A

Summarizing Data Succinctly with the Most Informative Itemsets

MICHAEL MAMPAEY, University of Antwerp
JILLES VREEKEN, University of Antwerp
NIKOLAJ TATTI, University of Antwerp

Knowledge discovery from data is an inherently iterative process. That is, what we know about the data greatly determines our expectations, and therefore, what results we would find interesting and/or surprising. Given new knowledge about the data, our expectations will change. Hence, in order to avoid redundant results, knowledge discovery algorithms ideally should follow such an iterative updating procedure.

With this in mind, we introduce a well-founded approach for succinctly summarizing data with the most informative itemsets; using a probabilistic maximum entropy model, we iteratively find the itemset that provides us the most novel information—that is, for which the frequency in the data surprises us the most—and in turn we update our model accordingly. As we use the Maximum Entropy principle to obtain unbiased probabilistic models, and only include those itemsets that are most informative with regard to the current model, the summaries we construct are guaranteed to be both descriptive and non-redundant.

The algorithm that we present, called MTV, can either discover the top-k most informative itemsets, or we can employ either the Bayesian Information Criterion (BIC) or the Minimum Description Length (MDL) principle to automatically identify the set of itemsets that together summarize the data well. In other words, our method will 'tell you what you need to know' about the data. Importantly, it is a one-phase algorithm: rather than picking itemsets from a user-provided candidate set, itemsets and their supports are mined on-the-fly. To further its applicability, we provide an efficient method to compute the maximum entropy distribution using Quick Inclusion-Exclusion.

Experiments on our method, using synthetic, benchmark, and real data, show that the discovered summaries are succinct, and correctly identify the key patterns in the data. The models they form attain high likelihoods, and inspection shows that they summarize the data well with increasingly specific, yet non-redundant itemsets.

Categories and Subject Descriptors: H.2.8 [**Database management**]: Data applications—*Data mining*

General Terms: Theory, Algorithms, Experimentation

Additional Key Words and Phrases: Frequent Itemsets, Pattern Sets, Summarization, Maximum Entropy, Minimum Description Length Principle, MDL, BIC, Inclusion-Exclusion

ACM Reference Format:
M. Mampaey, J. Vreeken, and N. Tatti, 2011. Summarizing Data Succinctly with the Most Informative Itemsets. ACM Trans. Knowl. Discov. Data. V, N, Article A (January 2012), 44 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

Michael Mampaey is supported by a Ph.D. grant of the Agency for Innovation by Science and Technology in Flanders (IWT). Jilles Vreeken is supported by a Post-Doctoral Fellowship of the Research Foundation – Flanders (FWO). Nikolaj Tatti is supported by a Post-Doctoral Fellowship of the Research Foundation – Flanders (FWO).
Authors' address: M. Mampaey, J. Vreeken, and N. Tatti, ADReM Research Group, Department of Mathematics and Computer Science, University of Antwerp, Middelheimlaan 1, 2020 Antwerp, Belgium.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Knowledge discovery from data is an inherently iterative process. That is, what we already know about the data greatly determines our expectations, and therefore, which results we would find interesting and/or surprising. Early on in the process of analyzing a database, for instance, we are happy to learn about the generalities underlying the data, while later on we will be more interested in the specifics that build upon these concepts. Essentially, this process comes down to summarization: we want to know what is interesting in the data, and we want this to be reported as succinctly as possible, without redundancy.

As a simple example, consider supermarket basket analysis. Say we just learned that *pasta* and *tomatoes* are sold together very often, and that we already knew that many people buy *wine*. Then it would not be very interesting to be told that the combination of these three items is also sold frequently; although we might not have been able to predict the sales numbers exactly, our estimate would most likely have come very close, and hence we can say that this pattern is redundant.

At the same time, at this stage of the analysis we are probably also not interested in highly detailed patterns, e.g., an itemset representing the many ingredients of an elaborate Italian dinner. While the frequency of this itemset may be surprising, the pattern is also highly specific, and may well be better explained by some more general patterns. Still, this itemset might be regarded as highly interesting further on in the discovery process, after we have learned those more general patterns, and if this is the case, we would like it to be reported at that time.

In a nutshell, that is the approach we adopt in this paper: we incrementally adjust our model as we iteratively discover informative patterns, in order to obtain a non-redundant summary of the data.

As natural as it may seem to update a knowledge model during the discovery process, and in particular to iteratively find results that are informative with regard to what we have learned so far, few pattern mining techniques actually follow such a dynamic approach. That is, while many methods provide a series of patterns in order of interestingness, most score these patterns using a static model; during this process the model, and hence the itemset scores, are not updated with the knowledge gained from previously discovered patterns. For instance, Tan et al. [2002] and Geng and Hamilton [2006] respectively study 21 and 38 interestingness measures, all of which are static, and most of which are based on the independence model [e.g., Brin et al. 1997; Aggarwal and Yu 1998]. The static approach inherently gives rise to a typical problem of traditional pattern mining: overwhelmingly large and highly redundant pattern sets.

Our objective is to mine succinct summaries of binary data, that is, to obtain a small, yet high-quality set of itemsets that describes key characteristics of the data at hand, in order to gain useful insight. This is motivated by the fact that many existing algorithms often return too large collections of patterns with considerable redundancy, as discussed above. The view that we take in this paper on succinctness and non-redundancy is therefore a fairly strict one.

While we are not the first to propose a method that updates its scoring model dynamically—examples include the swap randomization-based approach by Hanhijärvi et al. [2009] and the compression-based approach by Vreeken et al. [2011]—there are several differences with existing methods. For instance, the former requires generating many randomized databases in order to estimate frequencies, whereas our model is analytical, allowing for direct frequency calculation. The models for the latter are not probabilistic, and while non-redundant with respect to compression, can contain patterns that are variations of the same theme. We treat related work in more detail in Section 2, but let us first discuss the basic features of our approach.

To model the data, we use the powerful and versatile class of maximum entropy models. We construct a maximum entropy distribution that allows us to directly calculate the expected frequencies of itemsets. Then, at each iteration, we return the itemset that provides the most information, i.e., for which our frequency estimate was most off. We update our model with this new knowledge, and continue the process. The non-redundant model that contains the most important information is thus automatically identified. Therefore, we paraphrase our method as 'tell me what I need to know'.

While in general solving the maximum entropy model is infeasible, we show that in our setting it can be solved efficiently—depending on the amount of overlap between the selected patterns. Similarly, we give an efficient method for estimating frequencies from the model. Further, we provide an efficient convex heuristic for effectively pruning the search space when mining the most informative itemsets. This heuristic allows us to mine collections of candidate itemsets on the fly, instead of picking them from a larger candidate collection that has to be materialized beforehand.

Our approach does not require user-defined parameters such as a significance level or an error threshold. In practice, however, we allow the user to specify itemset constraints such as a minsup threshold or a maximum size, which reduces the size of the search space. Such constraints can be integrated quite easily into the algorithm.

We formalize the problem of identifying the most informative model both by the Bayesian Information Criterion (BIC), and by the Minimum Description Length (MDL) principle; both are well-known and well-understood model selection techniques that have natural interpretations. To heuristically approximate these ideal solutions, we introduce the MTV algorithm, for mining the most informative itemsets. Alternatively, by its iterative nature, MTV can also mine the top-k most informative itemsets. Finally, our approach easily allows the user to infuse background knowledge into the model (in the form of itemset frequencies, column margins, and/or row margins), to the end that redundancy with regard to what the user already knows can effectively be avoided.

Experiments on real and synthetic data show that our approach results in succinct, non-redundant data summaries using itemsets, and provide intuitive descriptions of the data. Since they only contain a small number of key patterns about the data, they can easily be inspected manually by the user, and since redundancy is reduced to a minimum, the user knows that every pattern he or she looks at will be informative.

An earlier version of this work appeared as Mampaey et al. [2011]. In this work we significantly extend said paper in the following ways. We define an MDL-based quality measure for a collection of itemsets, which is more expressive and supersedes the BIC score presented earlier. To solve the maximum entropy model, we introduce a new and faster algorithm employing Quick Inclusion-Exclusion to efficiently compute the sizes of the transaction blocks in partitions induced by an itemset collection. Moreover, we formulate how basic background information such as column margins and row margins (i.e., the density of each row and column), can be included into the model, such that results following from this background knowledge will not be regarded as informative, since the background knowledge of the user determines what he or she will find interesting. Further, we provide extensive experimental validation of our methods using fourteen real and synthetic datasets covering a wide range of data characteristics.

The remainder of this paper is organized as follows. First, Section 2 discusses related work. We cover notation and preliminaries in Section 3. Next, in Section 4 we give an introduction to Maximum Entropy models and how to compute them efficiently, and show how to measure the interestingness of a set of itemsets using BIC and MDL. We give a formal problem statement in Section 5. Subsequently, we present the MTV algorithm in Section 6. In Section 7 we report on the experimental evaluation of our method. We round up with a discussion in Section 8 and conclude in Section 9. For reasons of presentation some proofs have been placed in the Appendix.

2. RELATED WORK

Selecting or ranking interesting patterns is a well-studied topic in data mining. Existing techniques can roughly be split into two groups.

2.1. Static Approaches

The first group consists of techniques that measure how *surprising* the support of an itemset is compared against some null hypothesis: the more the observed frequency deviates from the expected value, the more interesting it is. In frequent itemset mining, for instance, one can consider the null hypothesis to be that no itemset occurs more than the minimum support threshold. Similarly, in tile mining [Geerts et al. 2004], one searches for itemsets with a large area (support multiplied by size); here the underlying assumption is that the area of any itemset is small. A simple and often-used probabilistic null hypothesis is the independence model [Brin et al. 1997; Aggarwal and Yu 1998]. More flexible models have been suggested, for example, Bayesian Networks [Jaroszewicz and Simovici 2004]. The major caveat of these approaches is that the null hypothesis is static and hence we keep rediscovering the same information. As a result, this will lead to pattern collections with high levels of redundancy.

Swap randomization was proposed by Gionis et al. [2007] and Hanhijärvi et al. [2009] as a means to assess the significance of data mining results through randomization. To this end, Gionis et al. [2007] gave an algorithm by which randomized data samples can be drawn by repeatedly swapping values locally, such that the background knowledge is maintained—essentially, a Markov chain is defined. Then, by repeatedly sampling such random datasets, one can assess the statistical significance of a result by calculating empirical p-values. While the original proposal only considered row and column margin as background knowledge, Hanhijärvi et al. [2009] extended the approach such that cluster structures and itemset frequencies can be maintained.

While a very elegant approach, swap randomization does suffer from some drawbacks. First of all, there are no theoretical results on the mixing time of the Markov chain, and hence one has to rely on heuristics (e.g., swap as many times as there are ones in the data). Second, as typically very many swaps are required to obtain a randomized sample of the data, and finding suitable swaps is nontrivial, the number of randomized datasets we can realistically obtain is limited, and hence so is the p-value resolution by which we measure the significance of results. As our approach is to model the data probabilistically by the Maximum Entropy principle, we do not suffer from convergence issues, and moreover, as our model is analytical in nature, we can calculate exact probabilities and p-values.

2.2. Dynamic Approaches

The alternative approach to measuring informativeness statically, is to rank and select itemsets using a dynamic hypothesis. That is, when new knowledge arrives, e.g., in the form of an interesting pattern, the model is updated such that we take this newly discovered information into account, and hence we avoid reporting redundant results. The method we present in this paper falls into this category.

Besides extending the possibilities for incorporating background knowledge into a static model, the aforementioned approach by Hanhijärvi et al. [2009] discusses that by iteratively updating the randomization model, redundancy is eliminated naturally.

The MINI algorithm by Gallo et al. [2007] also uses row and column margins to rank itemsets. It first orders all potentially interesting itemsets by computing their p-values according to these margins. Then, as itemsets are added, the p-values are recomputed, and the itemsets re-ordered according to their new p-values. However, this method does not allow querying, and requires a set of candidates to be mined beforehand.

KRIMP, by Siebes et al. [2006]; Vreeken et al. [2011], employs the MDL principle to select those itemsets that together compress the data best. As such, patterns that essentially describe the same part of the data are rejected. The models it finds are not probabilistic, and cannot straightforwardly be used to calculate probabilities (although Vreeken et al. [2007] showed data strongly resembling the original can be sampled from the resulting code tables). Further, while non-redundant from a compression point of view, many of the patterns it selects are variations of the same theme. The reason for this lies in the combination of the covering strategy and encoding KRIMP utilizes. The former prefers long itemsets over short ones, while the second assumes independence between all itemsets in the code table. Hence, for KRIMP it is sometimes cheaper to encode highly specific itemsets with *one* relatively long code, than to encode it with *multiple* slightly shorter codes, which may lead to the selection of overly specific itemsets. Other differences to our method are that KRIMP considers its candidates in a static order, and that it is unclear how to make it consider background knowledge. Recent extensions of KRIMP include the SLIM algorithm by Smets and Vreeken [2012], which finds good code tables directly from data by iteratively heuristically finding the optimal addition to the code table, and the GROEI algorithm by Siebes and Kersten [2011], which identifies the optimal set of k itemsets by beam search instead of identifying the optimal set overall.

2.3. Modelling by Maximum Entropy

The use of maximum entropy models in pattern mining has been proposed by several authors, e.g., [Tatti and Heikinheimo 2008; Tatti 2008; Wang and Parthasarathy 2006; Kontonasios and De Bie 2010; De Bie 2011b]. Discovering itemset collections with good BIC scores was suggested by Tatti and Heikinheimo [2008]. Alternatively, Tatti [2010] samples collections and bases the significance of an itemset on its occurrence in the discovered collections. However, in order to guarantee that the score can be computed, the authors restrict themselves to a particular type of collections: downward closed and decomposable collections of itemsets.

The method of Tatti [2008] uses local models. That is, to compute the support of an itemset X, the method only uses sub-itemsets of X, and outputs a p-value. Unlike our approach, it requires a threshold to determine whether X is important. Relatedly, Webb [2010] defines itemsets as *self-sufficient*, if their support differs significantly from what can be inferred from their sub- and supersets; therefore such a model is also local.

Wang and Parthasarathy [2006] incrementally build a maximum entropy model by adding itemsets that deviate more than a given error threshold. The approach ranks and adds itemsets in level-wise batches, i.e., first itemsets of size 1, then of size 2, and so on. This will, however, not prevent redundancy within batches of itemsets.

De Bie [2011b] proposed an alternative to swap-randomization for obtaining randomized datasets, by modeling the whole data by maximum entropy, using row and column sums as background information; besides faster, and more well-founded, unlike swap-randomization this approach does not suffer from convergence issues. Furthermore, by its analytical nature, exact p-values can be calculated. Kontonasios and De Bie [2010] used the model to analytically define an interestingness measure, Information Ratio, for noisy tiles, by considering both the expected density of a tile, and the complexity of transferring the true tile to the user.

Although both De Bie and ourselves model data by the Maximum Entropy principle, there exist important differences between the two approaches. The most elementary is that while we regard it a *bag of samples* from a distribution, De Bie considers the data as a *monolithic* entity. That is, De Bie models the whole binary matrix, while we construct probabilistic model for individual rows. Informally said, De Bie considers the location of a row in the matrix important, whereas we do not. Both these approaches

have different advantages. While our approach intuitively makes more sense when modeling, say, a supermarket basket dataset, where the data consists of individual, independent samples; the monolithic approach is more suited to model data where the individual rows have meaning, say, for a dataset containing mammal presences for geographic locations. Moreover, while for our models it is straightforward to include itemset frequencies (which does not include specifying transaction identifiers), such as *tomatoes* and *pasta* are sold in 80% of the transactions, this is currently not possible for the whole-dataset model. Additionally, while the De Bie framework in general allows the background knowledge of a user to be false, in this work we only consider background knowledge consistent with the data. As opposed to Kontonasios and De Bie [2010], we do not just rank patterns according to interestingness, but formalize model selection techniques such as BIC or MDL such that we can identify the optimal model, and hence avoid discovering overly complex models.

As overall comments to the methods described above, we note that in contrast to our approach, most of the above methods require the user to set one or several parameters to asses the informativeness of itemsets, e.g., a maximum error threshold or a significance level (although we do allow thresholding the support of candidate itemsets). Many also cannot easily be used to estimate the frequency of an itemset. Further, all of them are two-phase algorithms, i.e., they require that the user first provides a collection of candidate (frequent) itemsets to the algorithm, which must be completely mined and stored first, before the actual algorithm itself is run.

3. PRELIMINARIES AND NOTATION

This section provides some preliminaries and the notation that we will use throughout the rest of this paper.

By a *transaction* we mean a binary vector of size N generated by some unknown distribution. The ith element in a random transaction corresponds to an *attribute* or *item* a_i, a Bernoulli random variable. We denote the set of all items by $A = \{a_1, \ldots, a_N\}$. We denote the set of all possible transactions by $\mathcal{T} = \{0,1\}^N$.

The input of our method is a *binary dataset* D, which is a bag of $|D|$ transactions. Given the data D we define an empirical distribution

$$q_D(a_1 = v_1, \ldots, a_N = v_N) = |\{t \in D \mid t = v\}|/|D| \ .$$

An *itemset* X is a subset of A. For notational convenience, given a distribution p, an itemset $X = \{x_1, \ldots, x_L\}$, and a binary vector v of length L, we often use $p(X = v)$ to denote $p(x_1 = v_1, \ldots, x_L = v_L)$. If v consists entirely of 1's, we use the notation $p(X = 1)$. A transaction t is said to *support* an itemset X if it has 1's for all attributes that X identifies. As such, the support of an itemset X in a database D is defined as

$$\mathit{supp}(X) = |\{t \in D \mid \pi_{x_1}(t) = 1, \ldots, \pi_{x_L}(t) = 1\}| \ ,$$

where $\pi_x(t) \in \{0,1\}$ is the projection of transaction t onto item x. Analogously, the *frequency* of an itemset X in a dataset D is defined as

$$\mathit{fr}(X) = \mathit{supp}(X)/|D| = q_D(X = 1) \ .$$

An *indicator function* $S_X : \mathcal{T} \to \{0,1\}$ of an itemset X maps a transaction t to a binary value such that $S_X(t) = 1$ if and only if t supports X.

The *entropy* of a distribution p over $\mathcal{T}$ is defined as

$$H(p) = -\sum_{t \in \mathcal{T}} p(A = t) \log p(A = t) \ ,$$

where the base of the logarithm is 2, and by convention $0 \log 0 = 0$. The entropy of p is the expected number of bits needed to optimally encode a transaction t.

