Statistical Supports for Frequent Itemsets on Data Streams

Pierre-Alain Laur[1], Jean-Emile Symphor[1], Richard Nock[1], and Pascal Poncelet[2]

[1] GRIMAAG-Dépt Scientifique Interfacultaire,
Université Antilles-Guyane, Campus de Schoelcher,
B.P. 7209, 97275 Schoelcher Cedex, France
{palaur,rnock,je.symphor}@martinique.univ-ag.fr
[2] LG2IP-Ecole des Mines d'Alès,
Site EERIE, Nîmes, parc scientifique Georges Besse,
30035 Nîmes Cedex, France
pascal.poncelet@ema.fr

Abstract. A statistical technique is developed for estimating the support of itemsets on data streams, regardless of the size of the data stored. This technique, which is computationally ultra fast, does not depend on the algorithm used to build or maintain the itemsets. On frequent itemsets, it allows to maximize either the precision or the recall, as chosen by the user, while it does not damage the other criterion, and may even yield very good F_β-measures. Since the maximization of both criteria is statistically hard, this provides algorithms building frequent itemsets with an efficient alternative to find those that are true frequents, when only a partial storing of the data stream is technically available. Experiments demonstrate the potential of the technique.

1 Introduction

A growing body of works arising from researchers in Databases and Data Mining deals with data arriving in the form of continuous potentially infinite streams, *i.e.* an ordered sequence of item occurrences that arrives in timely manner. Many emerging and real applications generate data streams: trend analysis, fraud detection, intrusion detection, click stream, among others. In fraud detection, data miners try to detect suspicious changes in user behavior [6]. Trend analysis is an important problem that commercial applications have to deal with, which is to detect in the data stream significant trends, emerging buzz, and unusually high or low activity [10]. Security of network systems is becoming increasingly important as more and more sensitive informations are being stored and manipulated online. Intrusion detection has thus become a critical approach to help protect systems [9].

From now on, we consider *items* to be the unit information, and *itemsets* to be sets of items. An itemset is *θ-frequent* if it occurs in at least a fraction θ of the data stream (called its support), where θ is a user-specified parameter.

When it is not needed, we shall simply omit the θ parameter, and simply qualify as "frequent" some itemset. Because it is fast and represent a huge amount of information, the item flow prevents its exact storage in a database. Out of the uncertainty it generates, the problem becomes to store the information so as to keep valid its most crucial contents. One common example of such a content is the list of the most frequent items of itemsets encountered, a crucial issue in Data Mining that has recently attracted significant attention [8, 15, 12, 4, 16, 9, 13]. For example, algorithms concerned with applications such as answering iceberg query, computing iceberg cubes or identifying large networks flows, are mainly interested in maintaining frequent items (rather than directly itemsets).

When the database is subject to be updated regularly, maintaining frequent itemsets has been successfully addressed by various incremental algorithms, in order to mine association rules without having to build repeatedly everything from scratch [2, 23]. Most of these approaches focus on building efficient data structures and/or maintaining generalization. But, due to the high frequency and potentially huge information carried out in a timely fashion by data streams, these incremental approaches cannot easily handle them, unless they take the risk to make errors [22, 7] and/or fail at estimating supports, one of the two essential components of association rules algorithms. This is where our paper takes place.

More precisely, we address the following questions:

(a) is it possible to set up a method which replaces the *exact* support by a *statistical* support ensuring some desirable properties on support computations, and frequency estimations ? Ideally, we would like the resulting support to hold regardless of the algorithm used to build or maintain frequent items/itemsets (see *e.g.* [2, 23]), and rely on mild theoretical assumptions so as to be reliably implementable. For example, common statistical assumptions such as normality, homoscedasticity and others alike are clearly not desirable in our setting.
(b) how good is this statistical support, both from the theoretical and experimental standpoints ?

The rest of this paper is organized as follows. Section 2 goes deeper into presenting the problems of dealing with uncertainty in data streams, and gives an extensive statement of our problem. Section 3 presents our solution to this problem, and its properties. Section 4 presents experimental results, and Section 5 concludes the paper with future avenues for research.

2 Problem Statement

The huge size of data streams for real-world domains compared to the limited amounts of resources to mine them makes it necessary to cope with uncertainty

to achieve reasonable processing time and/or space. A significant body of previous works has addressed the accurate storing of the data stream history. This storage problem consists in finding compact data structures to reduce the size of the data kept out of the stream, while guaranteeing with high probability that the items *observed* as frequent from the data stream are still frequent inside the data structure [1, 3, 12].

