

ACME: An Associative Classifier based on Maximum Entropy Principle

Risi Thonangi, Vikram Pudi

Center for Data Engineering,
International Institute of Information Technology, Hyderabad
rishi@research.iiit.ac.in , vikram@iiit.ac.in

Abstract. Recent studies in classification have proposed ways of exploiting the association rule mining paradigm. These studies have performed extensive experiments to show their techniques to be both efficient and accurate. However, existing studies in this paradigm either do not provide any theoretical justification behind their approaches or assume independence between some *parameters*. In this work, we propose a new classifier based on association rule mining. Our classifier rests on the maximum entropy principle for its statistical basis and does not assume any independence not inferred from the given dataset. We use the classical generalized iterative scaling algorithm (GIS) to create our classification model. We show that GIS fails in some cases when itemsets are used as features and provide modifications to rectify this problem. We show that this modified GIS runs much faster than the original GIS. We also describe techniques to make GIS tractable for large feature spaces – we provide a new technique to divide a feature space into independent clusters each of which can be handled separately. Our experimental results show that our classifier is generally more accurate than the existing classification methods.

1 Introduction

Classification has been an age old problem. It involves labeling a query with one among a set of possible class labels by learning from available query-label pairs. Previous studies such as decision trees [20], rule learning [3], naive-bayes [8] and other statistical approaches [15] have developed heuristic/greedy search techniques for building classifiers. These techniques build a set of rules covering the given dataset and use them for prediction. Machine learning approaches like SVMs [5] do classification by learning boundaries between classes and checking on which side of the boundary the query lies.

Recent studies in classification have proposed ways to exploit the paradigm of association rule mining for the problem of classification. These methods mine high quality association rules and build classifiers based on them [16] [7] [18]. We refer to these approaches as *associative classifiers* and they have several advantages – (1) Frequent itemsets capture all the dominant relationships between items in a dataset. (2) Efficient itemset mining algorithms exist. (3) These classifiers naturally handle *missing values* and *outliers* as they only deal with *statistically significant* associations. This property translates well into the classification framework to make it robust. (4) Extensive performance studies [14] [17] have shown such classifiers to be generally more *accurate*.

However, existing studies in the associative classification paradigm either do not provide any theoretical justification behind their approaches [14] or assume independence between some *parameters* in the domain [18] [8].

In this paper we propose **ACME** - a new Associative Classifier based on Maximum Entropy. The maximum entropy principle is well-accepted in the statistics community. It states that given a collection of known facts about a probability distribution, choose a model for this distribution that is consistent with all the facts but otherwise is as uniform as possible. Hence, the chosen model does not assume any independence between its parameters that is not reflected in the given facts. In this paper, we use the *Generalized Iterative Scaling (GIS)* algorithm to compute the maximum entropy model.

Our main contributions in this work are as follows:

1. We develop ACME, a new associative classifier.
2. We show that the classical GIS algorithm fails in some cases when itemsets are used as features. In particular it fails in the presence of itemsets that are not *closed*¹.
3. We provide a modification to GIS to work around the above-mentioned problem when itemsets are used as features. The modified GIS is faster than the original GIS.
4. We describe techniques to make GIS tractable for large feature spaces – we provide a new technique to efficiently divide a feature space into independent clusters each of which can be handled separately.

The rest of this paper is organized as follows: Section 2 formally introduces the classification problem and in Section 3, we describe the overall design of the proposed ACME classifier. In Section 3.1, we describe the classical GIS algorithm that computes the max-entropy distribution. Section 4 describes the drawbacks of the classical Maximum Entropy model and in Section 5, we present the new Maximum Entropy model. In Section 5.1, we adapt the GIS algorithm to converge in the presence of itemsets that are not *closed*. In Section 6, we describe techniques to make GIS tractable for large feature spaces and describe related work in Section 7. In Section 8, we experimentally evaluate the accuracy of ACME and finally conclude our study in Section 9.