Finally, the *Kullback-Leibler divergence* [Cover and Thomas 2006] between two distributions p and q over $\mathcal{T}$ is defined as

$$KL(p \,\|\, q) = \sum_{t \in \mathcal{T}} p(A = t) \log \frac{p(A = t)}{q(A = t)} \ .$$

Intuitively, the KL divergence between two distributions p and q is the average number of *extra* bits required to encode data generated by p using a coding optimal for q. Since p defines the optimal coding distribution for this data, on average, it will always cost extra bits when we encode data generated by p using a coding distribution $q \neq p$, and hence $KL(p \,\|\, q)$ is non-negative. The KL divergence equals 0 if and only if p equals q.

4. IDENTIFYING THE BEST SUMMARY

Our goal is to discover the collection $\mathcal{C}$ of itemsets and frequencies that is the most informative for a dataset D, while being succinct and as little redundant as possible. Here, by informative we mean whether we are able to reliably describe, or predict, the data using these itemsets and their frequencies.

By non-redundancy we mean that, in terms of frequency, every element of $\mathcal{C}$ provides significant information for describing the data that cannot be inferred from the rest of the itemset frequencies in $\mathcal{C}$. This is equivalent to requiring that the frequency of an itemset $X \in \mathcal{C}$ should be surprising with respect to $\mathcal{C} \setminus X$. In other words, we do not want $\mathcal{C}$ to be unnecessarily complex as a collection, or capture spurious information, since we only want it to contain itemsets that we really need.

Informally, assume that we have a quality score $s(\mathcal{C}, D)$ which measures the quality of an itemset collection $\mathcal{C}$ with respect to D. Then our aim is to find that $\mathcal{C}$ with the highest score $s(\mathcal{C}, D)$. Analogously, if we only want to know k itemsets, we aim to find the collection $\mathcal{C}$ of size at most k, with the highest score $s(\mathcal{C}, D)$.

Next, we will detail how we define our models, how we define this score, provide theoretical evidence why it is a good choice, and discuss how to compute it efficiently.

EXAMPLE 4.1. *As a running example, assume that we have a binary dataset D with eight items, a to h. Furthermore, consider the set of itemsets $\mathcal{C} = \{abc, cd, def\}$ with frequencies 0.5, 0.4 and 0.8, respectively. Assume for the moment that based on $\mathcal{C}$, our method predicts that the frequency of the itemset agh is 0.19. Now, if we observe in the data that $fr(agh) = 0.18$, then we can safely say that agh is redundant with regard to what we already know, as it does not contribute a lot of novel information, and the slight deviation from the expected value may even be coincidental. On the other hand, if $fr(agh) = 0.7$, then the frequency of agh is surprising with respect to $\mathcal{C}$, and hence adding it to $\mathcal{C}$ would strongly increase the amount of information $\mathcal{C}$ gives us about the data; in other words $\mathcal{C} \cup \{agh\}$ provides a substantially improved description of D.*

4.1. Maximum Entropy Model

In our approach we make use of maximum entropy models. This is a class of probabilistic models that are identified by the Maximum Entropy principle [Csiszár 1975; Jaynes 1982]. This principle states that the best probabilistic model is the model that makes optimal use of the provided information, and that is fully unbiased (i.e., fully random, or, maximally entropic) otherwise. This property makes these models very suited for identifying informative patterns: by using maximum entropy models to measure the quality of a set of patterns, we know that our measurement only relies on the information we provide it, and that it will not be thrown off due to some spurious structure in the data. These models have a number of theoretically appealing properties, which we will discuss after a formal introduction.

Assume that we are given a collection of itemsets and corresponding frequencies

$$\langle \mathcal{C}, \Phi \rangle = \langle \{X_1, \ldots, X_k\}, \{f_1, \ldots, f_k\} \rangle \ , \tag{1}$$

where $X_i \subseteq A$ and $f_i \in [0,1]$, for $i = 1, \ldots, k$. Note that we do not require that the frequencies f_i of the itemsets are equal to the frequencies $fr(X_i)$ in the data. If this does hold, we will call $\langle \mathcal{C}, \Phi \rangle$ *consistent with* D. For notational convenience, we will at times omit writing the frequencies Φ, and simply use $\mathcal{C} = \{X_1, \ldots, X_k\}$, especially when it is clear what the corresponding frequencies are. Now, we consider those distributions over the set of all transactions $\mathcal{T}$ that satisfy the constraints imposed by $\langle \mathcal{C}, \Phi \rangle$. That is, we consider the following set of distributions

$$\mathcal{P}_{\langle \mathcal{C}, \Phi \rangle} = \{p \mid p(X_i = 1) = f_i, \text{ for } i = 1, \ldots, k\} \ . \tag{2}$$

In the case that $\mathcal{P}_{\langle \mathcal{C}, \Phi \rangle}$ is empty, we call $\langle \mathcal{C}, \Phi \rangle$ *inconsistent*. Among these distributions we are interested in only one, namely the unique distribution maximizing the entropy

$$p^*_{\langle \mathcal{C}, \Phi \rangle} = \arg\max_{p \in \mathcal{P}_{\langle \mathcal{C}, \Phi \rangle}} H(p) \ .$$

Again, for notational convenience we will often simply write $p^*_{\mathcal{C}}$, or even omit $\langle \mathcal{C}, \Phi \rangle$ altogether, especially when they are clear from the context.

The following famous theorem states that the maximum entropy model has an exponential form. This form will help us to discover the model and will be useful to compute the quality score of a model.

THEOREM 4.2 (THEOREM 3.1 IN [CSISZÁR 1975]). *Given a collection of itemsets and frequencies $\langle \mathcal{C}, \Phi \rangle = \langle \{X_1, \ldots, X_k\}, \{f_1, \ldots, f_k\} \rangle$, let $\mathcal{P}_{\langle \mathcal{C}, \Phi \rangle}$ be the set of distributions as defined in Eq. 2. If there is a distribution in $\mathcal{P}_{\langle \mathcal{C}, \Phi \rangle}$ that has only nonzero entries, then the maximum entropy distribution $p^*_{\langle \mathcal{C}, \Phi \rangle}$ can be written as*

$$p^*_{\langle \mathcal{C}, \Phi \rangle}(A = t) = u_0 \prod_{X \in \mathcal{C}} u_X^{S_X(t)} \ , \tag{3}$$

where $u_X \in \mathbb{R}$, and u_0 is a normalization factor such that $p^*_{\langle \mathcal{C}, \Phi \rangle}$ is a proper distribution.

Note that the normalization factor u_0 can be thought of as corresponding to the constraint that the empty itemset $\emptyset$ should have a frequency $f_\emptyset = 1$. Theorem 4.2 has a technical requirement that $\mathcal{P}$ needs to contain a distribution with nonzero entries. The easiest way to achieve this is to apply a Bayesian shift [Gelman et al. 2004] by redefining the frequencies $f'_i = (1 - \epsilon) f_i + \epsilon 2^{-|X_i|}$ for some small $\epsilon > 0$. This way the frequencies remain approximately the same, but no transaction has zero probability.

4.2. Identifying the Best Model

Here we describe how we can quantify the goodness of a pattern collection.

A natural first choice would be to directly measure the goodness of fit, using the log-likelihood of the maximum entropy model, that is,

$$\log p^*_{\langle \mathcal{C}, \Phi \rangle}(D) = \log \prod_{t \in D} p^*_{\langle \mathcal{C}, \Phi \rangle}(A = t) = \sum_{t \in D} \log p^*_{\langle \mathcal{C}, \Phi \rangle}(A = t) \ .$$

Note that if $\langle \mathcal{C}, \Phi \rangle$ is inconsistent, p^* does not exist. In this case we define the likelihood to be zero, and hence the log-likelihood to be $-\infty$.

The following corollary shows that for exponential models, we can easily calculate the log-likelihood.

COROLLARY 4.3 (OF THEOREM 4.2). *The log-likelihood of the maximum entropy distribution $p^*_{\langle \mathcal{C},\Phi\rangle}$ for a collection of itemsets and frequencies $\langle \mathcal{C},\Phi\rangle$ is equal to*

$$\log p^*_{\langle \mathcal{C},\Phi\rangle}(D) = |D|\left(\log u_0 + \sum_{(X_i,f_i)\in\langle \mathcal{C},\Phi\rangle} f_i \log u_{X_i}\right) .$$

PROOF. See the Appendix. □

Thus, to calculate the log-likelihood of a collection $\langle \mathcal{C},\Phi\rangle$, it suffices to compute the parameters u_X and u_0 of the corresponding distribution $p^*_{\langle \mathcal{C},\Phi\rangle}$.

The following theorem states that if we are searching for collections with a high likelihood, we can restrict ourselves to collections that are consistent with the data.

THEOREM 4.4. *For a fixed collection of itemsets $\mathcal{C} = \{X_1,\ldots,X_k\}$, the likelihood $p^*_{\langle \mathcal{C},\Phi\rangle}(D)$ is maximized if and only if $\langle \mathcal{C},\Phi\rangle$ is consistent with D, that is, $f_i = fr(X_i)$ for all $i = 1\ldots,k$.*

For our goal, maximum entropy models are theoretically superior over any other model. Let us discuss why. Let D_1 and D_2 be two datasets such that $fr(X \mid D_1) = fr(X \mid D_2)$ for any $X \in \mathcal{C}$. Let p^*_1 and p^*_2 be the corresponding maximum entropy models, then, by definition, $p^*_1 = p^*_2$. In other words, the model depends only the support of the chosen itemsets. This is a natural requirement, since we wish to measure the quality of the statistics in $\mathcal{C}$ and nothing else. Similarly, Corollary 4.3 implies that $p^*_1(D_2) = p^*_1(D_1)$. This is also a natural property because otherwise, the score would be depending on some statistic not included in $\mathcal{C}$. Informally said, the scores are equal if we cannot distinguish between D_1 and D_2 using the information $\mathcal{C}$ provides us. The next theorem states that among all such models, the maximum entropy model has the best likelihood, in other words, the maximum entropy model uses the available information as efficiently as possible.

THEOREM 4.5. *Assume a collection of itemsets $\mathcal{C} = \{X_1,\ldots,X_k\}$ and let $p^*_{\mathcal{C}}$ be the maximum entropy model, computed from a given dataset D. Assume also an alternative model $r(A = t \mid f_1,\ldots,f_k)$, where $f_i = fr(X_i \mid D)$, that is, a statistical model parametrized by the frequencies of $\mathcal{C}$. Assume that for any two datasets D_1 and D_2, where $fr(X \mid D_1) = fr(X \mid D_2)$ for any $X \in \mathcal{C}$, it holds that*

$$1/|D_1| \log r(D_1 \mid f_1,\ldots,f_k) = 1/|D_2| \log r(D_2 \mid f_1,\ldots,f_k) .$$

Then $p^*_{\mathcal{C}}(D) \geq r(D)$ for any dataset D.

PROOF. See the Appendix. □

Using only log-likelihood to evaluate a model, however, suffers from overfitting: larger collections of itemsets will always provide more information, hence allow for better estimates, and therefore have a better log-likelihood. Consequently, we need to prevent our method from overfitting. In order to do so, we will explore the Bayesian Information Criterion (BIC), and the Minimum Description Length (MDL) principle—both of which are well-known and well-founded model selection techniques. We start by discussing BIC, which is the least strict, and least involved of the two.

The Bayesian Information Criterion (BIC) measures the quality of a model by taking both its log-likelihood, and the number of parameters of said model into account. It favors models that fit the data well using few parameters; in our setting, the number of parameters of the model p^* corresponds exactly to the number of itemsets k. It has a strong theoretical support in Bayesian model selection [Schwarz 1978].

Definition 4.6. Given a collection of itemsets $\langle \mathcal{C}, \Phi \rangle = \langle \{X_1, \ldots, X_k\}, \{f_1, \ldots, f_k\} \rangle$, the BIC score with respect to a dataset D is defined as

$$\text{BIC}(\langle \mathcal{C}, \Phi \rangle, D) = -\log p^*_{\langle \mathcal{C}, \Phi \rangle}(D) + k/2 \log |D| \ . \tag{4}$$

The better a model fits the data, the higher its likelihood. On the other hand, the more parameters the model has—i.e., the more complex it is—the higher its penalty. Therefore, it is be possible that a model that fits the data slightly worse, but contains few parameters, is favored by BIC over a model that fits the data better, but is also more complex. From Corollary 4.3 we see that the first term of the BIC score is equal to $|D|H(p^*_{\langle \mathcal{C}, \Phi \rangle})$. Hence, the likelihood term grows faster than the penalty term with respect to the size of the D. As such, the more data (or evidence) we have, the more complicated the model is allowed to be.

COROLLARY 4.7 (OF THEOREM 4.4). *For a fixed collection of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$, the* BIC *score* $\text{BIC}(\langle \mathcal{C}, \Phi \rangle, D)$ *is minimized if and only if $\langle \mathcal{C}, \Phi \rangle$ is consistent with D, that is, $f_i = fr(X_i)$ for all i.*

PROOF. Follows directly from Theorem 4.4 and the fact that the BIC penalty term, $k/2 \log |D|$, does not depend on the frequencies f_i. □

While the BIC score helps to avoid overfitting, it is somewhat simplistic. That is, it only incorporates the *number* of itemsets to penalize a summary, and not their complexity. As stated in the introduction, if possible, we would typically rather be given some number of general patterns than the same number of highly involved patterns. MDL provides us a means to define a score that also takes into account the complexity of the itemsets in $\mathcal{C}$.

The Minimum Description Length (MDL) principle [Rissanen 1978; Grünwald 2005], like its close cousin MML (Minimum Message Length) [Wallace 2005], is a practical version of Kolmogorov Complexity [Li and Vitányi 1993]. All three embrace the slogan *Induction by Compression*. The MDL principle can be roughly described as follows.

Given a dataset D and a set of models $\mathcal{M}$ for D, the best model $M \in \mathcal{M}$ is the one that minimizes

$$L(M) + L(D \mid M)$$

in which

— $L(M)$ is the length, in bits, of the description of the model M, and
— $L(D \mid M)$ is the length, in bits, of the description of the data, encoded with M.

This is called two-part MDL, or *crude* MDL. This stands opposed to *refined* MDL, where model and data are encoded together [Grünwald 2007]. We use two-part MDL because we are specifically interested in the model: the set of itemsets that yields the best description length. Further, although refined MDL has stronger theoretical foundations, it cannot be computed except for some special cases. We should also point out that refined MDL is asymptotically equivalent to BIC if the number of transactions goes to infinity and the number of free parameters stays fixed. However, for moderate numbers of transactions there may be significant differences. Generally speaking, MDL tends to be more conservative than BIC [Grünwald 2007].

To use MDL, we have to define what our set of models $\mathcal{M}$ is, how a model M describes a database, and how all of this is encoded in bits. Intuitively, we want to favor itemset collections that are small, i.e., collections which can describe the data well, using few itemsets. At the same time, we also prefer collections with small itemsets over collections with large ones.

Definition 4.8. Given a collection of itemsets $\langle \mathcal{C}, \Phi \rangle = \langle \{X_1, \ldots, X_k\}, \{f_1, \ldots, f_k\} \rangle$, let $x = \sum_{i=1}^{k} |X_i|$. We define the MDL score of $\langle \mathcal{C}, \Phi \rangle$ with respect to the dataset D as

$$\text{MDL}(\langle \mathcal{C}, \Phi \rangle, D) = L(D \mid \langle \mathcal{C}, \Phi \rangle) + L(\langle \mathcal{C}, \Phi \rangle) \ , \tag{5}$$

where

$$L(D \mid \langle \mathcal{C}, \Phi \rangle) = -\log p^*_{\langle \mathcal{C}, \Phi \rangle}(D) \quad \text{and} \quad L(\langle \mathcal{C}, \Phi \rangle) = l_1 k + l_2 x + 1 \ ,$$

with

$$l_1 = \log |D| + N \log(1 + N^{-1}) + 1 \approx \log |D| + \log e + 1$$

and

$$l_2 = \log N \ .$$

Whenever D is clear form the context, we simply write $\text{MDL}(\langle \mathcal{C}, \Phi \rangle)$.

The first term is simply the negative log-likelihood of the model, which corresponds to the description length of the data given the maximum entropy model induced by $\mathcal{C}$. The second part is a penalty term, which corresponds to the description length of the model. It is a linear function of $k = |\mathcal{C}|$ and $x = \sum_i |X_i|$, of which the coefficients depend on N and $|D|$. How it is derived is explained further below. The smaller this score, the better the model. Given two collections with an equal amount of itemsets, the one containing fewer items is penalized less; conversely, if they have the same total number of items, the one that contains those items in fewer itemsets is favored. Consequently, the best model is identified as the model that provides a good balance between high likelihood and low complexity. Moreover, we automatically avoid redundancy, since models with redundant itemsets are penalized for being too complex, without sufficiently improving the likelihood.

With this quality score we evaluate collections of itemsets, rather than the (maximum entropy) distributions we construct from them. The reason for this is that we want to summarize the data with a succinct set of itemsets, not model it with a distribution. A single distribution, after all, may be described by many different collections of itemsets, simple or complex. Further, we assume that the set $\mathcal{M}$ of models consists of collections of itemsets which are represented as vectors, rather than as sets. This choice keeps the quality score function computationally simple and intuitive, and is not disadvantageous: if $\mathcal{C}$ contains duplicates, they simply increase the penalty term. Additionally, we impose no restrictions on the consistency of $\langle \mathcal{C}, \Phi \rangle$, that is, there are collections $\langle \mathcal{C}, \Phi \rangle$ for which $\mathcal{P}_{\langle \mathcal{C}, \Phi \rangle}$ is empty, and hence the maximum entropy distribution does not exist. As mentioned above, in this case we define the likelihood to be zero, and hence the description length is infinite.

