Prediction with Local Patterns using Cross-Entropy

Heikki Mannila
Microsoft Research
mannila@microsoft.com

Dmitry Pavlov
University of California at Irvine
pavlovd@ics.uci.edu

Padhraic Smyth
University of California at Irvine
smyth@ics.uci.edu

Abstract

Sets of local patterns in the forms of rules and co-occurrence counts are produced by many data mining methods such as association rule algorithms. While such patterns can yield useful insights it is not obvious how to synthesize local sparse information into a coherent global predictive model. We study the use of a cross-entropy approach to combining local patterns. Each local pattern is viewed as a constraint on an appropriate high-order joint distribution of interest. Typically, a set of patterns returned by a data mining algorithm under-constrains the high-order model. The cross-entropy criterion is used to select a specific distribution in this constrained family relative to a prior. We review the iterative-scaling algorithm which is an iterative technique for finding a joint distribution given constraints. We then illustrate the application of this method to two specific problems. The first problem is combining information about frequent itemsets. We show that the cross-entropy approach can be used for query selectivity estimation for 0/1 data sets. The results show that we can accurately answer a large class of queries using just a small set of aggregate information. The second problem involves sequence modeling using historical rules, with an application to protein sequences. We conclude that viewing local patterns as constraints on a high-order probability model is a useful and principled framework for prediction based on large sets of mined patterns.

1 Introduction

Several data mining and rule induction methods provide partial information specified as conditional probabilities (e.g., probabilistic rules) or joint probabilities (e..g., frequent itemsets). Such partial information provides useful insight into the data, but it is not obvious how to combine such pieces of knowledge for other purposes. We consider the discrete-valued problem, thus, we assume that a vector-valued random variable $x = (x_1, x_2, \ldots, x_d)$ takes values from some finite alphabet. We are interested in estimating the joint distribution $P(x) = P(x_1, x_2, \ldots, x_d)$, or some function of $P(x)$ such as a conditional distribution, given sets of local rules containing information of the form $P(x_i = a|x_j = b, x_k = c)$ and/or $P(x_i = a, x_j = b, x_k = c)$. In general, if for example each x_i can take m distinct values, specification of $P(x)$ will require $O(m^d)$ different entries in the joint probability table. It is clear that estimating the full joint distribution from data is often impractical given even moderate values of m and d, because of the exponential growth of the number of different probabilities which must be specified.

The primary focus of many current data mining algorithms is to extract low-order information in an efficient manner. For example, association rule algorithms [1] essentially compute all conditional probabilities of the form $p(x_i = 1 \mid x_{j_1} = 1 \wedge \cdots \wedge x_{j_k} = 1)$ which are above a given threshold and for which $p(x_{j_1} = 1 \wedge \cdots \wedge x_{j_k} = 1)$, the support, is also above a prespecified threshold. These types of patterns are local to small sets of variables and specific values of these variables. There are numerous similar rule-finding algorithms that can efficiently produce large lists of such local patterns. Typically the patterns are chosen based on the fact that they differ from the expected norm, i.e., they are informative in some sense relative to a uniform prior distribution on $P(x_1, x_2, \ldots, x_d)$.

We study the problem of how to combine such local partial information into a single global model which can be used for prediction. We retain the good properties of both worlds: we get understandable local patterns and still are able to use them in prediction or estimation tasks. While there are already some approaches to combining local patterns for specific problems like classification (e.g., [5]), we focus explicitly here on a probabilistic framework. We will not discuss how the patterns or rules can be found from the data since there is a large body of techniques available to do this already. Instead we focus on the problem of once given the patterns, how can one combine them for prediction?

Combining local patterns to form a global model for $P(x_1, x_2, \ldots, x_d)$ is worthwhile from two different perspectives. Firstly, from a data mining perspective, it offers a framework for linking sets of local patterns with the more well-established world of global models, where evaluation, prediction, estimation, model selection, and so forth, can be handled in a coherent and consistent manner. Secondly, from an applications perspective, the ability to quickly find local patterns and then use these to construct a model for $P(x_1, x_2, \ldots, x_d)$ can offer distinct advantages in accuracy, efficiency, and interpretability compared to more conventional approaches.

