A Study on the Performance of Large Bayes Classifier

Dimitris Meretakis, Hongjun Lu, and Beat Wüthrich

Computer Science Department, Hong Kong University of Science and Technology,
Clear Water Bay, Hong Kong, China
{meretaks, luhj, beat}@cs.ust.hk

Abstract. Large Bayes (LB) is a recently introduced classifier built from *frequent* and *interesting* itemsets. LB uses itemsets to create context-specific probabilistic models of the data and estimate the conditional probability $P(c_i|A)$ of each class c_i given a case A. In this paper we use chi-square tests to address several drawbacks of the originally proposed interestingness metric, namely: (i) the inability to capture certain really interesting patterns, (ii) the need for a user-defined and data dependent interestingness threshold, and (iii) the need to set a minimum support threshold. We also introduce some pruning criteria which allow for a trade-off between complexity and speed on one side and classification accuracy on the other. Our experimental results show that the modified LB outperforms the original LB, Naïve Bayes, C4.5 and TAN.

1 Introduction

Until recently association (*descriptive*) and classification (*predictive*) mining have been considered as disjoint research and application areas. Descriptive mining aims at the discovery of strong local patterns, so-called *itemsets* [1] that hopefully provide insights on the relationships among some of the attributes of the database. Predictive mining deals with databases that consist of *labeled* tuples. Each label represents a *class* and the aim is to discover a model of the data that can be used to determine the labels (classes) of previously unseen cases.

The use of association mining techniques for classification purposes has only recently been explored. Following this route we recently proposed Large Bayes (LB) classifier [5]. LB considers each attribute-value pair as a distinct *item* and assumes that the training set is a set of transactions. During the learning phase LB employs an Apriori-like [1] association mining algorithm to discover *interesting* and *frequent labeled* itemsets. In the context of classification, we define a labeled itemset l as a set of items together with the supports $l.sup_i$ for each possible class c_i. In other words a labeled itemset provides the observed probability distribution of the class variable given an assignment of values for the corresponding attributes: $l.sup_i = P(l,c_i)$.

A new case $A=\{a_1a_2\ldots a_n\}$, is assigned to the class c_i with the highest conditional probability $P(c_i|A)=P(A,c_i)/P(A)$. Since the denominator is constant with respect to c_i it can be ignored and the object is said to be in class c_i with the highest value $P(A,c_i)$. LB selects the longest subsets of A that are present in the set of discovered itemsets and uses them to incrementally build a *product approximation* of $P(A,c_i)$. For example, if $A=\{a_1a_2a_3a_4a_5\}$, a valid product approximation would be:

$$P(A,c_i)=P(a_2a_5c_i)P(a_3|a_5c_i)P(a_1|a_2a_3c_i)P(a_4|a_1a_5c_i).$$

R. López de Mántaras, E. Plaza (Eds.): ECML 2000, LNAI 1810, pp. 271-279, 2000.

Fig. 1 illustrates how this product approximation is incrementally generated from the set of longest itemsets by adding one itemset at each step. The formula is subsequently evaluated using the class supports of the selected itemsets and finally the class c_i with the highest probability $P(l,c_i)$ is assigned to A. Note that this process builds on the fly a local probabilistic model for the approximation of $P(A,c_i)$ that only holds for the particular classification query.

step	covered items	iset selected	product approximation	available itemsets
0	$\varnothing$	$\varnothing$	N/A	$\{a_1a_2a_3\},\{a_1a_4a_5\},\{a_2a_5\},\{a_3a_4\},\{a_3a_5\}$
1	$\{a_2a_5\}$	$\{a_2a_5\}$	$P(a_2a_5c_1)$	$\{a_1a_2a_3\},\{a_1a_4a_5\},\{a_3a_4\},\{a_3a_5\}$
2	$\{a_2a_3a_5\}$	$\{a_3a_5\}$	$P(a_2a_5c_1)P(a_3\|a_5c_1)$	$\{a_1a_2a_3\},\{a_1a_4a_5\},\{a_3a_4\}$
3	$\{a_1a_2a_3a_5\}$	$\{a_1a_2a_3\}$	$P(a_2a_5c_1)P(a_3\|a_5c_1)P(a_1\|a_2a_3c_1)$	$\{a_1a_4a_5\},\{a_3a_4\}$
4	$\{a_1a_2a_3a_4a_5\}$	$\{a_1a_4a_5\}$	$P(a_2a_5c_1)P(a_3\|a_5c_1)P(a_1\|a_2a_3c_1)P(a_4\|a_1a_5c_1)$	$\{a_3a_4\}$