We now describe the derivation of the penalty term, which equals

$$k \log |D| + x \log N + (k + 1) + kN \log(1 + N^{-1}) \ .$$

To describe an itemset we encode a support using $\log |D|$ bits and the actual items in the itemsets using $\log N$ bits. This gives us the first two terms. We use the third term to express whether there are more itemsets, one bit after each itemset[1]. and one extra bit to accommodate the case $\mathcal{C} = \emptyset$. The term $kN \log(1 + N^{-1})$ is a normalization factor, to ensure that the encoding is *optimal* for the prior distribution over all pattern collections. That is, the encoding corresponds to a distribution

$$\mathit{prior}(\langle \mathcal{C}, \Phi \rangle) = 2^{-l_1 k - l_2 x - 1} \ ,$$

[1]Although it is intuitive, and it provides good results in practice, using 1 bit to signal the end of an itemset is slightly inefficient; it could be further optimized by the decision tree encoding of Wallace and Patrick [1993].

which assigns high probability to simple summaries, and low probability to complex ones. The following equation shows that the above encoding is optimal for this prior.

$$\begin{aligned}\sum_{\langle\mathcal{C},\Phi\rangle} prior(\langle\mathcal{C},\Phi\rangle) &= \sum_{k=0}^{\infty}\sum_{x=0}^{kN}\binom{kN}{x}|D|^k 2^{-l_1 k - l_2 x - 1} \\ &= \sum_{k=0}^{\infty} 2^{-k-1}2^{-kN\log(1+N^{-1})}2^{-k\log|D|}|D|^k \sum_{x=0}^{kN}\binom{kN}{x}N^{-x} \\ &= \sum_{k=0}^{\infty} 2^{-k-1}2^{-kN\log(1+N^{-1})}(1+N^{-1})^{kN} \\ &= \sum_{k=0}^{\infty} 2^{-k-1} = 1\end{aligned}$$

The following corollary shows that for identifying the MDL optimal model, it suffices to only consider summaries that are consistent with the data.

COROLLARY 4.9 (OF THEOREM 4.4). *For a fixed collection of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$, the* MDL *score* $\text{MDL}(\langle\mathcal{C},\Phi\rangle, D)$ *is maximized if and only if $\langle\mathcal{C},\Phi\rangle$ is consistent with D, that is, $f_i = fr(X_i)$ for all i.*

PROOF. Follows directly from Theorem 4.4 and the fact that for a fixed $\mathcal{C}$, the frequencies f_i are encoded with a constant length $\log|D|$, and hence the penalty term is always the same. □

Therefore, in the rest of this paper we will assume that $\langle\mathcal{C},\Phi\rangle$ is always consistent with D, and hence we will omit Φ from notation.

4.3. Reducing Redundancy

Here we show that our score favors collections of itemsets that exhibit low redundancy, and make a theoretical link with some popular lossless redundancy reduction techniques from the pattern mining literature. Informally, we define redundancy as anything that does not deviate (much) from our expectation, or in other words is unsurprising given the information that we already have. The results below hold for MDL as well as for BIC, and hence we write s to denote either score.

A baseline technique for ranking itemsets is to compare the observed frequency against the expected value of some null hypothesis. The next theorem shows that if the observed frequency of an itemset X agrees with the expected value $p^*(X = 1)$, then X is redundant.

THEOREM 4.10. *Let $\mathcal{C}$ be a collection of itemsets and let p^* be the corresponding maximum entropy model. Let $X \notin \mathcal{C}$ be an itemset such that $fr(X) = p^*(X = 1)$. Then $s(\mathcal{C} \cup \{X\}, D) > s(\mathcal{C}, D)$.*

PROOF. We will prove the theorem by showing that the likelihood terms for both collections are equal. Define the collection $\mathcal{C}_1 = \mathcal{C} \cup \{X\}$ and let $\mathcal{P}_1$ be the corresponding set of distributions. Let p_1^* be the distribution maximizing the entropy in $\mathcal{P}_1$. Note that since $\mathcal{C} \subset \mathcal{C}_1$, we have $\mathcal{P}_1 \subseteq \mathcal{P}$ and hence $H(p_1^*) \leq H(p^*)$. On the other hand, the assumption in the theorem implies that $p^* \in \mathcal{P}_1$ and so $H(p^*) \leq H(p_1^*)$. Thus, $H(p^*) = H(p_1^*)$ and since the distribution maximizing the entropy is unique, we have $p^* = p_1^*$. This shows that the likelihood terms in $s(\mathcal{C}, D)$ and $s(\mathcal{C}_1, D)$ are equal. The penalty term in the latter is larger, which concludes the proof. □

Theorem 4.10 states that adding an itemset X to $\mathcal{C}$ improves the score only if its observed frequency deviates from the expected value. The amount of deviation required to lower the score, is determined by the penalty term. This gives us a convenient advantage over methods that are based solely on deviation, since they require a user-specified threshold.

Two useful corollaries follow from Theorem 4.10, which connect our approach to well-known techniques for removing redundancy from pattern set collections—so-called *condensed representations*. The first corollary relates our approach to *closed* itemsets [Pasquier et al. 1999], and *generator* itemsets (also known as *free* itemsets [Boulicaut et al. 2003]). An itemset is closed if all of its proper supersets have a strictly lower support. An itemset is called a generator if all of its proper subsets have a strictly higher support.

COROLLARY 4.11 (OF THEOREM 4.10). *Let $\mathcal{C}$ be a collection of itemsets. Assume that $X, Y \in \mathcal{C}$ such that $X \subset Y$ and $fr(X) = fr(Y) \neq 0$. Assume that $Z \notin \mathcal{C}$ such that $X \subset Z \subset Y$. Then $s(\mathcal{C} \cup \{Z\}, D) > s(\mathcal{C}, D)$.*

PROOF. Let $p \in \mathcal{P}$, as defined in Eq. 2. We have that $p(X = 1) = fr(X) = fr(Y) = p(Y = 1)$. Hence we must have $p(Z = 1) = fr(Z)$. Since $p^* \in \mathcal{P}$, it must hold that $p^*(Z = 1) = fr(Z)$. The result follows from Theorem 4.10. □

Corollary 4.11 implies that if the closure and a generator of an itemset Z are already in the collection, then adding Z will worsen the score. The second corollary provides a similar relation with *non-derivable* itemsets [Calders and Goethals 2007]. An itemset is called derivable if its support can be inferred exactly given the supports of all of its proper subsets.

COROLLARY 4.12 (OF THEOREM 4.10). *Let $\mathcal{C}$ be a collection of itemsets. Assume that $X \notin \mathcal{C}$ is a derivable itemset and all proper sub-itemsets of X are included in $\mathcal{C}$. Then $s(\mathcal{C} \cup \{X\}, D) > s(\mathcal{C}, D)$.*

PROOF. The proof of this corollary is similar to the proof of Corollary 4.11. □

An advantage of our method is that it can avoid redundancy in a very general way. The closed and non-derivable itemsets are two types of lossless representations, whereas our method additionally can give us lossy redundancy removal. For example, in the context of Corollary 4.11, we can choose to reject X from $\mathcal{C}$ even if $fr(X)$ does not equal $fr(Y)$ exactly, and as such we can prune redundancy more aggressively.

4.4. Efficiently Computing the Maximum Entropy Model

Computing the maximum entropy model comes down to finding the u_X and u_0 parameters from Theorem 4.2. To achieve this, we use the well-known Iterative Scaling procedure by Darroch and Ratcliff [1972], which is given here as Algorithm 1. Simply put, it iteratively updates the parameters of an exponential distribution, until it converges to the maximum entropy distribution p^* which satisfies a given set of constraints—itemset frequencies in our case. The distribution is initialized with the uniform distribution, which is done by setting the u_X parameters to 1, and $u_0 = 2^{-N}$ to properly normalize the distribution. Then, for each itemset $X \in \mathcal{C}$, we adjust the corresponding parameter u_X to enforce $p(X = 1) = fr(X)$ (line 5,6). This process is repeated in a round robin fashion until p converges, and it can be shown (see Darroch and Ratcliff [1972]) that p always converges to the maximum entropy distribution p^*. Typically the number of iterations required for convergence is low (usually $<$10 in our experiments).

ALGORITHM 1: ITERATIVESCALING($\mathcal{C}$)

input : itemset collection $\mathcal{C} = \{X_1, \ldots, X_k\}$, frequencies $fr(X_1), \ldots, fr(X_k)$
output : parameters u_X and u_0 of the maximum entropy distribution $p^*_\mathcal{C}$ satisfying $p^*_\mathcal{C}(X_i) = fr(X_i)$ for all i

```
1 initialize p
2 repeat
3     for each X in C do
4         compute p(X = 1)
5         u_X ← u_X · fr(X)/p(X=1) · (1−p(X=1))/(1−fr(X))
6         u_0 ← u_0 · (1−fr(X))/(1−p(X=1))
7     end
8 until p converges
9 return p
```

EXAMPLE 4.13. *In our running example, with $\mathcal{C} = \{abc, cd, def\}$, the maximum entropy model has three parameters u_1, u_2, u_3, and a normalization factor u_0. Initially we set $u_1 = u_2 = u_3 = 1$ and $u_0 = 2^{-N} = 2^{-8}$. Then, we iteratively loop over the itemsets and scale the parameters. For instance, for the first itemset abc with frequency 0.5, we first compute its current estimate to be $2^{-3} = 0.125$. Thus, we update the first parameter $u_1 = 1 \cdot (0.5/2^{-3}) \cdot ((1 - 2^{-3})/0.5) = 7$. The normalization factor becomes $u_0 = 2^{-8} \cdot 0.5/(1 - 2^{-3}) \approx 2.2 \cdot 10^{-3}$. Next, we do the same for cd, and so on. After a few iterations, the model parameters converge to $u_1 = 28.5$, $u_2 = 0.12$, $u_3 = 85.4$ and $u_0 = 3 \cdot 10^{-4}$.*

Straightforward as this procedure may be, the greatest computational bottleneck is the inference of the probability of an itemset on line 4 of the algorithm,

$$p(X = 1) = \sum_{\substack{t \in \mathcal{T} \\ S_X(t)=1}} p(A = t) \; . \tag{6}$$

Since this sum ranges over all possible transactions supporting X, it is infeasible to calculate by brute force, even for a moderate number N items. In fact, it has been shown that querying the maximum entropy model is **PP**-hard in general [Tatti 2006].

Therefore, in order to be able to query the model efficiently, we introduce a partitioning scheme, which makes use of the observation that many transactions have the same probability in the maximum entropy distribution. Remark that an itemset collection $\mathcal{C}$ partitions $\mathcal{T}$ into blocks of transactions which support the same set of itemsets. That is, two transactions t_1 and t_2 belong to the same block T if and only if $S_X(t_1) = S_X(t_2)$ for all X in $\mathcal{C}$. Therefore, we know that $p(A = t_1) = p(A = t_2)$ if p is of the form in Eq. 3. This property allows us to define $S_X(T) = S_X(t)$ for any $t \in T$ and $X \in \mathcal{C}$. We denote the partition of $\mathcal{T}$ induced by $\mathcal{C}$ as $\mathcal{T}_\mathcal{C}$. Then the probability of an itemset is

$$p(X = 1) = \sum_{\substack{T \in \mathcal{T}_\mathcal{C} \\ S_X(T)=1}} p(A \in T) \; . \tag{7}$$

The sum in Eq. 6 has been reduced to a sum over blocks of transactions, and the inference problem has been moved from the transaction space $\mathcal{T}$ to the block space $\mathcal{T}_\mathcal{C}$. In our setting we will see that $|\mathcal{T}_\mathcal{C}| \ll |\mathcal{T}|$, which makes inference a lot more feasible. In the worst case, this partition may contain up to $2^{|\mathcal{C}|}$ blocks, however, through the interplay of the itemsets, it can be as low as $|\mathcal{C}| + 1$. As explained below, we can exploit, or even choose to limit, the structure of $\mathcal{C}$, such that practical computation is guaranteed.

ALGORITHM 2: COMPUTEBLOCKSIZES($\mathcal{C}$)

input : itemset collection $\mathcal{C} = \{X_1, \ldots, X_k\}$
output : block sizes $e(T)$ for each T in $\mathcal{T}_\mathcal{C}$

for T in $\mathcal{T}_\mathcal{C}$ **do**
  $I \leftarrow \bigcup\{X \mid X \in sets(T;\mathcal{C})\}$
  $c(T) \leftarrow 2^{N-|I|}$
end
sort the blocks in $\mathcal{T}_\mathcal{C}$
for T_i in $\mathcal{T}_\mathcal{C}$ **do**
  $e(T_i) \leftarrow c(T_i)$
  for T_j in $\mathcal{T}_\mathcal{C}$, with $j < i$ **do**
    if $T_i \subset T_j$ **then**
      $e(T_i) \leftarrow e(T_i) - e(T_j)$
    end
  end
end
return $\mathcal{T}_\mathcal{C}$

All we must do now is obtain the block probabilities $p(A \in T)$. Since all transactions t in a block T have the same probability

$$p(A = t) = u_0 \prod_{X \in \mathcal{C}} u_X^{S_X(t)} \ ,$$

it suffices to compute the number of transactions in T to get $p(A \in T)$. So, let us define $e(T)$ to be the number of transactions in T, then

$$p(A \in T) = \sum_{t \in T} p(A = t) = e(T)\, u_0 \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \ .$$

Algorithm 2 describes COMPUTEBLOCKSIZES, as given in [Mampaey et al. 2011]. In order to compute the block sizes $e(T)$, we introduce a partial order on $\mathcal{T}_\mathcal{C}$. Let

$$sets(T;\mathcal{C}) = \{X \in \mathcal{C} \mid S_X(T) = 1\}$$

be the itemsets of $\mathcal{C}$ that occur in the transactions of T. Note that every block corresponds to a unique subset of $\mathcal{C}$; conversely a subset of $\mathcal{C}$ either corresponds to an empty block of transactions, or to a unique nonempty transaction block. We can now define the partial order on $\mathcal{T}_\mathcal{C}$ as follows,

$$T_1 \subseteq T_2 \quad \text{if and only if} \quad sets(T_1;\mathcal{C}) \subseteq sets(T_2;\mathcal{C}) \ .$$

In order to compute the size $e(T)$ of a block, we start from its *cumulative* size,

$$c(T) = \sum_{T' \supseteq T} e(T') \ , \tag{8}$$

which is the number of transactions that contain *at least* all the itemsets in $sets(T;\mathcal{C})$. For a given block T, let $I = \bigcup\{X \mid X \in sets(T;\mathcal{C})\}$. That is, I are the items that occur in all transactions of T. Then it holds that $c(T) = 2^{N-|I|}$, where N is the total number of items. To obtain the block sizes $e(T)$ from the cumulative sizes $c(T)$, we use the Inclusion-Exclusion principle. To that end, the blocks are topologically sorted such that if $T_2 \subset T_1$, then T_1 occurs before T_2. The algorithm then reversely iterates over the blocks in a double loop, subtracting block sizes, using the identity

$$e(T) = c(T) - \sum_{T' \supsetneq T} e(T') \ . \tag{9}$$

Table I: Transaction blocks for the running example above, with $X_1 = abc$, $X_2 = cd$, and $X_3 = def$.

X_1	X_2	X_3	$c(T)$	$e(T)$	$p(A = t)$
1	1	1	4	4	$u_0u_1u_2u_3$
1	1	0	16	12	$u_0u_1u_2$
1	0	0	32	16	u_0u_1
0	1	1	16	12	$u_0u_2u_3$
0	1	0	64	36	u_0u_2
0	0	1	32	16	u_0u_3
0	0	0	256	160	u_0

EXAMPLE 4.14. *Assume again that we have a dataset with eight items (a to h), and an itemset collection containing three itemsets $\mathcal{C} = \{abc, cd, def\}$ with frequencies 0.5, 0.4 and 0.8, respectively.*

Table I shows the sizes of the transaction blocks. Note that while there are 256 transactions in $\mathcal{T}$, there are only 7 blocks in $\mathcal{T}_\mathcal{C}$, whose sizes and probabilities are to be computed (the eighth combination 'abc and def but not cd' is clearly impossible).

Let us compute the size of the first three blocks. For the first block, $I = abcdef$ and therefore $c(T) = 4$, for the second block $I = abcd$, and for the third block $I = abc$. Since the first block is the maximum with respect to the order $\subseteq$, its cumulative size is simply its size, so $e(T) = 4$. For the second block, we subtract the first block, and obtain $e(T) = 16 - 4 = 12$. From the third block we subtract the first two blocks, and we have $e(T) = 32 - 12 - 4 = 16$. Now, to compute, say, $p(abc = 1)$, we simply need the sizes of the blocks containing abc, and the current model parameters,

$$p(abc = 1) = 4(u_0u_1u_2u_3) + 12(u_0u_1u_2) + 16(u_0u_1) \ .$$

Since the algorithm performs a double loop over all transaction blocks, the complexity of COMPUTEBLOCKSIZES equals $O(|\mathcal{T}_\mathcal{C}|^2)$. Note that topologically sorting the blocks (line 5) takes $O(|\mathcal{T}_\mathcal{C}| \log |\mathcal{T}_\mathcal{C}|) \leq O(|\mathcal{T}_\mathcal{C}|k)$, however, we can also simply ensure that the blocks are topologically sorted by construction.

In this work, we substantially improve upon the COMPUTEBLOCKSIZES algorithm, by using a generalized version of the Quick Inclusion-Exclusion (QIE) algorithm, introduced by Calders and Goethals [2006]. The new algorithm presented here, called QIEBLOCKSIZES, has a lower complexity than COMPUTEBLOCKSIZES. The idea behind Quick Inclusion-Exclusion is to reuse intermediate results to reduce the number of subtractions. The standard QIE algorithm computes the supports of all generalized itemsets based on some given itemset of size k (a generalized itemset is an itemset containing both positive and negative items, e.g., $a\bar{b}$ means a and not b), using the supports of all of its (positive) subsets. For instance, from the supports of ab, a, b, and the empty set, we can infer the support of $a\bar{b}$: $supp(a\bar{b}) = supp(\emptyset) - supp(a) - supp(b) + supp(ab)$. QIE therefore works on an array of size 2^k, which allows an implementation of the algorithm to employ efficient array indexing using integers, and makes it easy to locate subsets using bit operations on the indices—where a positive item is represented by a 1 and a negative item by a 0.

In our setting, we want to find the sizes of transaction blocks which correspond to subsets of $\mathcal{C}$, starting from the cumulative sizes of said blocks. We can represent each block T by a binary vector defined by the indicator functions S_X. However, an important difference with the QIE algorithm is that not every possible binary vector necessarily corresponds to a (nonempty) transaction block, i.e., it is possible that $|\mathcal{T}_\mathcal{C}| < 2^k$.

Clearly, if $|\mathcal{T}_\mathcal{C}| \ll 2^k$, it would be inefficient to use an array of size 2^k. Therefore, we must take this fact into account. Before we can discuss the algorithm itself, we first need to introduce the following definitions.

