether one is larger than another:

$$\begin{aligned}|&(\text{adjusted conf. of highest-scoring-correct rule})\\ &-(\text{adjusted conf. of highest-scoring-incorrect rule})|.\end{aligned} \tag{2}$$

However, as discussed in the introduction, if we adjust the loss function's K value to match the adjusted confidence K value, then we cannot fairly compare the algorithm's performance using two different values of K. An illustration of this point is that for large K, all adjusted confidence values are $\ll 1$, and for small K, then the adjusted confidence can be ≈ 1; differences in adjusted confidence for small K cannot be directly compared to those for large K. Since we want to directly compare performance as K is adjusted, we fix an evaluation measure that is separate from the choice of K. Specifically, we use the difference in adjusted confidence values with respect to a reference K_r:

$$\begin{aligned}|&(\{\text{adjusted conf.}\}_{K_r}\text{of highest-scoring-correct rule}_K)\\ &-(\{\text{adjusted conf.}\}_{K_r}\text{of highest-scoring-incorrect rule}_K)|.\end{aligned} \tag{3}$$

The reference K_r is a parameter of the loss function, whereas K is a parameter of an algorithm. We set $K_r = 0$ to measure loss using the difference in confidence, and $K = 0$ for an algorithm that chooses rules according to the confidence. As K gets farther from K_r, the algorithm is more distant from the way it is being evaluated, which leads to worse generalization.

3.1. Notation

We have a training set of m baskets $S = \{z_i\}_{1\ldots m}$ that are the customers past orders. The baskets are chosen randomly (iid) from a fixed (but unknown) probability distribution $\mathcal{D}$ over possible baskets. The generalization bound will be a guarantee on performance for a new randomly chosen basket. A basket z consists of an ordered (permuted) set of items, $z \in 2^{\mathcal{X}} \times \Pi$, where $2^{\mathcal{X}}$ is the set of all subsets of $\mathcal{X}$, and Π is the set of permutations over at most $|\mathcal{X}|$ elements. Denote $z \sim \mathcal{D}$ to mean that basket z is drawn randomly (iid) according to distribution $\mathcal{D}$ over the space of possible items in baskets and permutations over those items, $2^{\mathcal{X}} \times \Pi$. The t^{th} item added to the basket is written $z_{\cdot,t}$, where the dot is just a

placeholder for the generic basket z. The t^{th} element of the i^{th} basket in the training set is written $z_{i,t}$. We define the number of items in basket z by T_z and overload notation by defining T_i to be the number of items in the i^{th} training basket, $T_i := |z_i|$. Recall that *GenRules* produces only rules whose left-hand sides are in an allowed set A.

For the adjusted confidence algorithm, given a basket z and a particular time t, the algorithm uses the training set S to compute the adjusted confidences $f_{S,K}$. A *highest-scoring-correct* rule is a highest scoring rule that has the next item $z_{\cdot,t+1}$ on the right. The left side a^+_{SztK} of a highest-scoring-correct rule obeys:

$$a^+_{SztK} \in \operatorname*{argmax}_{a \subseteq \{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a \in A} f_{S,K}(a, z_{\cdot,t+1}).$$

(If $z_{\cdot,t+1}$ has never been purchased, the adjusted confidence for all rules $a \to z_{\cdot,t+1}$ is 0, and we choose the maximizing rule to be $\varnothing \to z_{\cdot,t+1}$.) A highest-scoring-correct rule correctly recommends the next item, and it is a best rule for the algorithm to choose.

The algorithm incurs an error when it recommends an incorrect item. A *highest-scoring-incorrect* rule is a highest scoring rule that does not have $z_{\cdot,t+1}$ on the right. It is denoted $a^-_{SztK} \to b^-_{SztK}$, and obeys:

$$[a^-_{SztK}, b^-_{SztK}] \in \operatorname*{argmax}_{\substack{a \subseteq \{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a \in A \\ b \in \mathcal{X} \setminus \{z_{\cdot,1},\ldots,z_{\cdot,t+1}\}}} f_{S,K}(a, b).$$

If there is more than one highest-scoring rule, one is chosen at random. (With the exception that all incorrect rules are tied at zero adjusted confidence, in which case the left side is taken as $\varnothing$ and the right side is chosen randomly). The adjusted confidence algorithm determines a^+_{SztK}, a^-_{SztK}, and b^-_{SztK}, whereas nature chooses $z_{\cdot,t+1}$.