2 Problem Statement and the Iterative Scaling Algorithm

In a data mining framework we will assume that our patterns and rules are statements about frequency counts in the data that can be represented as constraints. For example, if we know that the number of co-occurrences of $(x_j = 1, x_k = 1)$ in the training data is n_{jk}, then we represent this as a constraint by writing a constraint equation of the form

$$\sum_x P(x) k(x|i) = d_i. \tag{1}$$

where the ith indicator function $k(x|i)$ yields 1 if x satisfies the ith constraint and 0 otherwise and d_i denotes the fraction of elements that satisfy this constraint, $1 \leq i \leq C$, C being the total number of such constraints. In this case the indicator function is 1 when $x_j = 1$ and $x_k = 1$ and 0 otherwise, so that the left-hand side of the above equation sums to $P(x_j = 1, x_k = 1)$. The right-hand side d_i can be set to n_{jk}/n, where n is the total number of data points, thus constraining the probability on the left to take this maximum-likelihood value. Let the total number of such constraints be C (this represents the number of patterns returned by the data mining algorithm). We can view the constraints as requirements that are imposed on the otherwise unknown joint probability distribution $P(x)$.

Typically this will *underconstrain* $P(x)$, i.e., there is an infinite set of probability distributions which satisfy the constraints. To choose a specific $P(x)$ we invoke the principle that the $P(x)$ which is chosen from within this set is the one closest to a specified prior $\pi(x)$ in a cross-entropy sense, i.e., $P_{CE} = \arg\min_P CE(P, \pi)$, where $CE(P, \pi) = \sum_x P(x) \log \frac{P(x)}{\pi(x)}$ is the cross-entropy between P and π. In this paper we will choose $\pi(x)$ to be uniform, and thus, minimizing $CE(P, \pi)$ is equivalent to maximizing the entropy of P subject to the given constraints. Maximum entropy has a long history as a criterion for model selection that we will not dwell on here (e.g., [4]): in a certain sense it picks the distribution which adds the least additional information beyond that already given.

Assume one of the constraints in Equation 1 is $k(x|0) = 1$ for all x and $d_0 = 1$ to ensure that the distribution sums up to 1. The method of undetermined Lagrangian multipliers can then be used to find P_{CE} satisfying the constraints. A slightly surprising (and useful) theorem is that P_{CE} can be expressed in the following general functional form:

$$P_{CE}(x) = \pi(x) \mu_0 \prod_{i=1}^{C} \mu_i^{k(x|i)}$$

where the μ_i, $i = 0, \ldots, C$ are positive constants satisfying for all i

$$\mu_0 \sum_x \pi(x) k(x \mid i) \prod_{j=1}^{C} \mu_j^{k(x|j)} = d_i. \tag{2}$$

The general problem is thus reduced to the problem of finding a set of μ_i from the set of Equations 2. It can be shown that a solution to Equation 2 exists and is unique if and only if the constraints do not contradict each other [3]. For example, if all the constraints only consist of relative frequency counts in the data (as is the case in results presented below), then they will be consistent automatically.

The generalized iterative scaling algorithm is well known in the statistical literature as an iterative technique which converges to the solution P_{CE} for problems of this general form (see, for instance, [3]). The algorithm can be used to obtain a numerical solution for the parameters satisfying the given constraints. A high-level outline of the algorithm is as follows.

```
1.  Choose an initial approximation to $P(x)$
2.  While (Not all Constraints are Satisfied)
        For (i varying over all constraints)
            Update $\mu_0$;
            Update $\mu_i$;
        End;
    EndWhile;
3.  Output the constants $\mu_i$
```

The update rules $\mu_{i,t}$ corresponding to constraint i at iteration t will be:

$$S_{i,t} = \sum_x P_t(x) k(x \mid i) \tag{3}$$

$$\mu_{i,t+1} = \mu_{i,t} \frac{d_i}{S_{i,t}}, \tag{4}$$

where P_t refers to our estimate of P at iteration t. This algorithm is well-known in statistics (and related applications such as statistical language modeling), however, to our knowledge this paper is the first illustration of its use in a data mining context.

3 Query Selectivity Estimation

In this section we present our first application, query selectivity estimation for 0/1 relations. The method actually amounts to combining association rules for prediction. Suppose we are given a table r of 0s and 1s; denote the schema (set of column headers) of the table by R. A *conjunctive query* Q over R is an expression of the form $A_1 = b_1 \wedge A_2 = b_2 \cdots \wedge A_k = b_k$, where $k \geq 1$, for each i we have $A_i \in R$ and $b_i \in \{0, 1\}$. The size of the answer to the query Q is the number of rows of r that satisfy Q, and the query selectivity is the fraction of the rows that satisfy the queries. Query selectivity estimation tries to find the approximate size of the answer to a conjunctive query without actually computing the query. This problem has wide applications in database query optimization, and scores of methods have been developed for this task [10, 8].