Definition 4.15. Given a collection of itemsets $\mathcal{C} = \langle X_1, \ldots, X_k \rangle$ and an integer $j \in \{0, \ldots, k\}$, the *j-prefix* of $\mathcal{C}$ is defined as

$$\mathcal{C}_j = \{X_1, \ldots, X_j\} .$$

For a subcollection $\mathcal{G}$ of $\mathcal{C}$, we define the $closure : 2^\mathcal{C} \to 2^\mathcal{C}$ as

$$closure(\mathcal{G}) = \{X_i \in \mathcal{C} \mid X_i \subseteq \bigcup_{X \in \mathcal{G}} X\} .$$

The *j-closure* of $\mathcal{G}$ is defined as

$$\begin{aligned} closure(\mathcal{G}, j) &= \mathcal{G} \cup \{X_i \in \mathcal{C} \mid X_i \notin \mathcal{C}_j \text{ and } X_i \subseteq \bigcup_{X \in \mathcal{G}} X\} \\ &= \mathcal{G} \cup (closure(\mathcal{G}) \setminus \mathcal{C}_j) \end{aligned}$$

The following lemma states that there is a one-to-one mapping between the closed subsets $\mathcal{G}$ of $\mathcal{C}$, and the transaction blocks of $\mathcal{T}_\mathcal{C}$.

LEMMA 4.16. *Let $\mathcal{G}$ be an itemset collection. Then $\mathcal{G} = closure(\mathcal{G})$ if and only if there exists a block T in $\mathcal{T}_\mathcal{C}$ such that $\mathcal{G} = sets(T; \mathcal{C})$.*

PROOF. Assume that $\mathcal{G} = closure(\mathcal{G})$. Let $U = \bigcup_{X \in \mathcal{G}} X$ and let $t \in \mathcal{T}$ be such that $t_i = 1$, if $a_i \in U$, and $t_i = 0$ otherwise. Let $T \in \mathcal{T}_\mathcal{C}$ be the block containing t. If $X \in \mathcal{G}$, then $S_X(t) = 1$. On the other hand, if $S_X(t) = 1$, then $X \subseteq U$ and consequently $X \in closure(\mathcal{G}) = \mathcal{G}$. Hence, $\mathcal{G} = sets(T; \mathcal{C})$.

Assume now that there is a T such that $\mathcal{G} = sets(T; \mathcal{C})$, let $t \in T$ and $U = \bigcup_{X \in \mathcal{G}} X$. It follows that $S_U(t) = 1$. Let $X \in closure(\mathcal{G})$, then $X \subseteq U$ and $S_X(t) = 1$. Hence, $X \in sets(T; \mathcal{C}) = \mathcal{G}$. Since $\mathcal{G} \subseteq closure(\mathcal{G})$, the lemma follows. □

Using the above lemma, we can introduce the following function, which maps subsets of $\mathcal{C}$ to their corresponding blocks.

Definition 4.17. For a subset $\mathcal{G}$ of a collection of itemsets $\mathcal{C}$, we define

$$block(\mathcal{G}) = \begin{cases} T \in \mathcal{T}_\mathcal{C} \text{ s.t. } sets(T; \mathcal{C}) = \mathcal{G} & \text{if } closure(\mathcal{G}) = \mathcal{G} , \\ \emptyset & \text{otherwise} . \end{cases}$$

That is, if $\mathcal{G}$ is closed, the $block$ function simply maps it to the corresponding block in $\mathcal{T}_\mathcal{C}$. If $\mathcal{G}$ is not closed, it is mapped to the empty transaction block. Note that $block(sets(T; \mathcal{C})) = T$ for all $T \in \mathcal{T}_\mathcal{C}$.

QIEBLOCKSIZES is given as Algorithm 3. As before, we first compute the cumulative size of every block T in $\mathcal{T}_\mathcal{C}$ (line 3). Then, for each itemset X_i (line 5), the algorithm subtracts from each block T for which $X_i \notin \mathcal{G} = sets(T; \mathcal{C})$, the current size of the block T' corresponding to $\mathcal{G}' = closure(\mathcal{G} \cup \{X_i\}, i-1)$ if T' exists in $\mathcal{T}_\mathcal{C}$, i.e., the size of $T' = block(\mathcal{G}')$.

The following theorem states that QIEBLOCKSIZES correctly computes the sizes of all blocks of transactions in $\mathcal{T}_\mathcal{C}$. For the proof, please refer to the Appendix.

THEOREM 4.18. *Given a collection of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$, let $\mathcal{T}_\mathcal{C}$ be the corresponding partition with respect to $\mathcal{C}$. The algorithm* QIEBLOCKSIZES *correctly computes the block sizes $e(T)$ for $T \in \mathcal{T}_\mathcal{C}$.*

ALGORITHM 3: QIEBLOCKSIZES($\mathcal{C}$)

input : itemset collection $\mathcal{C} = \{X_1, \ldots, X_k\}$
output : block sizes $e(T)$ for each T in $\mathcal{T}_\mathcal{C}$

```
for T in T_C do
    I ← ⋃{X | X ∈ sets(T; C)}
    c(T) ← 2^{N−|I|}
end
for i = 1, ..., k do
    foreach T in T_C do
        G ← sets(T; C)
        if X_i ∉ G then
            G' ← closure(G ∪ {X_i}, i − 1)
            T' ← block(G')
            if T' ≠ ∅ then
                e(T) ← e(T) − e(T')
            end
        end
    end
end
```

EXAMPLE 4.19. *Let us apply* QIEBLOCKSIZES *to our running example. Recall that* $\mathcal{C} = \{abc, cd, def\}$ *(see Table I). For brevity, we restrict ourselves to the first three blocks. In step 1, the first three blocks remain unaffected, since they all contain* X_1*. In step 2, only the third block does not contain* X_2*; we subtract the second block from it, to obtain* $32 - 16 = 16$*. In step 3, we subtract the first block from the second block, and get* $16 - 4 = 12$*. From the third block we do not have to subtract anything, since the* 3*-closure of the corresponding block does not appear in* $\mathcal{T}_\mathcal{C}$*. We have thus calculated the sizes of the first three blocks using two subtractions, rather than three, as was previously required in Example 4.14.*

Finally, we can significantly optimize the algorithm as follows. Assume that we can divide $\mathcal{C}$ into two disjoint groups $\mathcal{C}_1$ and $\mathcal{C}_2$, such that if $X_1 \in \mathcal{C}_1$ and $X_2 \in \mathcal{C}_2$, then $X_1 \cap X_2 = \emptyset$. Let $B = \bigcup \mathcal{C}_1$ be the set of items occurring in $\mathcal{C}_1$. Theorem 4.2 implies that $p^*(A) = p^*(B)p^*(A \setminus B)$. In other words, the maximum entropy distribution can be factorized into two *independent* distributions, namely $p^*(B)$ and $p^*(A \setminus B)$, and more importantly, the factor $p^*(B)$ depends only on $\mathcal{C}_1$. Consequently, if we wish to compute the probability $p^*(X = 1)$ such that $X \subset B$, we can ignore all variables outside B and all itemsets outside $\mathcal{C}_1$. The number of computations to be performed by QIEBLOCKSIZES can now be greatly reduced, since in the case of independence it holds that $|\mathcal{T}_\mathcal{C}| = |\mathcal{T}_{\mathcal{C}_1}| \times |\mathcal{T}_{\mathcal{C}_2}|$, and we can simply compute the block sizes for $\mathcal{T}_{\mathcal{C}_1}$ and $\mathcal{T}_{\mathcal{C}_2}$ separately. Naturally, this decomposition can also be applied when there are more than two disjoint groups of itemsets.

Moreover, in order to *guarantee* that we can apply the above separation, we could reduce the solution space slightly by imposing a limit on the number of items (or itemsets) per group, such that the number of blocks for each group remains small. Alternatively, we could first partition the items of the dataset into smaller, approximately independent groups, and subsequently apply the algorithm for each group separately—for which the approach of Mampaey and Vreeken [2010] of identifying the optimal partitioning by MDL would be a logical choice.

4.5. Querying the Model

We have seen how we can efficiently query the probability of an itemset $X \in \mathcal{C}$ when given the maximum entropy distribution $p^*_{\mathcal{C}}$. In order to compute the probability of an arbitrary itemset Y that is not a member of $\mathcal{C}$, we do the following. We first set $\mathcal{C}' = \mathcal{C} \cup \{Y\}$ and compute the block probabilities $e(T')$ for all T' in $\mathcal{T}_{\mathcal{C}'}$ by calling QIEBLOCKSIZES. Then, we can simply use the parameters of $p^*_{\mathcal{C}}$ to compute $p^*_{\mathcal{C}}(Y = 1)$ as follows,

$$p^*_{\mathcal{C}}(Y = 1) = \sum_{\substack{T \in \mathcal{T}_{\mathcal{C}'} \\ S_Y(T)=1}} e(T) \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \ .$$

Thus, to obtain the probability of an itemset, it suffices to compute the block probabilities in $\mathcal{T}_{\mathcal{C}'}$, for which we know that $|\mathcal{T}_{\mathcal{C}'}| \leq 2|\mathcal{T}_{\mathcal{C}}|$.

4.6. Computational Complexity

Let us analyze the complexity of the ITERATIVESCALING algorithm. To this end, we define $nb(\mathcal{C}) = |\mathcal{T}_{\mathcal{C}}|$ as the number of blocks in $\mathcal{T}_{\mathcal{C}}$. The computational complexity of QIEBLOCKSIZES is

$$O\left(k \cdot nb(\mathcal{C}) \log nb(\mathcal{C})\right) \ ,$$

for a given collection $\mathcal{C}$, with $|\mathcal{C}| = k$. The logarithmic factor comes from looking for the block T' on line 11. Note that $nb(\mathcal{C}) \leq 2^k$, and hence $\log nb(\mathcal{C}) \leq k$. Assume now that we can partition $\mathcal{C}$ into L disjoint parts $\mathcal{C} = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_L$, such that if $X \in \mathcal{C}_i$ and $Y \in \mathcal{C}_j$ then $X \cap Y = \emptyset$. As mentioned in Section 4.4, we can now simply compute L independent distributions at a lower total cost. Denoting $B_i = \bigcup_{X \in \mathcal{C}_i} X$, it holds that $nb(\mathcal{C}_i) \leq \min\left(2^{|\mathcal{C}_i|}, 2^{|B_i|}\right)$. If $\mathcal{C}_i$ cannot be partitioned further, this usually means that either $|\mathcal{C}_i|$ is small, or the itemsets in $\mathcal{C}_i$ overlap a lot and $nb(\mathcal{C}_i) \ll 2^{|\mathcal{C}_i|}$. The total execution time of ITERATIVESCALING is

$$O\left(K \sum_{i=1}^{L} |C_i| nb(\mathcal{C}_i) \log nb(\mathcal{C}_i)\right) \ ,$$

where K is the number of iterations, which is usually low. The complexity of estimating the frequency of an itemset requires running QIEBLOCKSIZES once and, hence equals

$$O\left(\sum_{i=1}^{L} |C_i| nb(\mathcal{C}_i) \log nb(\mathcal{C}_i)\right) \ .$$

4.7. Including Background Knowledge into the Model

Typically, when analyzing data we have some basic background knowledge about the data. For instance, we may already know the individual frequencies of the items, i.e., the *column margins*. These margins supply some basic information about the data, for instance whether *tomatoes* are sold often or not in a supermarket database. These individual frequencies are intuitive and easy to calculate, yet already provide information on whether some combinations of items are more or less likely to occur frequently. For this reason, many existing techniques use the independence model as a basis to discover interesting patterns, [e.g., Brin et al. 1997; Aggarwal and Yu 1998]. Another form of background information that is often used are the *row margins* of the data, that is, the probabilities that a transaction contains a certain number of items, e.g., [Gionis et al. 2007; Hanhijärvi et al. 2009; Kontonasios and De Bie 2010; Tatti and Mampaey 2010]. If we know that most transactions are rather small, large itemsets are likely to have low frequencies.

Clearly, when we analyze data we want to incorporate this background knowledge, since otherwise we would simply rediscover it. If we do include it in our analysis, we discover itemsets that are interesting *with respect to* what we already know. Therefore, we extend the BIC and MDL quality scores of Definition 4.8 to incorporate background knowledge, say, $\mathcal{B}$. Although in this section we focus on row and column margins as forms of background knowledge, many other patterns or count statistics that can be expressed as linear constraints on transactions could be used, for instance, a set of association rules and their confidences. Therefore, we intentionally omit the specification of $\mathcal{B}$.

Definition 4.20. Given a dataset D and some background knowledge $\mathcal{B}$, we define the BIC score for a collection of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$, with respect to $\mathcal{B}$ as

$$\text{BIC}(\mathcal{C}, D; \mathcal{B}) = -\log p^*_{\mathcal{B},\mathcal{C}}(D) + k/2 \log |D| \ . \tag{10}$$

Similarly, we define the MDL score of $\mathcal{C}$ with respect to $\mathcal{B}$ as

$$\text{MDL}(\mathcal{C}, D; \mathcal{B}) = -\log p^*_{\mathcal{B},\mathcal{C}}(D) + l_1 k + l_2 x + 1 \ , \tag{11}$$

where $p^*_{\mathcal{B},\mathcal{C}}$ is the maximum entropy distribution satisfying the background knowledge $\mathcal{B}$ and $p^*_{\mathcal{B},\mathcal{C}}(X = 1) = fr(X)$ for all $X \in \mathcal{C}$, and l_1 and l_2 are the same as in Definition 4.8.

Note that while the background knowledge is included in the log-likelihood term of $s(\mathcal{C}, D; \mathcal{B})$ (where s denotes either BIC or MDL), it is not included in the penalty term. We choose not to do so because we will assume that our background knowledge is consistent with the data and invariable. We could alternatively define

$$s(\mathcal{C}, D, \mathcal{B}) = s(\mathcal{C}, D; \mathcal{B}) + L(\mathcal{B}) \ ,$$

where $L(\mathcal{B})$ is some term which penalizes $\mathcal{B}$, however, since this term would be equal for all $\mathcal{C}$, for simplicity it might as well be omitted. If we were to compare different models that use different background knowledge, though, it must be included.

In this section, we show how to include row and column margins as background knowledge into our algorithm without, however, blowing up its computational complexity. If we were, for instance, to naively add all singleton itemsets $\mathcal{I}$ and their frequencies to an itemset collection $\mathcal{C}$, the number of transaction blocks in the corresponding partition would become $|\mathcal{T}_{\mathcal{C}\cup\mathcal{I}}| = |\mathcal{T}| = 2^N$, by which we would be back at square one. Therefore, we will consider the row and column margins separately from $\mathcal{C}$ in our computations.

First, let us consider using only column margins, viz., item frequencies. With these, we build an independence model, while with $\mathcal{C}$ we partition the transactions $\mathcal{T}$ as above; then we simply combine the two to obtain the maximum entropy distribution. As before, the maximum entropy model has an exponential form:

$$p^*_{\mathcal{C}'}(A = t) = u_0 \prod_{X \in \mathcal{C}} u_X^{S_X(t)} \prod_{i \in \mathcal{I}} v_i^{S_i(t)} \ . \tag{12}$$

The second part of the product defines an independence distribution

$$v(A = t) = v_0 \prod_i v_i^{S_i(t)} \ , \tag{13}$$

where v_0 is a normalization factor. It is not difficult to see that $v(a_i = 1) = v_i/(1 + v_i)$, for all for $a_i \in A$. It should be noted that while $p^*(a_i = 1) = fr(a_i)$, in general it does not necessarily hold that $v(a_i = 1) = fr(a_i)$. Now we can write

$$p^*_{\mathcal{C}'}(A \in T) = v(A \in T) \frac{u_0}{v_0} \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \ . \tag{14}$$

Thus, we simply need to compute $v(A \in T)$ for each block T, which is computed very similarly to $e(T)$, using QIEBLOCKSIZES. Note that $e(A = t)$ is in fact nothing more than a uniform distribution over $\mathcal{T}$, multiplied by 2^N. To compute $v(A \in T)$, we simply initiate the algorithm with the cumulative sizes of the blocks with respect to v, which are equal to

$$c(A \in T) = v_0 \prod_{i \in I} v_i \ ,$$

where $I = \bigcup sets(T; \mathcal{C})$. Hence, we can include the item frequencies at a negligible additional cost. To update the v_i parameters, we must query the probability of a single item. We can achieve this by simply adding the corresponding singleton to $\mathcal{C}$, in exactly the same way as described in Section 4.5.

Next, we also include row margins in the background information. Let us define the indicator functions $S^j(t) : \mathcal{T} \to \{0, 1\}$ for $j \in \{0, \ldots, N\}$ such that $S^j(t) = 1$ if and only if the number of ones in t, denoted as $|t|$, is equal to j. Further, for any distribution p on A, let us write $p(|A| = j)$ to indicate the probability that a transaction contains j items. Again, the maximum entropy distribution has an exponential form,

$$p^*_{\mathcal{B},\mathcal{C}}(A = t) = u_0 \prod_{X \in \mathcal{C}} u_X^{S_X(t)} \prod_{i \in \mathcal{I}} v_i^{S_i(t)} \prod_{j=0}^{N} w_j^{S^j(t)} \ . \tag{15}$$

The row and column margins define a distribution

$$w(A = t) = v_0 \prod_{i \in \mathcal{I}} v_i^{S_i(t)} \prod_{j=0}^{N} w_j^{S^j(t)} \ , \tag{16}$$

where v_0 is a normalization factor. Now, for the probabilities $p^*(A \in T)$, we have

$$\begin{aligned} p^*(A \in T, |A| = j) &= w(A \in T, |A| = j) \frac{u_0}{v_0} \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \\ &= w_j v(A \in T, |A| = j) \frac{u_0}{v_0} \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \end{aligned}$$

for $j = 0, \ldots, N$, and we marginalize over j to obtain

$$\begin{aligned} p^*(A \in T) &= \sum_{j=0}^{N} p^*(A \in T, |A| = j) \\ &= \frac{u_0}{v_0} \prod_{X \in \mathcal{C}} u_X^{S_X(T)} \sum_{j=0}^{N} w_j v(A \in T, |A| = j) \end{aligned}$$

As above, we compute the probabilities $v(A \in T, |A| = j)$ using QIEBLOCKSIZES. Let $I = \bigcup \{X \mid X \in sets(\mathcal{C}; T)\}$, then the corresponding cumulative probability becomes

$$c(A \in T, |A| = j) = v_0 \prod_{i \in I} v_i \cdot v(|A| = j \mid I = 1)$$

Computing the probabilities $v(|A| = j)$, and similarly $v(|A| = j \mid I = 1)$, can be done from scratch in $O(N^2)$ time and $O(N)$ space, using the following recurrence relation,

$$v(|A_i| = j) = v(a_i) \cdot v(|A_{i-1}| = j - 1) + (1 - v(a_i)) \cdot v(|A_{i-1}| = j) \ ,$$

where $A_i = \{a_1, \ldots, a_i\}$. Starting from $A_0 = \emptyset$, the COMPUTESIZEPROBABILITIES algorithm adds each item a_i until we have computed all probabilities $v(|A_N| = j)$ where

$A_N = A$; see Algorithm 4. The time complexity can be reduced to $O(N)$, by applying the updates that ITERATIVESCALING performs on $v(a_i)$, to the probabilities $v(|A| = j)$ as well, this is done by UPDATESIZEPROBABILITIES in Algorithm 5. The algorithm first removes the item, and then re-adds it with the updates probability. For further details we refer to Tatti and Mampaey [2010]. Computing item frequencies is done by adding singletons to $\mathcal{C}$. To compute the row margin probabilities $p^*(|A| = j)$, we simply marginalize over $\mathcal{T}_\mathcal{C}$,

$$p^*(|A| = j) = \sum_{T \in \mathcal{T}_\mathcal{C}} p^*(A \in T, |A| = j) \ .$$

Hence, including the row margins increases the time and space complexity of model computation and inference by a factor of N.