2 Problem Definition

Let $I = \{i_1, i_2, \dots, i_n\}$ be the set of items that can appear in a transaction and $L = \{c_1, c_2, \dots, c_l\}$ be a set of l classes. A transaction containing items $\{i_{k_1}, i_{k_2}, \dots, i_{k_j}\}$ is denoted by x_r where r is a *whole number* whose k_1, k_2, \dots, k_j bits are set to 1 and the remaining bits are set to 0. Hence the set $X = \{x_0, x_1, \dots, x_{2^n-1}\}$ represents the set of all possible transactions. The training dataset D is a set of transactions, each of which is labeled with one of the m classes.

Given a transaction x to classify (i.e. x is a *query*), we label it with class c_i for which the posterior probability $p(c_i|x)$ is maximum. *Bayes formula* allows us to compute this probability from the prior probability $p(c_i)$ and the class-conditioned probability $p(x|c_i)$ as follows

$$p(c_i|x) = \frac{p(x|c_i) * p(c_i)}{p(x)}$$

¹ An itemset is not closed iff it has the same frequency as one of its supersets

Since the denominator $p(x)$ is common for all the classes, it is ignored. $p(c_i)$ is the relative frequency of class c_i in D , which is trivial to calculate. Hence, the classification problem is translated to the correct estimation of $p(x|c_i)$, given the dataset D .

We can calculate $p(x|c_i)$ by directly measuring the frequency of x in the transactions in D which belong to the class c_i . But, because most training datasets are not large enough, this method would not be accurate if x is infrequent or non-existent in the dataset.

3 The ACME Classifier

The ACME classifier estimates $p(x|c_i)$ in the following way. Let $S = \{s_1, s_2, \dots, s_{|S|}\}$ be the union of the frequent itemsets extracted from each class c_i .

$$S = \{ s_j \mid \exists c_i \text{ s.t. } P(s_j|c_i) \geq \sigma \} \quad (1)$$

$$\text{where } P(s_j|c_i) = \sum_{x \in X \wedge s_j \subseteq x} p(x|c_i)$$

$P(s_j|c_i)$ denotes the support of s_j in class c_i and σ is the user given support threshold. Each $s_j \in S$ is called an *fset*. We use the itemsets in set S as parameters to model each class, such that each itemset s_j together with its support $P(s_j|c_i)$ in class c_i forms a *constraint* that needs to be satisfied by the statistical model for that particular class. Thus, for each class c_i we have a set of constraints $C_i = \{(s_j, P(s_j|c_i)) \mid s_j \in S\}$. The probability distribution that we build for class c_i must satisfy these constraints. However, there could be multiple distributions satisfying these constraints. We follow the *maximum entropy principle* [10] and among these distributions select the one with the highest entropy. This is referred to as the Maximum Entropy model. It is unique and can be expressed in the following product form [22]:

$$p(x|c_i) = \pi \prod_{j=1}^{|C_i|} \mu_j^{f_j(x)} \quad (2)$$

$$\begin{aligned} \text{where } f_j(x) &= 1 \quad \text{if } s_j \subseteq x \\ &= 0 \quad \text{otherwise} \end{aligned}$$

In Equation 2, π is a normalization constant which ensures $\sum_{x \in X} p(x|c_i) = 1$. There are well-defined iterative algorithms to compute μ_j 's. In this paper we use the Generalized Iterative Scaling (GIS) algorithm [6] described in Section 3.1. The model given in Equation 2 is referred to as the classical Maximum Entropy model.

Maximum entropy modeling does not differentiate between a class-variable and a normal variable. Infact, the model given in Equation 2 is called as conditional maximum entropy model as it models the domain for each class-variable. For brevity, we refer to it as "maximum entropy model". In the ensuing discussion we drop the *conditional* in $P(s_j|c_i)$ and write it as $P(s_j)$ and likewise wherever required. The *conditional* is implicitly assumed.

In ACME, the training phase would involve finding the set of constraints S and computing μ values for all the classes. The computed μ values for each class are stored and are used in the actual classification phase. The procedure to classify a given transaction x is to first extract all the itemsets in S that are subsets of x . These are the *features* of x . Then compute Equation 2 for each class and select that class which maximizes this equation.