A typical solution to the query selectivity problem is to maintain for each attribute $A_i \in R$ information about the selectivity $c(A_i, b_i)$ of the conditions $A_i = b$ for $b_i = 0, 1$ (obviously, for binary data only one number per attribute is needed). The selectivity $c(A_i, b_i)$ can be viewed as a constraint on the margin of the variable A_i. Assuming independence of attributes, we can estimate the selectivity of query Q by $\prod_{i=1}^{k} c(A_i, b_i)$. This independence model is in fact the maximum entropy distribution given only counts on individual attributes, i.e., it can be viewed as a special case of the more general approach presented below.

To use the cross-entropy approach to give better estimates we need additional information to constrain the distribution. The *frequent sets* from association rules [2] provide a way of defining useful marginal information. Given a set of attributes $X \subseteq R$, the *frequency* $f(X)$ of X in r is defined as the fraction of rows of r that have a 1 in all the columns of X: $f(X) = |\{t \in r \mid t[A] = 1 \text{ for all } A \in X\}|/n$, where n is the number of rows in table r. Given a threshold σ, the collection of σ-frequent sets is the collection of all subsets of R (and their frequencies) whose frequency is at least σ.

We use the cross-entropy approach to obtain improved estimates for query selectivity as follows. Let S be the collection of σ-frequent sets for some given σ. Given a conjunctive query $A_1 = b_1 \wedge A_2 = b_2 \cdots \wedge A_k = b_k$, first find all frequent sets $X \in \mathcal{S}$ such that $X \subseteq \{A_1, A_2, \ldots, A_k\}$, i.e., those frequent sets which are subsets of the query attributes. Using these marginal counts, then build the maximum entropy distribution for the state space consisting of all vectors $x = (x_1, \ldots, x_k)$ of length k, where each x_i is either 0 or 1. After this has been done, answer the conjunctive query is simple: we just find $P(x)$ for the particular x corresponding to the query Q.

The method is applicable to any type of queries, not just conjunctive queries. As we build an estimate of the whole joint distribution on the k attributes occurring in the query, using this estimate we can compute approximate answers to any query. The running time per query of the method has the form $O(Fk + IC2^k)$, where k is the number of variables in the query, F is the size of the frequent set collection, C is the number of frequent sets that are included in the set of variables of the query, and I is the number of iterations (typically 10 to 20 for the experiments below). The exponentiality in k makes the method impractical for about $k > 10$, but for queries with a small number of variables the method is quite fast. Note that the running time is independent of the size of the original data set.

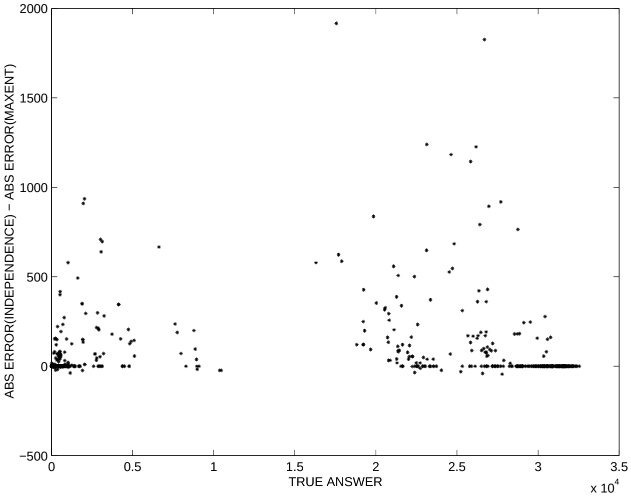

Figure 1: Relative performance of the independence method and the maximum entropy method as a function of the true answer.

For brevity, we present results only on one data set. The dataset contains information about the buying behavior of certain customers. It has 249 attributes and 32711 rows. The data are relatively sparse, containing 98654 1s, and hence we would need about 130 kB to represent the data (one byte for each one in the data set, plus some overhead to indicate where each observation starts). A textual representation takes 335 kB.