ALGORITHM 4: COMPUTESIZEPROBABILITIES(v)

input : independence distribution v over A, with probabilities $v_i = v(a_i = 1)$ for $i = 1, \ldots, N$
output : probabilities $g_j = v(|A| = j)$ for $j = 0, \ldots, N$

$g_0 \leftarrow 1$
for $j = 1, \ldots, N$ **do**
  $g_j \leftarrow 0$
end
for $i = 1, \ldots, N$ **do**
  for $j = i, \ldots, 1$ **do**
    $g_j \leftarrow v_i \cdot g_{j-1} + (1 - v_i) \cdot g_j$
  end
  $g_0 \leftarrow (1 - v_i) \cdot g_0$
end
return g

ALGORITHM 5: UPDATESIZEPROBABILITIES(v, g, a_i, x)

input : probabilities $g_j = v(|A| = j)$ for independence distribution v, with parameters $v_i = v(a_i = 1)$, updated probability v'_i for item a_i
output : updated probabilities $g_j = v(|A| = j)$ for $j = 0, \ldots, N$

$g_0 \leftarrow g_0/(1 - v_i)$
for $j = 1, \ldots, N$ **do**
  $g_j \leftarrow (g_j - v_i g_{j-1})/(1 - v_i)$
end
update v such that $v(a_i = 1) = v'_i$
for $j = N, \ldots, 1$ **do**
  $g_j \leftarrow v'_i \cdot g_{j-1} + (1 - v'_i) \cdot g_j$
end
$g_0 \leftarrow (1 - v'_i) \cdot g_0$
return g

5. PROBLEM STATEMENTS

In this section we identify four different problems that we intend to solve using the theory introduced above. We assume some given set $\mathcal{B}$ that represents our background knowledge, e.g., the individual item frequencies, some arbitrary collection of itemsets, or simply the empty set. We start simple, with a size constraint k and a collection $\mathcal{F}$ of

potentially interesting itemsets to choose from, for instance, frequent itemsets, closed itemsets, itemsets of a certain size, etc.

PROBLEM 1 (MOST INFORMATIVE k-SUBSET). *Given a dataset D, a set $\mathcal{B}$ that represents our background knowledge, an integer k, and a collection of potentially interesting itemsets $\mathcal{F}$, find the subset $\mathcal{C} \subseteq \mathcal{F}$ with $|\mathcal{C}| \leq k$ such that $s(\mathcal{C}, D; \mathcal{B})$ is minimal.*

Note that if we choose $k = 1$, this problem reduces to 'Find the Most Interesting Itemset in $\mathcal{F}$', which means simply scoring $\mathcal{C} = \{X\}$ with respect to $\mathcal{B}$ for each set $X \in \mathcal{F}$, and selecting the best one. Further, these scores provide a ranking of the itemsets $X \in \mathcal{F}$ with regard to what we already know, that is, $\mathcal{B}$.

Now, if we do not want to set k ourselves, we can rely on either BIC or MDL to identify the best-fitting, least-redundant model, a problem we state as the following.

PROBLEM 2 (MOST INFORMATIVE SUBSET). *Given a dataset D, a set $\mathcal{B}$ that represents our background knowledge, and a collection of potentially interesting itemsets $\mathcal{F}$, find the subset $\mathcal{C} \subseteq \mathcal{F}$ such that $s(\mathcal{C}, D; \mathcal{B})$ is minimal.*

When we do not want to constrain ourselves to a particular itemset collection $\mathcal{F}$, we simply use all itemsets. Problem 1 then generalizes to the following.

PROBLEM 3 (k MOST INFORMATIVE ITEMSETS). *Given a dataset D, an integer k, and a set $\mathcal{B}$ that represents our background knowledge, find the collection of itemsets $\mathcal{C}$, with $|\mathcal{C}| \leq k$, such that $s(\mathcal{C}, D; \mathcal{B})$ is minimal.*

Similarly, and most generally, we can simply consider finding the best collection of itemsets altogether.

PROBLEM 4 (MOST INFORMATIVE ITEMSETS). *Given a dataset D and a set $\mathcal{B}$ that represents our background knowledge, find the collection of itemsets $\mathcal{C}$ such that $s(\mathcal{C}, D; \mathcal{B})$ is minimal.*

Note that these problem statements do not require $\mathcal{F}$ to be explicitly available beforehand (let alone the complete set of itemsets), i.e., it does not have to be mined or materialized in advance (we postpone the details of this to Section 6.2).

Next, we discuss how we can efficiently mine sets of itemsets to solve the above problems.

6. MINING INFORMATIVE SUCCINCT SUMMARIES

In Section 4 we described how to compute the maximum entropy model and its BIC or MDL quality score *given* a set of itemsets. Finding the *optimal* collection as stated in Section 5, however, is clearly nontrivial. The size of the search space is

$$\sum_{j=0}^{k} \binom{|\mathcal{F}|}{j} \leq 2^{|\mathcal{F}|} .$$

If we do not restrict the candidate itemsets, then the number of all (non-singleton) itemsets is $|\mathcal{F}| = 2^N - N - 1$. Moreover, our quality scores are not monotonic, nor is there to our knowledge some easily exploitable structure, which prevents us from straightforwardly exploring the search space.

Therefore, we resort to using a heuristic, greedy approach. Starting with a set of background knowledge—for instance the column margins—we incrementally construct our summary by iteratively adding the itemset that reduces the score function the most. The algorithm stops either when k interesting itemsets are found, or when the score no longer decreases. The pseudo-code for our MTV algorithm, which mines Maximally informaTiVe itemset summaries, is given in Algorithm 6.

ALGORITHM 6: MTV(D, $\mathcal{B}$, k)

input : binary dataset D, background knowledge $\mathcal{B}$, integer $k \leq \infty$
output : itemset collection $\mathcal{C}$

1 $\mathcal{I} \leftarrow$ items in D
2 **while** $s(\mathcal{C}, D; \mathcal{B})$ decreases **and** $|\mathcal{C}| < k$ **do**
3 $\quad X \leftarrow$ FINDMOSTINFORMATIVEITEMSET($\emptyset, \mathcal{I}, \emptyset$)
4 $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \{X\}$
5 $\quad p^*_{\mathcal{B},\mathcal{C}} \leftarrow$ ITERATIVESCALING($\mathcal{C}$)
6 $\quad$ compute $s(\mathcal{C}, D; \mathcal{B})$
7 **end**
8 **return** $\mathcal{C}$

Due to its incremental nature, we note that we can apply an optimization to the algorithm. When we call ITERATIVESCALING on line 5, rather than computing p^* from scratch, we can initialize the algorithm with the parameters of the previous p^* (line 1 of Algorithm 1), instead of with the uniform distribution. In doing so, the ITERATIVESCALING procedure converges faster. Further, we can also reuse part of the computations from QIEBLOCKSIZES.

6.1. A Heuristic for Scoring Itemsets

Finding the most informative itemset to add to the current collection is practically infeasible, since it involves solving the maximum entropy model for each and every candidate. This remains infeasible even if we restrict the search space (for example, using only frequent itemsets). Therefore, instead of selecting the candidate that optimizes the BIC or MDL score directly, we select the candidate that maximizes a heuristic which expresses the divergence between its frequency and its estimate. To derive and motivate this heuristic we first present the following theorem.

THEOREM 6.1. *Given an itemset collection $\mathcal{C}$, a dataset D, and a candidate collection of itemsets $\mathcal{F}$. Let s denote either* BIC *or* MDL*. It holds that*

$$\begin{aligned}\arg\min_{X \in \mathcal{F}} s(\mathcal{C} \cup \{X\}) &= \arg\max_{X \in \mathcal{F}} KL\Big(p^*_{\mathcal{C} \cup \{X\}} \,\|\, p^*_{\mathcal{C}}\Big) - r(X) \\ &= \arg\min_{X \in \mathcal{F}} KL\Big(q_D \,\|\, p^*_{\mathcal{C} \cup \{X\}}\Big) + r(X)\end{aligned}$$

where

$$r(X) = \begin{cases} 0 & \text{if} \quad s = \text{BIC} \\ |X| \log N / |D| & \text{if} \quad s = \text{MDL} \end{cases}$$

PROOF. Let us write $\mathcal{C}' = \mathcal{C} \cup \{X\}$. Corollary 4.3 states that $-\log p^*_{\mathcal{C}'}(D) = |D| H(p^*_{\mathcal{C}'})$. In addition, we can show with a straightforward calculation that

$$KL(p^*_{\mathcal{C}'} \,\|\, p^*_{\mathcal{C}}) = H(p^*_{\mathcal{C}}) - H(p^*_{\mathcal{C}'}) \ .$$

For BIC the difference in the penalty terms of $s(\mathcal{C})$ and $s(\mathcal{C}')$ is equal to $1/2 \log |D|$, which is identical for all itemsets X, and hence may be eliminated from the $\arg\max$. For MDL, the difference in penalty terms can similarly be reduced to $|X| \log N$. The second equality follows similarly. □

Thus, we search for the itemset X for which the new distribution diverges maximally from the previous one, or equivalently, brings us as close to the empirical distribution as possible—taking into account the penalty term for MDL. Note that for BIC, since $r(X) = 0$, the algorithm simply tries to maximize the likelihood of the model, and

the penalty term functions as a stopping criterion; the algorithm terminates when the increase in likelihood (i.e., decrease of the negative log-likelihood) is not sufficient to counter the increase of the penalty. When we use MDL, on the other hand, $r(X)$ represents a part of the penalty term, and hence this guides the algorithm in its search.

The heuristic we employ uses an approximation of the above KL divergence, and is in fact a simpler KL divergence itself. In the expression

$$KL(p^*_{\mathcal{C}'} \,\|\, p^*_{\mathcal{C}}) = \sum_{t \in \mathcal{T}} p^*_{\mathcal{C}'}(A = t) \log \frac{p^*_{\mathcal{C}'}(A = t)}{p^*_{\mathcal{C}}(A = t)} \tag{17}$$

we merge the terms containing X into one term, and the terms not containing X into another term. To differentiate between these two divergences, let us write the function $kl : [0, 1] \times [0, 1] \to \mathbb{R}^+$ as follows,

$$kl(x, y) = x \log \frac{x}{y} + (1 - x) \log \frac{1 - x}{1 - y} \,, \tag{18}$$

then we approximate Eq. 17 by $kl\left(fr(X), p^*_{\mathcal{C}}(X = 1)\right)$. We will write the latter simply as $kl(X)$ when fr and p^* are clear from the context. To compute this heuristic, we only need the frequency of X, and its estimate according to the current p^* distribution. This gives us a measure of the divergence between $fr(X)$ and $p^*_{\mathcal{C}}(X = 1)$, i.e., its surprisingness given the current model.

The following theorem shows the relation between KL and kl.

THEOREM 6.2. *For an itemset collection $\mathcal{C}$ and an itemset X, it holds that*

$$0 \leq kl(X) \leq KL\Big(p^*_{\mathcal{C} \cup \{X\}} \,\|\, p^*_{\mathcal{C}}\Big) \,.$$

Moreover, $kl(X) = 0$ if and only if $KL\Big(p^*_{\mathcal{C} \cup \{X\}} \,\|\, p^*_{\mathcal{C}}\Big) = 0$, i.e., when $fr(X) = p^*_{\mathcal{C}}(X = 1)$.

PROOF. Both inequalities follow directly from the log-sum inequality, which states that for any nonnegative numbers a_i, b_i, with $a = \sum_i a_i$ and $b = \sum_i b_i$, it holds that

$$\sum_i a_i \log \frac{a_i}{b_i} \geq a \log \frac{a}{b} \,.$$

For equality to zero, we have $kl(fr(X), p^*_{\mathcal{C}}(X = 1)) = 0$ if and only if $fr(X) = p^*_{\mathcal{C}}(X = 1)$. In this case it holds that $p^*_{\mathcal{C}'} = p^*_{\mathcal{C}}$ which is true if and only if $KL(p^*_{\mathcal{C}'} \,\|\, p^*_{\mathcal{C}}) = 0$. □

Using Theorem 6.1, the heuristic function we employ is defined as

$$h(X) = kl(fr(X), p^*_{\mathcal{C}}(X = 1)) - r(X) \tag{19}$$

and we will make use of the following assumption:

$$\arg\min_{X \in \mathcal{F}} s(\mathcal{C} \cup \{X\}) = \arg\max_{X \in \mathcal{F}} h(X) \,.$$

Note that h has an elegant interpretation: it is equal to the Kullback-Leibler divergence after exactly one step in the ITERATIVESCALING algorithm—when initializing p^* with the parameters from the previous model, as discussed above. Since the KL divergence increases monotonically during the Iterative Scaling procedure, if the total divergence is large, then we expect to already see this in the first step of the procedure.

6.2. Searching for the Most Informative Itemset

To find the itemset maximizing $h(X)$, we take a depth-first branch-and-bound approach. We exploit the fact that kl is a convex function, and employ the bound introduced by Nijssen et al. [2009] to prune large parts of the search space as follows.

ALGORITHM 7: FINDMOSTINFORMATIVEITEMSET(X, Y, Z)

input : itemset X, remaining items Y, currently best set Z
output : itemset between X and XY maximizing h, or Z

```
1  compute fr(X) and p*(X)
2  if h(X) = kl(fr(X), p*(X)) − r(X) > h(Z) then
3  |  Z ← X
4  end
5  compute fr(XY) and p*(XY)
6  b ← max{kl(fr(X), p*(XY)), kl(fr(XY), p*(X))} − r(X)
7  if b > h(Z) then
8  |  for y ∈ Y do
9  |  |  Y ← Y \ {y}
10 |  |  Z ← FindMostInformativeItemset(X ∪ {y}, Y, Z)
11 |  end
12 end
13 return Z
```

Say that for a candidate itemset X in the search space, its maximal possible extension in the branch below it is $X \cup Y$ (denoted XY), then for any itemset W such that $X \subseteq W \subseteq XY$ it holds that

$$h(W) = kl(W) - r(W) \leq \max\Big\{ kl\big(fr(X), p^*(XY)\big), kl\big(fr(XY), p^*(X)\big) \Big\} - r(X)\,. \quad (20)$$

If this upper bound is lower than the best value of the heuristic seen so far, we know that no (local) extension W of X can ever become the best itemset with respect to the heuristic, and therefore we can safely prune the branch of the search space below X. The FINDMOSTINFORMATIVEITEMSET algorithm is given in Algorithm 7.

An advantage of this approach is that we do not need to collect the frequencies of all candidate itemsets beforehand. Instead, we just compute them on the fly when we need them (line 1). For instance, if we wish to pick itemsets from a collection $\mathcal{F}$ of frequent itemsets for some minimum support threshold, we can integrate the support counting with the depth-first traversal of the algorithm, rather than first mining and storing $\mathcal{F}$ in its entirety. Since for real datasets and nontrivial minimal support thresholds billions of frequent itemsets are easily discovered, this indubitably makes our approach more practical.

7. EXPERIMENTS

In this section we experimentally evaluate our method and empirically validate the quality of the returned summaries.

7.1. Setup

We implemented our algorithm in C++, and provide the source code for research purposes.[2] All experiments were executed on a 2.67GHz (six-core) Intel Xeon machine with 12GB of memory, running Linux. All reported timings are of the single-threaded implementation of our algorithm.

We evaluate our method on three synthetic datasets, as well as on eleven real datasets. Their basic characteristics are given in Table II.

The *Independent* data has independent items with random frequencies between 0.2 and 0.8. In the *Markov* dataset each item is a noisy copy of the previous one, with a random copy probability between 0.5 and 0.8. The *Mosaic* dataset is generated by

[2] http://www.adrem.ua.ac.be/implementations

randomly planting five itemsets of size 5 with random frequencies between 0.2 and 0.5, in a database with 1% noise.

The *Abstracts* dataset contains the abstracts of all accepted papers at the ICDM conference up to 2007, where all words have been stemmed and stop words have been removed [Kontonasios and De Bie 2010].

The *Accidents*, *Kosarak*, *Mushroom*, and *Retail* datasets were obtained from the FIMI dataset repository [Goethals and Zaki 2004; Geurts et al. 2003; Brijs et al. 1999].

The *Chess (kr–k)* and *Plants* datasets were obtained from the UCI ML Repository [Frank and Asuncion 2010], the former was converted into binary form, by creating an item for each attribute-value pair. The latter contains a list of plants, and the U.S. and Canadian states where they occur.

The *DNA Amplification* data contains information on DNA copy number amplifications [Myllykangas et al. 2006]. Such copies activate oncogenes and are hallmarks of nearly all advanced tumors. Amplified genes represent attractive targets for therapy, diagnostics and prognostics. In this dataset items are genes, and transactions correspond to patients.

The *Lotto* dataset was obtained from the website of the Belgian National Lottery, and contains the results of all lottery draws between May 1983 and May 2011.[3] Each draw consist of seven numbers (six plus one bonus ball) out of a total of 42.

The *Mammals* presence data consists of presence records of European mammals within geographical areas of 50×50 kilometers [Mitchell-Jones et al. 1999].[4] Heikinheimo et al. [2007] analyzed the distribution of these mammals over these areas, and showed they form environmentally distinct and spatially coherent clusters.

The *MCADD* data was obtained from the Antwerp University Hospital. Medium-Chain Acyl-coenzyme A Dehydrogenase Deficiency (MCADD) [Baumgartner et al. 2005; Van den Bulcke et al. 2011] is a deficiency newborn babies are screened for during a Guthrie test on a heel prick blood sample. The instances are represented by a set of 21 features: 12 different acylcarnitine concentrations measured by tandem mass spectrometry (TMS), together with 4 of their calculated ratios and 5 other biochemical parameters, each of which we discretized using k-means clustering with a maximum of 10 clusters per feature.