Our setting is a bit more downstream, as we question the forecasting on the data stream future. The data stored is used to mine information. Ideally, this information is sought to be accurate not only on the data stored, but also on the whole data stream itself. For example, it is not enough to observe some item as frequent in the data stored; it is much more important (*e.g.* from a business standpoint, when building association rules) to *predict* if it is really frequent in the whole data stream. Similarly, it is not enough to observe that some itemsets does not meet the observed frequency requirements to argue that it is not frequent. It is much more important to be able to conclude whether it is *really* not frequent on the whole data stream.

In this paper, we consider itemsets rather than simple items [12] to be build and stored out of the data stream. This makes the forecasting problem even more pregnant, as there is a natural generalization/specialization relationship between itemsets which does not exist between ordinary items, and this relationship has a great impact on estimating frequencies.

From the estimation standpoint, there are two sources of error:

1. it is possible that some itemsets observed as frequent might in fact not be frequent anymore from a longer observation of the data stream, even without a drift of the observation odds;
2. on the other hand, some itemsets observed as not frequent may well in fact be frequent from a longer history of the data stream.

Should it rely on frequencies estimations, any loss (money, business opportunity, etc.) due to the imperfection of the information stored is incurred by at least one of these two sources of error. The point is that it is statistically hard to nullify both of them from a subset, even very large, of the whole data stream [21]. It is also generally impossible to capture the missing informations from the data stream to make a fully accurate prediction. Our paper is aimed at obtaining a solution to the following problem, which is a convenient relaxation of this unsatisfiable goal:

(a) the user chooses a source of error, and fixes some related parameters;
(b) the source of error chosen is nullified with high probability, while the other one incurs a limited loss.

It turns out that in many domains, the relative importance of the two sources of error is not the same, and one may be much more important to control than the other one: fraud detection, trends detection, data privacy, among others [22,

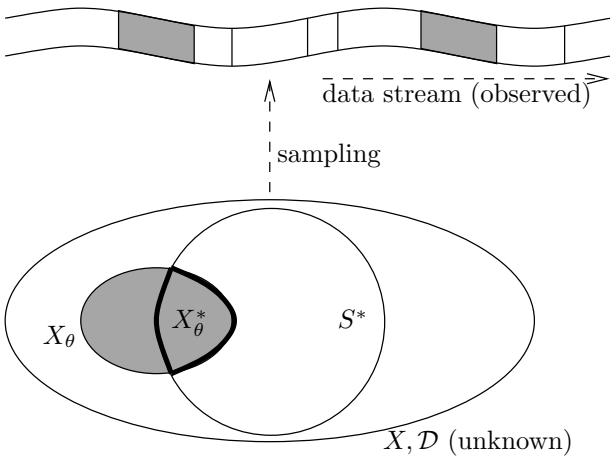

Fig. 1. Sampling the domain X through distribution $\mathcal{D}$ yields the datastream. Our objective is to find the intersection between the subset of X we build (S^*) and the θ-frequents of X (X_θ). This set is called X^*_θ (see text for details).

20]. For these domains, our approach may be a very convenient way to cope with uncertainty in finding frequent itemsets. Obviously, we also want the computational cost of the solution to be affordable for data mining purposes.

Now, let us skip to a slightly more formal presentation. The data stream is supposed to be obtained from the repetitive sampling of a potentially huge *domain* X which contains all possible itemsets, see Figure 1. Each itemset is sampled independently through a distribution $\mathcal{D}$ for which we make absolutely *no* assumption, except that it remains fixed: there is no distribution drift through time. The reader may find relevant empirical studies on concept drift for supervised mining in [6, 24]. The user specifies a real $0 < \theta < 1$, the *theoretical* support, and ideally wishes to recover all the *true* θ-frequent patterns of X, that is, each itemset that generalizes at least a fraction θ of all possible itemsets of X with respect to $\mathcal{D}$. This set is called X_θ in Figure 1, and formally defined below.

Definition 1.

$$\forall 0 \leq \theta \leq 1, X_\theta = \{T \in X : \rho_X(T) \geq \theta\} \ , \tag{1}$$

with $\rho_X(T) = \sum_{T' \in X : T \leq_t T'} \mathcal{D}(T')$*, and* $T \leq_t T'$ *means that* T *generalizes* T'*, according to some accurate user-based definition of generalization.*

The recovery of X_θ faces two problems. Apart from our statistical estimation problem, there is a combinatorial problem which comes from the fact that X is typically huge, even when finite. The set of observed itemsets which we have sampled from X in the data stream, hereafter called S, has a size $|S| = m$ (with $|.|$ denoting the cardinal) which is typically of minute order compared to

$|X|$. In a framework like ours, admitting a natural generalization/specialization relationship between the data stored, we usually reduce this difference with some algorithm returning a superset S^* of S, having size $|S^*| = m^* > m$. Typically, S^* contains additional generalizations of the elements of S [17]. This is not the purpose of this paper to cover the numerous aspects and approaches to this combinatorial problem; the key point is that S^* is usually still not large enough to cover X_θ, regardless of the way it is built (see Figure 1), so that the pregnancy of our statistical estimation problem remains the same. Notice however that it has to be addressed on a potentially very large set S^*, which implies that any solution has to be efficient from both the statistical *and* computational standpoints.

