Goal-Oriented Clustering

David Maxwell Chickering
dmax@microsoft.com

David Heckerman
heckerma@microsoft.com

Christopher Meek
meek@microsoft.com

John C. Platt
platt@microsoft.com

Bo Thiesson
thiesson@microsoft.com

May 2000

Technical Report
MSR-TR-00-82

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Abstract

We introduce goal-oriented clustering, a process that clusters items with the explicit knowledge that the ultimate use of the clusters is prediction. In this approach, we use data on a set of target variables (those we want to predict) and a set of input variables (those we do not want to predict) to learn a graphical (generative) model with a single hidden layer of discrete variables $\mathbf{H}$. The states of $\mathbf{H}$ correspond to clusters. We describe a generalized EM algorithm for learning the parameters of this class of models and provide a convergence guarantee. We compare our goal-oriented approach to a standard clustering approach on the task of targeted advertising on a web site.

1 Introduction

Clustering, the process of assigning similar data points to the same group, is used frequently in science, engineering, and marketing applications (e.g., [2, 8]). Traditional clustering algorithms include similarity-based approaches such as hierarchical agglomerative clustering (HAC) and K-means (e.g., [5]) and model-based approaches that cluster by learning mixture models with EM (e.g., [1, 2]).

A common application of clustering is exploratory data analysis. For example, Cheeseman et al. [3] discovered new star types by clustering spectral data. In addition, especially in the marketing arena, clustering is used for prediction. For example, a company that has a web site can cluster users by the pages they visit on the site. Marketing executives for that company can then view use patterns in each cluster and identify labels for the clusters such as "high-tech users", "teen-age users", and so on. Advertising campaigns and/or product promotions can then be tailored to each cluster based on the marketing executives interpretation of those clusters. In this process, the marketing executives are trying to find clusters based on web use with the goal of predicting response to advertising and/or promotions.

Unfortunately, traditional clustering algorithms do not have this goal-oriented focus. In the marketing example, suppose we have both web-use and advertising response data. When we apply one of the traditional clustering methods just described to data that includes observations of both web use and advertising response, both pieces of information are treated on an equal footing. None of these methods use the additional information that only the response needs to be predicted.

In this paper, we introduce the process of *goal-oriented clustering*—the process of forming clusters for prediction. The specific goal-oriented-clustering approach that we take is model-based. That is, we cluster by learning a model. To describe the model, we need some notation. We denote a variable by a capitalized token (e.g., X, X_i, Θ), and the state or value of a corresponding variable by that same token in lower case (e.g., x, x_i, θ). We denote a

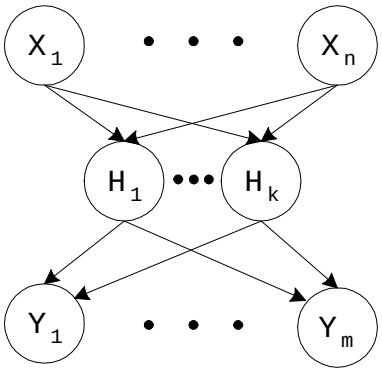

Figure 1: Structure of goal-oriented-clustering model.

set of variables by a bold-face capitalized token (e.g., $\mathbf{X}, \mathbf{X}_i, \mathbf{Pa}_i$). We use a corresponding bold-face lower-case letter (e.g., $\mathbf{x}, \mathbf{x}_i, \mathbf{pa}_i$) to denote an assignment of state or value to each variable in a given set. We use $p(X = x|Y = y)$ (or $p(x|y)$ as a shorthand) to denote the probability that $X = x$ given $Y = y$. We also use $p(x|y)$ to denote the probability distribution for X given Y (both mass functions and density functions). Whether $p(x|y)$ refers to a probability, a probability density, or a probability distribution will be clear from context.

Our goal-oriented-clustering model consists of a set of target variables $\mathbf{Y} = \{Y_1, \ldots, Y_m\}$ (the variables we want to predict), a set of input variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ (other variables that we do not want to predict), and a set of hidden variables $\mathbf{H} = \{H_1, \ldots, H_k\}$. Our model is a graphical (generative) model for these variables with structure shown in Figure 1. The model is equivalent to a fully connected single-layer neural network. Viewed as a graphical model, this model encodes a conditional distribution of $p(\mathbf{H}, \mathbf{Y}|\mathbf{X}, \Theta) = \prod_{i=1}^{k} p(H_i|X_1, \ldots, X_n, \Theta_H) \prod_{j=1}^{m} p(Y_j|H_1, \ldots, H_l, \Theta_Y)$ where $\Theta = (\Theta_H, \Theta_Y)$ are the parameters of the model.