3.1 Computing Parameters of the Maximum Entropy Model

The μ_j 's in Equation 2 are estimated by a procedure called the Generalized Iterative Scaling (GIS) [22]. This is an iterative method that improves the estimation of the parameters with each iteration. The algorithm stops when there is no significant change in the μ_j values. This solution is globally optimal as it has been proved that the search space of μ_j 's over which we are searching for the final solution is concave leading to the conclusion that every locally optimal solution is globally optimal [6]. In this section, we present the GIS algorithm and discuss some issues about it.

The GIS algorithm runs with the constraint set C on the domain X computing a model which has the highest entropy and satisfies all the constraints. We call X and C , the parameters of GIS. The GIS algorithm first initializes all the μ_j 's to 1, and executes the following procedure until convergence:

$$\mu_j^{(n+1)} = \mu_j^{(n)} \left[\frac{P(s_j)}{P^{(n)}(s_j)} \right] \quad (3)$$

where

$$P^{(n)}(s_j) = \sum_{x \in X} p^{(n)}(x) f_j(x)$$

$$p^{(n)}(x) = \pi \prod_{j=1}^{|C|} (\mu_j^{(n)})^{f_j(x)}.$$

The variable n in the above system of equations denotes the iteration number. $P^{(n)}(s_j)$ is the expected support of s_j in the n 'th iteration while $P(s_j)$ is the actual support of s_j calculated from the training dataset. Convergence is achieved when the expected and actual supports of every s_j are nearly equal.

Note that for the GIS to converge, minor modifications to the above formalism (in Equation 3) are required. Details of this are available in [21] and are not required for the discussion in this paper.

Time Complexity Every time Equation 3 needs to be executed, $P^{(n)}(s_j)$ is to be calculated from the distribution $p^{(n)}$. This step takes $O(|X|)$, and to execute it for all s_j it takes $O(|X| * |C|)$. If the algorithm requires m iterations for the distribution p to converge, the time complexity of GIS can be given as $O(m * |X| * |C|)$. Under practical circumstances, the number of iterations m is hard-coded and the algorithm is made to

stop once it reaches those many iterations without waiting for the distribution to fully converge² [22].

One drawback in the above approach is that the probability model expressed in Equation 2 *fails* in some cases [22]. In particular, it fails in the presence of itemsets that are not *closed*. In Section 5.1 we discuss this drawback in detail, and in Section 5 we propose a solution to this problem.

Another drawback in the above approach is that it is not tractable when the set of items I is very large. This is because the complexity of the GIS algorithm is exponential w.r.t. $|I|$. We describe a technique to overcome this drawback in Section 6. It involves partitioning I into independent clusters of items so that GIS can be run on each cluster separately.

4 Failure of Approach due to Non-Closed Itemsets

In this section, we explain why the classical Maximum Entropy model as given by the product form in Equation 2 fails in some cases. An itemset is not *closed* *iff* it has the same frequency as one of its supersets. The formal definition of a *non-closed* itemset follows.

Definition 1. An itemset $s_u \in S$ is non-closed *iff*

$$P(s_u) \neq 0 \text{ and} \\ \exists s_v \in S \text{ s.t. } s_u \subset s_v \wedge P(s_u) = P(s_v)$$

The presence of s_u in a transaction x means that s_v will also be present in x . We denote such a pair by $[s_u, s_v]$, a fully-confident itemset pair. \square

A major disadvantage with the Maximum Entropy model is that Equation 2 does not have a solution when the system of constraints have *non-closed* itemsets in them. We prove this formally in Theorem 1.

Theorem 1. Equation 2 does not have a solution when the system of constraints have non-closed itemsets.

Proof. Let $s_u \in S$ be a non-closed itemset. From Definition 1:

$$P(s_u) > 0 \tag{4}$$

Since s_u is a non-closed itemset, $\exists s_v \in S \text{ s.t. } s_u \subset s_v \wedge P(s_u) = P(s_v)$ which means probability of any transaction that contains s_u but not s_v is 0. Let x be a transaction s.t. no other items except the ones in s_u are present in it. It means:

$$i_k \in s_u \Leftrightarrow i_k \in x$$

where i_k is an item in I . Since s_v is strictly a superset of s_u , $s_v \not\subseteq x$. Hence $p(x) = 0$. From Equation 2:

$$p(x) = \pi \prod_{s_w \subseteq s_u} \mu_w = 0 \Rightarrow \because \pi \neq 0, \exists s_w \in S \wedge s_w \subseteq s_u \text{ s.t. } \mu_w = 0$$

² In our experiments, the GIS was run for 100 iterations.