Because we want it to be independent from the algorithm used to build S^* out of S, our statistical estimation problem can be formalized as follows:

- approximate as best as possible the following set:

$$X^*_\theta = X_\theta \cap S^* \ , \tag{2}$$

  for any S and S^* (see Figures 1 and 2).

Remark that $\forall T \in S^*$, we cannot compute exactly $\rho_X(T)$, since we do not know X and $\mathcal{D}$. Rather, we have access to its best unbiased estimator $\rho_S(T)$, which can be easily computed from S:

$$\forall T \in S^*, \rho_S(T) = \sum_{T' \in S : T \leq_t T'} w(T') \ , \tag{3}$$

with $w(T')$ the weight (observed frequency) of T' in S. We adopt the following approach to solve our problem, which is both simple and computationally attractive:

- find some $0 < \theta' < 1$ and approximate the set X^*_θ by the set of *observed* θ'-frequent of S^*, that is:

$$S^*_{\theta'} = \{T \in S^* : \rho_S(T) \geq \theta'\} \ . \tag{4}$$

Before computing θ', we first turn to the formal criteria appreciating the goodness-of-fit of $S^*_{\theta'}$ in regard to our two sources of error. These two sources of error, committed with respect to X^*_θ, come from the two subsets of the symmetric difference with $S^*_{\theta'}$, as presented in Figure 2. To quantify them, let us define:

$$TP = \sum_{T \in S^*_{\theta'} \cap X^*_\theta} \mathcal{D}(T) \ , \tag{5}$$

$$FP = \sum_{T \in S^*_{\theta'} \setminus X^*_\theta} \mathcal{D}(T) \ , \tag{6}$$

$$FN = \sum_{T \in X^*_\theta \setminus S^*_{\theta'}} \mathcal{D}(T) \ , \tag{7}$$

$$TN = \sum_{T \in S^* \setminus (S^*_{\theta'} \cup X^*_\theta)} \mathcal{D}(T) \ . \tag{8}$$

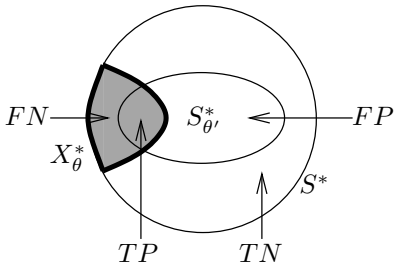

Fig. 2. The set we build to approximate X^*_θ ($S^*_{\theta'}$) contains the **observed** θ'-frequents of S. The error committed with respect to X^*_θ can be appreciated through four sets whose weights are denoted TP, TN, FP, FN (see text for details).

The *precision* allows to quantify the proportion of estimated θ-frequent that are in fact not true θ-frequents, out of $S^*_{\theta'}$:

$$\mathtt{P} = TP/(TP + FP) \ . \tag{9}$$

Maximizing $\mathtt{P}$ is equivalent to minimizing our first source of error. Symmetrically, the *recall* allows to quantify the proportion of true θ-frequent that are missed in $S^*_{\theta'}$:

$$\mathtt{R} = TP/(TP + FN) \ . \tag{10}$$

Maximizing $\mathtt{R}$ is equivalent to minimizing our second source of error. In the experimental section, we shall also make use of a well known quantity in information retrieval, which is a weighted harmonic average of precision and recall, the $\mathtt{F}_\beta$-measure. In this measure, we can adjust the importance of one source of error against the other one:

$$\mathtt{F}_\beta = (1 + \beta^2)\mathtt{PR}/(\mathtt{R} + \beta^2\mathtt{P}) \ , \tag{11}$$

with $\beta > 0$ some user-fixed real. $\beta = 1$ makes the two sources of error equivalent; the first source of error becomes more important as $\beta \to 0$, and the second one becomes more important as $\beta \to \infty$.

Now, let us return to fixing the value of θ'. A naive approach to approximate X^*_θ would typically be to fix $\theta' = \theta$, and thus keep exactly the observed θ-frequent of S^*. Unfortunately, the main and only interesting property of $S^*_{\theta'}$ is that it converges with probability 1 to X^*_θ as $m \to \infty$ from the Borel-Cantelli Lemma [5]. Glivenko-Cantelli's Theorem gives a rate of convergence as a function of m, but this is only useful to yield the maximization of $\mathtt{P}$ and $\mathtt{R}$ in the limit.

The next Section presents our approach to solving our problem.