The clusters in this model correspond to the possible states of $\mathbf{H}$. For example, in a model where $\mathbf{H}$ consists of two ternary variables, there are nine clusters. In typical applications of clustering, the number input variables is large (greater than 100) whereas the number of clusters is small (less than 100). Consequently, our model typically has a bottleneck architecture, in which we compress the information in $\mathbf{X}$ to predict $\mathbf{Y}$.

Although the restriction is not necessary, for the work we describe in this paper, we assume a softmax logistic model for each $p(H_i|X_1, \ldots, X_n, \Theta_H)$ and an unconstrained multinomial distribution for each $p(Y_j|H_1, \ldots, H_l, \Theta_Y)$. The latter distributions can be uncon-

strained because the number of states of $\mathbf{H}$ is limited.

The paper is organized as follows. In Section 2, we describe a conditional version of the generalized EM algoritm for learning parameters of the goal-oriented-clustering model, and prove that this algorithm achieves a local maximum. In Section 3, we present a specific application of goal-oriented clustering: targeted advertising with inventory management. Unlike the marketing application described in this introduction, this new application can be evaluted quantitatively. In Section 4, we demonstrate that goal-oriented-clustering outpeforms ordinary clustering for this application.

2 Learning methods

A goal-oriented-clustering model can be learned with a variety of standard techniques including gradient descent, stochastic gradient descent, and generalized EM (GEM). In this work, we use a GEM algorithm because it requires no parameter tuning and has a convergence guarantee (as we show). In this section, we derive a GEM algorithm for learning the MAP parameters of a goal-oriented-clustering model with fixed structure.

As opposed to the standard unconditional Q function, $Q(\theta : \theta^n)$, our GEM algorithm is based on a *conditional Q function*:

$$Q(\theta : \theta^n|\mathbf{x}) = \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x}, \mathbf{y}, \theta^n) \log p(\mathbf{h}, \mathbf{y}|\mathbf{x}, \theta) p(\theta)$$

where $p(\theta)$ is the prior over the model parameters and $p(\mathbf{h}, \mathbf{y}|\mathbf{x}, \theta)$ is the conditional distribution encoded by the model. The E step of the GEM algorithm computes the $Q(\theta : \theta^n|\mathbf{x})$ function by performing inference in a graphical model to compute $p(\mathbf{h}|\mathbf{x}, \mathbf{y}, \theta^n)$. The M step then chooses θ to improve $Q(\theta : \theta^n|\mathbf{x})$ with respect to $Q(\theta^n : \theta^n|\mathbf{x})$.

It is not hard to show that improving $Q(\theta : \theta^n|\mathbf{x})$ improves the MAP (or ML estimates if $p(\theta)$ is flat) for the conditional probability $p(\mathbf{y}|\mathbf{x}, \theta)$. Namely,

$$\begin{aligned} Q(\theta : \theta^n|\mathbf{x}) &= \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x}, \mathbf{y}, \theta^n) \log p(\mathbf{y}|\mathbf{x}, \theta) p(\theta) + \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x}, \mathbf{y}, \theta^n) \log p(\mathbf{h}|\mathbf{x}, \mathbf{y}, \theta) \\ &= \log p(\mathbf{y}|\mathbf{x}, \theta) p(\theta) + H(\theta : \theta^n). \end{aligned}$$

As a consequence of Jensen's inequality $H(\theta : \theta^n) \leq H(\theta^n : \theta^n)$. Thus, choosing θ such that $Q(\theta : \theta^n) > Q(\theta^n : \theta^n)$ improves the MAP estimate for θ (or ML estimate).

When learning the parameters of a goal-oriented-clustering model from i.i.d. data using our GEM algorithm, our conditional Q function becomes

$$Q(\theta : \theta^n|\mathbf{x}) = \sum_{\text{case } i} p(\mathbf{h}^i|\mathbf{y}^i, \mathbf{x}^i, \theta^n) \log p(\mathbf{h}^i, \mathbf{y}^i|\mathbf{x}^i, \theta) p(\theta)$$

We assume that Θ_H and Θ_Y are independent a priori, allowing us to factor the Q function as follows:

$$\begin{aligned} Q(\theta : \theta^n|\mathbf{x}) &= \sum_{\text{case } i} p(\mathbf{h}^i|\mathbf{y}^i, \mathbf{x}^i, \theta^n) \log p(\mathbf{h}_j|\mathbf{x}_j, \Theta_\mathbf{H}) p(\Theta_\mathbf{H}) + \\ &\quad \sum_{\text{case } i} p(\mathbf{h}^i|\mathbf{y}^i, \mathbf{x}^i, \theta^n) \log p(\mathbf{y}_j|\mathbf{h}_j, \Theta_\mathbf{Y}) p(\Theta_\mathbf{Y}) \end{aligned} \quad (1)$$

Because the parameters $\Theta_\mathbf{H}$ and $\Theta_\mathbf{Y}$ are a priori independent, they also must be variationally independent, and hence can be maximized independently. The second term can be maximized in closed form when using the conjugate Dirichlet prior for the multinomial likelihoods. The first term can be improved with (e.g.) a line search in the direction of the gradient. In our experiments, we use a standard golden section search (e.g., [6]).