Using the independence assumption requires that we store 249 numbers, about 1 kB, i.e., the fraction of 1s for each attribute. For the maximum entropy method we experimented by computing frequent sets using a threshold of $\sigma = 0.005$ (selected to be small enough to ensure that non-trivial combinations of frequent sets are generated). This yielded 453 frequent sets for a total of 702 constraints, and the space needed for this representation of the dataset is about 4 kB. Thus the maximum entropy method uses about 4 times as much space as the pure independence assumption, but about a fraction of 1/30 of the space needed for the whole dataset. The threshold of 0.005 means that we know the size of each "1s only" query $A_1 = 1 \wedge \cdots \wedge A_k = 1$, provided the answer is at least 165.

We generated random queries with $k = 4$ and $k = 5$ conjuncts by selecting attributes A_i at random and adding the conjunct $A_i = 0$ with probability 0.8 and the conjunct $A_i = 1$ with probability 0.2. Due to this query generation method, the queries tend to have either a relatively small or relatively large answer.

The maximum entropy method produces consistently very good estimates of the actual query size. The method performs significantly better than the estimation based only on the the frequencies of single attributes. For about 3/4 of the queries the query attributes were such that maximum entropy method had exactly the same information as the independent attribute method, and in those cases the methods produced, of course, the same answer. In the remaining 25% of the cases the maximum entropy method typically produced much better approximations. Figure 1 shows for each of 1000 queries the difference between the absolute errors for the maximum entropy method and for the attribute independence method. The figure shows that the maximum entropy method only rarely produces an estimate that is worse than the estimate produced by the attribute independence method, whereas for a large set of queries the use of the maximum entropy method gives significantly better answers.

As mentioned in the beginning of this section, the approach we have described can obviously be used to combine association rules for prediction. While some work has been done on finding predictive association rules (see, e.g., [7]), we are not aware of work on actually combining association rules.

Given a set $\{X_i \Rightarrow A\}$ of association rules, let k be the number of distinct attributes $\{B_1, \ldots, B_k\}$ occurring in the sets X_i. We build the joint distribution over the 2^k different states. When we encounter an input row t with values $b_1, \ldots, b_k$ for the attributes B_i, we use the corresponding entry in the joint distribution to predict the value of A. For brevity, we omit the details.

4 Modeling Sequences with Probabilistic Rules

In this section we apply the methodology of the earlier sections to probabilistic modeling of *sequential data*. Again we consider discrete-valued variables with alphabet Σ. For any symbol w occurring in the sequence we define the *L-history* of w, $H_L(w)$, as the L symbols preceding w in the sequence. For discrete-valued sequences, there exist several data mining algorithms which can extract *sequential rules* of the general form "if event A happens in $H_L(w)$ then w will occur at position t with probability p," or equivalently, conditional probability statements of the form $p(w|A \in H_L(w)) = p$. Provided with a set of such probabilistic rules from a training data set, we can view such rules as constraints on the unknown joint distribution $p(w, H_L(w))$. Thus, here we are interested in estimating the distributions of the form $P_L(w, H_L(w))$ or $P(w|H_L(w))$.

As in the non-sequential multivariate case there are many measures for defining interesting rules and many algorithms for finding such sets from data. We used the average mutual information between A and w as a simple measure to find and rank different possible rules (since we are primarily interested here in how to combine such patterns rather than how they are found). We also limited our attention to simple events of the form "symbol v occurs k positions before w" in the sequence. The maximum value of k is defined to be the history length. For example, for history-length 1 we would only be considering bigram terms $p(w(t)|w(t-1))$ in our model.

The problem of estimating $p(w|H_L(w))$ given constraints in the form of rules described above is in its general form equivalent to the problem discussed in Section 2 of choosing a specific distribution from a constrained family of distributions. Once again one can invoke the maximum entropy principle to choose among distributions and use the iterative scaling algorithm to solve the associated optimization problem. For sequential data, the problem is actually a little more complex in form than the multivariate case sketched in Section 2. We use the same approach as used with "trigger-models" in natural language modeling [9]. For full details see [6], where we also show that each iteration of the iterative scaling algorithm scales roughly as NC where N is the length of the training sequence and C is the total number of constraints (rules).