Fig. 1. Incremental construction of a product approximation for $P(a_1\,a_2\,a_3\,a_4\,a_5\,c_i)$

The key factor in this process is the selection of interesting itemsets. In [5] we used an interestingness metric that was an adaptation of the well known cross-entropy between two probability distributions. To overcome the drawbacks of this approach we use chi-square (χ^2) tests to identify interesting itemsets. In section 4 we show experimentally that this approach leads to significant performance improvements. Moreover, we deal with the problem of setting the correct minimum support and interestingness thresholds for each data set. Although the settings we suggested in [5] work relatively well in practice, they are empirically determined and lack intuitive justification. The χ^2 test besides stemming directly from statistical theory also provides intuitive interpretation to the thresholds.

We also discuss the effect of two other pruning criteria on the performance of the classifier, namely pruning based on (a) the support and (b) the conditional entropy of the class given an itemset. Use of these criteria often leads to the generation of smaller classifiers often without significant sacrifice in the classification accuracy.

2 An Overview of Large Bayes Classifier

We will briefly outline the original LB algorithm, which is described in more details in [5]. Large Bayes is a classifier build from labeled itemsets, denoted as *itemsets* in the sequel. Consider a domain where instances are represented as instantiations of a vector $\boldsymbol{A}=\{A_1,A_2,\ldots,A_n\}$ of n discrete variables, where each variable A_i takes values from $val(A_i)$ and each instance is labeled with one of the $|val(C)|$ possible class labels, where C is the class-variable. A labeled itemset l with its class supports $l.sup_i$ provides the probabilities of joint occurrence $P(l,c_i)$ for l and each class c_i. The learning phase of Large Bayes aims to discover such itemsets that are frequent and interesting. As usual, an itemset is *frequent*, if its support is above the user defined minimum support threshold *minsup*: $\frac{1}{|D|}\cdot\sum_{i=1..|val(c)|} l.count_i \geq minsup$.

We can derive an estimation of the class-supports of an itemset l using two subsets of l where one item is missing. Consider for example the itemset $l=\{a_1,a_2,a_3\}$. Its class-supports $l.sup_i = P(l,c_i) = P(a_1,a_2,a_3,c_i)$ can be estimated using $l_1 = \{a_1,a_2\}$ and l_2

$= \{a_1,a_3\}$ by implicitly making certain independence assumptions: $P(a_1,a_2,a_3,c_i) = P(a_1,a_2,c_i)P(a_3|a_1,c_i) = P(l_1,c_i)P(l_2,c_i)/P(l_1 \cap l_2,c_i)$. Roughly speaking, if this estimation is accurate then l itself is not interesting, since it does not provide any more information than its subsets l_1 and l_2. The quality of the approximation is quantified with an interestingness measure $I(l)$ that returns zero if $P(l,c_i)$ is actually equal to $P(l_1,c_i)P(l_2,c_i)/P(l_1 \cap l_2,c_i)$ and increases with their difference. An itemset is interesting if $I(l) > \tau$, where τ is a user-defined threshold. Fig. 2 presents the learning phase, which performs an Apriori-like bottom-up search and discovers the set F of itemsets that will be used to classify new cases.

```
GenItemsets(D)
In : The database D of training cases
Out: The set F of itemsets l and
     their class counts l.count_i

F_1 ={{a_j}| a_j is non class attribute}
Determine l.count_i ∀l∈ F_1, ∀class i
for (k=2; F_k-1 ≠ ∅; k++) {
  C_k = genCandidates(F_k-1)
  For all tuples t ∈ D {
    C_t = subsets(C_k, t);
    i = class of t;
    for all candidates l∈C_t
      l.count_i++;
  }
  F_k =selectFreguentAndInteresting(C_k) }
Return F = ∪_k F_k
```