If the adjusted confidence of the rule $a^-_{SztK} \to b^-_{SztK}$ is larger than that of $a^+_{SztK} \to z_{\cdot,t+1}$, it means that the algorithm recommended the wrong item. The loss function below counts the proportion of times this happens for each basket, and is defined with respect to K_r.

$$\ell_{0\text{-}1,K_r}(f_{S,K}, z) := \frac{1}{T_z} \sum_{t=0}^{T_z-1} \begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^-_{SztK}, b^-_{SztK}) \leq 0 \\ 0 & \text{otherwise.} \end{cases}$$

We will now define the true error. The true error is an expectation of the loss function with respect to $\mathcal{D}$, and is a random variable since the training set S is random, $S \sim \mathcal{D}^m$.

$$\text{TrueErr}(f_{S,K}, K_r) := \mathbb{E}_{z \sim \mathcal{D}} \ell_{0\text{-}1,K_r}(f_{S,K}, z).$$

We upper bound the true error by using a different loss ℓ_{γ,K_r} that is a continuous upper bound on the 0-1 loss $\ell_{0\text{-}1,K_r}$. It is defined with respect to K_r and another parameter $\gamma > 0$ as follows:

$$\ell_{\gamma,K_r}(f_{S,K}, z) := \frac{1}{T_z} \sum_{t=0}^{T_z-1} c_\gamma(f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^-_{SztK}, b^-_{SztK})), \text{ where}$$

$$c_\gamma(y) = \begin{cases} 1 & \text{for } y \leq 0 \\ 1 - y/\gamma & \text{for } 0 \leq y \leq \gamma \\ 0 & \text{for } y \geq \gamma. \end{cases}$$

As γ approaches 0, this loss approaches the 0-1 loss. Also, $\ell_{0-1,K_r}(f_{S,K}, z) \leq \ell_{\gamma,K_r}(f_{S,K}, z)$. We define TrueErr_γ using this loss:

$$\text{TrueErr}_\gamma(f_{S,K}, K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell_{\gamma,K_r}(f_{S,K}, z),$$

where $\text{TrueErr} \leq \text{TrueErr}_\gamma$. The first set of results below bound TrueErr by considering the difference between TrueErr_γ and its empirical counterpart that we define next.

We overload notation by replacing z for a generic basket with i for a training basket. The left-hand side a^+_{SitK} of a highest-scoring-correct rule for basket z_i at time t obeys :

$$a^+_{SitK} \in \underset{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A}{\text{argmax}} f_{S,K}(a, z_{i,t+1}),$$

similarly, a highest-scoring-incorrect rule for z_i at time t has:

$$[a^-_{SitK}, b^-_{SitK}] \in \underset{\substack{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{i,1},\ldots,z_{i,t+1}\}}}{\text{argmax}} f_{S,K}(a, b).$$

The empirical error is an average of the loss over the training baskets:

$$\text{EmpErr}_\gamma(f_{S,K}, K_r) := \frac{1}{m}\sum_{\text{baskets } i=1}^{m} \ell_{\gamma,K_r}(f_{S,K}, z_i).$$

For the max confidence, min support algorithm, we substitute θ where K appears in the notation. Again we use z when referring to a randomly drawn basket and i to refer to a specific training basket z_i. For instance, for training basket z_i, we define $a^+_{Sit\theta}, a^-_{Sit\theta}$, and $b^-_{Sit\theta}$ by:

$$a^+_{Sit\theta} \in \underset{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A}{\text{argmax}} \bar{f}_{S,\theta}(a, z_{i,t+1}),$$

$$[a^-_{Sit\theta}, b^-_{Sit\theta}] \in \underset{\substack{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{i,1},\ldots,z_{i,t+1}\}}}{\text{argmax}} \bar{f}_{S,\theta}(a, b), \text{ and similarly,}$$

$$\ell_{0-1,K_r}(\bar{f}_{S,\theta}, z_i) := \frac{1}{T_i}\sum_{t=0}^{T_i-1}\begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{Sit\theta}, z_{i,t+1}) - f_{S,K_r}(a^-_{Sit\theta}, b^-_{Sit\theta}) \leq 0 \\ 0 & \text{otherwise.}\end{cases}$$

$$\ell_{\gamma,K_r}(\bar{f}_{S,\theta}, z_i) := \frac{1}{T_i}\sum_{t=0}^{T_i-1} c_\gamma(f_{S,K_r}(a^+_{Sit\theta}, z_{i,t+1}) - f_{S,K_r}(a^-_{Sit\theta}, b^-_{Sit\theta}))$$

and $\text{TrueErr}(\bar{f}_{S,\theta}, K_r)$ and $\text{TrueErr}_\gamma(\bar{f}_{S,\theta}, K_r)$ are defined analogously as expectations of the losses, and $\text{EmpErr}_\gamma(\bar{f}_{S,\theta}, K_r)$ is again an average of the loss over the training baskets.