This means $\forall t \in X$ with $s_u \subseteq t$, $p(t) = 0$ as s_w will be present in the product form of $p(t)$ since $s_w \subseteq s_u$. From the above statement,

$$P(s_u) = \sum_{s_u \subseteq x} p(x) = 0$$

which contradicts Equation 4. \therefore Equation 2 does not have a solution when the system of constraints have *non-closed* itemsets. ■

Hence, in cases when the system of constraints have non-closed constraints, the exact solution does not exist in the form of the probability model given in Equation 2 and the model parameters will not converge under the GIS algorithm. We propose a modification to the maximum entropy model enabling it to accommodate *non-closed* itemsets in its system of constraints and give a proof of convergence for this new model.

5 The Modified Maximum Entropy Model

The modified product form of the Maximum Entropy model is given as:

$$\begin{aligned} p(x) &= 0 \quad \text{if } \exists [s_u, s_v] \text{ s.t. } s_u \subseteq x \wedge s_v \not\subseteq x \\ &= \pi \prod_{i=1}^{|C'|} \mu_i^{f_i(x)} \quad \text{otherwise} \end{aligned} \quad (5)$$

C' in the above equation is the set of closed constraints in C . The non-closed constraints in C are only used to determine whether the probability of the transaction is 0 or not. When we find that the transaction has positive probability, its value $p(x)$ is calculated using the closed constraints in C .

Let $X' \subset X$ be the set of transactions for which $p(x) > 0$. We call such transactions as *closed* transactions. Transactions in the set $\{X - X'\}$ are called non-closed transactions.

Theorem 2. Equation 5 refers to the Maximum Entropy model satisfying the given constraints inclusive of the closed-itemsets.

Proof. Same as that of the classical GIS given in [22], except that the transaction set would be X' instead of X and constraint set is C' instead of C . ■

5.1 Computing parameters for the Modified Maximum Entropy Model

Instead of the usual *parameters* X and C , the GIS algorithm is run with C' on the domain X' to build the new Maximum Entropy model. It first initializes all the μ_j 's in C' to 1 and executes the GIS procedure until convergence:

$$\mu_j^{(n+1)} = \mu_j^{(n)} \left[\frac{P(s_j)}{P^{(n)}(s_j)} \right] \quad \mu_j \in C' \quad (6)$$

where

$$P^{(n)}(s_j) = \sum_{x' \in X'} p^{(n)}(x') f_j(x')$$

$$p^{(n)}(x') = \pi \prod_{j=1}^{|C'|} (\mu_j^{(n)})^{f_j(x')} \quad \forall x' \in X' \quad (7)$$

Theorem 3. *The modified GIS as given in Equations 6 and 7 converges to the desired Maximum Entropy model.*

Proof. Same as that of the classical GIS given in [6], except that the transaction set would be X' instead of X and constraint set is C' instead of C . ■

Time Complexity The time complexity of the modified GIS algorithm is $O(m * |X'| * |C'|)$. Notice that $|C'|$ and $|X'|$ will be smaller than or equal to $|C|$ and $|X|$ respectively. In cases where some *non-closed* itemsets are removed from C , the size of X' will be significantly smaller than the decrease in the size of C . This is because of the reason that for every *non-closed* itemset removed from C , an exponential number of transactions are removed from X . Hence, this new modified GIS algorithm not only computes the correct Maximum Entropy model but also has the advantage of running much faster than the classical GIS algorithm.

6 Improving the execution time of GIS

The running time complexity of GIS is $O(m * |C| * 2^{|I|})$, where m is the number of iterations required by the GIS for convergence. $|C|$ is typically exponential in $|I|$ as there is one constraint for every frequent itemset mined: The set of frequent itemsets are *generally* exponential. Hence, the two important parameters which influence the execution time of GIS are $|C|$ and $|I|$.