3 Choosing θ'

Informally, our approach boils down to picking a θ' different from θ, so as to maximize either the precision or the recall, as chosen by the user. Clearly, extremal values for θ' would do the job, but they would yield very poor values for

F_β, and also be completely useless for data mining purposes. For example, we could choose $\theta' = 0$, and would obtain $S^*_0 = S^*$, and thus $\mathtt{R} = 1$. However, in this case, we would also have $\mathtt{P} = |X^*_\theta|/|S^*|$, a too small value for many domains and values of θ, and we would also keep all elements of S^* as true θ-frequents, a clearly huge drawback for mining issues. We could also choose $\theta' = 1$, so as to be sure to maximize $\mathtt{P}$ this time; however, we would also have $\mathtt{R} = 0$, and would keep *no* element of S^* as θ-frequent. Again this would not be very useful for mining issues.

These extremal examples show the principle of our approach. Should we want to maximize the precision, we would pick a θ' larger than θ to guarantee with high probability that $\mathtt{P} = 1$, yet while keeping large enough values for $\mathtt{R}$ (or F_β), and a set $S^*_{\theta'}$ not too small to contain significant informations. There is obviously a statistical barrier which prevents θ' to be too close to θ to keep the constraint $\mathtt{P} = 1$ (*Cf* Section 2, last §). The objective is to be the closest to this barrier, which statistically guarantees the largest recall values under the constraint.

The same principle holds for the maximization of the recall. This time, we pick a θ' smaller than θ, and hopefully close enough to the statistical barrier for θ' under which we have $\mathtt{R} = 1$. This time, we aim at keeping large enough values for the precision, and a set $S^*_{\theta'}$ not too large to be mined efficiently.

The following Theorem states explicitly our bound for the maximal precision. Its key feature is that it holds regardless of the domain, the distribution of the itemsets, the size of S^*, or the user-fixed parameters (support, statistical risk). It relies *only* on a rather mild assumption for sampling the itemsets out of the data stream (namely, its independence).

Theorem 1. $\forall X, \forall \mathcal{D}, \forall m > 0, \forall 0 \leq \theta \leq 1, \forall 0 < \delta \leq 1$, *suppose we pick some* ε *satisfying:*

$$\varepsilon \geq \sqrt{\frac{1}{2m} \ln \frac{|S^*|}{\delta}} \ .$$

If we fix $\theta' = \theta + \varepsilon$ *in eq. (4), then* $\mathtt{P} = 1$ *with probability at least* $1 - \delta$.

Now, we state the equivalent Theorem for the maximal recall.

Theorem 2. $\forall X, \forall \mathcal{D}, \forall m > 0, \forall 0 \leq \theta \leq 1, \forall 0 < \delta \leq 1$, *suppose we pick some* ε *satisfying:*

$$\varepsilon \geq \sqrt{\frac{1}{2m} \ln \frac{|S^*|}{\delta}} \ .$$

If we fix $\theta' = \theta - \varepsilon$ *in eq. (4), then* $\mathtt{R} = 1$ *with probability at least* $1 - \delta$.

These Theorems are proven using standard tools on concentration inequalities [18]; due to the lack of space, we skip their proofs. The main point is that the values of θ' seem to be very close to the statistical barriers [21, 14] that still guarantee the maximal values for the precision or recall.

Database	DB size	Total items	Max. size	Avg. size
Accidents	340183	468	51	34
Retail	88163	16470	76	11
Kosarak	990002	41270	2498	9

Fig. 3. Databases used. For each of them, we give, from left to right, the whole number of transactions of the database, the whole number of items, the maximum size of a transaction, and the average size of a transaction.

4 Experiments

Since our method for statistical supports does not depend on the algorithms used to build the frequent itemsets or the association rules, and since its computational cost is not larger than that of computing ordinary frequencies, we do not evaluate it in terms of speed. Rather, we focus on evaluating how our statistical support can be helpful to mine frequent itemsets on a data stream, given a fragment of this stream.

This experimental Section is split in four subsections: we first explain the measures to evaluate the quality of the frequent itemsets kept. Then, we present the databases used. The third subsection presents the experimental setup, and the last one presents and discusses the results.

4.1 Performance measures

We previously define in eqs. (9), (10) and (11) the following information retrieval measures: the precision (`P`), the recall (`R`) and a weighted harmonic average of both, the F_β measure. These measures are widely spread to evaluate statistical predictive models. We have chosen to use them in our experiments as we thought they are the most reliable to evaluate our method.

4.2 Data used

In order to evaluate our predictive method we have chosen three real life databases from the Frequent itemsets Mining Dataset Repository [11], whose principal goal is to evaluate and compare association rules algorithms. We explain in the experimental set-up subsection below how these sets are used as data streams to evaluate our method; we focus here on the content of these sets, as shown in Figure 3.

