Chapter 8
DepMiner: A Method and a System for the Extraction of Significant Dependencies

Rosa Meo[1] and Leonardo D'Ambrosi[2]

[1] University of Torino, Italy
[2] Regional Agency for Health Care Services - A.Re.S.S. Piemonte, Italy

Abstract. We propose *DepMiner*, a method implementing a simple but effective model for the evaluation of itemsets, and in general for the evaluation of the dependencies between the values assumed by a set of variables on a domain of finite values. This method is based on $\varDelta$, the departure of the probability of an observed event from a referential probability of the same event. The observed probability is the probability that the variables assume in the database given values; the referential probability, is the probability of the same event estimated in the condition of maximum entropy.

DepMiner is able to distinguish between dependencies among the variables intrinsic to the itemset and dependencies "inherited" from the subsets: thus it is suitable to evaluate the utility of an itemset w.r.t. its subsets. The method is powerful: at the same time it detects significant positive dependencies as well as negative ones suitable to identify rare itemsets. Since $\varDelta$ is anti-monotonic it can be embedded efficiently in algorithms. The system returns itemsets ranked by $\varDelta$ and presents the histogram of $\varDelta$ distribution. Parameters that govern the method, such as minimum support for itemsets and thresholds of $\varDelta$ are automatically determined by the system. The system uses the thresholds for $\varDelta$ to identify the statistically significant itemsets. Thus it succeeds to reduce the volume of results more then competitive methods.

1 Introduction

In statistics, machine learning and data mining the problem of the determination of set of variables whose values are correlated represents an important knowledge for the user in many fields such as in feature selection, database design and schema reverse engineering, market basket analysis, information retrieval, machine translation, biology, etc. Often in the scientific literature, the study of the dependence between variables is limited to pairs [9,19]. Much previous research focused on finding correlated pairs but finding correlations among more than two variables is essential for problems in many commercial and sociological studies (e.g., for collaborations and interaction networks), medical and biological (e.g., interaction among drugs and proteins) and scientific domains. Thus, instead of correlated items we should find correlated itemsets in which all items are correlated with each other.

In practical cases, often it happens that the set of returned itemsets is large and much of the information is redundant because many itemsets are returned together with many

D.E. Holmes, L.C. Jain (Eds.): Data Mining: Found. & Intell. Paradigms, ISRL 23, pp. 209–222.
springerlink.com

of their subsets. The attempt to reduce the redundancy in the result set answers to two major challenges in frequent-pattern mining: the first is to reduce the often overwhelming size of the mining results and the other is to eliminate redundancy in the information content and the overlapping between the itemsets.

Deciding which itemsets are redundant is not easy and straightforward. It might depend on the applications. For instance, the inclusion in the set of more itemsets with some common items could be acceptable because the itemsets might have different meaning. Instead, the inclusion in the result of both subsets and their supersets is not acceptable if the supersets do not add new information to the information carried by the subsets. In literature, redundant itemsets are detected in many different and sometimes opposite ways. For instance, [25] considers the correlation among the items as strong only when all the items are considered together and when removing any items greatly reduces the correlation. Therefore, the subsets of these itemsets must have instead a weak correlation. On the opposite side, [5] considers interesting an itemset if all its subsets are closely related to all other subsets, and no irrelevant items can be removed. This kind of approach is often adopted in feature selection by step-wise, forward methods [6,12].

In data mining there exist computationally efficient methods to discover significant dependencies and correlations among the items of a frequent pattern [1,3,16]. In order to determine the dependencies in k-itemsets with $k > 2$, either they make the multi-way independence assumption or they evaluate the contribution to the overall itemset of each variable separately [7,23,25]. The difficulty stems from the fact that there is not an easy way to determine a referential probability of an itemset I that represents a condition of independence among the subsets if we do not suppose independence among all the single variables in I. But the multi-way independence condition gives a problem: according to this definition of independence, if a dependence already exists in a subset of I, this dependence is "inherited" from the subset to I and to all the supersets of I [3]. Thus we do not have a way to distinguish if an intrinsic dependence exists in an itemset I in addition to the dependencies inherited from its subsets.

We can solve the problem in terms of quantity of information that an itemset provides: we are interested only in itemsets that add any information to their subsets. If instead, an itemset can be foreseen given the observation of its subsets, it does not carry any new information in addition to the subsets; therefore it can be considered as redundant and it is not interesting. We proposed in [13] a solution based on the maximum entropy. The entropy of an itemset I is computed by an estimation of the probability of I computed on the basis of the probability of its subsets. The probability of I at which the entropy is maximum (denoted by $P_E(I)$) corresponds to the probability that the itemset I would have in the condition in which it carries the maximum amount of information in addition to its subsets. The interest measure that we proposed for an itemset I is the departure of the probability of I w.r.t. the referential value computed at maximum entropy: $\Delta(I) = P(I) - P_E(I)$. The more the departure between the two probabilities, the less the itemset can be correctly foreseen from the observation of its subsets. This departure identifies a dependence between the items and tells us that this dependence is not due to the subsets only. As a consequence the itemset represents a non redundant itemset that must be included in the result. In Section 3 we summarize how Δ is computed.