3.2. Generalization Analysis for Large m

The choice of minimum support threshold θ or the choice of parameter K matters mainly in the regime where m is small. For the max confidence, min support algorithm, when m is large, then all itemsets that would be chosen by the customer have appeared more times than the minimum support threshold with high probability. For the adjusted confidence algorithm, when m is large, prediction ability is guaranteed as follows.

Theorem 1 *(Generalization Bound for Adjusted Confidence Algorithm, Large m)*
For set of rules A and $K \geq 0$, $K_r \geq 0$, with probability $1-\delta$ (with respect to training set $S \sim \mathcal{D}^m$),

$$\text{TrueErr}(f_{S,K}, K_r) \leq \text{EmpErr}_\gamma(f_{S,K}, K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + 6\beta\right]}$$
$$\text{where } \beta = \frac{2|\mathcal{A}|}{\gamma}\left[\frac{1}{(m-1)p_{\min A} + K} + \frac{|K_r - K|\frac{m}{m+K}}{(m-1)p_{\min A} + K_r}\right] + \mathcal{O}\left(\frac{1}{m^2}\right),$$

and where $\mathcal{A} = \{a \in A : P_z(a \subseteq z) > 0\}$ are the itemsets that have some probability of being chosen. Out of these, any itemset that is the least likely to be chosen has probability $p_{\min A}$:

$$p_{\min A} := \min_{a \in \mathcal{A}} P_{z \sim \mathcal{D}}(a \subseteq z).$$

A special case is where $K_r = K = 0$: the algorithm chooses the rule with maximum confidence, and accuracy is then judged by the difference in confidence values between the highest-scoring-incorrect rule and the highest-scoring-correct rule. The expression reduces to:

Corollary 2 *(Generalization Bound for Maximum Confidence Setting, Large m)*
With probability $1-\delta$ (with respect to $S \sim \mathcal{D}^m$),

$$\text{TrueErr}(f_{S,0}, 0) \leq \text{EmpErr}_\gamma(f_{S,0}, 0) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + \frac{12|\mathcal{A}|}{\gamma(m-1)p_{\min A}}\right] + \mathcal{O}\left(\frac{1}{m^2}\right)}.$$

The use of the pointwise hypothesis stability within this proof is the key to providing a decay of order $\sqrt{(1/m)}$. Now that this bound is established, we move to the small sample case, where the minimum support is the force that provides generalization.

3.3. Generalization Analysis for Small m

The first small sample result is a general bound for the max confidence, min support algorithm, that is, Algorithm 2.

Theorem 3 *(Generalization Bound for Max Confidence, Min Support)*
For $\theta \geq 1$, $K_r \geq 0$, with probability $1-\delta$ (with respect to $S \sim \mathcal{D}^m$), $m > \theta$,

$$\text{TrueErr}(\bar{f}_{S,\theta}, K_r) \leq \text{EmpErr}_\gamma(\bar{f}_{S,\theta}, K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + 6\beta\right]}$$
$$\text{where } \beta = \frac{2}{\gamma}\left[\frac{1}{\theta} + K_r\left(\frac{1}{\theta + K_r}\right)\left(1 + \frac{1}{\theta}\right)\right].$$

Figure 1 shows β as a function of θ for several different values of K_r. The special case of interest is when $K_r = 0$, so that the loss is judged with respect to differences in confidence, as follows:

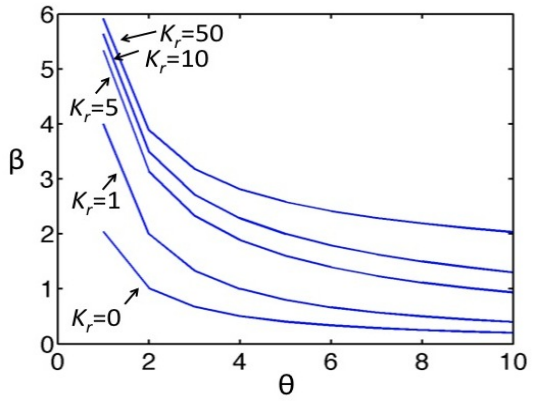

Figure 1: β vs. θ from Theorem 3, with $\gamma = 1$. The different curves are different values of $K_r = 0$, 1, 5, 10, 50 from bottom to top.