In this section, we discuss methods to overcome the effects of these two parameter values. In Section 6.1 we provide a technique to prune the constraint set C and in Section 6.2 we describe a procedure to split I into smaller mutually-exclusive and collectively exhaustive sub-parts, each of which can be handled separately to produce the global distribution.

6.1 Pruning Constraints

Below, we define the term *confidence* of an itemset s_j for a class c_i .

Definition 2. *We call $P(c_i|s_j)$ as the confidence of seeing class c_i in transactions containing the itemset s_j . $P(c_i|s_j)$ (confidence of s_j in c_i) is computed as $P(c_i|s_j) = \frac{P(s_j \cup \{c_i\})}{P(s_j)}$ □*

As discussed earlier, the set of constraints are exponential in $|I|$ and pruning them to a handful of *interesting* itemsets will improve the execution time of GIS. We give an interestingness measure that can be employed for the purpose of pruning.

- The *confidence*, defined in Definition 2, is considered a measure of interestingness as the higher the confidence value the more the association between s_j and c_i . We denote this interestingness measure by $\mathcal{I}(s_j)$.

$$\mathcal{I}(s_j) = \max_{c_i} P(c_i|s_j) \quad (8)$$

An itemset s_j is included in the set of constraints of all the classes, if there exists atleast one class c_i such that s_j has high confidence in c_i (i.e. $P(c_i|s_j) > \text{minconf}$, for a given minimum confidence threshold minconf). Notice that an itemset s_j is either included as a constraint in all the classes or is not included in any of the classes. Since $\sum_{c_i} P(c_i|s_j) = 1$, a large value for a particular $P(c_i|s_j)$ would imply that $\forall c_k \neq c_i, P(c_k|s_j)$ will be very less. If $P(c_i|s_j) > \text{minconf}$, it means that s_j is a good distinguishing factor between the classes. By including s_j in all the classes, we ensure that this information remains in the system of constraints.

Note that an effect of this pruning is that the new distribution that will be computed using the GIS is not the actual distribution of the data. However, it is a good representative in that the distributions computed for all classes are equally affected by this pruning. Hence, the outcome of the classifier is not changed.

6.2 Decomposing the domain I

Using the interestingness measure \mathcal{I} for pruning, the effects of large $|C|$ values on the running time of GIS can be decreased. However the algorithm still has to go through the entire possible transaction set X for every iteration. The transaction set X is the power-set of I , and hence it's size can be quite large for even modest values of $|I|$. To decrease the effects of $|I|$ on the running time of GIS, we divide it into *clusters* $\{I_1, I_2, \dots, I_k\}$ which are mutually exclusive and collectively exhaustive so that the GIS algorithm can be applied to each cluster and the final global distribution can be built by combining the outputs of GIS for these clusters.

Let C_i'' denote the final set of constraints for class c_i obtained after pruning C_i . The division of the set of items I into $\{I_1, I_2, \dots, I_k\}$ is in such a way that a constraint in C_i'' with itemset s_j is fully contained in only one I_i . Such a division of I ensures that I_i and I_j , where $i \neq j$, will not have any item in common and $\bigcup_i I_i = I$. The items of I_i are said to be independent of items in I_j , for every i and j , since there exist no constraints which overlap with both I_i and I_j . If there were items in I_i which had dependency relationships with items in I_j , then there should be atleast one constraint containing these items which would have stayed through the pruning process to remain in the final set of constraints. The final global distribution over I can be obtained by combining the local distributions of $\{I_1, I_2, \dots, I_k\}$ in a naive-bayes fashion.

In the above discussion, I_i and I_j will be independent of each other if there is no constraint which has an overlap with both I_i and I_j . Two cases can exist for such a constraint's absence:

1. The constraint was pruned away based on the interestingness measure \mathcal{I} .
2. The constraint was not frequent enough in any of the classes to enter the system of constraints.

If a constraint was not frequent enough, the statistical relationships it encodes are not strongly represented in the dataset. In such a case, we assume that these relationships are not important and the classifier will not lose much by pruning them away and using only those relationships which are strongly represented in the dataset. On the other hand, if the constraint was removed based on the interestingness measure \mathcal{I} , it means that the constraint was not *distinguishing* enough between the classes. This means, although the itemset was frequent enough in at least one of the classes, it is not important for the task of classification.