$\Delta(I)$ decreases with the increase in the cardinality of itemsets. As a consequence, Δ is not a suitable measure to compare itemsets of different cardinality. For this purpose in this chapter we propose Δ_n, a version of Δ normalized w.r.t. the probability of the itemset:

$$\Delta_n(I) = \frac{P(I) - P_E(I)}{P(I)}$$

$\Delta_n(I)$ takes both positive and negative values, in the range from $[-\infty, 1]$. Specifically, if the value is positive, it means a positive dependence, i.e., an itemset that is more frequent than expected; if the value if negative it means a negative dependence, i.e., an itemset that occurs rarer than expected.

In this chapter we present a method for the computation of the interesting and non redundant itemsets based on the above observations. $\Delta_n(I)$ is used as a score function to rank the itemsets. In Section 4 we show how we succeeded to determine the significance level of $\Delta_n(I)$ and to point out to the user the significant itemsets.

Δ computation occurs from an initial, intermediate result composed of frequent itemsets. Another contribution of this chapter is to show how the minimum frequency threshold can be set without the explicit intervention of the user. In fact, the determination of the frequency threshold is well-known to be difficult for the user.

The rest of the chapter is organized as follows. In Section 2 we review the related works. In Section 3 we summarize how Δ is computed. Section 4 shows how to determine the significance level of $\Delta_n(I)$. Though $\Delta_n(I)$ does not satisfy the anti-monotonicity property, in Section 5 we prove an alternative property that guarantees that $\Delta_n(I)$ can be computed efficiently in algorithms as it were anti-monotone. In Section 6 we show how the system determines the minimum support threshold. Section 7 describes the system implementation and the computation flow. Section 8 presents an empirical evaluation study on the results of the system on some common datasets. In this Section DepMiner is compared with other methods for the evaluation of the itemsets, such as [4,7]. The obtained results show that DepMiner identifies the dependencies overcoming the restrictive assumptions of multi-way independence and that the identified significant itemsets are a little portion of the results returned by the other methods. This means that DepMiner is able to compress much without discarding any significant itemsets. Finally, Section 9 draws the conclusions.

2 Related Work

In statistics the problem of the determination of dependent variables is a classical, traditional problem and it has been solved in many ways. The most common approaches are derived by the statistical hypothesis tests such as the tests based on the χ^2 statistics, the Fisher's exact tests [22] and Likelihood-ratio tests [5].

[9] is a deep study on the association between categorical variables and proposes a survey on the measures of association between variables. In machine learning the discovery of dependencies in a multivariate problem (structure learning) is solved by application of neural networks or Bayesian learning methods like MCMC [2] which are NP-hard problems in the number of variables.

In data mining there exist methods to discover significant dependencies and correlations between the items of a frequent pattern with computationally efficient algorithms. [3] is one of the first attempts to discover significant association rules (called dependence rules) by means of the χ^2 test. They point out that the satisfaction of the dependence condition is down-ward closed, i.e., it is monotonic. In other words, once that the dependence is raised in an itemset it will be raised in all the supersets of the itemset. Thus, if the test on the existence of dependencies is checked by a test based on χ^2, the test is not sensible to the addition of further items to the initial dependent set, either the items are independent or not. Thus, χ^2 does not appear to be a suitable measure to determine the effective contribution of an item to the dependence.

[1] proposes collective strength as an itemset interest measure. Collective strength makes use of the ratio between the probability that the itemset appears w.r.t. the expected one under the hypothesis of multiway independence among the items. In turn this ratio is compared with the analogous ratio computed between the probability that at least one item violates the itemset and the expected probability of the same event under the condition of independence. There is violation of the itemset when there is at least one of its items that appears separately w.r.t. the other items in the set.

[16] proposes other measures based on the minimum and maximum confidence that would be obtained by the association rules generated from the itemset. Furthermore, it proposes bond as a further, more restrictive measure defined as the ratio of the number of occurrences of the itemset and of any of its items. Notice, that, in order to determine the dependencies in k-itemsets with $k > 2$, the most of these approaches make the multi-way independence assumption or they evaluate the contribution to the overall itemset of each variable separately.

[7] ranks the frequent itemsets according to the unlikelihood that they appear under the hypothesis that all the items in the itemset are independent. [23] has the aim to rank the most relevant itemsets (by a suitable measure application dependent) and selects the significant portion of the ranking based on measures of lack of redundancy. According to this study, significance and relevance are strictly connected and are combined in a unique measure.

[4] is one of the most well-known studies that allows the reduction of the number of the itemsets in the result; it allows a lossless compression of the result because all the information on the frequent itemsets lacking from the result can be restored from the result set. A similar aim as regards to the reduction in the number of returned itemsets is proposed by [18] with a criterion based on minimum description length. Each itemset is used to represent the portion of the database in which it occurs and therefore it compresses it. Then, the resulting set of itemsets is interesting if it yields a good and lossless compression of the database. Another work on pattern summarization is [24] in which the authors define a proximity metric between the patterns according to the overlapping between the portions of database in which they occur.