Corollary 4 *(Generalization Bound for Max Confidence, Min Support, $K_r = 0$) For $\theta \geq 1$, with probability $1 - \delta$ (with respect to $S \sim \mathcal{D}^m$), $m > \theta$,*

$$\text{TrueErr}(\bar{f}_{S,\theta}, 0) \leq \text{EmpErr}_\gamma(\bar{f}_{S,\theta}, 0) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + \frac{12}{\theta\gamma}\right]}.$$

It is common to use a minimum support threshold that is a fraction of m, for instance, $\theta = 0.1 \times m$. In that case, the bound again scales with $\sqrt{(1/m)}$. Note that there is no generalization guarantee when $\theta = 0$; the minimum support threshold enables generalization in the small m case.

Now we discuss the adjusted confidence algorithm for small m setting. In the proof of the following theorem, if we were to use the definitions established above, the bound does not simplify beyond a certain point and is difficult to read at an intuitive level. From that bound, it would not be easy to see what are the important quantities for the learning process, and how they scale. In what follows, we redefine the loss function slightly, so that it approximates a 0-1 loss from below instead of from above. This provides a concise and intuitive bound. Almost the same proof can be used to create both versions of the bound, only the last steps are different.

Define a *highest-scoring* rule $a^*_{SztK} \rightarrow b^*_{SztK}$ as a rule that achieves the maximum adjusted confidence, over all of the possible rules. It will either be equal to $a^+_{SztK} \rightarrow z_{\cdot,t+1}$ or $a^-_{SztK} \rightarrow b^-_{SztK}$, depending on which has the larger adjusted confidence:

$$[a^*_{SztK}, b^*_{SztK}] \in \underset{\substack{a \subseteq \{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a \in A \\ b \in \mathcal{X} \setminus \{z_{\cdot,1},\ldots,z_{\cdot,t}\}}}{\text{argmax}} f_{S,K}(a, b).$$

Note that b^*_{SztK} can be equal to $z_{\cdot,t+1}$ whereas b^-_{SztK} cannot. The notation for a^*_{SitK} and b^*_{SitK} is similar, and the new loss is:

$$\ell^{\text{new}}_{0-1,K_r}(f_{S,K}, z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK}) < 0 \\ 0 & \text{otherwise.} \end{cases}$$

By definition, the difference $f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK})$ can never be strictly positive. The continuous approximation is:

$$\ell^{\text{new}}_{\gamma,K_r}(f_{S,K}, z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1} c^{\text{new}}_\gamma(f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK})), \text{ where}$$

$$c^{\text{new}}_\gamma(y) = \begin{cases} 1 & \text{for } y \le -\gamma \\ -y/\gamma & \text{for } -\gamma \le y \le 0 \\ 0 & \text{for } y \ge 0. \end{cases}$$

As γ approaches 0, the c_γ loss approaches the 0-1 loss. We define $\text{TrueErr}^{\text{new}}_\gamma$ and $\text{EmpErr}^{\text{new}}_\gamma$:

$$\text{TrueErr}^{\text{new}}_\gamma(f_{S,K}, K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell^{\text{new}}_{\gamma,K_r}(f_{S,K}, z),$$

$$\text{EmpErr}^{\text{new}}_\gamma(f_{S,K}, K_r) := \frac{1}{m}\sum_{i=1}^{m}\ell^{\text{new}}_{\gamma,K_r}(f_{S,K}, z_i).$$

The minimum support threshold condition we used earlier is replaced by a weaker condition on the support. This weaker condition has the benefit of allowing more rules to be used in order to achieve a better empirical error; however, it is more difficult to get a generalization guarantee. This support condition is derived from the fact that the adjusted confidence of the highest-scoring rule $a^*_{SitK} \to b^*_{SitK}$ exceeds that of the highest-scoring-correct rule $a^+_{SitK} \to z_{i,t+1}$, which exceeds that of the marginal rule $\varnothing \to z_{i,t+1}$:

$$\frac{\#a^*_{SitK}}{\#a^*_{SitK}+K} \ge \frac{\#(a^*_{SitK}\cup b^*_{SitK})}{\#a^*_{SitK}+K} \ge \frac{\#(a^+_{SitK}\cup z_{i,t+1})}{\#a^+_{SitK}+K} \ge \frac{\#z_{i,t+1}}{m+K}. \tag{4}$$

This leads to a lower bound on the support $\#a^*_{SitK}$:

$$\#a^*_{SitK} \ge K\left(\frac{\#z_{i,t+1}}{m+K-\#z_{i,t+1}}\right). \tag{5}$$

This is not a hard minimum support threshold, yet since the support generally increases as K increases, the bound will give a better guarantee for large K. Note that in the original notation, we would replace the condition (4) with $\frac{\#a^-_{SitK}}{\#a^-_{SitK}+K} \ge \frac{\#(a^-_{SitK}\cup b^-_{SitK})}{\#a^-_{SitK}+K} \ge \frac{\#b^-_{SitK}}{m+K}$ and proceed with analogous steps in the proof.