We applied this approach to modeling of protein sequence data. The purpose of this experiment was to explore the utility of probabilistic rules for modeling sequence structure beyond simple bigram models. We picked a well-known set of 585 hemoglobin protein sequences (about 85,000 symbols in total) available on-line at `http://www.isb-sib.ch/` and used various combinations of prior, bigram, and rule-based models for modeling the conditional distribution of the current symbol given its history. The size of the alphabet Σ is 20. We generated a 10-fold cross-validated estimate of the out-of-sample cross-entropy for each model, namely, the sum of the negative log probabilities for each observed symbol in the test sequence conditioned on its history. The higher the probabilities a model assigns to observed symbols the lower (and better) the cross-entropy score will be. A simple baseline score can be defined as the cross-entropy of a model with a uniform distribution for $p(w)$, namely $\log|\Sigma| = \log(20) = 4.3219$ bits. Thus, all decreases in reported cross-entropy are relative to this baseline uncertainty.

The upper panel of Table 2 shows the average relative decrease in cross-entropy using only priors and a bigram model. As expected, bigrams do better than priors alone on both the training and test data. The two pairs of columns in the bottom panel of Table 2 show the results for combining rules with priors, and combining rules with priors and bigrams, respectively, where we tried rules with different history lengths. In the "rules with priors" there were no single-lag (bigram) terms in the model, i.e., all rule events were at least 2 positions back. The rules clearly provide additional information beyond priors or bigrams. The relative decrease in entropy using rules is close to 1 bit and almost 5 times higher than for the priors alone, and 2.5 times higher than for all bigrams alone (upper

Priors		Priors and bigrams	
Train	Test	Train	Test
5.94%	5.92%	13.05%	11.82%

History length	Priors and rules		Priors, bigrams and rules	
	Train	Test	Train	Test
2	11.84%	10.66%	16.71%	14.55%
3	14.80%	13.28%	19.18%	16.86%
6	18.62%	16.64%	22.39%	19.80%
11	20.63%	18.14%	24.06%	21.12%
16	21.61%	18.60%	25.22%	21.54%
21	22.12%	18.67%	25.68%	21.65%

Table 1: Relative percentage decrease in train and test cross-validated cross-entropy for different models, where $|\Sigma| = 20$ and at most the top 6 rules with mutual information greater than 0.001 bit are retained for each symbol w.

panel). Interestingly, models based on priors and rules only (without bigrams, columns 2 and 3 of the lower panel in table 3) consistently outperform the full bigram model both in and out-of-sample. For models with rules, the performance of the model improves as the history length increases, and the methodology appears relatively robust to overfitting on this data.

In [6] we report a variety of other experiments of this nature on this data set using rule-based probability models. The major drawback of the technique is the computational complexity of iterative scaling; for example, online modeling using the approach presented here would be impractical. Possible generalizations are numerous, such as allowing a more general language for rules with multiple symbols, disjunctions, and so forth. Parameters such as history lengths, numbers of rules, bigrams, etc., could all be chosen automatically in principle using cross-validated estimates of cross-entropy.

5 Conclusions

Data mining algorithms are useful for efficiently finding patterns in the form of conjunctive expressions (rules) with attached frequencies of occurrence from large data sets. However, the problem of combining these patterns into a coherent global model has been relatively unexplored. In this paper we proposed a straightforward approach to this problem by viewing the local patterns as constraints on an unknown high-order distribution. The iterative scaling procedure can then be used to find the the distribution that is closest in a cross-entropy sense to a specified prior. We illustrated the application of this idea on two different problems, query selectivity estimation and sequence modeling. In both cases the method using local patterns provided significant improvements over conventional alternatives. The method we have proposed is applicable to various other settings.

Acknowledgments

The work of DP and PS described in this paper was supported by the National Science Foundation under Grant IRI-9703120.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'93)*, pages 207 – 216, Washington, D.C., USA, May 1993. ACM.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307 – 328. AAAI Press, Menlo Park, CA, 1996.

[3] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[4] E. T. Jaynes. *Probability Theory - the Logic of Science.* Physics, Washington University, St. Louis, MO 63130, USA, 1994.

[5] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 80–96, 1998.

[6] H. Mannila, D. Pavlov, and P. Smyth. Prediction with local patterns using cross-entropy. Technical Report UCI-ICS-TR-99-28, Information and Computer Science, University of California, Irvine, 1999.

[7] N. Megiddo and R. Srikant. Discovering predictive association rules. In *Proc. of the 4th Int'l Conference on Knowledge Discovery in Databases and Data Mining*, 1998.

[8] R. Ramakrishnan. *Database Management Systems.* McGraw-Hill, 1997.

[9] R. Rosenfeld. A maximum entropy approach to adaptive statistical language modelling. *Computer Speech and Language*, 10(3):187–228, July 1996.

[10] J. D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume I. Computer Science Press, Rockville, MD, USA, 1988.