The first database of Fig. 3, named "Accidents", is obtained from the National Institute of Statistics (NIS) for the region of Flanders (Belgium) for the period 1991-2000. More specifically, the data are obtained from the Belgian

Database	θ	sampling1	sampling2	δ
Accidents	[.3, .9] / .05	[.01,.1] / .01	[.1, 1] /.03	[.01, .11] / .02
Retail	[.05, .1] / .01	[.01,.1] / .01	[.1, 1] /.03	[.01, .11] / .02
Kosarak	[.05,.1] / .01	[.01,.1] / .01	[.1, 1] /.03	[.01, .11] / .02

Fig. 4. Range of parameters for the experiments. For each parameter, the range of values it takes is given on the form $[a, b]/c$, where a is the starting value, c is the increment, and d is the last value. Thus, the set of values is $\{a, a + c, a + 2c, ..., b\}$. θ is the minimum theoretical support, δ is the risk parameter. The columns "sampling1" and "sampling2" give the two scales of percentages of the database sampled out of the data stream (see text for details).

"Analysis Form for Traffic Accidents", a form that has to be filled out by a police officer for each traffic accident that occurs with injured or deadly wounded casualties on a public road in Belgium.

The second data set, named "Retail", is supplied by an anonymous Belgian retail supermarket store. The data are collected over a period ranging from December 1999 till November 2000. Each record contains informations about the customer ID, the date of purchase and the various items bought.

The third data set, named "Kosarak", contains anonymized click-stream data of a Hungarian on-line news portal.

4.3 Experimental setup

In order to analyze the correctness of our statistical supports, we need to evaluate as many situation as possible, that is, we need to use our method with a range as large as possible for each of the free parameters. These parameters that vary during our experiments are described in Fig. 4.

Better than using a real data stream, which would possibly skew the performance evaluations of the statistical supports, we have chosen to simulate data streams assuming the complete knowledge of the domains, thus allowing to compute exact values for the performance measurements. More precisely, using the databases of Fig. 3, we simulate data streams by sampling each database into fragments. For example, we could consider that data arrive in a timely manner from the "Accidents" database, and that only 20% of the whole data is available and stored. So we pick 20% of the transactions of this database, we consider that it is the data stored, and and check the $\mathtt{P}, \mathtt{R}$ and F_β values with the whole database. Obviously, 20% is an arbitrary percentage, and we have in fact chosen to sample the database on a broad range of percentages. In fact, we have used two scales. The first allows a fine sampling of the database, for small values ranging from 1% to 10% by steps of 1% (column "sampling1" in Fig. 4), and typically gives an idea of what may happens for very large, fast data streams.We have completed this first range with a coarse range of samplings, from 10% to

100% by steps of 3% (column "sampling2" in Fig. 4), which gives a basic idea of the average and limit behaviors of our method.

Finally, δ has been chosen to range through a somewhat usual interval of values for common statistical risks, *i.e.* starting from 1% and stopping at 11% by steps of 2% (see Fig. 4).

On the top of our experiments, we have chosen to use a very convenient association rule mining algorithm, `kdci` [19]. Given the very large number of tests to do for each database, we have written a test generator, which automatically crosses the parameters, and makes all experiments for all possible tuples of parameters. This represents thousands of runs over all databases.

4.4 Results and discussion

Due to the very large number of experiments and the lack of space to report them all, we have separately chosen to put the detail of the experiments and resulting plots into web pages, that make navigation more convenient and understandable. The interested reader can found these results and plots on a separate webpage[3]. Here, we have chosen to report some plots we consider as representative, and make a synthesis out of the whole results.

Figure 5 shows result from experiments on the Accidents and Retail databases. Each plot describes for one database and one support value, either the precision or recall of the three methods which consist in keeping $S^*_{\theta-\varepsilon}$, S^*_{θ}, and $S^*_{\theta+\varepsilon}$. Notice that the value of the risk parameter is kept constant for the plots we show, *i.e.* $\delta = .05$.

A first glance at these plots, or the other ones we have computed, on whichever of the three databases, reveals that their behavior is almost always the same. Namely:

- the precision increases with θ' (eq. 4), while the recall decreases with θ',
- the precision equals or approaches 1 for a large majority of storing sizes when $\theta' = \theta + \varepsilon$,
- the recall equals or approaches 1 for a large majority of storing sizes when $\theta' = \theta + \varepsilon$.

These observations are in accordance with the theoretical results of Section 3. There is another phenomenon we may observe: for example, the recall associated to $\theta' = \theta + \varepsilon$ is not that far from the recall for $\theta' = \theta$. Similarly, the precision associated to $\theta' = \theta - \varepsilon$ is not that far from the precision for $\theta' = \theta$. This shows that the maximization of the precision or recall is obtained at a reduced degradation of the other parameter. We also remark that the precision plots tend to be better than the recall plots. This is not really surprising, as advocated in Section 3, since the range of values for the precision is smaller than for the recall.