Theorem 5 *(Generalization Bound for Adjusted Confidence Algorithm, Small m)*

$$\text{TrueErr}^{\text{new}}_\gamma(f_{S,K}, K_r) \le \text{EmpErr}^{\text{new}}_\gamma(f_{S,K}, K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m}+6\beta\right]} \text{ where}$$

$$\begin{aligned}\beta &= \frac{2}{\gamma}\frac{1}{K}\left(1-\frac{(m-1)p_{\min}}{m+K}\right) \\ &\quad + \frac{2}{\gamma}|K_r-K|\mathbb{E}_{\zeta\sim Bin(m-1,p_{\min})}\frac{1}{K\left(\frac{\zeta}{m+K-\zeta-1}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\zeta}{m+K}\right)\right),\end{aligned}$$

and where $Q = \{x \in \mathcal{X} : P_{z\sim D}(x \in z) > 0\}$ are the items that have some probability of being chosen by the customer. Out of these, any item that is the least likely to be chosen has probability $p_{\min}$:

$$p_{\min} := \min_{x\in Q} P_{z\sim\mathcal{D}}(x \in z).$$

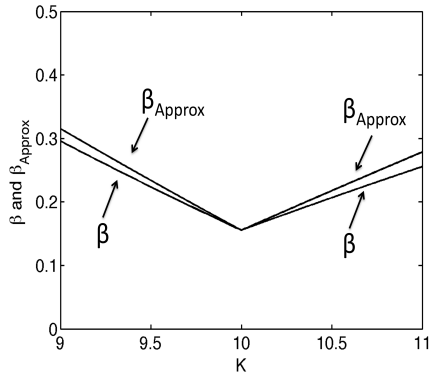

Figure 2: β and β_{Approx} vs K, where $K_r = 10$, $p_{\min} = 0.3$, $m = 20$, $\gamma = 1$.

The stability β has two main terms. The first term decreases generally as $1/K$. The second term arises from the error in measuring loss with K_r rather than K. In order to interpret β, consider the following approximation to the expectation in the bound, which assumes that m is large and that $m \gg K \gg 0$, and that $\zeta \approx mp_{\min}$:

$$\beta \approx \frac{2}{\gamma}\frac{1}{K}\left(1 - \frac{(m-1)p_{\min}}{m+K}\right) + \frac{2}{\gamma}|K_r - K|\frac{1}{K\frac{p_{\min}}{1-p_{\min}} + K_r}. \tag{6}$$

Intuitively, if either K is close to K_r or $p_{\min}$ is large (close to 1) then this term becomes small. Figure 2 shows an example plot of β and the approximation using (6), which we denote by β_{Approx}.

One can observe that if $K_r > K$, then both terms tend to improve (decrease) with increasing K. When $K_r < K$, then the two terms can compete as K increases.

3.4. Summary of Bounds

We have provided probabilistic guarantees on performance that show the following: 1) For large m, the association rule-based algorithms for sequential event prediction have a performance guarantee of the same order as for classical supervised learning problems (classification, ranking, regression, density estimation). 2) For small m, the minimum support threshold guarantees generalization (at the expense of removing important rules). 3) The adjusted confidence provides a much weaker support threshold, allowing important rules to be used, while still being able to generalize. 4) All generalization guarantees depend on the way the goodness of the algorithm is measured (the choice of K_r in the loss function). There are two terms in the small sample size bound for the adjusted confidence. Generally, one term decreases (becomes better) as K increases, and the other term decreases as K gets closer to K_r.

4. Experiments

All datasets chosen for these experiments are publicly available from the UCI machine learning repository (Frank and Asuncion, 2010), and from the IBM Quest Market-Basket Synthetic Data Generator (Agrawal and Srikant, 1994). To obtain formatted market-basket

Input: $(S, B, \mathcal{X})$, that is, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, set of items $\mathcal{X}$

Output: Set of all rules where a_j is an item in the basket B (or the empty set) and b_j is not in B. That is, rules $\{a_j \to b_j\}_j$ such that $b_j \in \mathcal{X} \backslash B$ and either $a_j \in B$ or $a_j = \varnothing$.