Fig. 2. Algorithm *genItemsets*

```
Classify(F,A)
In : The set F of discovered itemsets a new
     instance A
Out: The classification c_i of A

cov = ∅ \\ the subset of A already covered
nom = ∅  \\ set of itemsets in nominator
den = ∅  \\ set of itemsets in denominator
B = {l∈F | l⊆A and ¬∃l'∈F: l'⊆A and l⊂l'}
for  (k=1; cov ⊂ A; k++){
  l_k = pickNext( cov, B )
  nom = nom ∪ {l_k}
  den = den ∪ {l_k ∩ cov}
  cov = cov ∪ l_k }
output that class c_i with maximal P(A,c_i):
```

$$P(A,c_i) = P(c_i) \cdot \prod_{l \in nom} P(l,c_i) \Big/ \prod_{h \in den} P(h,c_i)$$

Fig. 3. Algorithm *classify*

Given a particular instance A to be classified, the set F' of the longest and most interesting itemsets in F which are subsets of A are selected. The itemsets of F' are then used to incrementally construct a product approximation for $P(A,c_i)$. The procedure *classify()* that performs this task is presented in Fig. 3 while Fig. 4 presents the selection criteria for the next itemset to be inserted in the product approximation.

```
pickNext( cov, B )
T = { l∈B: |l- covered | ≥1};
Return an itemset l_k∈T such that for all other itemsets l_j ∈T:
1.|l_k-covered| < |l_j-covered|, or
2.|l_k-covered| = |l_j-covered| and |l_k| > |l_j| , or
3.|l_k-covered| = |l_j-covered| and |l_k| = |l_j| and |I(l_k) > I(l_j)
```

Fig. 4. Procedure *pickNext*

The resulting formula is the local model build on the fly by LB to classify A. This model implies some conditional independence assumptions among the variables but they are context-specific in the sense that different classification queries (i.e. different values of A) will produce different models making different independence assumptions. [5,6] discuss this in more detail. Finally, the formula $P(A,c_i)$ is evaluated for each c_i and A is labeled with the class c_i that maximizes $P(A,c_i)$.

3 Improving Large Bayes

A key factor affecting the performance of Large Bayes is the accurate identification of interesting itemsets. The interestingness of an itemset l is defined in terms of the error

when estimating $P(l,c_i)$ using subsets of l. Let l be an itemset of size $|l|$ and l_j , l_k be two ($|l|$-1)-itemsets obtained from l by omitting the j^{th} and k^{th} item respectively. We can use l_j, l_k to produce an estimate $P_{j,k}(l,c_i)$ of $P(l,c_i)$:

$$P_{est}(l,c_i) = P_{j,k}(l,c_i) = \frac{P(l_j,c_i)\cdot P(l_k,c_i)}{P(l_j \cap l_k,c_i)} \quad (1)$$

Our goal is to keep those itemsets only, for which the corresponding observed probabilities differ much from the estimated ones. Information-theoretic metrics such as the cross-entropy (or Kullback-Leibler distance) are widely used [4] as a measure of the distance between the observed and the estimated probability distributions:

$$D_{KL}(P,P_{est}) = \sum_{l,c_i} P(l,c_i)\log\frac{P(l,c_i)}{P_{est}(l,c_i)} \quad (2)$$

In our case, however, the goal is to measure the distance between specific elements of the probability distribution. Consider for example a case with two variables A_1 and A_2 and $|val(A_1)| = |val(A_2)| = 4$. The corresponding sixteen 2-itemsets define the complete observed joint probability distribution $P(A_1,A_2,C)$. A high value of such metrics suggests that the class-supports of the corresponding itemsets cannot be accurately approximated *on average* by the class-supports of their subsets. To measure the accuracy of the approximation for individual itemsets in [5] we defined the interestingness $I(l|l_j,l_k)$ of l with respect to its subsets l_j and l_k as:

$$I(l\,|\,l_j,l_k) = \sum_{c_i}\left|P(l,c_i)\log\frac{P(l,c_i)}{P_{est}(l,c_i)}\right| \quad (3)$$