The core of our method is parameter-free. That is, it will select itemsets from the complete space of possible itemsets. In practice, however, it may not always be feasible or desirable to consider *all* itemsets. For dense or large datasets, for instance, we might want to explicitly exclude low-frequency itemsets, or itemsets containing many items. General speaking, choosing a larger candidate space, yields a larger search space, and hence potentially better models. In our experiments we therefore consider collections of frequent itemsets $\mathcal{F}$ mined at support thresholds as low as feasible. The actual thresholds and corresponding size of $\mathcal{F}$ are depicted in Table II. Note that although the minimum support threshold is used to limit the size of $\mathcal{F}$, it can be seen as an additional parameter to the algorithm. To ensure efficient computation, we impose a maximum of 10 items per group, as described at the end of Section 4.4. Further, we terminate the algorithm if the runtime exceeds two hours. For the sparse datasets with many transactions (the synthetic ones, *Accidents*, *Kosarak*, and *Retail*), we mine and store the supports of the itemsets, rather than computing them on the fly. Caching the supports is faster in these cases, since for large datasets support counting is relatively expensive. The runtimes reported below include this additional step.

[3] http://www.nationaleloterij.be

[4] The full version of the *Mammals* dataset is available for research purposes from the Societas Europaea Mammalogica at http://www.european-mammals.org

Table II: The synthetic and real datasets used in the experiments. Shown for each dataset are the number of items $|A|$, the number of transactions $|D|$, the minimum support threshold for the set of candidate frequent itemsets $\mathcal{F}$ and its size.

	Data Properties		*Candidate Collection*	
Dataset	$\|A\|$	$\|D\|$	$minsup$	$\|\mathcal{F}\|$
Independent	50	100 000	5 000	1 055 921
Markov	50	100 000	5 000	377 011
Mosaic	50	100 000	5 000	101 463
Abstracts	3 933	859	10	75 061
Accidents	468	340 183	50 000	2 881 487
Chess (kr–k)	58	28 056	5	114 148
DNA Amplification	391	4 590	5	$4.57{\cdot}10^{12}$
Kosarak	41 270	990 002	1 000	711 424
Lotto	42	2 386	1	139 127
Mammals	121	2 183	200	93 808 244
MCADD	198	31 924	50	1 317 234
Mushroom	119	8 124	100	66 076 586
Plants	70	34 781	2 000	913 440
Retail	16 470	88 162	10	189 400

In all experiments, we set the background information $\mathcal{B}$ to contain the column margins, i.e., we start from the independence model. For *Chess (kr–k)* and *Mushroom*, we also perform experiments which include the row margins, since their original form is categorical, and hence we know that each transaction has a fixed size. For the *Lotto* data we additionally use *only* the row margins, which implicitly assumes all numbers to be equiprobable, and uses the fact that each draw consist of seven numbers.

7.2. A First Look at the Results

In Tables III and IV we present the scores and sizes of the discovered summaries, the time required to compute them, and for comparison we include the score of the background model, using BIC and MDL respectively as quality score s. For most of the datasets we consider, the algorithm identifies the optimum quite rapidly, i.e., within minutes. For these datasets, the number of discovered itemsets, k, is indicated in bold. For three of the datasets, i.e., the dense *MCADD* dataset, and the large *Kosarak* and *Retail* datasets, the algorithm did not find the optimum within two hours.

For both BIC and MDL, we note that the number of itemsets in the summaries is very small, and hence manual inspection of the results by an expert is feasible. We see that for most datasets the score (and thus relatedly the negative log-likelihood) decreases a lot, implying that these summaries model the data well. Moreover, for highly structured datasets, such as *DNA*, *Kosarak*, and *Plants*, this improvement is very large, and only a handful of itemsets are required to describe their main structure.

Comparing the two tables, we see that the number of itemsets selected by BIC compared to MDL tends to be the same or a bit higher, indicating that BIC is more permissive in adding itemsets—an expected result. If we (crudely) interpret the raw BIC and MDL scores as negative log-likelihoods, we see that BIC achieves lower, and hence better scores. This follows naturally from the larger collections of itemsets that BIC selects; the larger the collection, the more information it provides, and hence the higher the likelihood.

Table III: Statistics of the discovered summaries using BIC, with respect to the background information $\mathcal{B}$. Shown are the number of itemsets k, the wall clock time, the score of $\mathcal{C}$, and the score of the empty collection. (Lower scores are better.) Values for k identified as optimal by BIC are given in boldface.

Dataset	k	*time*	$\text{BIC}(\mathcal{C}, D; \mathcal{B})$	$\text{BIC}(\emptyset, D; \mathcal{B})$
Independent	**7**	2m14s	4 494 219	4 494 242
Markov	**62**	15m44s	4 518 101	4 999 963
Mosaic	**16**	6m06s	807 603	2 167 861
Abstracts	**220**	27m21s	233 483	237 771
Accidents	**74**	18m35s	24 592 869	25 979 244
Chess (kr–k)	**67**	83m32s	766 235	786 651
DNA Amplification	**204**	4m31s	79 164	183 121
Kosarak	261	120m36s	66 385 663	70 250 533
Lotto	**29**	0m31s	64 970	65 099
Mammals	**76**	25m23s	99 733	119 461
MCADD	80	121m02s	2 709 240	2 840 837
Mushroom	**80**	28m11s	358 369	441 130
Plants	**94**	66m56s	732 145	1 271 950
Retail	62	121m46s	8 352 161	8 437 118

Table IV: Statistics of the discovered summaries using MDL, with respect to the background information $\mathcal{B}$. Shown are the number of itemsets k, the wall clock time, the description length of $\mathcal{C}$, and the description length of the empty collection. (Lower scores are better). Values for k identified as optimal by MDL are given in boldface.

Dataset	k	*time*	$\text{MDL}(\mathcal{C}, D; \mathcal{B})$	$\text{MDL}(\emptyset, D; \mathcal{B})$
Independent	**0**	1m57s	4 494 243	4 494 243
Markov	**62**	16m16s	4 519 723	4 999 964
Mosaic	**15**	7m23s	808 831	2 167 861
Abstracts	**29**	1m26s	236 522	237 772
Accidents	**75**	19m35s	24 488 943	25 979 245
Chess (kr–k)	**54**	76m20s	767 756	786 652
DNA Amplification	**120**	1m06s	96 967	183 122
Kosarak	262	120m53s	66 394 995	70 250 534
Lotto	**0**	0m01s	65 100	65 100
Mammals	**53**	2m17s	102 127	119 461
MCADD	82	123m34s	2 700 924	2 840 838
Mushroom	**74**	22m00s	361 891	441 131
Plants	**91**	63m9s	734 558	1 271 951
Retail	57	122m44s	8 357 129	8 437 119

7.3. Summary Evaluation

Next, we inspect the discovered data summaries in closer detail.

For the *Independent* dataset we see that using MDL the returned summary is empty, i.e., no itemset can improve on the description length of the background model, which is the independence model. In general, MDL does not aim to find any underlying 'true' model, but simply the model that it considers best, given the available data. In this

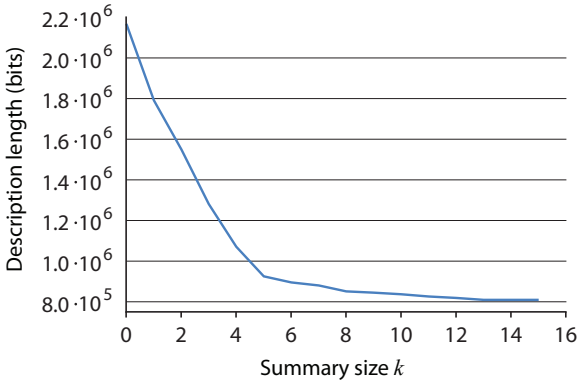

Fig. 1: The description length of the *Mosaic* dataset, as a function of the summary size k. The first five discovered itemsets correspond to the process which generated the dataset. The minimum MDL score is attained at $k = 15$.

case, however, we see that the discovered model correctly corresponds to the process by which we generated the data. Using BIC, on the other hand, the algorithm discovers 7 itemsets. These itemsets have an observed frequency in the data which is slightly higher or lower than their predicted frequencies under the independence assumption. Among the 7 itemsets, the highest absolute difference between those two frequencies is lower than 0.3%. While these itemsets are each significant by themselves, after correcting their p-values to account for Type I error (see Section 7.5), we find that none of them are statistically significant.

For the *Markov* data, we see that all itemsets in the discovered summary are very small (about half of them of size 2, the rest mostly of size 3 or 4), and they all consist of consecutive items. Since the items in this dataset form a Markov chain, this is very much in accordance with the underlying generating model. In this case, the summaries for BIC and MDL are identical. Knowing that the data is generated by a Markov chain, we can compute the MDL score of the 'best' model. Given that $\mathcal{B}$ contains the singleton frequencies, the model that fully captures the Markov chain contains all 49 itemsets of size two containing consecutive items. The description length for this model is 4 511 624, which is close to what our algorithm discovered.

The third synthetic dataset, *Mosaic*, contains 5 itemsets embedded in a 1%-noisy database. The five first itemsets returned, both by BIC and MDL, are exactly those itemsets. These sets are responsible for the better part of the decrease of the score, as can be seen in Figure 1, which for MDL depicts the description length as a function of the summary size. After these first five highly informative itemsets, a further ten additional itemsets are discovered that help explain the overlap between the itemsets—which, because we construct a probabilistic model using itemset frequencies cannot be inferred otherwise—and while these further itemsets do help in decreasing the score, their effect is much less strong than for the first sets. After discovering fifteen itemsets, the next best itemset does not decrease the log likelihood sufficiently to warrant the MDL model complexity penalty, and the algorithm terminates.

For *Lotto*, we see that using MDL, our algorithm fails to discover any informative itemsets.[5] Using BIC, we find a summary of 29 itemsets, which are all combinations of 3 numbers, having been drawn between one and three times in the past twenty-odd

[5]This is a desired result, assuming of course that the lottery is fair.

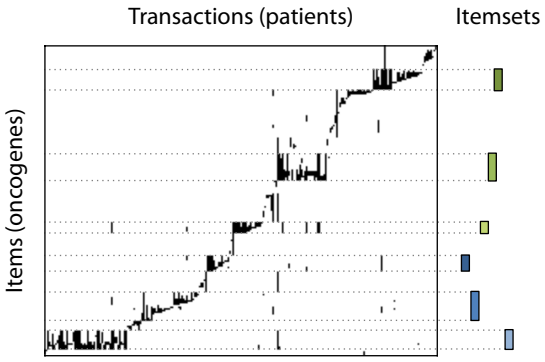

Fig. 2: Detail of the *DNA Amplification* dataset (left), along with six of the discovered itemsets (right).

years. While for each itemset individually this is significantly lower than the expected absolute support (which is about 11), when we adjust for Type I error, they are no longer statistically significant. This gives evidence that in our setup, and with these data sizes, BIC may not be strict enough. We additionally ran experiments using only row margins as background knowledge, i.e., using the fact that every transaction contains exactly seven items, the numbers of each lottery draw. Then, according to the maximum entropy background distribution, this implies that every number has exactly the same probability of being picked in a particular draw—namely ⅙. If our algorithm should find that a ball is drawn significantly more or less often, then it would be included in the model. Further, if there were any meaningful correlation between some of the numbers (positive or negative), these numbers would be included in a reported itemset. Using MDL, our algorithm again finds no itemsets of interest. Using BIC, we find 7 itemsets (in a timespan of two hours, after which the algorithm was terminated), which are all combinations of five or six numbers, with absolute supports between 2 and 4, which is higher than expected. After applying Bonferroni adjustment, none of these itemsets turn out to be significant.

In the case of the *DNA Amplification* data, our algorithm finds that the data can be described using 120 itemsets. As this dataset is banded, it contains a lot of structure [Garriga et al. 2011]. Our method correctly discovers these bands, i.e., blocks of consecutive items corresponding to related genes, which lie on the same chromosomes. The first few dozen sets are large, and describe the general structure of the data. Then, as we continue, we start to encounter smaller itemsets, which describe more detailed nuances in the correlations between the genes. Figure 2 depicts a detail of the *DNA* dataset, together with a few of the itemsets from the discovered summary.

For the *Chess (kr–k)* and *Mushroom* datasets, we also ran experiments including the row margins, since we know that these datasets originated from categorical data, and hence the margins of these datasets are fixed. As remarked in Section 4.7, this increases the runtime of the algorithm by a factor $N = |A|$. For both BIC and MDL, the algorithm therefore took longer to execute, and was terminated after two hours for both datasets. The size of the summaries in all cases was equal to 6. Comparing between them reveals that some (but not all) of the itemsets also occur in the summaries that did not use the row margins as background knowledge, which leads us to conclude that the use of row margins as background information for these datasets does not have a substantial effect on the discovered summaries, at least not for the top-six.

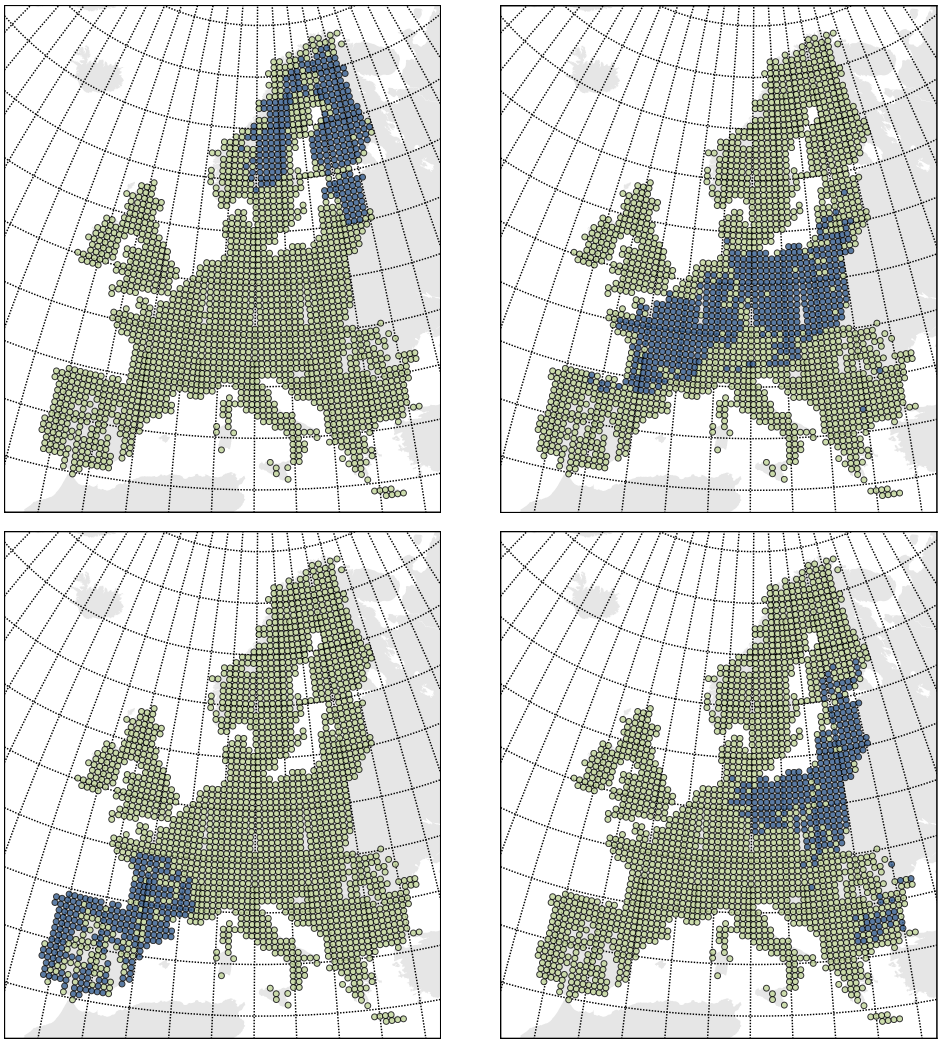

Fig. 3: For the *Mammals* dataset, for four of the itemsets discovered by our method, we depict the transactions (i.e., locations) that support that itemset in blue (dark). A transaction supports the itemset if all of the mammals identified by the set have been recorded in the data to occur in that area.

The items in the *Mammals* data are European mammals, and the transactions correspond to their occurrences in geographical areas of (50×50) km^2. The discovered itemsets represent sets of mammals that are known to co-exist in certain regions. For instance, one such set contains the Eurasian elk (moose), the mountain hare, the Eurasian lynx, and the brown bear, which are all animals living in colder, northern territories. Going back to the transactions of the data, we can also look at the areas where these itemsets occur. Figure 3 depicts the geographical areas for some of the discovered itemsets. Some clear regions can be recognized, e.g., Scandinavia (for the aforementioned itemset), Central Europe, the Iberian peninsula, or Eastern Europe.

Heikinheimo et al. [2007] considered the same data, annotated with environmental data, and applied mixture modelling and k-means clustering to identify regions with similar mammal presence. When comparing to these expert-validated results, we observe a strong correlation to the regions (i.e., itemsets) MTV discovers. Although our method does not hard-cluster the map, we observe that our itemsets relate to clusters are various level of detail; the region identified in the top-right of Fig. 3 shows strong correlation to the large 'mainland' cluster of [Heikinheimo et al. 2007] at $k = 2, 3$ whereas the mammals identified in the bottom-left plot of Fig. 3 correspond to 2 clusters that only show up at $k \in [11, 13]$.

The *Plants* dataset is similar to the *Mammals* data, except than now the plants form transactions, and the items represent the U.S. and Canadian states where they occur. The discovered itemsets, then, are states that exhibit similar vegetation. Naturally, these are typically bordering states. For instance, some of the first discovered itemsets are {CT, IL, IN, MA, MD, NJ, NY, OH, PA, VA} (North-Western states), {AL, AR, GA, KY, LA, MO, MS, NC, SC, TN} (South-Western states), and {CO, ID, MT, OR, UT, WA, WY} (North-Eastern states).

Finally, for the ***MCADD*** data, we find about 80 itemsets after running the algorithm for two hours. The attributes in this dataset are measured fatty acid concentrations, ratios between these, and some further biochemical parameters, which have all been discretized. The items therefore are attribute-value pairs. In the discovered summary, we see that the selected itemsets mostly consist of items corresponding to a few known key attributes. Furthermore, we can identify strongly correlated attributes by regarding those combinations of attributes within the itemsets selected in the summary. As an example, one of the first itemsets of the summary corresponds to particular values for attributes $\{MCADD, C8, \frac{C8}{C2}, \frac{C8}{C10}, \frac{C8}{C12}\}$, respectively the class label, an acid, and some of its calculated ratios. This acid and its ratios, and the identified values, are commonly used diagnostic criteria for screening MCADD, and were also discovered in previous in-depth studies [Baumgartner et al. 2005; Van den Bulcke et al. 2011], and similarly for other itemsets in the summary.