[3] http://www.univ-ag.fr/grimaag/statisticalsupports/

A close look at small storing sizes of the streams (before 10%) also reveals a more erratic behavior without convergence to maximal precision or recall. The behavior for the Retail and Kosarak databases are also the same. This behavior is not linked to the statistical support, but to the databases used. Indeed, the data stream is simulated out of each database. So, small databases lead to even smaller storing sizes, and frequent itemsets kept out of small databases are in fact trickier to predict than for bigger databases. This point is important as, from a real-world standpoint, we tend to store very large databases, so we may expect this phenomenon to be reduced. We are actually leading experiments to validate this point.

On the smallest databases, such as Retail and Kosarak, another phenomenon seems to appear. First of all, during the experiments, because of the small values for θ, some tests have not be performed because $\theta - \varepsilon$ was < 0, which is uninteresting as advocated in Section 3. Furthermore, the greater difference observed between the curves seems to stem out from the different sizes of databases. For example, the Retail database is smaller than the Accidents database by a factor 3. In addition, the number of frequent itemsets found in this database is not more than a hundred. For the sake of comparison, the Accidents database for the smallest θ gives hundreds of thousands frequent itemsets. This, we think, explains the greater differences between the curves: they are mostly a small database phenomenon, and may not be expected from larger databases, or even real-world data streams.

In Figure 6, two sets of two plots taken from the Accidents database plot the F_β measure, against the size of the stream used (in %). The values of β have been chosen different from 1, since it would make no real sense to put the same weight into precision and recall, given that we put our primary emphasis on the maximization of a single criterion. The values have also been chosen not too small or too large to yield a reasonable prominence of one criterion (.2 and 1.8, see Figure 6). In each plot, the F_β value displays the advantage of choosing $\theta' = \theta \pm \varepsilon$ against the choice $\theta' = \theta$. This is all the more interesting as this is obtained while statistically guaranteeing the *maximal* value for whichever of the precision or recall criterion chosen by the user.

5 Conclusion

There are three main contributions in this paper. First, we discuss the replacement of the conventional minimal support requirement for finding frequent itemsets by a statistical support, in cases where storing the entire data is impossible (such as for data streams). Then, we provide a method to compute this statistical support, while keeping relevant statistical properties. Last, we validate experimentally our approach.

There are a number of possible extensions to this work. First, technically speaking, our statistical support is optimal from a qualitative standpoint, since maximizing both the precision and recall is, as we have advocated, statistically hard. We feel that the bounds on θ' we have obtained also yield a statistical support which is *quantitatively* optimal. Previous works on the statistical validity of prediction rules tend to show that it is the case [14, 18, 21], but this remains to be formally shown.

The most promising extensions to this work certainly concern the application of the technique to relevant data mining subfields, such as sequential pattern mining or building borders in incremental mining. One very promising research direction would also be to integrate our approach with those exploring data structures to maintain *items* that are *observed* as frequent with maximal recall [12]. In the framework of data streams, where they are particularly relevant, it would be much more efficient from a statistical standpoint to keep the *itemsets* that are *truly* frequent, better than simply observed as frequent, thus killing two birds in one shot for minimizing approximation errors. Because of the technical machinery used in these papers, mixing the approaches into a global technique for reducing the error in maintaining frequent itemsets out of data streams may be more than simply interesting: it seems to be very natural.

References

1. M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proc. of the 29 th International Colloquium on Automata, Languages, and Programming*, pages 693–703, 2002.
2. D. Cheung, J. Han, V. Ng, and C. Wong. Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. In *Proc. of the* 12^{th} *International Conference on Data Engineering*, pages 106–114, New Orleans, Louisiana, February 1996.
3. G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. of the* 22^{nd} *ACM Symposium on the Principle of Database Systems*, pages 296–306. ACM Press, 2003.
4. E. Demaine, A. Lopez-Ortizand, and J.-I. Munro. Frequency Estimation of Internet Packet Streams with Limited Space. In *Proc. of the* 10^{th} *European Symposium on Algorithms*, pages 348–360, 2002.
5. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition.* Springer, 1996.
6. W. Fan, Y.-A. Huang, H. Wang, and P.-S. Yu. Active mining of data streams. In *Proc. of the* 4^{th} *SIAM International Conference on Data Mining*, pages 457–461, 2004.
7. M. Garofalakis, J. Gehrke, and R. Rastogi. Querying and Mining Data Streams: You only get One Look. In *Turorial notes of the 28th International Conference on Very Large Databases (VLDB'02)*, pages 682–693, 2002.
8. C. Giannella, J. Han, J. Pei, X. Yan, and P.-S. Yu. *Mining Frequent Patterns in Data Streams at Multiple Time Granularities*, chapter 6. Data Mining: Next Generation Challenges and Future Directions. H. Karguta, A. Joshi, K. Sivakumar and Y. Yesha (Eds.). MIT/AAAI Press, 2004.