This ad-hoc measure presents certain drawbacks with respect to its ability to identify interesting local patterns. Consider for example a domain with two classes and an itemset l for which $P(l,c_1) = 0$, $P(l,c_2)=0.15$ and $P_{est}(l,c_1) = 0.1$, $P_{est}(l,c_2)=0.15$. Although this is indeed a very interesting itemset since the estimated probability for c_1 greatly differs from observed one, $I(l|l_j,l_k) = 0$ and l is discarded as non-interesting. In addition, our interestingness measure (but also every information-theoretic measure) suffers from the fact that it ignores the sample size and assumes that the sample probability distribution is equal to the population probability distribution thus ignoring the possibility that the differences occurred purely because of chance.

In the sequel we describe the application of chi-square (χ^2) tests to overcome these problems. We reduce the problem of deciding whether an itemset is interesting to applying a hypothesis-testing procedure on the following hypotheses:

H_0: $P(l,c_i) = P_{est}(l,c_i)$, i.e. l is not-interesting

H_1: $P(l,c_i) \neq P_{est}(l,c_i)$, i.e. l is interesting

To test the hypotheses we calculate the χ^2 test statistic with $|C|$ degrees of freedom. ($|D|$ is the database size, $|C|$ the number of classes):

$$\chi_l^2 = \sum_{j=1}^{|C|}\frac{\left(P(l,c_i)\cdot|D| - P_{est}(l,c_i)\cdot|D|\right)^2}{P_{est}(l,c_i)\cdot|D|} = \sum_{j=1}^{|C|}\frac{\left(P(l,c_i)-P_{est}(l,c_i)\right)^2}{P_{est}(l,c_i)} * |D| \quad (4)$$

If $\chi_l^2 > \chi^2_{p,|C|}$ the null hypothesis H_0 is rejected and l is considered interesting. The statistical-significance threshold p is user-defined but should in general be high i.e. $p<0.05$ since discovering non-interesting itemsets does not improve the accuracy and unnecessarily increases the complexity of the resulting classifier. The degrees of freedom for the test are $|C|$ since the sums of the expected and the observed

frequencies of an itemset are generally different [8]. If the degrees of freedom are two or less, Yates correction is applied (subtracting 0.5 from the absolute difference in Eq. (4) before squaring, when this difference exceeds 0.5).

A problem associated with χ^2 tests is that the estimated frequencies $P_{est}(l, c_i) \cdot |D|$ in each term of in Eq. (4) should be not too small otherwise the test is sensitive to errors. To overcome this problem we apply a merging step before calculating the χ^2-statistic. During this step the class with the smallest frequency is merged with the immediate larger class to form a composite class containing the sum of the frequencies. The corresponding observed frequencies are merged also and the degrees of freedom (*df*) are reduced by one. The merging phase stops when all expected frequencies are large enough or when all class-frequencies are merged. Following standard statistical practice we set the minimum value of an expected frequency to 5 if $df = 2$ and 3 if $df>2$, otherwise it is merged.

As a result of the merging step, each itemset l has a value χ_l^2 that refers to different degrees of freedom df_l. To compare these values with the minimum required threshold $\chi^2_{p,|C|}$ we need a degrees-of-freedom-independent test. For that reason we take advantage of the fact that the value $t = \sqrt{2 \cdot \chi^2} - \sqrt{2 \cdot df - 1}$ approximately follows the normal distribution and therefore the modified requirement for an itemset to be interesting becomes:

$$t = \sqrt{2 \cdot \chi_l^2} - \sqrt{2 \cdot df_l - 1} > \sqrt{2 \cdot \chi^2_{p,|C|}} - \sqrt{2 \cdot |C| - 1} \quad \textbf{(5)}$$

Note that although t can now take negative values as well, the requirement for an itemset to be interesting remains that its t value is bigger than the threshold of equation (5) which is determined by the required statistical significance level p and the number of classes $|C|$.