7.4. Comparison with Other Methods

In Table V, we give the top-10 itemsets in the *Abstracts* dataset, as discovered by our algorithm (using MDL), the method by Kontonasios and De Bie [2010], the compression-based KRIMP algorithm [Vreeken et al. 2011], and Tiling [Geerts et al. 2004]. We see that our algorithm discovers important data mining topics such as *support vector machines*, *naive bayes*, and *frequent itemset mining*. Further, there is little overlap between the itemsets, and there is no variations-on-the-same-theme type of redundancy.

The results of the Information-Theoretic Noisy Tiles algorithm by Kontonasios and De Bie [2010], based on the Information Ratio of tiles, are different from ours, but seem to be more or less similar in quality for this particular dataset.

The KRIMP algorithm does not provide its resulting itemsets in an order, so in order to compare between the different methods, following its MDL approach, we selected the top-10 itemsets from the code table that have the highest usage, and hence, shortest associated code. From a compression point of view, the items in these sets co-occur often, and thus result in small codes for the itemsets. Arguably, this does not necessarily make them the most interesting, however, and we observe that some rather general terms such as *state [of the] art* or *consider problem* are ranked highly.

Finally, for Tiling we provide the top-10 tiles of at least two items, i.e., the ten tiles whose support times their size is highest. Without the minimum size constraint, only singleton itemsets are returned, which, although objectively covering the largest area individually, are not very informative. Still, the largest discovered tiles are of size two,

Table V: The top-10 itemsets of the *Abstracts* dataset for our MTV algorithm using MDL (top left), Kontonasios and De Bie [2010] (top right), KRIMP [Vreeken et al. 2011] (bottom left), and Tiling [Geerts et al. 2004] (bottom right).

MTV
support vector machin svm
associ rule mine
nearest neighbor
frequent itemset mine
naiv bay
linear discrimin analysi lda
cluster high dimension
state art
frequent pattern mine algorithm
synthet real

Information-Theoretic Noisy Tiles
support vector machin
effici discov frequent pattern mine algorithm
associ rule mine algorithm database
train learn classifi perform set
frequent itemset
mine high dimensional cluster
synthetic real
time seri
decis tree classifi
problem propos approach experiment result

KRIMP
algorithm experiment result set
demonstr space
larg databas
consid problem
knowledg discoveri
experiment demonstr
rule mine associ databas
algorithm base approach cluster
state art
global local

Tiling
algorithm mine
algorithm base
result set
approach problem
propos method
experiment result
algorithm perform
model base
set method
algorithm gener

and contain quite some redundancy, for instance, the top-10 contains only 13 (out of 20) distinct items.

Each of these methods formalize the task of summarization differently, and hence optimize objective scores quite different from ours. It therefore is impossible to fairly and straightforwardly quantitatively compare to these methods; neither construct a itemset-frequency-prediction model as we do, and not even result itemsets together with their frequencies. Tiling and Information Theoretic Tiling both result in tiles, sets of itemsets with tid-lists, while KRIMP code tables contain itemsets linked to relative *usage* frequencies of covering the data without overlap. It hence makes as little sense to score KRIMP itemsets by our score, as it would to consider our itemsets as a code table.

7.5. Significant Itemsets

Next, we investigate the significance of the itemsets that are included in the summaries we discover, as well as the significance of the itemsets in $\mathcal{F}$ that were not included. To properly compute the p-values of the itemsets, we employ holdout evaluation, as described by Webb [2007]. That is, we equally split each dataset into an exploratory (training) set D_e and a holdout (test) set D_h, apply our algorithm to the exploratory data, and evaluate the itemsets on the holdout set. Since D_e contains less data than D, the discovered models tend to contain fewer itemsets (i.e., as there is less data to fit the model on, BIC and MDL both allow for less complex models).

The significance of an itemset X is evaluated as follows. We compute its estimated probability $p^*(X = 1)$ (using either $\mathcal{B}$, or $\mathcal{B}$ and $\mathcal{C}$, consistent with D_e). This is the null hypothesis. Then, we calculate the two-tailed p-value given the observed frequency in

the holdout data, $fr(X)$. Let us write $d = |D_h| = |D|/2$, $f = d \cdot fr(X)$, and $p = p^*(X = 1)$. The p-value of X expresses the probability of observing an empirical frequency $q_{D_h}(X)$ at least as extreme (i.e., improbable) as the observed frequency $fr(X)$, with respect to the model, according to a binomial distribution parametrized by d and p

$$B(d; p)(f) = \binom{d}{f} p^f (1-p)^{(d-f)} \ .$$

Assuming that $fr(X) \geq p^*(X = 1)$, we calculate the two-tailed p-value of X as

$$\begin{aligned} \text{p-value} &= \text{Prob}(q_{D_h}(X) \geq f \mid p) + \text{Prob}(q_{D_h}(X) \leq f' \mid p) \\ &= \sum_{i=f}^{d} \binom{d}{i} p^i (1-p)^{d-i} + \sum_{i=0}^{f'} \binom{d}{i} p^i (1-p)^{d-i} \end{aligned}$$

where

$$f' = \max\{f' \in [0, dp) \mid B(d; p)(f') \leq B(d; p)(f)\} \ .$$

The p-value for the case $fr(X) < p^*(X = 1)$ is defined similarly.

It is expected that the itemsets that our algorithm discovers are significant with respect to the background model. Simply put, if the frequency of an itemset is close to its expected value, it will have a high p-value, and thus it will not be significant. Moreover, it will also have a low heuristic value $h(X)$, and hence it will not greatly improve the model. The following demonstrates this connection between kl (and hence h), and the p-value. Using Stirling's approximation, we can write the logarithm of the binomial probability $B(d; p)(f)$ as follows.

$$\begin{aligned} \log \binom{d}{f} p^f (1-p)^{d-f} &\approx d \log d - f \log f - (d-f) \log(d-f) \\ &\quad + f \log p + (d-f) \log(1-p) \\ &= -f \log \frac{f}{dp} - (d-f) \log \frac{d-f}{d(1-p)} \\ &= -d \cdot kl(X) \end{aligned}$$

Therefore, when we maximize the heuristic h, we also indirectly minimize the p-value. Note, however, that in our algorithm we do not employ a significance level as a parameter; we simply let BIC or MDL decide when to stop adding itemsets. Further, we can also take the complexity of the itemsets into account.

In Table VI we show the number of significant selected and unselected itemsets. Since we are testing multiple hypotheses, we apply Bonferroni adjustment, to avoid Type I error—falsely rejecting a true hypothesis [Shaffer 1995]. That is, if we were to test, say, 100 true hypotheses at significance level 0.05, then by chance we expect to falsely reject five of them. Therefore, at significance level α, each p-value is compared against the adjusted threshold α/n where n is the number of tests performed.

The first column of Table VI shows the number of itemsets in $\mathcal{C}$ that are significant with respect to the background knowledge $\mathcal{B}$. In general, we see that all itemsets are significant. However, for the *Accidents* dataset, e.g., we find two itemsets that are not significant with respect to $\mathcal{B}$. Nevertheless, they are not redundant in this case; upon inspection, it turns out that these itemsets (say, X_1 and X_2) are subsets of another itemset (say, Y) in $\mathcal{C}$ that was significant. After adding Y to the model, the estimates of X_1 and X_2 change, causing them to become significant with respect to the intermediate model. A similar observation is made for the *Chess (kr–k)* and *Mushroom* datasets. In the second column, we therefore also show the number of itemsets $X_{i+1} \in \mathcal{C}$ that are

significant with respect to the previous model $\mathcal{C}_i$. In this case we see that indeed all itemsets are significant, from which we conclude that all itemsets really contribute to the summary, and are not redundant.

Next, we compute the p-values of the itemsets in the candidate set $\mathcal{F}$ that were not included in $\mathcal{C}$. Since for all datasets the candidate set $\mathcal{F}$ is very large, we uniformly sample 1 000 itemsets from $\mathcal{F} \setminus \mathcal{C}$, and compute their p-values with respect to $p^*_{\mathcal{B},\mathcal{C}}$. We see that for seven of the datasets, there are few significant itemsets, which means that $\mathcal{C}$ captures the data quite well. For two datasets (*Mushroom* and *Retail*), a few hundred are significant, but still many are not. For the five remaining datasets, however, we see that almost all itemsets are significant. Nonetheless, this does not automatically mean that the discovered models are poor. That is, apart from considering the deviation of an itemset's frequency, we also consider its complexity and the complexity of a model as a whole. This means that even if the observed frequency of an itemset is surprising to some degree, it may be too complex to include it; upon inspection, we indeed find that among the sampled itemsets, there tend to be many large ones.

Table VI: The number of significant itemsets in the discovered summary $\mathcal{C}$, with respect to background knowledge $\mathcal{B}$ and each intermediate model $\mathcal{C}_i$, and the number of significant itemsets among 1 000 itemsets sampled from $\mathcal{F} \setminus \mathcal{C}$, denoted by $\mathcal{S}$. We use a significance level of 0.05, and Bonferroni-adjusted p-values.

	$X \in \mathcal{C}$			$X \in \mathcal{S}$	
	# signif. w.r.t. $\mathcal{B}$	*# signif. w.r.t.* $\mathcal{C}_i$	$\lvert\mathcal{C}\rvert$	*# signif. w.r.t.* $\mathcal{C}$	$\lvert\mathcal{S}\rvert$
Independent	0	0	**0**	0	1 000
Markov	64	64	**64**	6	1 000
Mosaic	14	14	**14**	0	1 000
Abstracts	12	12	**12**	14	1 000
Accidents	69	71	**71**	883	1 000
Chess (kr–k)	42	42	**43**	37	1 000
DNA Amplification	87	87	**87**	990	1 000
Kosarak	268	268	268	993	1 000
Lotto	0	0	**0**	0	1 000
Mammals	39	39	**39**	986	1 000
MCADD	61	61	61	89	1 000
Mushroom	59	63	**63**	139	1 000
Plants	87	87	**87**	998	1 000
Retail	65	65	65	302	1 000

8. DISCUSSION

The approach introduced in this paper fulfills several intuitive expectations one might have about summarization, such as succinctness, providing a characteristic description of the data, and having little redundancy. The experiments show that quantitatively we can achieve good BIC and MDL scores with only a handful of itemsets, and that these results are highly qualitative and meaningful; moreover, we can discover them in a relatively short amount of time. In practice, we see that the results using MDL are slightly better than those using BIC; the former tends to be a bit more conservative, in the sense that it does not discover spurious itemsets. Furthermore, using our MDL

score, we have more control over the complexity of the summary, since it takes into account not only the summary size but also the sizes of the itemsets within it.

MDL does not provide a free lunch. First of all, although highly desirable, it is not trivial to bound the score. For Kolmogorov complexity, which MDL approximates, we know this is incomputable. For our models, however, we have no proof one way or another. Furthermore, although MDL gives a principled way to construct an encoding, this involves many choices that determine what structure is rewarded. As such, we do not claim our encoding is suited for all goals, nor that it cannot be improved. Since we take a greedy approach, and in each step optimize a heuristic function, we also do not necessarily find the globally optimal solution with respect to MDL or BIC. However, we argue that in practice it is not strictly necessary to find an optimal summary, but to find one that provides a significant amount of novel information.

In this paper we consider data mining as an iterative process. By starting off with what we already know—our background knowledge—we can identify those patterns that are the surprising to us. Simply finding the itemset that is most surprising, is a problem that Hanhijärvi et al. [2009] describe as 'tell me something I don't know'. When we repeat this process, in the end, we will have identified a group of itemsets that 'tell me all there is to know' about the data. Clearly, this group strongly overfits the data. This is where the MDL principle provides a solution, as it automatically identifies the most informative group. Hence, we paraphrase our approach as 'tell me what I need to know'. As such, by our Information Theoretic approach it can be seen as an instantiation of the general iterative framework recently proposed by De Bie [2011a].

The view that we take here on succinctness and non-redundancy is fairly strict. Arguably, there are settings conceivable where limited redundancy (at the cost of brevity) can give some robustness to a technique, or provide alternative insights by restating facts differently. However, this is not the intention of this paper, and we furthermore argue that our method can perfectly be complemented by techniques such as redescription mining [Zaki and Ramakrishnan 2005].

Data mining is not only an iterative process, but also an interactive one. The MTV algorithm above simply returns a set of patterns to the user, with respect to his or her background knowledge. However, in practice we might want to dynamically guide the exploration and summarization process. That is, we may want to add or remove certain itemsets, next let the algorithm add some more itemsets, etc. Our algorithm can easily be embedded into an interactive environment. Deleting an itemset from a summary is quite straightforward; we just collapse the transaction partition, and rerun the iterative scaling algorithm to discover the parameters of the simplified model.

Even though in this paper we significantly extend upon Mampaey et al. [2011], there are some further improvements possible for our method. For instance, one problem setting in which our method is applicable, is that of finding the best specialization of a given itemset X. That is, to identify the superset Y of X that provides the best score $s(\mathcal{C} \cup \{Y\})$. This setup allows experts to interactively discover interesting itemsets. As part of future work, we are currently investigating this in practice for finding patterns in proteomics and mass-spectrometry data.

The experiments showed that we discover high-quality summaries, and that we can efficiently compute the maximum entropy model for a given collection of itemsets in practice—even though the latter is an NP-hard problem in general. In specific cases, however, there is room for further optimization. For computational reasons, we split up the distribution into smaller groups, by restricting the number of items per group. However, this does imply that exactly modeling a (very) long Markov chain, for instance, is not possible, as only parts up to n items will be modeled. While a complete Markov chain can easily be described by our model, computing it may prove to be difficult in practice. It would be interesting to see then, how modeling techniques as de-

composition, for instance, using junction trees [Cowell et al. 1999], could be combined with our methods.

We see several further possibilities for improving upon our current, unoptimized, implementation of MTV. For example, massive parallelization can be applied to the search for the most informative itemset. This requires computing itemset probabilities with respect to a single model, for many candidates, which can easily be done in parallel. This can also benefit pruning in this phase, since we can use the maximum heuristic value over all processes. Another option is to simplify the computation of the row margin distribution, which adds a factor N to the runtime of the algorithm. Rather than computing the probability that a transaction contains a certain number of items, we could group these probabilities in to a coarser granularity, to reduce this factor.

9. CONCLUSION

We introduced a well-founded method for iteratively mining non-redundant collections of itemsets that form high-quality summaries of transaction data. By employing the Maximum Entropy principle, we obtain unbiased probabilistic models of the data, through which we can identify informative itemsets, and subsequently iteratively build succinct summaries of the data by updating our model accordingly. As such, unlike static interestingness models, our approach does not return patterns that are redundant with regard to what we have learned, or already knew as background knowledge, and hence the result is kept succinct and maximally informative.

To this end, we presented the MTV algorithm for mining informative summaries, which can either mine the top-k most informative itemsets, or by employing either the Bayesian Information Criterion (BIC) or the Minimum Description Length (MDL) principle, we can automatically identify the set of itemsets that as a whole provides a high quality summary of the data. Hence, informally said, our method 'tells you what you need to know' about the data.

Although in the general case modeling by Maximum Entropy is NP-hard, we showed that in our case we can do so efficiently using Quick Inclusion-Exclusion. Furthermore, MTV is a one-phase algorithm; rather than picking itemsets from a pre-materialized user-provided candidate set, it calculates on-the-fly the supports for only those itemsets that stand a chance of being included in the summary.

Experiments show that we discover succinct summaries, which correctly identify important patterns in the data. The resulting models attain high log-likelihoods using only few itemsets, are easy to interpret, contain significant itemsets with low p-values, and for a wide range of datasets are shown to give highly intuitive results.

ACKNOWLEDGMENTS

The authors wish to thank Tijl De Bie for pre-processing and making available of the ICDM Abstracts data, the i-ICT of Antwerp University Hospital (UZA) for providing the MCADD data and expertise, and last, but not least, the anonymous referees whose insightful comments and suggestions helped to improve the paper considerably.

APPENDIX

A. PROOF OF COROLLARY 4.3

COROLLARY 4.3. *The log-likelihood of the maximum entropy distribution $p^*_{\langle \mathcal{C}, \Phi \rangle}$ for a collection of itemsets and frequencies $\langle \mathcal{C}, \Phi \rangle$ is equal to*

$$\log p^*_{\langle \mathcal{C}, \Phi \rangle}(D) = |D| \left(\log u_0 + \sum_{(X_i, f_i) \in \langle \mathcal{C}, \Phi \rangle} f_i \log u_{X_i} \right).$$

PROOF. From Theorem 4.2, we have that $p^*_{\langle \mathcal{C},\Phi\rangle}(A=t) = u_0 \prod_{X\in\mathcal{C}} u_X^{S_X(t)}$. Hence

$$\begin{aligned}
\log p^*_{\langle \mathcal{C},\Phi\rangle}(D) &= \sum_{t\in D} \log p^*_{\langle \mathcal{C},\Phi\rangle}(A=t) \\
&= \sum_{t\in D} \left(\log u_0 + \sum_{X_i\in\mathcal{C}} S_{X_i}(t) \log u_{X_i} \right) \\
&= |D| \left(\log u_0 + \sum_{(X_i,f_i)\in\langle \mathcal{C},\Phi\rangle} f_i \log u_{X_i} \right) \\
&= -|D| H\left(p^*_{\langle \mathcal{C},\Phi\rangle} \right) .
\end{aligned}$$

The transition from the second to the third line follows from the fact that $\sum_{t\in D} S_{X_i}(t) = supp(X)$. □

B. PROOF OF THEOREM 4.5

THEOREM 4.5. *Assume a collection of itemsets $\mathcal{C} = \{X_1,\ldots,X_k\}$ and let $p^*_{\mathcal{C}}$ be the maximum entropy model, computed from a given dataset D. Assume also an alternative model $r(A = t \mid f_1,\ldots,f_k)$, where $f_i = fr(X_i \mid D)$, that is, a statistical model parametrized by the frequencies of $\mathcal{C}$. Assume that for any two datasets D_1 and D_2, where $fr(X \mid D_1) = fr(X \mid D_2)$ for any $X \in \mathcal{C}$, it holds that*

$$1/|D_1| \log r(D_1 \mid f_1,\ldots,f_k) = 1/|D_2| \log r(D_2 \mid f_1,\ldots,f_k) .$$

Then $p^*_{\mathcal{C}}(D) \geq r(D)$ for any dataset D.