9. L. Golab and M. Tamer Ozsu. Issues in Data Stream Management. *ACM SIGMOD Record*, 2(2):5–14, June 2003.
10. S. Gollapudi and D. Sivakumar. Framework and Algorithms for Trend Analysis in Massive Temporal Data Sets. In *Proc. of the 13 th International Conference on Information and Knowledge Management*, pages 168–177, 2004.
11. Frequent itemset mining dataset repository — http://fimi.cs.helsinki.fi/data, 2005.
12. C. Jin, W. Qian, C. Sha, J.-X. Yu, and A. Zhou. Dynamically maintaining frequent items over a data stream. In *Proc. of the 12 th International Conference on Information and Knowledge Management*, pages 287–294. ACM Press, 2003.
13. R.-M. Karp, S. Shenker, and C.-H. Papadimitriou. A Simple Algorithm for Finding Frequent Elements in Streams and Bags. *ACM Transactions on Database Systems*, 28(1):51–55, March 2003.
14. M. J. Kearns and Y. Mansour. A Fast, Bottom-up Decision Tree Pruning algorithm with Near-Optimal generalization. In *Proc. of the 15 th International Conference on Machine Learning*, pages 269–277, 1998.
15. H.-F. Li, S.-Y. Lee, and M.-K. Shan. An Efficient Algorithm for Mining Frequent Itemsets over the Entire History of Data Streams. In *Proc. of the* 1^{st} *Int. Workshop on Knowledge Discovery in Data Streams*, Pisa, Italy, 2004.
16. G. Manku and R. Motwani. Approximate Frequency Counts over Data Streams. In *Proc. of the 28th International Conference on Very Large Databases*, pages 346–357, Hong Kong, China, 2002.
17. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
18. R. Nock and F. Nielsen. Statistical Region Merging. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(11):1452–1458, 2004.
19. S. Orlando, P. Palmerini, R. Perego, C. Silvestri, and F. Silvestri. kDCI: a multi-strategy algorithm for mining frequent sets. In *Proc. of the Workshop on Frequent Itemset Mining Implementations, in conjunction with ICDM 2003*, 2003.
20. S.-J. Rizvi and J.-R. Haritsa. Maintaining Data Privacy in Association Rule Mining. In *Proc. of the 28th International Conference on Very Large Databases*, pages 682–693, 2002.
21. V. Vapnik. *Statistical Learning Theory*. John Wiley, 1998.
22. A. Veloso, B. Gusmao, W. Meira, M. Carvalho, S. Parthasarathy, and M.-J. Zaki. Efficiently Mining Approximate Models of Associations in Evolving Databases. In *Proc. of the 6th European Conference on the Principles and Practice of Knowledge Discovery in Databases*, pages 435–448, 2002.
23. A. Veloso, W. Meira, M. Carvalho, B. Possas, S. Parthasarathy, and M.-J. Zaki. Mining Frequent Itemsets in Evolving Databases. In *Proc. of the* 2^{nd} *SIAM International Conference on Data Mining*, pages 31–41, Arlington, April 2002.
24. H. Wang, W. Fan, P.-S. Yu, and J. Han. Mining concept-drifting data streams with ensemble classifiers. In *Proc. of the 9th International Conference on Knowledge Discovery in Databases*, pages 226–235, 2003.