3.1 Pruning Criteria: Trading off Accuracy for Simplicity and Speed

A difficult challenge in the design of classifiers is preventing overfitting and generating simple models. Simple models are not only easily interpretable but also generalize better in unseen data and are faster to build and evaluate. In LB overfitting translates to the discovery of "too many itemsets" and this is particularly true in domains with many multi-valued attributes, where the search space is huge. χ^2 tests significantly reduce the number of discovered itemsets to a tractable amount. However, there are some other prunning criteria that can potentially reduce the number of itemsets and accelerate both the learning and classification phase.

Support-based pruning is used by many classification methods including decision trees, where a leaf is not expanded if it contains less than a minimum number of cases. In section 4 we evaluate the effect of support pruning on the accuracy of LB.

A somehow more effective pruning criterion is the conditional entropy of the class C given an itemset l:

$$H(C \mid l) = -\sum_{j=1}^{|C|} P(c_i \mid l) \cdot \log P(c_i \mid l)$$

Conditional entropy takes values ranging from zero (if l only appears with a single class) to $log(|C|)$, if l's appearances are uniformly distributed among the classes. If $H(C|l)$ is very small l bears almost certainty about a class and therefore needs not be expanded. In the next section we show that the

introduction of a relatively low conditional entropy threshold often reduces the size of the classifier without significantly affecting its accuracy.

4 Experimental Results

To evaluate the performance of LB with the χ^2 tests (LB-chi2), we use 23 data sets from the UCI ML Repository [7] with a special preference on the largest and more challenging ones in terms of achievable classification accuracy. We compared LB-chi2 with the originally proposed version of LB, the Naïve Bayes classifier [2] (since in the extreme case if only 1-itemsets are used LB reduces to NB), Quinlan's Decision Tree classifier C4.5 [9], and TAN [4]; a Bayesian Network classifier that relaxes the independence assumptions of NB by using some pairs of attributes.

Accuracy was measured either using 10-fold cross validation (CV-10) for small data sets or the holdout method (training and testing set split) for the larger ones. The train and test set splits and the cv-folds were the same for all results reported. Since all methods except of C4.5 only deal with discrete attributes, we used entropy-based [3] discretization for all continuous attributes. No discretization was applied for C4.5.

The factor most affecting the results is the p-value of the χ^2 tests. We experimented with 0.01, 0.025, 0.005, 0.001 and 0.0005, and selected 0.005 as the most effective one. Higher p-values slowly deteriorated the accuracy and tended to produce more complex, larger and slower classifiers. This is natural since high p-values cause more itemsets to be characterized as interesting. On the other hand, values below p=0.005 caused most of the itemsets to be rejected as non-interesting and generated simplistic classifiers with poor accuracy. The effects of the varying p-values on the average accuracy and classifier size can be seen on figure 5.

Table **1** provides a comparison of the algorithms in the 23 data sets according to five criteria. LB-chi2 outperforms all others according to all criteria indicating that it is indeed a very accurate classifier. The criteria used are: (1) Average Accuracy of the classifiers, (2) Average Rank (Smallest values indicate better performance on average), (3) The number of wins–losses of LB-chi2 against other algorithms, and the statistical significance of the improvement of LB-chi2 against each algorithm using (4) a one-sided paired t-test and (5) a Wilcoxon paired, signed, one-sided, rank test.

Table 1. Comparison of the classifiers according to various criteria

		NB	C4.5	TAN	LB	LB-chi2
1	**Average Accuracy**	0.8187	0.8147	0.8376	0.8332	0.8434
2	**Average Rank**	3.695652	3.73913	2.73913	2.652174	1.913043
3	**No wins vs.:**	19 – 4	19 - 4	16 - 6	15-7	--
4	**1-side Paired t-test**	0.9995	0.9991	0.9940	0.9828	--
5	**Wilcoxon paired signed rank test**	>0.995	>0.995	>0.99	>0.975	--