7 Related Work

The surprising ability of naive-bayes (NB) at classification has spawned many extensions to it, most of which try to decrease the assumptions they make about the statistical characteristics of the features. TAN (Tree Augmented Naive Bayesian classifier) [4] learns a restricted tree-structured bayesian network taking into account only the most important correlations between pairs of attributes. The Selective Bayesian Classifier [12] deals with highly correlated features by incorporating only some attributes into the final decision process. The Semi-Naive Bayesian Classifier [11] iteratively joins pairs of attributes to relax the strongest independence assumptions.

The above methods try to decrease the strong independence assumptions made by NB. Two variables x, y are conditionally independent given z if for all values of x, y and z , $p(x|y, z) = p(x|z)$. The conditional independence fails, even if a few instantiations of x, y and z don't satisfy this condition. If such a case occurs, the above algorithms would consider storing all elements of the joint probability distribution $p(x, y, z)$ to obtain more accurate estimations. This method is error prone as enough data might not exist to provide reliable probability estimations for each $p(x, y, z)$. [2] introduced *context-specific independence*, which describes independence relations among variables that hold in certain contexts only. *Large Bayes* [18] considers only relationships between variable instantiations. Classification in Large Bayes is done by incrementally building a product approximation of $p(x|c_i)$ with the available subsets of x . The approximation strategy judiciously uses subsets of x available at hand to decrease the independence assumptions it is making. However, all the above methods still assume independence between some pairs of attributes even though they are much lesser than NB. Our classifier, does not assume any relationships between items and uses the well accepted Maximum Entropy framework to effectively handle the independences inferred from sets of features. By representing relationships with itemsets, this classifier considers only independences between variable instantiations without trying to generalize.

Our work is somewhat similar to [16] [14] in that they use association rules to capture relationships. CBA [16] picks out the most *confident* rule from the mined rules for classification. Such single rule based classifiers are prone to unreliable estimates. CMAR [14] overcomes this problem by considering multiple rules at the time of classification. This method gives a formula to compare between groups of rules of each class, but a statistical justification for this formula is not provided.

The principle of Maximum Entropy has been widely used for a variety of natural language tasks including language modeling [23], text segmentation [1], part-of-speech

tagging [22] and prepositional phrase attachment [22]. Ours is the first approach, to our knowledge, which uses it for categorical data classification with mined frequent itemsets as constraints.

The paper [13] discussed the Maximum Entropy model’s failure to accommodate *non-closed* constraints and proposed solutions to this problem. It applied Good-Turing discounting to the observed constraint frequencies and ran the classical GIS algorithm on these constraints. For example, a constraint’s frequency will be changed to $f_j - \epsilon$, instead of its usual observed frequency f_j before including it for GIS algorithm. In this approach there is no guarantee that the constraints would remain consistent with each other after they have been changed. It also reports results on a *fuzzy maximum entropy framework* in which the objective is to maximize the sum of the entropy function and a penalty function which is heuristically selected to penalize deviations from unreliable (less frequent) constraints. Our solution to this problem, fixes the Maximum Entropy model in a simple fashion, without adding any additional complexity, such that the GIS algorithm will converge and will generate the *correct* maximum entropy distribution. Further, this new solution runs much faster than the classical GIS algorithm (see the Section 8).

8 Experiments

In this section, we show the evaluation of ACME classifier on 11 datasets chosen from the UCI Machine-Learning Repository [19]. We compare ACME with four other state-of-the-art classifiers: (1) the widely known and used decision tree classifier C4.5 [20] (2) a recently proposed associative classifier CBA [16] (3) the naive-bayes classifier and (4) the Tree-Augmented naive-bayes (TAN) classifier [4].

Continuous variables in the dataset were first discretized to interval attributes using entropy based discretization as given in [9]. The same discretized datasets were used for all the classification methods, except for C4.5 which accepts continuous valued variables directly. In all the experiments, accuracy is measured using the 10-fold cross validation. We implemented the classification methods naive-bayes and ACME in C++ and the classifiers TAN and C4.5 were borrowed from the *weka* machine learning software [24]. The results for the CBA classifier were borrowed from [14]. Experiments were run on an Intel Celeron dual-processor machine with two 3.0GHz processors and running RedHat Linux 9.0.