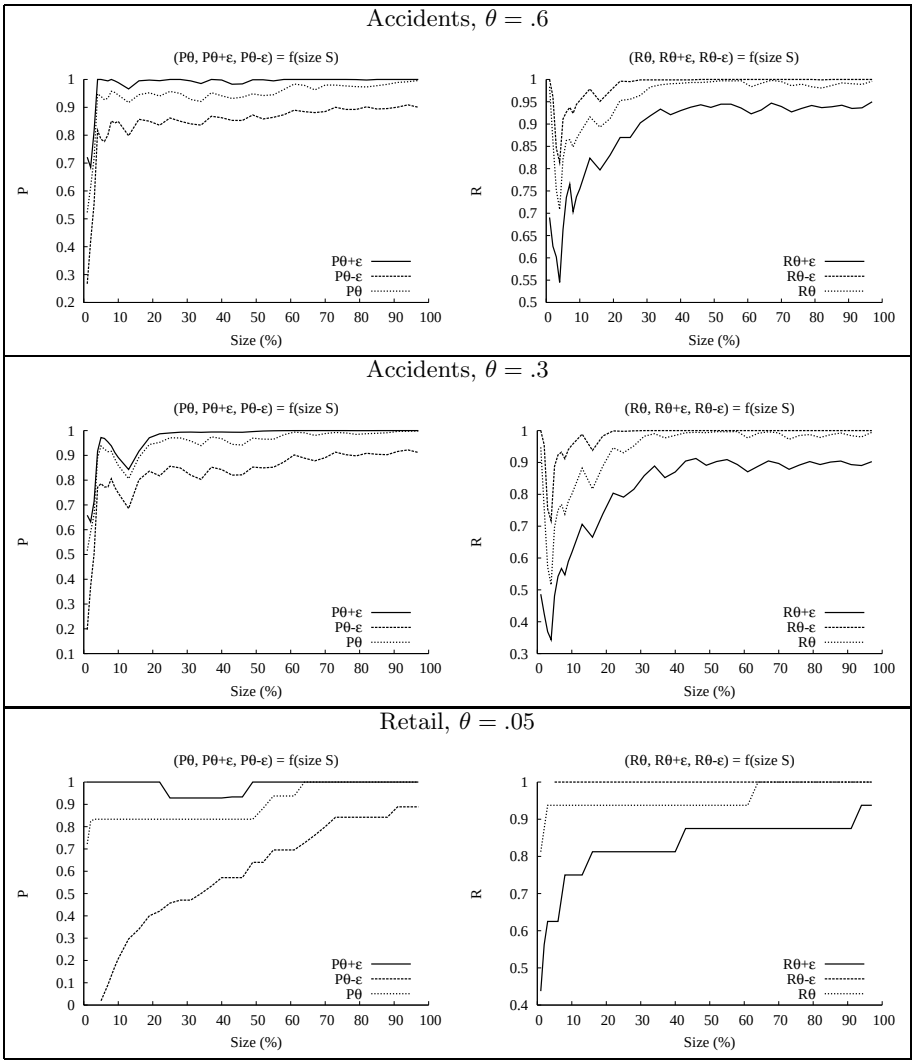

Fig. 5. Three examples of plots for two of our databases, with $\delta = .05$. For three different values of θ, we give the precision (left plot) and recall (right plot) for the three methods consisting in picking $S^*_{\theta-\varepsilon}, S^*_\theta, S^*_{\theta+\varepsilon}$. The x-axis denotes the percentage of the data kept out of the simulated data stream (see text for details).

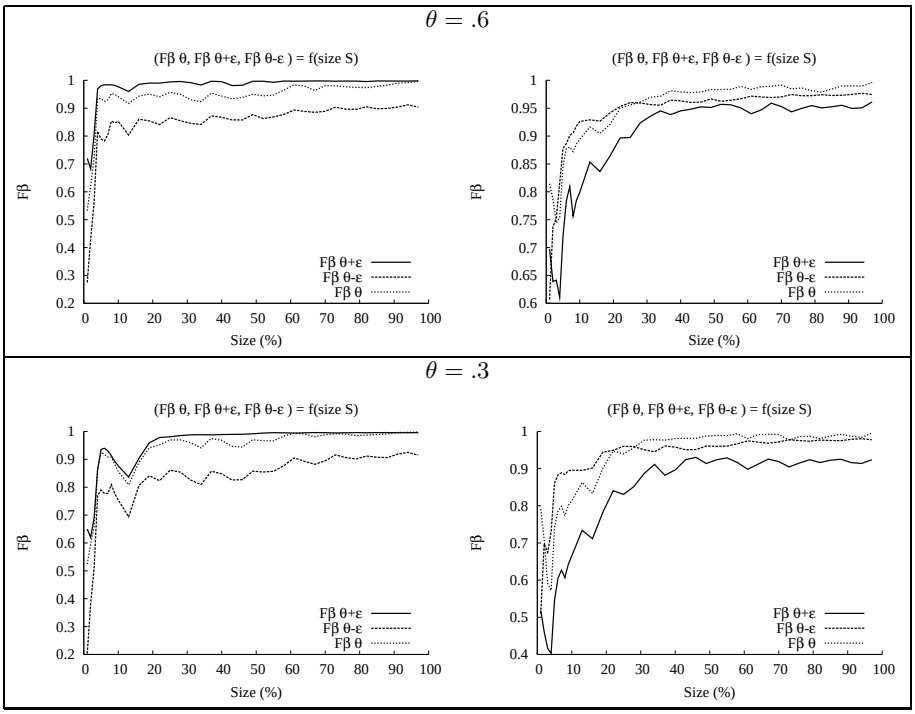

Fig. 6. Two sets of plots of the F_β value from the Accidents database, with $\beta = .2$ for the left plots and $\beta = 1.8$ for the right plots (see text for details).