Algorithm 4: *Subroutine GenRules*, simplest version that considers only "marginal" rules.

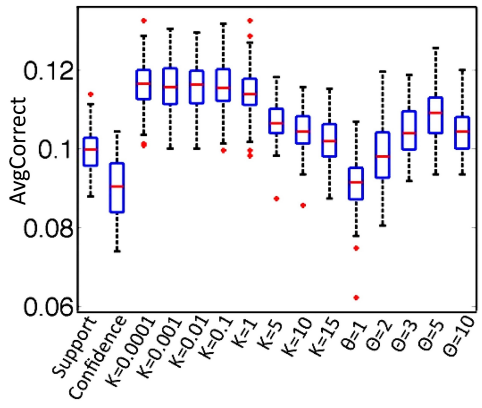

Figure 3: Boxplots of AvgCorrect values for Mushroom dataset.

Algorithm	mean ± standard dev.
Support	0.0996 ± 0.0051
Confidence	0.0902 ± 0.0075
K=0.0001	**0.1164 ± 0.0061**
K=0.001	**0.1158 ± 0.0062**
K=0.01	**0.1161 ± 0.0061**
K=0.1	**0.116 ± 0.0058**
K=1	0.1142 ± 0.0062
K=5	0.1069 ± 0.0052
K=10	0.1044 ± 0.0054
K=15	0.1024 ± 0.0053
θ=1	0.0909 ± 0.007
θ=2	0.0986 ± 0.0077
θ=3	0.1048 ± 0.0064
θ=5	0.1088 ± 0.0069
θ=10	0.1042 ± 0.0057

Figure 4: Means and standard deviations for Mushroom dataset. Bold indicates no significant difference from the best algorithm.

data, categorical data were converted into binary features (one feature per category). Each feature represents an item, and each example represents a basket. The feature value (0 or 1) indicates the presence of an item. Training baskets and test baskets were chosen randomly without replacement from the full dataset. Since these data do not come naturally with a time ordering, items in the basket were randomly permuted to attain an order. At each iteration, rules were formed from one item or the empty item on the left, and one item on the right (See *GenRules* as Algorithm 4). Recommendations of one item were made using the following 15 algorithms: highest support, highest confidence, highest adjusted confidence for eight K levels, max confidence, min support algorithm for five support threshold levels θ. All 15 algorithms were evaluated by the average fraction of correct recommendations (AvgCorrect) per basket. As recommendations were made, it was common to have ties where multiple items are equally good to recommend, in which case the tie was broken at random; AvgCorrect is similar to $\ell_{0-1,K}$ except for this way of dealing with ties.

The parameters of the experiment are: number of training baskets (20 in all cases), number of test baskets (100 in all cases), values of K for the adjusted confidence algorithm (0.0001, 0.001, 0.01, 0.1, 1, 5, 10, 15), and values of θ for the max confidence, min support algorithm (1, 2, 3, 5, 10). Note that two of these algorithms are the same: the

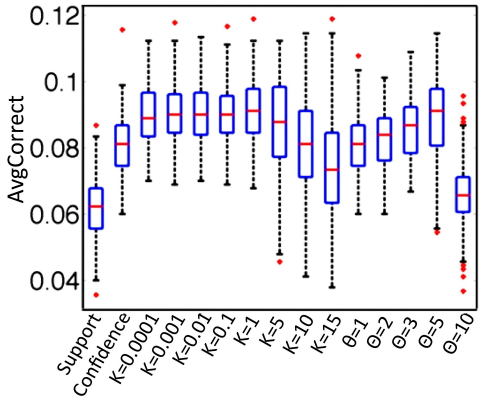

Figure 5: Boxplots of AvgCorrect values for Nursery dataset.

Algorithm	mean ± standard dev.
Support	0.0619 ± 0.0098
Confidence	0.081 ± 0.0094
K=0.0001	0.0898 ± 0.0091
K=0.001	**0.0902 ± 0.0093**
K=0.01	**0.0902 ± 0.0085**
K=0.1	**0.0903 ± 0.0095**
K=1	**0.0909 ± 0.0096**
K=5	0.0869 ± 0.0139
K=10	0.0804 ± 0.0154
K=15	0.0747 ± 0.0154
θ=1	0.0811 ± 0.0088
θ=2	0.0819 ± 0.0094
θ=3	0.0858 ± 0.0095
θ=5	0.0883 ± 0.0137
θ=10	0.0654 ± 0.0111