Table 2. Summary Table of datasets and results. |A| = number of attributes, |I|=number of distinct items (attribute-value pairs) after discretization, |C| = number of classes, Miss = presence of missing values. Last two colums indicate the training/testing time of LB-chi2 in sec

		Data set Properties						Accuracy					Time (s) LB-chi2	
	Data Set	|A|	|I|	|C|	Mis	# Train	# Test	NB	C4.5	TAN	LB	LB-chi2	Train	Test
1	**Adult**	14	147	2	Yes	32561	16281	0.8412	0.854	0.8571	0.8511	**0.8668**	48.81	37.04
2	**Australian**	14	48	2	No	690	CV-10	0.8565	0.8428	0.8522	0.8565	**0.8609**	0.18	0.03
3	**Breast**	10	28	2	Yes	699	CV-10	0.97	0.9542	0.9671	0.9686	**0.9714**	0.11	0.02
4	**Chess**	36	73	2	No	2130	1066	0.8715	**0.995**	0.9212	0.9024	0.9418	1.99	2.20
5	**Cleve**	13	27	2	Yes	303	CV-10	**0.8278**	0.7229	0.8122	0.8219	0.8255	0.07	0.01
6	**Flare**	10	27	2	No	1066	CV-10	0.7946	0.8116	**0.8264**	0.8152	0.818	0.18	0.03
7	**German**	20	60	2	No	999	CV-10	0.741	0.717	0.727	0.748	**0.75**	0.42	0.07
8	**Heart**	13	17	2	No	270	CV-10	0.8222	0.7669	**0.8333**	0.8222	0.8185	0.05	0.01
9	**Hepatitis**	19	32	2	Yes	155	CV-10	0.8392	0.8	0.8188	**0.845**	0.8446	0.05	0.01
10	**Letter**	16	146	26	No	15000	5000	0.7494	0.777	0.8572	0.764	**0.8594**	109.29	56.90
11	**Lymph**	18	49	4	No	148	CV-10	0.8186	0.7839	0.8376	0.8457	**0.8524**	0.07	0.02
12	**Pendigits**	16	151	10	No	7494	3499	0.8350	0.923	0.9360	0.9182	**0.9403**	44.23	22.58
13	**Pima**	8	15	2	No	768	CV-10	**0.759**	0.711	0.7577	0.7577	0.7564	0.06	0.02
14	**Pima Diabetes**	8	14	2	No	768	CV-10	0.7513	0.7173	0.7656	**0.7669**	0.763	0.07	0.02
15	**Satimage**	36	384	6	No	4435	2000	0.818	0.852	0.872	0.839	**0.8785**	392.60	83.75
16	**Segment**	19	147	7	No	1540	770	0.9182	**0.958**	0.9351	0.9416	0.9429	2.28	1.16
17	**Shuttle-small**	9	50	7	No	3866	1934	0.987	0.995	**0.9964**	0.9938	0.9948	1.40	0.78
18	**Sleep**	13	113	6	No	70606	35305	0.6781	0.7310	0.7306	0.7195	**0.7353**	476.71	621.97
19	**Splice**	59	287	3	No	2126	1064	**0.9464**	0.933	0.9463	**0.9464**	0.9408	3.24	3.07
20	**Vehicle**	18	69	4	No	846	CV-10	0.6112	0.6982	0.7092	0.688	**0.7187**	1.24	0.23
21	**Vote Records**	16	48	2	No	435	CV-10	0.9034	**0.9566**	0.9332	0.9472	0.9334	0.13	0.04
22	**Waveform-21**	21	44	3	No	300	4700	0.7851	0.704	0.7913	**0.7943**	0.7913	0.1	2.724
23	**Yeast**	8	18	10	No	1484	CV-10	0.5805	0.5573	0.5721	**0.5816**	**0.5816**	0.15	0.04

Table 2 provides information about the data sets, lists the accuracies of the classifiers and shows the training and testing time of LB-chi2 on all data sets (Measured on a 400MHz Pentium WinNT PC). Noticeably, the biggest improvements in accuracy against the original LB came mostly from the largest data sets; this indicates the inability of the originally used interestingness metric in such cases.