[25] proposes the use of multi-information, an extension of mutual information to more than two variables. It considers an itemsets as correlated only if multi-information is higher than a given threshold and if any proper subset has instead a weak amount of multi-information. In multi-information the contribution of each single item to the itemset is considered. A similar approach is proposed for feature selection in classification

by [6,12]. On the opposite side, [11] proposes to use entropy and the quantity of information lead by a set of features for the identification of the set of k features that are maximally independent. Maximum entropy is seen as a guarantee of lack of redundancy in the set. The aim of this kind of itemset is at optimising the independence of items within the set. According to this approach, any single feature is added to the set only if it provides an additional distinctive power.

On the contrary, [5] proposes the following criterion of "fully-correlation" for the determination of the interest of an itemset: an itemset is fully-correlated if all its subsets are closely related to all other subsets, and no irrelevant items can be removed from the set. As regards to the selection criterion, it selects only the maximal fully-correlated itemsets in the following way: if there is no other item that can be added to the itemset to generate a new fully-correlated itemset, then the itemset is maximal fully-correlated.

[10] deals with diversity measures used as heuristic measures of interest for ranking summaries generated from a single dataset. Summaries are composed by a set of attribute-value pairs where attributes can be generalized to many levels of granularity according to taxonomic hierarchies.

A similar approach to the method proposed here and based on Δ, is proposed in [20] in which the analysis is based on K-L divergence.

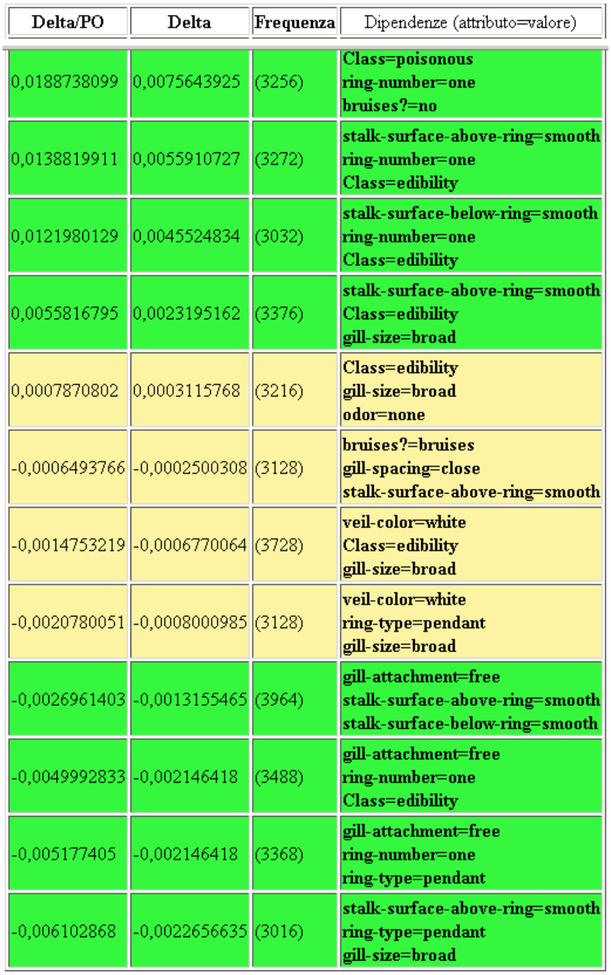
3 Estimation of the Referential Probability

Suppose itemset $I = \{i_1, i_2, \ldots, i_k\}$.

Entropy $H(I) = -\sum P(i_1^*, i_2^*, \cdots, i_k^*) \log[P(i_1^*, i_2^*, \cdots, i_k^*)]$ where we denote by i_j^* the item i_j taken affirmed or negated. Summation ranges over the probabilities of all the combinations of the k items taken affirmed or negated. $H(I)$ is not computed by assumption that singletons are independent but taking in consideration the actual probability of occurrence of each subset of I, as observed from the database. The exclusion-inclusion principle [4] is adopted to compute the entropy of I starting from the probability of the subsets of I. Thus, if the dependence in an itemset I is intrinsic, due to the synergy between all its items, then its probability departs with respect to its estimate given only on the basis of the observed probabilities of its subsets. As a result, thanks to $\Delta_n(I)$ we make emerge the intrinsic, actual dependencies, existing among all the items in I.

4 Setting a Threshold for Δ

Another problem that we have to solve is how large must be Δ_n such that an itemset is deemed significant. We use a null model in which there are not dependencies between the variables. The null model is generated empirically via a randomization of the original dataset. Randomization is generally accepted as a way to allow a statistical test on significance of results [8]. Randomization occurs by independently shuffling the variable values among the examples. As a result, the new dataset will have the same marginal probabilities of the single variables but the dependencies between them are spoiled.