Figure 6: Means and standard deviations for Nursery dataset.

max confidence algorithm is the same as the max confidence, min support algorithm for θ=1. Datasets are: Car Evaluation (25 items, 1728 baskets), Chess King-Rook vs King-Pawn, (75 items, 3196 baskets), MONK's problems (19 items, 1711 baskets) Mushroom (119 items, 8124 baskets), Nursery (32 items, 12960 baskets), Plants (70 items, 34781 baskets), T20I18D10KN22CR50 (22 items, 10000 baskets). Each experiment (training, test, evaluation for all 15 algorithms) was performed 100 times, (totaling $100\times100\times15 = 150{,}000$ test basket evaluations per dataset, for each of 7 datasets). In Figures 3 through 6, the distribution of AvgCorrect values for each algorithm on datasets Mushroom and Nursery is shown via boxplot, along with the mean and standard deviation of AvgCorrect values for each algorithm. Bold indicates that the mean is not significantly different from that of the algorithm with the largest mean value; that is, bold indicates the highest scores. Similar experimental results for the other datasets are in the longer version (Rudin et al., 2011).

Figure 7 summarizes the results of all of the experiments by totaling the number of datasets for which each algorithm achieved one of the highest scores. The best performing algorithms were $K = 0.01$ and $K = 0.1$, both algorithms achieving one of the top scores for 6 out of 7 of the datasets. The single dataset for which these algorithms did not achieve one the best scores was the very dense dataset T20I18D10KN22CR50, where the algorithms requiring a higher support (the max support algorithm, and also the adjusted confidence algorithm for $K = 5, 10$, and 15) achieved the highest AvgCorrect score. In that case, the $K = 0.01$ and $K = 0.1$ algorithms still performed better than the max confidence, min support algorithms for the parameters we tried.

The adjusted confidence algorithm with a very small K is similar to using the max confidence algorithm, except that whenever there is a tie, the tie is broken in favor of the rule with largest support. It seems that in most of the datasets we chose, this type of algorithm performed the best, which indicates two things. First, that for some datasets, increasing K too much can have the same effect as a too-large minimum support threshold: large values of K could potentially remove the best rules, leading to too much bias, where

the algorithm cannot explain enough of the variance in the data. Second, when comparing rules, it is important not to break ties at random as in the max confidence, min support algorithm, but instead to use the support of the rules. Another observation is that the performance levels of the adjusted confidence algorithm vary less than those of the max confidence, min support algorithm. In other words, our experiments indicate that a less-than-perfect choice of K for the adjusted confidence algorithm is likely to perform better than a less-than-perfect choice of θ for the max confidence, min support algorithm.

5. Related Work and Ongoing Work

The usefulness of association rules and their impact on even a wider range of practical applications remains limited due to problems arising from the minimum support threshold. Most prior work relies on this strong requirement; exceptions include works of Li et al. (1999); Koh (2008) and DuMouchel and Pregibon (2001). Some work (Cohen et al., 2001; Wang et al., 2001) aims to find high confidence rules, ignoring the support altogether. Association rules are generally used as an exploratory tool rather than a predictive tool, which is in contrast with our work. On the other hand, it is clearly possible to use the adjusted confidence as an "interestingness" measure for database exploration. Lin et al. (2002) also construct a recommender system using rules, having a minimum confidence threshold and then an adjustable minimum support threshold. Lawrence et al. (2001) provide a recommender system for a grocery store, but the setting differs from ours in that they always recommend items that have never been previously purchased.

Algorithm	No. datasets
Support	1
Confidence	1
K=0.0001	4
K=0.001	5
K=0.01	6
K=0.1	6
K=1	2
K=5	2
K=10	2
K=15	2
θ=1	1
θ=2	1
θ=3	1
θ=5	0
θ=10	1

Figure 7: Summary of experiments: for each algorithm, the number of datasets where it performed comparably with the best algorithm.

In terms of Bayesian analysis, DuMouchel and Pregibon (2001) present a Bayesian approach to the identification of interesting itemsets. While not a rule mining algorithm per se, the approach could be extended to produce rules. Breese et al. (1998) present a number of different algorithms for collaborative filtering, including two Bayesian approaches. One of their Bayesian approaches clusters users while the other constructs a Bayesian network. Condliff et al. (1999) present a hierarchical Bayesian approach to collaborative filtering that "borrows strength" across users. Neither Breese et al. nor Condliff et al. focus on repeated purchases but both present ideas that may have relevance to future versions of our approach.