Figure 1 gives the characteristics of the ten datasets used in the evaluation of ACME. The column *fsets* in this figure, gives the size of the union of all the frequent itemsets extracted from every class. As given in Equation 1, this set is denoted as S . The next column “largest cluster (I_c)” gives the size of the largest cluster in all the classes. Column “closed trans.” gives the percentage of transactions in the cluster I_c which had non-zero probability (see Section 5). Only these transactions were used in the learning phase, instead of the entire domain X . Apart from the transactions which are not *closed*, we also removed transactions which satisfy a constraint whose support was 0. The μ values of such constraints were set to 0 and they were removed from the learning phase. The last column gives the number of constraints in the largest cluster (I_c) among all the classes. If a dataset has multiple classes which contain the largest cluster, it is

Dataset	Attrs.	Classes	size	<i>fsets</i>	largest cluster(I_c)	<i>closed trans.</i>	Cons. in I_c
Australian	14	2	690	354	17	10.1%	263
Breast	10	2	699	189	17	8.89%	180
Cleve*	13	2	303	246	15	9.96%	204
Diabetes	8	2	768	85	15	7.42%	61
German	20	2	1000	54	20	9.66%	44
Heart*	13	2	270	115	9	55.1%	95
Hepatitis	19	2	155	32.1k	17	1.3%	153
Lymph*	18	4	148	29	9	12.1%	14
Pima	8	2	768	87	15	8.6%	55
Waveform	21	3	300	99	18	1.3%	24

Fig. 1. Characteristics of the Datasets

marked with a ‘*’. For these datasets, the last two columns give the average calculated over these clusters.

Looking at Figure 1 we can see that even if the decrease in the size of the set of constraints C is modest, the decrease in the size of X is significant. The decrease in the size of X is nearly 99% for datasets *Waveform* and *Hepatitis* as the decrease in the size of their constraint set C is very large. Since the complexity of GIS is proportional to $|X|$ (calculated in Section 3.1), the new GIS algorithm will run much faster than the classical GIS algorithm in the presence of *non-closed* constraints.

If an attribute a is discretized into $\{a_1, a_2, \dots, a_z\}$ binary variables, utmost one of these binary variables can be present in any transaction. This fact is not represented in the set of constraints for any class. We explicitly add constraints of the form $(\{a_i, a_j\}, 0)$, for every a_i and a_j , to the set of constraints for every class to ensure the presence of this information. Each constraint means that the probability of seeing a_i and a_j together in a transaction is 0. We call these constraints as *zero-constraints*. This implementation detail improved ACME’s accuracy. None of the classifiers that we know, including the ones with which ACME is compared in this section, had the provision to improve their accuracy by using the *zero-constraints*.

In our experiments, the parameters for all the classifiers were set to the standard values as reported in the literature. For CBA the minimum support was set to 0.01 (i.e. 1%) and the minimum confidence is set to 50% with the pruning option enabled. The minimum support threshold for ACME was set to 0.1 and the threshold confidence is set to 0.8. TAN classifier is run with default settings as used in the *weka* toolkit.

Figure 2 gives a detailed description of the datasets employed in the experiments and the accuracies of various classifiers. The winners for each dataset are highlighted in bold font. As can be seen from this table, ACME outperforms all the other classifiers. Out of the ten datasets employed for the testing, ACME wins in five cases. NB wins in 3 datasets and the remaining algorithms perform the best on one dataset each. However, ACME has an additional advantage of handling missing values. Even for other datasets in which ACME does not win, it comes close to the winner.