Delta/PO	Delta	Frequenza	Dipendenze (attributo=valore)

0,0188738099	0,0075643925	(3256)	Class=poisonous
			ring-number=one
			bruises?=no

0,0138819911	0,0055910727	(3272)	stalk-surface-above-ring=smooth
			ring-number=one
			Class=edibility

0,0121980129	0,0045524834	(3032)	stalk-surface-below-ring=smooth
			ring-number=one
			Class=edibility

0,0055816795	0,0023195162	(3376)	stalk-surface-above-ring=smooth
			Class=edibility
			gill-size=broad

0,0007870802	0,0003115768	(3216)	Class=edibility
			gill-size=broad
			odor=none

-0,0006493766	-0,0002500308	(3128)	bruises?=bruises
			gill-spacing=close
			stalk-surface-above-ring=smooth

-0,0014753219	-0,0006770064	(3728)	veil-color=white
			Class=edibility
			gill-size=broad

-0,0020780051	-0,0008000985	(3128)	veil-color=white
			ring-type=pendant
			gill-size=broad

-0,0026961403	-0,0013155465	(3964)	gill-attachment=free
			stalk-surface-above-ring=smooth
			stalk-surface-below-ring=smooth

-0,0049992833	-0,002146418	(3488)	gill-attachment=free
			ring-number=one
			Class=edibility

-0,005177405	-0,002146418	(3368)	gill-attachment=free
			ring-number=one
			ring-type=pendant

-0,006102868	-0,0022656635	(3016)	stalk-surface-above-ring=smooth
			ring-type=pendant
			gill-size=broad

Fig. 1. Screen-shot: itemsets ranking with significant itemsets in green (*Mushroom*)

Broadly speaking, as a successive step (without discussing the optimizations that will be described in Section 5), we compute itemsets both from the real and the randomized data. (Figure 4 shows the computation flow described in detail in Section 7). Next, we compute the minimum negative value of Δ_n and the maximum positive value of Δ_n in randomized data. Then, we will use the minimum value of Δ_n in randomized data as an upper bound (denoted by UB) for rarer itemsets in real data and use the maximum value of Δ_n in randomized data as a lower bound (denoted by LB) for the more frequent itemsets in real data. This is a a sort of statistical test on Δ_n and accept as dependent an itemset if its Δ_n is higher (resp. lower) than the maximum (resp. minimum) Δ_n of the itemsets extracted from the randomized data.

An intuition behind the statistical test is the following. Given the high number of itemsets extracted from randomized data (usually, in the number of hundreds), if none of the itemsets has reached a so high (resp. low) value of Δ_n it means that it is unlikely that the observed value (and the correspondent itemset) occurs for chance in real data - as it happens indeed in randomized data. Otherwise, if the itemset had occurred for chance also in real data, then the observed values of Δ_n, in real and randomized data, would have been much more similar! If instead a value of Δ_n occurred in real data so departed from the observed values in random data, this constitutes an evidence of the fact that the itemset did not occur for chance. Therefore, an intrinsic dependence exists in that itemset and makes it emerge above the others. Thus the maximum value of Δ_n observed in randomized data constitutes a lower bound of accepted values in real data. Similarly, for the minimum (negative) value.

Consider the dataset *Mushroom*. After randomization, we observed the maximum value of $\Delta_n = 0.04$ while the minimum value is $\Delta_n = -0.03$. In real data, the maximum is $\Delta_n = 0.85$ and the minimum is $\Delta_n = -0.45$. Thus it is evident that in *Mushroom* the positive dependencies are more abundant and more marked while the negative dependencies are few and less evident. In Figure 1 we show one screen shot of our system prototype, *DepMiner*, with the ranking of itemsets extracted from *Mushroom*. In green the significant itemsets are shown (i.e., the itemsets with a value of Δ_n exceeding the observed range of values in randomized data). In yellow, instead, the other itemsets. Notice that both rarer and more frequent itemsets are interesting. Thus, DepMiner is not biased toward frequent or rare itemsets as it happens for many other systems of pattern mining.

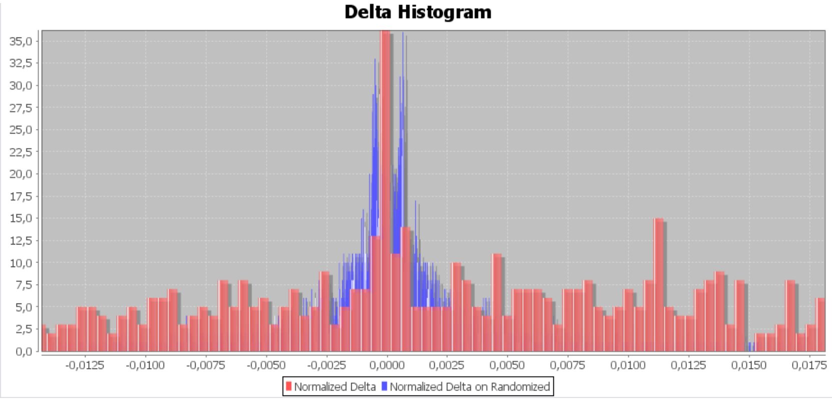


Fig. 2. Histograms of Delta on *Mushroom* (in red) and on its randomization (in blue)

5 Embedding Δ_n in Algorithms

A third problem is how to embed Δ_n in the algorithms. In fact Δ_n does not satisfy an anti monotonicity property that is useful to make efficient the exploration and pruning of the search space of the itemsets. We solved the problem by discovery of the following property.