Note that, given Equation 1, we can view the M step of the GEM algorithm as an improvement of the complete-data posterior, $p(\mathbf{h}, \mathbf{y}|\mathbf{x}, \theta)p(\theta)$, in which the data is completed with expected counts for the hidden variables.

3 Application

The common use of goal-oriented clustering discussed in the introduction—namely, the construction of clusters for directing hand-tailored ad campaigns and promotions—is difficult to evaluate. In this section, we introduce another application of goal-oriented clustering that can be evaluated quantitatively. In the following section, we present an evaluation.

The application is targeted advertising on a web site. The basic idea is to show users ads that will yield the most utility to the advertiser. An extremely useful measure of utility is the probability that a user will purchase an item on the site if shown the ad. Because this utility is difficult to measure in practice, many sites settle for targeting ads so as to maximize click through. We adopt this goal here.

If delivering high utility ads to users based on their properties were the only goal of targeted advertising, a simple approach would be sufficient. In particular, we could build a classifier that predicts click through for each ad given user properties as inputs, and show to a user the ad most likely to be clicked by that user. Many advertisers who buy advertising space on web sites, however, place an additional constraint on the web site. Namely, they require that each of their ads be shown a certain number of times—that is, each ad has an advertiser imposed quota. The problem of satisfying quotas for ad delivery is often called *inventory management*.

Our solution to this problem, discussed in [4] is as follows. First, we show ads randomly to users and gather data for a sufficient period of time to build (goal-oriented) clusters. Next, we learn the clusters using input variables that correspond to user properties (e.g., what web pages they have visited) and target variables that correspond to click behavior on ads—click through or not when shown—one variable per ad. Then, for each ad i and cluster j, we use the model and click behavior to determine the probability p_{ij} that a user currently in cluster j will click on ad i. Note that cluster membership can change as a user navigates through the site.

Next, we construct an ad schedule for the remainder of the campaign that—rather than shows ads randomly—shows ads to users based on the cluster membership of those users so as to maximize expected click through subject to the quotas. In particular, let x_{ij} denote the number of ad i to be shown (in the remainder of the campaign) to a user in cluster j. In this notation, the maximization of click through becomes

$$\max \sum_{ij} p_{ij}\ x_{ij} \tag{2}$$

and the ad-quota constraint takes the form

$$\sum_{j} x_{ij} = q_i \tag{3}$$

where q_i is the quota for ad i. In addition, there is a constaint having to do with the fact that a limited number of users in each cluster j, denoted c_j, is expected to visit the web site during the remainder of the campaign. That is,

$$\sum_{i} x_{ij} \leq c_j \tag{4}$$

The maximization in Expression 2 subject to the constraints in Equations 3 and 4 defines a linear program, which can be easily solved.

There are two key requirements that make this approach work. One, the clusters that are produced must be relatively small in number so that the quantities c_j can be accurately estimated from historical data. (We assume that the character of users visiting the site do not change quickly over time.) Two, the clusters that are produced must be able to predict ad click through. Goal-oriented clustering helps us realize both requirements.

4 Experiments

In this section, we compare the performance of clusters built with goal-oriented clustering versus those built with traditional clustering on the task of targeted advertising with inventory management.

Our data consists of two days of logs (December 21 and 22, 1998) from the msnbc.com web site. We use the first day of data to build cluster models, estimate probabilities p_{ij}, and create allocations x_{ij}. Target variables correspond to click behavior on ads and input variables correspond to pages visited by the user.

After a model is learned, we assign each user at the time of an ad view to the cluster with the highest posterior probability given the user's current page visits for the day. In principle, we could also use the user's previous ad-click behavior, but this information is not available to the ad-delivery system so we do not use this information for cluster assignment. We use these cluster assignments and the associated click behavior to determine the probabilities p_{ij}. We use a maximum-likelihood estimate.

Our evaluation is passive in that we do not actually change the allocation of ads. Instead, we estimate the overall number of clicks had the schedule been implemented. The estimation procedure is straightforward under the assumption that the probabilities p_{ij} do not change as more or less ads are shown in clusters. In particular, the expected number of clicks is simply $\sum_{ij} p_{ij}\ x_{ij}$.