In current work, we are designing a Bayesian framework that estimates K for the adjusted confidence by "borrowing strength" across both users and items (McCormick et al., 2011). We are also looking at different approaches to the online grocery store problem,

where we allow the predictions to alter the sequence in which items are placed into the basket (Letham et al., 2011).

6. Conclusion

This work synthesizes tools from several fields to analyze association rules in a supervised learning framework. This analysis is necessarily different from that of classical supervised learning analysis; association rules provide two mechanisms for generalization: a large sample, and a minimum support of rules. We considered two simple algorithms, both that create a bound on the support, regulating a tradeoff between accuracy on the training set and generalization ability. We have also demonstrated that the adjusted confidence introduced here has several advantages over the minimum support threshold that is commonly considered in association rule mining.

Acknowledgments

C. Rudin is also at the Center for Computational Learning Systems, Columbia University. This work was performed partly while E. Kogan was at Fresh Direct. We would like to acknowledge support for this project from the National Science Foundation under grant IIS-1053407.

References

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int'l Conf. Very Large Data Bases, (VLDB)*, pages 487–499. Morgan Kaufmann, 1994.

Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD Int'l Conf. on Management of Data*, pages 207–216, 1993.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

John S. Breese, David Heckerman, and Carl Myers Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. Uncertainty in Artificial Intelligence (UAI)*, pages 43–52, 1998.

Edith Cohen, Mayur Datar, Shinji Fujiwara, Aristides Gionis, Piotr Indyk, Rajeev Motwani, Jeffrey D. Ullman, and Cheng Yang. Finding interesting associations without support pruning. *IEEE Trans. Knowl. Data Eng.*, 13(1):64–78, 2001.

Michelle Keim Condliff, David D. Lewis, David Madigan, and Christian Posse. Bayesian mixed-effects models for recommender systems. In *ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

Luc Devroye and T. J. Wagner. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory*, 25(5):601–604, 1979.

William DuMouchel and Daryl Pregibon. Empirical bayes screening for multi-item associations. In *Proc. ACM SIGKDD Int'l Conf. on Knowl. Discovery and Data Mining*, pages 67–76, 2001.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Xiangji Huang, Aijun An, Nick Cercone, and Gary Promhouse. Discovery of interesting association rules from Livelink web log data. In *Proc. IEEE Int'l Conf. on Data Mining (ICDM)*, 2002.

Xiang-Rong Jiang and Le Gruenwald. Microarray gene expression data association rules mining based on BSC-tree and FIS-tree. *Data & Knowl. Eng.*, 53(1):3–29, 2005.

Yun Sing Koh. Mining non-coincidental rules without a user defined support threshold. In *Advances in Knowl. Discovery and Data Mining, 12th Pacific-Asia Conf., (PAKDD)*, pages 910–915, 2008.

Ron Kohavi, Llew Mason, Rajesh Parekh, and Zijian Zheng. Lessons and challenges from mining retail e-commerce data. *Machine Learning*, 57(1-2):83–113, 2004.

R.D. Lawrence, G.S. Almasi, V. Kotlyar, M.S. Viveros, and S.S. Duri. Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, 5(1-2): 11–32, 2001.

Ben Letham, Cynthia Rudin, and David Madigan. A supervised ranking approach to sequential event prediction. In Preparation, 2011.

Jinyan Li, Xiuzhen Zhang, Guozhu Dong, Kotagiri Ramamohanarao, and Qun Sun. Efficient mining of high confidence association rules without support thresholds. In *Proc. Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 406–411, 1999.

Weiyang Lin, Sergio A. Alvarez, and Carolina Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, 6(1): 83–105, 2002.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD)*, 1998.

Tyler McCormick, Cynthia Rudin, and David Madigan. A hierarchical model for association rule mining of sequential events: An approach to automated medical symptom prediction. *SSRN eLibrary*, 2011. URL `http://ssrn.com/paper=1736062`.

W. H. Rogers and T. J. Wagner. A finite sample distribution-free performance bound for local discrimination rules. *Annals of Statistics*, 6(3):506–514, 1978.

Cynthia Rudin, Benjamin Letham, Eugene Kogan, and David Madigan. A learning theory framework for association rules and sequential events. In Preparation, 2011.

Adriano Veloso, Humberto Mossri de Almeida, Marcos André Gonçalves, and Wagner Meira Jr. Learning to rank at query-time using association rules. In *Proc. Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 267–274, 2008.

Ke Wang, Yu He, David W. Cheung, and Francis Y. L. Chin. Mining confident rules without support requirement. In *Proc. Conf. on Information and Knowledge Management (CIKM)*, pages 89–96, 2001.