Theorem 1. *Let $minsup$ be the minimum frequency threshold set in the FIMI (Frequent Itemset Mining) algorithm. Let Δ_{nu} denote the upper bound of Δ_n from the randomized data and Δ_{nl} the lower bound.*

$LB = \Delta_{\mathrm{nu}} \cdot \mathrm{minsup}$ is the minimum threshold for positive dependencies while $UB = \Delta_{\mathrm{nl}} \cdot \mathrm{minsup}$ is the maximum threshold for negative dependencies.

While traversing deeper the item-trie containing candidate itemsets it is sufficient to compute Δ_n for an itemset I if:

$\Delta(I) > LB$ or $\Delta(I) < UB$. Otherwise, we can prune I and its children from the item-trie.

Proof. Let be C a child of itemset I. From [4,17] it results that $\Delta(I) >= \Delta(C)$ since Δ is contained in the range of values of probabilities of any non derivable itemset (if it was derivable it had $\Delta = 0$).

$\Delta(I) < LB$ can be rewritten as $\Delta(I) = \Delta_n(I) \cdot P(I) < LB = \Delta_{nu} \cdot \mathrm{minsup}$. We obtain $\Delta_n(I) < \Delta_{\mathrm{nu}} \cdot \frac{\mathrm{minsup}}{P(I)}$. Since $P(I) >= \mathrm{minsup}$ (otherwise we would have pruned earlier I from the item-trie) it results that $\Delta_n(I) < \Delta_{\mathrm{nu}}$. As a consequence, I can be pruned. In addition, since $\Delta(C) <= \Delta(I) < LB$ we can prune C too. Similar reasoning applies to the other bound.

6 Determination of the Itemsets Minimum Support Threshold

The new model for the determination of the itemsets minimum support allows the replacement of the minimum support threshold imposed as a requirement by the user. As we know, that the minimum support threshold presents some drawbacks due to the fact that a fixed and unique value of support threshold for all the sets (that does not depend on the cardinality of the sets or on their probability density function) is not realistic. Another problem is the fact that the support threshold is given by the user who may not know how to set it. On the contrary, he/she may know how greater is the accuracy of the measure (or the error he/she wants to allow in the inference of the probability of the itemset from the sample database). Therefore, in DepMiner we allow the user to set a different minimum threshold of probability for each itemset: this decision is taken on the basis of the estimated probability of occurrence of the itemset (from the principle of maximum likelihood) and of an error tolerance in this estimation given by the user.

We judge the relevance of the information obtained from the database with a criterion based on the Bayes' Theorem. This one allows us to make an estimate of the probability distribution function of an itemset (a priori probability) starting from the set of observations obtained by the sample database (a posteriori probability). Our criteria is very simple. We consider the probability of an itemset a random variable and make an estimate of this probability on the basis of the observations (obtained a posteriori) from the database. If the most likely value of probability of occurrence of an itemset in the database gives values whose confidence interval width is comparable with the error allowed by the user, we can conclude that the probability estimation is not reliable.

In [15] we proved that this reliability property is, as the itemset support, a property that is anti-monotone, and allows us to stop the lattice traversal in depth, in practice in an equivalent manner as the itemset support.

We apply the theory for the inference of the proportion $\frac{K}{N}$ that gives the more likely value of the probability of an itemset. Since in data mining the sample size is always big, we can approximate the binomial distribution, that is the probability distribution function for the proportion estimation, with a normal distribution with mean $\frac{K}{N}$ and variance $\frac{K}{N}(1-\frac{K}{N})$. The theory of the confidence interval for the proportion gives the following formula:

$$p_o - Z\sqrt{\frac{p_o(1-p_o)}{N}} \leq p \leq p_o + Z\sqrt{\frac{p_o(1-p_o)}{N}} \tag{1}$$

where we denote by p the real probability of the itemset, by p_o the observed proportion (a posteriori observation) that constitutes the estimation on the sample of this probability, by N the sample size and by Z the critical value in the normal distribution corresponding to the confidence level imposed by the user (usually denoted by α). The usual values of Z are 1.96 or 2.58 corresponding to a probability of making an erroneous inference on the proportion with a dataset composed by a collection of random and independent samples equal respectively to 0.05 and 0.01. From the theory, we have that W, the width of the confidence interval, is the maximum error in the estimation of the proportion, and is given by:

$$W = 2Z\sqrt{\frac{p_o(1-p_o)}{N}} \tag{2}$$

Since we allow the user to set both Z and the maximum relative error in the probability estimation that she wants to allow, the relative error is given by $e_r = \frac{W}{p_o}$ and results equal to:

$$e_r = \frac{2Z}{\sqrt{N}}\sqrt{\frac{(1-p_o)}{p_o}} \tag{3}$$

Range of observable probabilities with a relative error. e_r is a monotonic decreasing function with the observable probabilities p_o. It means that we can set a certain value of e_r that is the error tolerance the user wishes to allow and we can set the confidence level in the inference of the proportion, that determines a critical Z value (usually set to $Z = 2.58$ corresponding to $\alpha = 0.01$). Given N, the number of samples that is fixed by the given database, the probabilities that are observable from the database within these constraints are higher than a threshold value, given by the diagram in Figure 3. It plots the lower bound to the observable probabilities in datasets of given size N, in correspondence to different values of e_r. It is evident that this methodology of setting the minimum support threshold is statistically reliable and provides the user a guide to set this minimum value that usually is difficult to set. Above this support limit the observable probabilities are statistically reliable: it means that the probability of making an error greater than the error tolerance in this inference is controlled and it is lower than the confidence level. Furthermore, the estimation error is within the user established error tolerance. On the contrary, under this support limit, the estimation of itemsets probabilities is too risky and subjected to a too higher error (outside of the error tolerance).

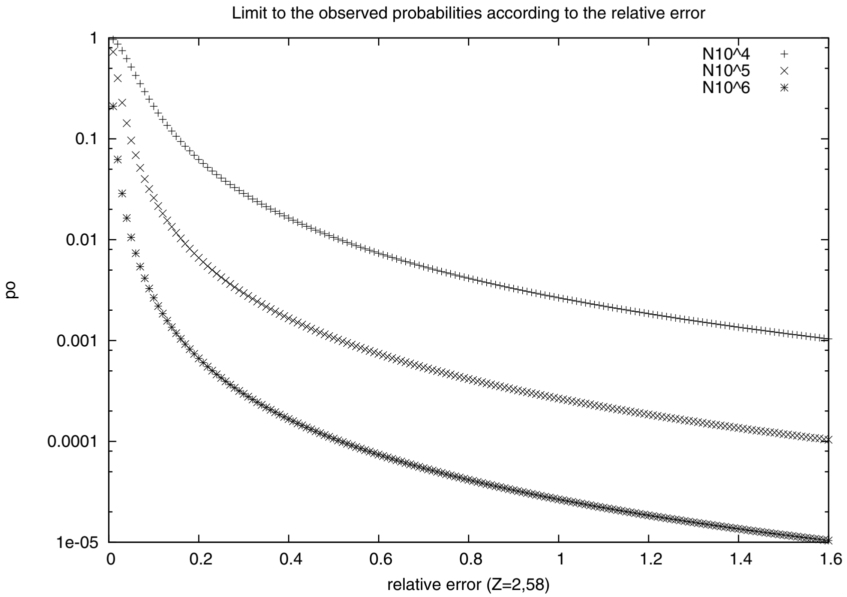


Fig. 3. Observable probabilities by relative error

7 System Description

DepMiner is implemented in java (1.6.0.12) and runs on a laptop. It uses Apache POI HSSF library for I/O. The core of the algorithm for frequent itemsets extraction is LCM FIMI algorithm (4.0) [21], the winner of the FIMI'04 competition. This algorithm is treated as a black-box and could be substituted by any other algorithm supporting the same I/O format. Figure 4 shows the computation flow of the system. Since, FIMI algorithm is treated as a black box, for this reason it is represented as a grey box in the flow. The algorithm for the computation of Δ_n performs the following tasks.

1. sets the *minsup* threshold for FIMI algorithm according to the explanation of Section 6;
2. randomizes the database;
3. runs the FIMI algorithm on the randomized database and reads its result;
4. builds the item-trie from the result of FIMI algorithm;
5. explores the item-trie in a level-wise fashion and computes for each itemset Δ. Δ computation is implemented in java by the algorithm described in [14]);
6. for each itemset, it computes Δ_n and stores the lower and upper bound for Δ_n found (LB and UB);
7. repeats steps 3-5 for the real database; for step 5 it performs pruning of itemsets by enforcing anti-monotonicity of Δ (it prunes if Δ is in the range (LB,UB));
8. enforces the property of Δ_n seen in Section 5 and computes Δ_n only if allowed (if Δ is outside of (LB,UB));
9. produces the itemsets ranking on the basis of Δ_n.