PROOF. There exists a sequence of finite datasets D_j such that $fr(X_i \mid D_j) = f_i$ and $q_{D_j} \to p^*_{\mathcal{C}}$. To see this, first note that f_i are rational numbers so that the set of distributions $\mathcal{P}_{\mathcal{C}}$ is a polytope with faces defined by rational equations. In other words, there is a sequence of distributions p_j with only rational entries reaching p^*. Since a distribution with rational entries can be represented by a finite dataset, we can have a sequence D_j such that $q_{D_j} = p_j$.

Now as j goes to infinity, we have

$$\frac{1}{|D|}\log r(D) = \frac{1}{|D_j|}\log r(D_j) = \sum_{t\in\mathcal{T}} q_{D_j}(A=t)\log r(A=t) \to \sum_{t\in\mathcal{T}} p^*_{\mathcal{C}}(A=t)\log r(A=t) .$$

Since the left side does not depend on j we actually have a constant sequence. Hence,

$$0 \leq KL(p^*_{\mathcal{C}} \,\|\, r) = H(p^*_{\mathcal{C}}) - \sum_{t\in\mathcal{T}} p^*_{\mathcal{C}}(A=t)\log r(A=t) = 1/|D|(\log p^*_{\mathcal{C}}(D) - \log r(D)) ,$$

which proves the theorem. □

C. PROOF OF THEOREM 4.18

First, we state two lemmas, which are then used in the proof of the subsequent theorem.

LEMMA C.1. *Let $\mathcal{G}$ and $\mathcal{H}$ be itemset collections such that $\mathcal{G} \subseteq \mathcal{H} \subseteq closure(\mathcal{G})$, then $closure(\mathcal{H}) = closure(\mathcal{G})$.*

PROOF. Write $\mathcal{F} = closure(\mathcal{G})$ and let $U = \bigcup_{X\in\mathcal{G}} X$, $V = \bigcup_{X\in\mathcal{H}} X$, and $W = \bigcup_{X\in\mathcal{F}} X$. By definition we have $U \subseteq V$ which implies that $closure(\mathcal{G}) \subseteq closure(\mathcal{H})$. Also since $V \subseteq W$, we have $closure(\mathcal{H}) \subseteq closure(\mathcal{F}) = closure(\mathcal{G})$, where the second equality follows from the idempotency of closure. □

LEMMA C.2. *Let $\mathcal{G}$ be an itemset collection and let $Y \notin \mathcal{G}$ be an itemset. Assume that there is a $t \in \mathcal{T}$ such that $S_X(t)$ for every $X \in \mathcal{G}$ and $S_Y(t) = 0$. Then $Y \notin closure(\mathcal{G})$.*

PROOF. $S_Y(t) = 0$ implies that there is $a_i \in Y$ such that $t_i = 0$. Note that $a_i \notin X$ for any $X \in \mathcal{G}$, otherwise $S_X(t) = 0$. This implies that $Y \notin \bigcup_{X \in \mathcal{G}} X$ which proves the lemma. □

With the lemmas above, we can now prove the main theorem.

THEOREM 4.18. *Given a collection of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$, let $\mathcal{T}_\mathcal{C}$ be the corresponding partition with respect to $\mathcal{C}$. The algorithm* QIEBLOCKSIZES *correctly computes the block sizes $e(T)$ for $T \in \mathcal{T}_\mathcal{C}$.*

PROOF. Let us denote $e^0(T) = c(T)$ for the initialized value, and let $e^i(T)$ be the value of $e(T)$ after the execution of the ith iteration of QIEBLOCKSIZES for $i = 1, \ldots, k$.
Let us write

$$S^i(\mathcal{G}) = \{t \in \mathcal{T} \mid S_X(t) = 1 \text{ for } X \in \mathcal{G} \text{ and } S_X(t) = 0 \text{ for } X \in \mathcal{C}_i \setminus \mathcal{G}\} \ .$$

The following properties hold for S^i:

(1) $S^i(\mathcal{G}) \subseteq S^{i-1}(\mathcal{G})$ for any $\mathcal{G}$ and $i > 0$.
(2) If $X_i \notin \mathcal{G}$, then $S^{i-1}(\mathcal{G}) = S^i(\mathcal{G}) \cup S^{i-1}(\mathcal{G} \cup \{X_i\})$, and $S^{i-1}(\mathcal{G} \cup \{X_i\}) \cap S^i(\mathcal{G}) = \emptyset$.
(3) $S^i(\mathcal{G}) = S^i(closure(\mathcal{G}, i))$ for any $\mathcal{G}$.
(4) If $X_i \in \mathcal{G}$, then $S^i(\mathcal{G}) = S^{i-1}(\mathcal{G})$.

Note that $\left|S^k(sets(T; \mathcal{C}))\right| = e(T)$, hence to prove the theorem we will show by induction that $e^i(T) = \left|S^i(sets(T; \mathcal{C}))\right|$ for $i = 0, \ldots, k$ and for $T \in \mathcal{T}_\mathcal{C}$.
For $i = 0$ the statement clearly holds. Let $i > 0$ and make an induction assumption that $e^{i-1}(T) = \left|S^{i-1}(sets(T; \mathcal{C}))\right|$. If $X_i \in \mathcal{G}$, then by definition $e^i(T) = e^{i-1}(T)$. Property 4 now implies that $S^i(\mathcal{G}) = S^{i-1}(\mathcal{G})$, proving the induction step.
Assume that $X_i \notin \mathcal{G}$. Let us define $\mathcal{F} = \mathcal{G} \cup \{X_i\}$ and let $\mathcal{G}' = closure(\mathcal{F}, i-1)$ and $\mathcal{H} = closure(\mathcal{G}')$. Since it holds that $\mathcal{F} \subseteq \mathcal{G}' \subseteq closure(\mathcal{F})$, Lemma C.1 implies that $\mathcal{H} = closure(\mathcal{F})$.
Assume that $T' = block(\mathcal{G}') = \emptyset$. Then we have $e^i(T) = e^{i-1}(T)$, hence we need to show that $S^i(\mathcal{G}) = S^{i-1}(\mathcal{G})$. Assume otherwise. We will now show that $\mathcal{G}' = \mathcal{H}$ and apply Lemma 4.16 to conclude that $block(\mathcal{G}') \neq \emptyset$, which contradicts our assumption.
First note that if there would exist an $X \in \mathcal{H} \setminus \mathcal{G}'$, then $X \in \mathcal{C}_{i-1}$, since

$$\begin{aligned} \mathcal{H} \setminus \mathcal{G}' &= \mathcal{H} \setminus (\mathcal{F} \cup (closure(\mathcal{F}) \setminus \mathcal{C}_{i-1})) \\ &= \mathcal{H} \setminus (\mathcal{F} \cup (\mathcal{H} \setminus \mathcal{C}_{i-1})) \\ &\subseteq \mathcal{H} \setminus (\mathcal{H} \setminus \mathcal{C}_{i-1}) \\ &\subseteq \mathcal{C}_{i-1} \ . \end{aligned}$$

Since $S^i(\mathcal{G}) \subseteq S^{i-1}(\mathcal{G})$, we can choose by our assumption $t \in S^{i-1}(\mathcal{G}) \setminus S^i(\mathcal{G})$. Since $t \in S^{i-1}(\mathcal{G})$, we have $S_X(t) = 1$ for every $X \in \mathcal{G}$. Also $S_{X_i}(t) = 1$, otherwise $t \in S^i(\mathcal{G})$. Lemma C.2 now implies that for every $X \in \mathcal{C}_{i-1} \setminus \mathcal{F}$, it holds that $X \notin closure(\mathcal{F}) = \mathcal{H}$. This proves that $\mathcal{H} = \mathcal{G}'$, and Lemma 4.16 now provides the needed contradiction. Consequently, $S^{i-1}(\mathcal{G}) = S^i(\mathcal{G})$.
Assume now that $T' = block(\mathcal{G}') \neq \emptyset$, i.e., there exists a $T' \in \mathcal{T}_\mathcal{C}$ such that $\mathcal{G}' = sets(T', \mathcal{C})$. The induction assumption now guarantees that $e^{i-1}(\mathcal{G}') = \left|S^{i-1}(\mathcal{G}')\right|$.

We have

$$\begin{aligned} \left|S^i(\mathcal{G})\right| &= \left|S^{i-1}(\mathcal{G})\right| - \left|S^{i-1}(\mathcal{F})\right| && \text{Property 2} \\ &= \left|S^{i-1}(\mathcal{G})\right| - \left|S^{i-1}(\mathcal{G}')\right| && \text{Property 3} \\ &= e^{i-1}(\mathcal{G}) - e^{i-1}(\mathcal{G}') \ , && \text{induction assumption} \end{aligned}$$

which proves the theorem. □

REFERENCES

AGGARWAL, C. C. AND YU, P. S. 1998. A new framework for itemset generation. In *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. ACM, 18–24.

BAUMGARTNER, C., BÖHM, C., AND BAUMGARTNER, D. 2005. Modelling of classification rules on metabolic patterns including machine learning and expert knowledge. *Biomedical Informatics 38,* 2, 89–98.

BOULICAUT, J., BYKOWSKI, A., AND RIGOTTI, C. 2003. Free-sets: a condensed representation of boolean data for the approximation of frequency queries. *Data Mining and Knowledge Discovery 7,* 1, 5–22.

BRIJS, T., SWINNEN, G., VANHOOF, K., AND WETS, G. 1999. Using association rules for product assortment decisions: A case study. In *Proceedings of the 5th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Diego, CA*. ACM, 254–260.

BRIN, S., MOTWANI, R., AND SILVERSTEIN, C. 1997. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Tucson, AZ*. ACM, 265–276.

CALDERS, T. AND GOETHALS, B. 2006. Quick inclusion-exclusion. In *Proceedings of the ECML PKDD Workshop on Knowledge Discovery in Inductive Databases (KDID)*. Springer, 86–103.

CALDERS, T. AND GOETHALS, B. 2007. Non-derivable itemset mining. *Data Mining and Knowledge Discovery 14,* 1, 171–206.

COVER, T. M. AND THOMAS, J. A. 2006. *Elements of Information Theory*. Wiley-Interscience New York.

COWELL, R. G., DAWID, A. P., LAURITZEN, S. L., AND SPIEGELHALTER, D. J. 1999. Probabilistic networks and expert systems. In *Statistics for Engineering and Information Science*, M. Jordan, S. L. L. nad Jeral F. Lawless, and V. Nair, Eds. Springer-Verlag.

CSISZÁR, I. 1975. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability 3,* 1, 146–158.

DARROCH, J. AND RATCLIFF, D. 1972. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics 43,* 5, 1470–1480.

DE BIE, T. 2011a. An information theoretic framework for data mining. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Diego, CA*. ACM, 564–572.

DE BIE, T. 2011b. Maximum entropy models and subjective interestingness: an application to tiles in binary databases. *Data Mining and Knowledge Discovery 23,* 3, 407–446.

FRANK, A. AND ASUNCION, A. 2010. UCI machine learning repository. http://archive.ics.uci.edu/ml.

GALLO, A., CRISTIANINI, N., AND DE BIE, T. 2007. MINI: Mining informative non-redundant itemsets. In *Proceedings of the European Conference on Machine*

Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Warsaw, Poland. Springer, 438–445.

GARRIGA, G. C., JUNTTILA, E., AND MANNILA, H. 2011. Banded structure in binary matrices. *Knowledge and Information Systems 28,* 1, 197–226.

GEERTS, F., GOETHALS, B., AND MIELIKÄINEN, T. 2004. Tiling databases. In *Proceedings of Discovery Science*. 278–289.

GELMAN, A., CARLIN, J., STERN, H., AND RUBIN, D. 2004. *Bayesian data analysis*. CRC press.

GENG, L. AND HAMILTON, H. J. 2006. Interestingness measures for data mining: A survey. *ACM Computing Surveys 38,* 3, 9.

GEURTS, K., WETS, G., BRIJS, T., AND VANHOOF, K. 2003. Profiling high frequency accident locations using association rules. In *Proceedings of the 82nd Annual Transportation Research Board, Washington DC. (USA), January 12-16*. National Academy, 1–18.

GIONIS, A., MANNILA, H., MIELIKÄINEN, T., AND TSAPARAS, P. 2007. Assessing data mining results via swap randomization. *ACM Transactions on Knowledge Discovery from Data 1,* 3, 167–176.

GOETHALS, B. AND ZAKI, M. 2004. Frequent itemset mining dataset repository (FIMI). http://fimi.ua.ac.be/.

GRÜNWALD, P. 2005. Minimum description length tutorial. In *Advances in Minimum Description Length*, P. Grünwald and I. Myung, Eds. MIT Press.

GRÜNWALD, P. 2007. *The Minimum Description Length Principle*. MIT Press.

HANHIJÄRVI, S., OJALA, M., VUOKKO, N., PUOLAMÄKI, K., TATTI, N., AND MANNILA, H. 2009. Tell me something I don't know: randomization strategies for iterative data mining. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France*. ACM, 379–388.

HEIKINHEIMO, H., FORTELIUS, M., ERRONEN, J., AND MANNILA, H. 2007. Biogeography of European land mammals shows environmentally distinct and spatially coherent clusters. *Journal of Biogeography 34,* 6, 1053–1064.

JAROSZEWICZ, S. AND SIMOVICI, D. A. 2004. Interestingness of frequent itemsets using bayesian networks as background knowledge. In *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Seattle, WA*. ACM, 178–186.

JAYNES, E. 1982. On the rationale of maximum-entropy methods. *Proceedings of the IEEE 70,* 9, 939–952.

KONTONASIOS, K.-N. AND DE BIE, T. 2010. An information-theoretic approach to finding noisy tiles in binary databases. In *Proceedings of the 10th SIAM International Conference on Data Mining (SDM), Columbus, OH*. SIAM, 153–164.

LI, M. AND VITÁNYI, P. 1993. *An Introduction to Kolmogorov Complexity and its Applications*. Springer.

MAMPAEY, M., TATTI, N., AND VREEKEN, J. 2011. Tell me what I need to know: Succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Diego, CA*. ACM, 573–581.

MAMPAEY, M. AND VREEKEN, J. 2010. Summarising data by clustering items. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Barcelona, Spain*. Springer, 321–336.

MITCHELL-JONES, A., AMORI, G., BOGDANOWICZ, W., KRYSTUFEK, B., REIJNDERS, P. H., SPITZENBERGER, F., STUBBE, M., THISSEN, J., VOHRALIK, V., AND ZIMA, J. 1999. *The Atlas of European Mammals*. Academic Press.

MYLLYKANGAS, S., HIMBERG, J., BÖHLING, T., NAGY, B., HOLLMÉN, J., AND KNUU-

TILA, S. 2006. DNA copy number amplification profiling of human neoplasms. *Oncogene 25,* 55, 7324–7332.

NIJSSEN, S., GUNS, T., AND DE RAEDT, L. 2009. Correlated itemset mining in ROC space: a constraint programming approach. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France*. Springer, 647–656.

PASQUIER, N., BASTIDE, Y., TAOUIL, R., AND LAKHAL, L. 1999. Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th International Conference on Database Theory (ICDT), Jerusalem, Israel*. ACM, 398–416.

RISSANEN, J. 1978. Modeling by shortest data description. *Automatica 14,* 1, 465–471.

SCHWARZ, G. 1978. Estimating the dimension of a model. *The Annals of Statistics 6,* 2, 461–464.

SHAFFER, J. 1995. Multiple hypothesis testing. *The Annual Review of Psychology 46,* 1, 561–584.

SIEBES, A. AND KERSTEN, R. 2011. A structure function for transaction data. In *Proceedings of the 11th SIAM International Conference on Data Mining (SDM), Mesa, AZ*. SIAM, 558–569.

SIEBES, A., VREEKEN, J., AND VAN LEEUWEN, M. 2006. Item sets that compress. In *Proceedings of the 6th SIAM International Conference on Data Mining (SDM), Bethesda, MD*. SIAM, 393–404.

SMETS, K. AND VREEKEN, J. 2012. SLIM: Directly mining descriptive patterns. In *Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA*. Society for Industrial and Applied Mathematics (SIAM), 236–247.

TAN, P., KUMAR, V., AND SRIVASTAVA, J. 2002. Selecting the right interestingness measure for association patterns. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Edmonton, Alberta*. ACM, 32–41.

TATTI, N. 2006. Computational complexity of queries based on itemsets. *Information Processing Letters 98,* 5, 183–187.

TATTI, N. 2008. Maximum entropy based significance of itemsets. *Knowledge and Information Systems 17,* 1, 57–77.

TATTI, N. 2010. Probably the best itemsets. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC*. ACM, 293–302.

TATTI, N. AND HEIKINHEIMO, H. 2008. Decomposable families of itemsets. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Antwerp, Belgium*. Springer, 472–487.

TATTI, N. AND MAMPAEY, M. 2010. Using background knowledge to rank itemsets. *Data Mining and Knowledge Discovery 21,* 2, 293–309.

VAN DEN BULCKE, T., VANDEN BROUCKE, P., VAN HOOF, V., WOUTERS, K., VANDEN BROUCKE, S., SMITS, G., SMITS, E., PROESMANS, S., VAN GENECHTEN, T., AND EYSKENS, F. 2011. Data mining methods for classification of Medium-Chain Acyl-CoA dehydrogenase deficiency (MCADD) using non-derivatized tandem MS neonatal screening data. *Journal of Biomedical Informatics 44,* 2, 319–325.

VREEKEN, J., VAN LEEUWEN, M., AND SIEBES, A. 2007. Preserving privacy through data generation. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM), Omaha, NE*. IEEE, 685–690.

VREEKEN, J., VAN LEEUWEN, M., AND SIEBES, A. 2011. KRIMP: Mining itemsets that compress. *Data Mining and Knowledge Discovery 23,* 1, 169–214.

WALLACE, C. 2005. *Statistical and inductive inference by minimum message length*. Springer-Verlag.

WALLACE, C. S. AND PATRICK, J. D. 1993. Coding decision trees. *Machine Learning 11,* 1, 7–22.
WANG, C. AND PARTHASARATHY, S. 2006. Summarizing itemset patterns using probabilistic models. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Philadelphia, PA*. 730–735.
WEBB, G. I. 2007. Discovering significant patterns. *Machine Learning 68,* 1, 1–33.
WEBB, G. I. 2010. Self-sufficient itemsets: An approach to screening potentially interesting associations between items. *ACM Transactions on Knowledge Discovery from Data 4,* 1, 1–20.
ZAKI, M. J. AND RAMAKRISHNAN, N. 2005. Reasoning about sets using redescription mining. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL*. ACM, 364–373.

Received July 2011; revised April 2012; accepted May 2012