An important point that needs to be noted is, for the *Hepatitis* and *Waveform* datasets, eventhough the new modified GIS algorithm is made to run on only 1.3% of

Dataset	NB	C4.5	CBA	TAN	ACME
Australian	84.34	85.50	84.9	84.9	85.36
Breast	97.28	95.13	96.3	96.56	96.49
Cleve	83.82	76.23	82.8	82.5	83.82
Diabetes	75.78	72.39	74.5	76.30	77.86
German	70.0	70.9	73.4	73.0	71.3
Heart	83.70	80.0	81.87	81.85	82.96
Hepatitis	80.22	81.93	81.8	81.29	82.58
Lymph	83.10	77.0	77.8	84.45	78.4
Pima	76.17	74.34	72.9	76.30	77.89
Waveform	80.82	76.44	80.0	81.52	83.08

Fig. 2. Accuracies of various Classifiers on UCI ML Datasets

the entire possible domain, it is achieving significant results. ACME was the winner for these two datasets. The reason behind this is that the eliminated portion of the domain is actually set to the correct probability 0. This substantiates our earlier argument that not only ACME runs faster, but also that the distribution it generates will be more closer to the distribution we are trying to mimic. Another important point to note is that the *naïve-bayes* classifier is performing better than the remaining classifiers.

9 Conclusions

In this paper we proposed a new classifier based on the paradigm of association rules. Though recent classifiers involving this paradigm have shown their techniques to be accurate, their approaches to build a classifier either assume not-observed statistical relationships in the dataset or have no statistical basis. Our classifier ACME uses the well-known Maximum Entropy principle to build a statistical model for the purpose of classification. We show that the classical Maximum Entropy model cannot have a solution for some input cases and propose a new Maximum Entropy model which can fit in all the input cases. We gave a modification to GIS, an algorithm to compute the classical Maximum Entropy model, to compute the new model. The modified GIS, apart from computing the correct Maximum Entropy model, also runs at least as fast as the original GIS. We also described techniques to make GIS tractable for large feature spaces – we provided a new technique to divide a feature space into independent clusters each of which can be handled separately. Finally, our experimental results show that our classifier is generally more accurate than the existing classification methods.

References

1. D. Beferman, A. Berger, and J. Lafferty. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210, 1999.
2. C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian-networks. In *Uncertainty in Artificial Intelligence(UAI)*, 1996.

3. P. Clark and T. Niblett. The **cn2** induction algorithm. *Machine Learning*, 2:261–283, 1989.
4. P. Clark and T. Niblett. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.
5. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.
6. J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.
7. G. Dong, X. Zhang, L. Wong, and J. Li. Classification by aggregating emerging patterns. In *Discovery Science*, Dec. 1999.
8. R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.
9. U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Intl. Joint Conf. on Artificial Intelligence(IJCAI)*, pages 1022–1029, 1993.
10. I. Good. Maximum entropy for hypothesis formulation, especially for multidimensional contingency tables. *Annals of Mathematical Statistics*, 34:911–934, 1963.
11. I. Kononenko. Semi-naive bayesian classifier. In *European Working Session on Learnign*, pages 206–219, 1991.
12. P. Langley and S. Sage. Induction of selective-bayesian classifiers. In *Uncertainty in Artificial Intelligence(UAI)*, pages 399–406, 1994.
13. R. Lau. Adaptive statistical language modeling. Master’s thesis, Massachusetts Institute of Technology, Cambridge, MA, 1994.
14. W. Li, J. Han, and J. Pei. **CMAR**: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, 2001.
15. T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40(3):203–228, Sept. 2000.
16. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proc. of 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, Aug. 1998.
17. D. Meretakis, H. Lu, and B. Wuthrich. A study on the performance of large bayes classifier. In *ECML*, pages 271–279. LNAI, 2000.
18. D. Meretakis and B. Wuthrich. Extending naive-bayes classifiers using long itemsets. In *KDD*, pages 165–174, 1999.
19. C. Merz and P. Murphy. **UCI** repository of machine learning databases, 1996. <http://cs.uci.edu/mlearn/MLRepository.html>.
20. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
21. A. Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing. Technical Report IRCS Report 97-98, Institute for Research in Cognitive Science, University of Pennsylvania, May 1997.
22. A. Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, Institute for Research in Cognitive Science, University of Pennsylvania, 1998.
23. R. Rosenfeld. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. PhD thesis, Carnegie Mellon University, 1994.
24. I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2 edition, 2005.