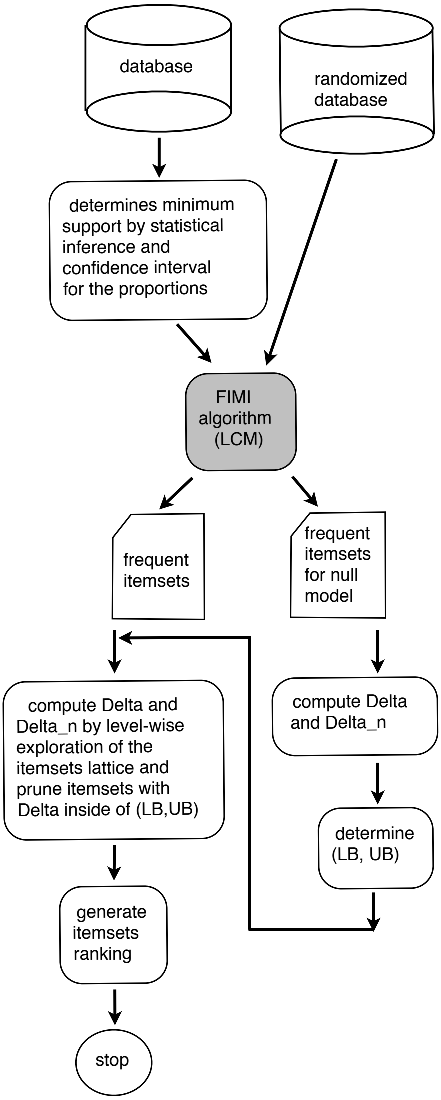

database
randomized database
determines minimum support by statistical inference and confidence interval for the proportions
FIMI algorithm (LCM)
frequent itemsets
frequent itemsets for null model
compute Delta and Delta_n by level-wise exploration of the itemsets lattice and prune itemsets with Delta inside of (LB,UB)
compute Delta and Delta_n
determine (LB, UB)
generate itemsets ranking
stop

Fig. 4. The computation flow

The output of the itemsets ranking is implemented as a web page in HTML. GUI is implemented on JFreeChart, an open source library in java for the rendering of graphics and diagrams. It allows also to order the list of the itemsets by different criteria in an interactive way, such as by items, or by ascending/descending values of Δ_n: this is useful for the user to explore the results, according to her/his desire to observe the rarer itemsets (with a negative Δ_n) or the more frequent ones (with a positive Δ_n).

In Figure 2 we present another screen-shot of *DepMiner*. It shows in red color the histogram of Δ_n on real data and in blue the histogram of Δ_n on randomized data.

The user can zoom on specific areas of the histogram and observe in more detail the characterization of the dependencies existing in the dataset by the distribution of $\varDelta_n$. It is instructive to observe the different distributions obtained in sparse and dense data. Usually dense data have higher values of $\varDelta_n$, while in sparse data $\varDelta_n$ values are lower and more scattered.

8 Experimental Evaluation

We run a set of experiments on 5 real datasets (from FIMI and UCI Machine Learning repositories) and on 2 real datasets coming from NASDAQ stock exchange index (from January 2001 to May 2009) and from the Italian lottery (with data on the numbers drawn from 1939). The lottery dataset is important in order to check the behavior of $\varDelta_n$ on complete random data where even the marginals were uniform. Experiments were run on a CPU Intel Core 2 Duo T8100, 3GB RAM, SO Win Vista Business (SP1).

In Table 1 we include the total number of examples, $minsup$ threshold adopted by LCM, the total number of itemsets generated (N), execution times to compute $\varDelta_n$ (in seconds).

We performed two experiments: the first one on the compression capability and the second one on the capability of DepMiner to determine the dependencies in contrast to methods that assume the multi-way independence condition.

1. In this experiment, to be further conservative, we compare DepMiner results at many levels. We denote as itemsets clearly non independent, the itemset whose $\varDelta_n \neq 0$. Thus we include in the results both these latter ones and the itemsets whose $\varDelta_n$ is acceptable by the significance test on the lower and upper bounds obtained in randomized data. We include in Table 1 three ratios: the ratio between the number of itemsets with $\varDelta_n \neq 0$ and N (denoted by `Dep/N`), the ratio between the significant dependencies and N (denoted by SDep/N) and the ratio between non derivable itemsets (NDI) obtained by the competitor method [4] and N (denoted by NDI/N).

   These ratios quantify the volume of found dependencies in data. They clearly demonstrate the increased ability of DepMiner to reduce redundancies in the result than NDI.
2. In the second experiment we compare the results of DepMiner with MINI [7], the second competitor that we adopted in order to determine the difference between

Table 1. Experimental results

dataset	minsup	itemsets (N)	Dep/N	SDep/N	NDI/N	time(s)	γ(DM,MINI)	γ(RDM,RMINI)
Accidents	35%	65,500	13.5%	0.1%	22.93%	4294	-0.84	0.16
Chess	75%	20,582	1.2%	0.34%	2.11%	135	-0.95	-0.91
Nasdaq	0.14%	242	95.8%	46.69%	100%	107	-0.05	0.27
Kosarak	1.01%	21,934	82.5%	10.39%	95.55%	2221	-0.56	0.28
Mushroom	22.15%	14,189	1.48%	1.03%	5.84%	115	-0.94	-0.29
Retail	4.53%	22,524	79.7%	5.9%	99.56%	1322	0.02	0.55
Lottery	0.006%	91,499	99.1%	0%	100%	5804	0.81	0.77

DepMiner and another method of ranking based on the multi-way independence assumption.

The last columns of Table 1 report the result of a comparison between DepMiner ranking (denoted by DM) and MINI.

As said, our method considers only intrinsic dependencies in the itemsets and makes an estimate of independence of the items by means of the maximum entropy: it corresponds to a condition of minimum amount of information on the items given the knowledge about the other items in the itemset. The adopted referential probability for itemset I coincides with the hypothesis of independence of the singletons only if I has cardinality 2.