The p-value for chi2-LB was set to 0.005 and to facilitate more accurate χ^2 tests the minimum support was set to *max{10, 2*|c|}*. Although this is a minimum requirement in order for the test statistic to be accurate this can be further increased in order to reduce both the training time and the size of the classifier as discussed below.

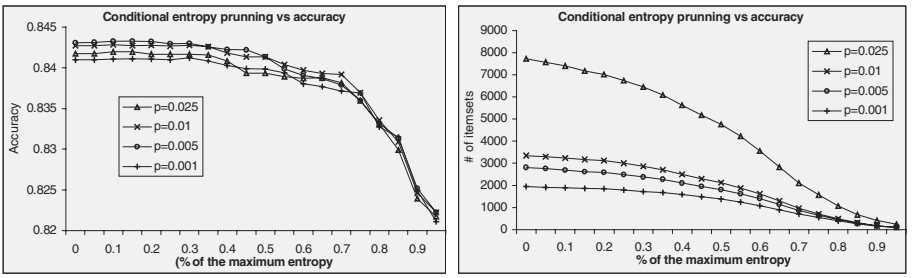

Fig. 5. Effect of conditional entropy pruning on (a) accuracy and (b) size of classifier for four different p-value thresholds. x-axis contains the threshold as a percentage of the maximum conditional entropy $log|C|$

Figure 5 illustrates the effect of conditional entropy pruning on the average accuracy (5a) and size (5b) of LB. Since the number of classes is diferent among the datasets the minimum threshold *minH* is expressed as a percentage of the maximum conditional entropy $\log|C|$. In a 4-class domain, for example, a value of 0.2 implies that $minH = 0.2 \cdot \log 4 = 0.4$. Values for *minH* of up to $0.3 \cdot \log|C|$ have little impact on the accuracy while at the same time reducing the size of the classifier. The rightmost values of the graph correspond to maximum pruning where only 1-itemsets are used and therefore represent the accuracy and size of Naïve Bayes classifier.

Support pruning has a more drastic effect on the size of the classifier as can be seen in Figure 6. This is particularly true on large data sets like "sleep" where 10 occurrences for an itemset *l* represent a probability *P(l)=0.0001*. Increasing the minsup threshold to 0.005 in this data set reduced the number of itemsets discovered from 31000 to 7500 while the accuracy fell only slightly, from 0.7336 to 0.727.

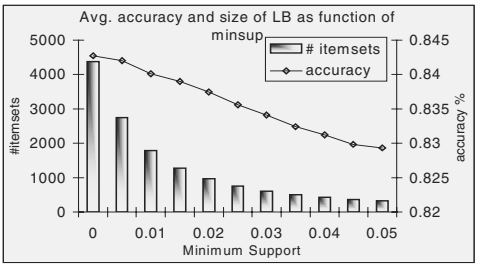

Fig. 6. Effect of support pruning on average accuracy and size of LB

References

1. R. Agrawal, R. Srikant, Fast algorithms for mining association rules, *VLDB-94*, 1994.
2. R.Duda, P.Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1973.
3. U.M.Fayyad, K.B.Irani, Multi-Interval discretization of continuous-valued attributes for classification learning, *13th IJCAI*, 1022-1027, 1993.

4. N.Friedman, D. Geiger, M. Goldszmidt, Bayesian Network Classifiers*, Machine Learning, 29*, 131-163, 1997.
5. D. Meretakis, B.Wüthrich, Extending Naïve Bayes Classifiers Using Long Itemsets, *KDD-99*, pp 165-174, San Diego, USA, 1999.
6. D.Meretakis, B.Wüthrich, Classification as Mining and Use of Labeled Itemsets, *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery* (DMKD'99), Philadelphia, USA, 1999.
7. C.J.Merz, P.Murphy, UCI repository of machine learning databases, 1996 (http:://www.cs.uci.edu/~mlearn/MLRepository.html).
8. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical Recipes in C*, 2nd Ed, Cambridge University Press, 1992.
9. J.R.Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.