We can estimate overall number of clicks given the schedule x_{ij} using the same probabilities p_{ij} that produced the schedule. Nonetheless, click through rates are small (often less than 1%). Thus, to obtain a more realistic estimate of the performance of our schedule, we estimate overall number of clicks based on probabilities p_{ij} obtained from the *second* day of data. These probabilities are determined using the same procedure as that for the training data.

As described above, we apply the GEM algorithm to estimate the parameters of a goal-oriented-clustering model. To initialize the GEM algorithm, we randomly choose the $\theta_{\mathbf{H}}$ from a diagonal Gaussian distribution with zero mean and identical variances equal to $1/n$. We initialize the parameters of the multinomial models θ_Y by estimating the parameters for a single-component cluster model and then randomly perturbing the parameter values by a small amount to obtain sets of parameters for each instantiation of $\mathbf{H}$ (see [7] for details). We run the GEM algorithm until the relative improvement in the log likelihood is less than 10^{-5}. Prior to training the model, we rescaled all input values of the network so as to have zero mean and unit variance.

The specific model we use in our final evaluation is a goal-oriented-clustering model with a single four-state hidden variable. For the weights of the softmax logistic model, we use a Gaussian prior with mean zero and variance 0.05, and use a prior count of one for each cell for the multinomial distributions. Both the number of states for the hidden variable and the variance of the prior were selected on the basis of the improvement in click through for a fifty–fifty split of the training data.

Table 1: Lifts for models.

Model & learning algorithm	Lift
One-variable softmax GEM	1.28
One-variable softmax SGD	1.29
Multi-variable sigmoid SGD	1.26
Standard clustering EM	0.99

For comparison, we evaluate ad allocation based on a standard cluster model. In particular, we use EM to learn a mixture model for the input variables only, in which each mixture component assumes the mutual independence of the variables (e.g., [2]).

In addition, we evaluate goal-oriented-clustering models trained with stochastic gradient descent (SGD). In particular, on-line back-propagation was used to update the parameters after every case applied to the network. A step size of 0.1 was used with no momentum. In addition, a very mild weight decay term was added to the hidden-to-output layer. Two different goal-oriented models were trained in this fashion: (1) a model with a single hidden variable with a softmax non-linearity (a model equivalent to that trained with GEM) and (2) a model with a binary hidden-layer. In both cases, the cross-entropy error function was used, and the optimal size of the networks and early stopping point were chosen with a train–test split of the training data.

We report our results in terms of *lift*: the estimated overall number of clicks given the new schedule divided by the overall number of clicks given the original schedule. Note that the original schedule is made by advertising executives whose goal for many ads is to optimize click through. Thus, these lifts can be viewed as improvements over manual scheduling. The results are shown in Table 1. All three goal-oriented-clustering models produce lifts over 1.25, whereas the traditional clustering model yields no lift. We note that, as advertising is currently the predominant source of income on the web, a 25% improvement in click through can dramatically improve revenue.

In summary, we have introduced the process of goal-oriented clustering as well as an architecture and GEM learning algorithm for this process. We have shown the utility of this approach with respect to standard clustering on a targeted-advertising application. In addition, we have seen that SGD can also be used to successfully train a goal-oriented-clustering model. Overall, however, we prefer the GEM algorithm because, as mentioned, it requires no parameter tuning and has a convergence guarantee.

Acknowledgments

We thank Mitch Goldman and Mikky Anderson for their comments on this work and their help in obtaining the msnbc.com datasets.

References

[1] J. Banfield and A. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803–821, 1993.

[2] P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In U. Fayyad, G. Piatesky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI Press, Menlo Park, CA, 1995.

[3] P. Cheeseman, J. Stutz, M. Self, W. Taylor, J. Goebel, K. Volk, and H. Walker. Automatic classification of spectra from the infrared astonomical satellite (iras). Technical Report 1217, NASA Ames, 1989.

[4] D. M. Chickering and D. Heckerman. Targeted advertising with inventory management. Technical Report MSR-TR-00-49, Microsoft Research, Redmond, WA, 2000.

[5] R. Duda and P. Hart, editors. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[6] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, New York, 1992.

[7] B. Thiesson, C. Meek, D. Chickering, and D. Heckerman. Computationally efficient methods for selecting among mixtures of graphical models, with discussion. In *Bayesian Statistics 6: Proceedings of the Sixth Valencia International Meeting*, pages 631–656. Clarendon Press, Oxford, 1999.

[8] M. Wedel and W. Kamakura. *Market Segmentation: Conceptual and methodological foundations*. Kluwer Academic Publishers, 1998.