In order to measure the correlation between our ranking (DepMiner) and MINI's we adopted an objective measure, known as γ [9]: $\gamma = \frac{n_c - n_d}{n_c + n_d}$. n_c denotes the number of itemsets pairs on which the methods agree (are ranked in the same way by both of them) while n_d is the total number of pairs for which the methods disagree. γ ranges in $[-1, +1]$ and is 0 if there is independence.

Since the methods differ in the referential probability estimate, γ will quantify the impact of this difference. The difference is in the fact that MINI tends to observe an increased amount of dependencies in itemsets due to the fact that it considers also dependencies inherited by a subset to all its supersets.

In Table 1 we also compared the two rankings computed on randomized data (denoted by RDM and RMINI). We can notice that all the values reported by γ denote disagreement and generally have low values. The amount of discrepancies decreases (γ increases) if we move from real data to randomized data (since the high-order dependencies are spoiled during randomization). Furthermore, on complete random data (*Lottery*) the two methods agree ($\gamma = 0.8$) since DepMiner agrees on the hypothesis of independence among the singletons! In addition, we do not observe any significant change in γ if we randomize *Lottery*.

9 Conclusions

We have presented *DepMiner*, a method for the extraction of significant dependencies between the values assumed by database variables. We quantify the volume of these dependencies by the histogram of $Delta$. DepMiner gave good results by comparison of the rankings with [7] by γ and of its capability to compress results with NDI [4].

In DepMiner the user can set the parameter values guided by the system, explore the results in an interactive way, change the itemsets ranking criteria and zoom details in the statistics reports on the dependencies.

DepMiner web site is: http://www.leodambrosi.it/depminer/. From the site it is possible to download a presentation video.

References

1. Aggarwal, C.C., Yu, P.S.: A new framework for itemset generation. In: Proc. PODS (1998)
2. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer Science (2006)

3. Brin, S., Motwani, R., Silverstein, C.: Beyond market baskets: Generalizing association rules to correlations. In: Proc. SIGMOD (1997)
4. Calders, T., Goethals, B.: Non-derivable itemset mining. Data Min. Knowl. Discov. 14(1) (2007)
5. Duan, L., Street, W.N.: Finding maximal fully-correlated itemsets in large databases. In: IEEE International Conference on Data Mining, pp. 770–775 (2009)
6. Fleuret, F.: Fast binary feature selection with conditional mutual information. Journal of Machine Learning Research 5, 1531–1555 (2004)
7. Gallo, A., Bie, T.D., Cristianini, N.: Mini: Mining informative non-redundant itemsets. In: PKDD (2007)
8. Gionis, A., Mannila, H., Mielikäinen, T., Tsaparas, P.: Assessing data mining results via swap randomization. In: Proc. KDD (2006)
9. Goodman, Kruskal: Measures of association for cross classifications. J. Amer. Stat. Ass. 49(268) (1954)
10. Hilderman, R.J., Hamilton, H.J.: Measuring the interestingness of discovered knowledge: A principled approach. Intell. Data Anal. 7, 347–382 (2003)
11. Knobbe, A.J., Ho, E.K.Y.: Maximally informative k-itemsets and their efficient discovery. In: KDD, pp. 237–244 (2006)
12. Liu, Z.Z.H.: Searching for interacting features. In: The 20th International Joint Conference on AI, IJCAI 2007 (2007)
13. Meo, R.: Theory of dependence values. TODS 45(3) (2000)
14. Meo, R.: Maximum independence and mutual information. TOIT 48(1) (2002)
15. Meo, R., Ienco, D.: Replacing support in association rule mining. In: Sing, Y., Rountree, N. (eds.) Rare Association Rule Mining and Knowledge Discovery: Technologies for Infrequent and Critical Event Detection. IGI Global publisher (2008)
16. Omiecinski, E.: Alternative interest measures for mining associations in databases. TKDE 15(1) (2003)
17. Savinov, A.: Mining dependence rules by finding largest support quota. In: Proc. SAC (2004)
18. Siebes, A., Vreeken, J., van Leeuwen, M.: Item sets that compress. In: SDM (2006)
19. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: Proc. KDD (2002)
20. Tatti, N.: Maximum entropy based significance of itemsets. In: Proc. ICDM (2007)
21. Uno, T., Asai, T., Uchida, Y., Arimura, H.: Lcm v2. In: FIMI 2004,
22. Webb, G.I.: Discovering significant rules. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 434–443 (2006)
23. Xin, D., Cheng, H., Yan, X., Han, J.: Extracting redundancy-aware top-k patterns. In: KDD (2006)
24. Xin, D., Han, J., Yan, X., Cheng, H.: Mining compressed frequent-pattern sets. In: In VLDB, pp. 709–720 (2005)
25. Zhang, X., Pan, F., Wang, W., Nobel, A.B.: Mining non-redundant high order correlations in binary data. PVLDB 1(1) (2008)