

# The Learning-Curve Sampling Method Applied to Model-Based Clustering

**Christopher Meek**

MEEK@MICROSOFT.COM

**Bo Thiesson**

THIESSON@MICROSOFT.COM

**David Heckerman**

HECKERMA@MICROSOFT.COM

*Microsoft Research, One Microsoft Way, Redmond, WA 98052-6399, USA*

**Editor:** Leslie Pack Kaelbling

## Abstract

We examine the learning-curve sampling method, an approach for applying machine-learning algorithms to large data sets. The approach is based on the observation that the computational cost of learning a model increases as a function of the sample size of the training data, whereas the accuracy of a model has diminishing improvements as a function of sample size. Thus, the learning-curve sampling method monitors the increasing costs and performance as larger and larger amounts of data are used for training, and terminates learning when future costs outweigh future benefits. In this paper, we formalize the learning-curve sampling method and its associated cost-benefit tradeoff in terms of decision theory. In addition, we describe the application of the learning-curve sampling method to the task of model-based clustering via the expectation-maximization (EM) algorithm. In experiments on three real data sets, we show that the learning-curve sampling method produces models that are nearly as accurate as those trained on complete data sets, but with dramatically reduced learning times. Finally, we describe an extension of the basic learning-curve approach for model-based clustering that results in an additional speedup. This extension is based on the observation that the shape of the learning curve for a given model and data set is roughly independent of the number of EM iterations used during training. Thus, we run EM for only a few iterations to decide how many cases to use for training, and then run EM to full convergence once the number of cases is selected.

**Keywords:** Learning-curve sampling method, clustering, scalability, decision theory, sampling

## 1. Introduction

In situations where one has access to massive amounts of data, the cost of building a statistical model can be significant if not insurmountable. A common practice is to build the model using a (usually random) sample of the examples or *cases* in the data. In so doing, however, the choice of the number of cases to use is far from clear. In this paper, we examine the learning-curve sampling method, an algorithmic approach to choosing an appropriate sample size for training.

Learning-curve sampling methods rely on two basic observations: (1) the computational cost of learning a model increases as a function of the size of the training data, and (2) the performance/accuracy of a model has diminishing improvements as a function of the size of the training data. The curve describing the performance as a function of the sample size of

the training data is often called the *learning curve*. The typical shape of a learning curve is concave (down) with performance approaching some limiting behavior. Thus, a learning-curve sampling method monitors the increasing costs and performance as larger and larger amounts of data are used for training, and terminates learning when the increasing costs outweigh the benefit of increasing performance.

In this paper, we formalize the learning-curve approach to sampling in terms of decision theory. In particular, we describe the use of utility as an overall measure of the quality of a learning method, and derive algorithms for selecting sample size from the principle of maximum expected utility.

In addition, we describe several high-utility learning-curve sampling methods for the particular task of model-based clustering. We investigate a simple but widely used class of models for clustering—namely, finite mixture models—with a fixed, pre-specified number of components and use the Expectation–Maximization (EM) algorithm to learn the model parameters. In experiments on three real data sets, we show that the learning-curve sampling method produces models of high utility—ones that are nearly as accurate as those trained on complete data sets, but with dramatically reduced run times.

Also in this paper, we describe an extension of the basic learning-curve approach for model-based clustering that results in higher utilities by further reducing run time without loss in accuracy. Our approach is based on the observation that the shape of the learning curve for a given model and data set is roughly independent of the number of EM iterations used during training. Thus, we run EM for only a few iterations to decide how many cases to use for training, and then run EM to full convergence once the number of cases is selected.

The paper is organized as follows. In Section 2, we present our decision-theoretic formulation of the learning-curve sampling method. In Section 3, we describe a model-based approach to clustering that uses the EM algorithm. In Section 4, we apply the learning-curve sampling method to the task of model-based clustering and, in Section 5, provide extensions for improving the method. In Section 6, we present an empirical study that demonstrates that our learning-curve sampling methods provide order-of-magnitude speedups on real data and have higher utilities than alternative methods. Finally, in Section 7 we describe related work and, in Section 8, we provide a summary and directions for future work.

## 2. A Decision-Theoretic Formulation of the Learning-Curve Sampling Method

As we have discussed, the basic idea of a learning-curve sampling method is to iteratively apply a training algorithm to larger and larger subsets of the data, until the future expected costs outweigh the future expected benefits associated with the training. In this section, we present a decision-theoretic formulation of this general approach. Given a data set  $D$ , let  $D_1, D_2, \dots, D_n = D$  denote the sequence of data sets that are examined in the process of finding the appropriate sample size, denoted  $N_{lc}$ . We shall assume that the data sets are nested—that is,  $D_i \subset D_{i+1}$ —so that  $|D_i| < |D_{i+1}|$ , where  $|D_i|$  is the number of cases in data set  $D_i$ . We shall also assume that, apart from this nesting, the cases in each  $D_i$  are randomly selected from  $D$ .

When we examine data set  $D_i$ , we apply some *training method* to it, gather information about that application—for example, the accuracy of the resulting model and the time it

took to learn that model—and then determine whether to examine additional data sets. Expressed in decision-theoretic terms, our determination of when to stop is a sequential decision problem. At stage  $i$  in this decision problem, we have just run the training method on data subset  $D_i$ . At this point, we have two alternatives: (1) stop—that is, output the learned model, or (2) continue. The choice to continue brings us to stage  $i+1$  of the decision problem. Decision theory (e.g., Howard, 1966) tells us that we should choose the alternative with the maximum expected utility (MEU), where expectation is taken with respect to our uncertainty (encoded in terms of probability) of the possible outcomes that follow from the alternatives. Thus, once we have identified the possible sequences of data sets, the utilities associated with the possible outcomes, and the uncertainties of those outcomes, our decision is determined. Let us consider each of these ingredients.

The first ingredient, the sequence of data sets, may be fixed—that is, chosen in advance—or chosen adaptively as data sets are examined. Two types of fixed sequences have been considered. John & Langley (1996) consider incrementally adding a constant number of cases. Provost, Jensen & Oates (1999) consider incrementally adding a geometrically increasing number of cases. As argued by Provost et al., when one does not have an accurate guess as to the “correct” number of cases to achieve the proper cost/benefit tradeoff, the method of incrementally adding a fixed number of cases can require an unreasonable number of iterations when a large number of cases is needed. In contrast, when using a geometric schedule, one can quickly reach an appropriate sample size. For instance, if the cost of training is roughly linear in the number of cases, then using a geometric schedule to train on data sets of size  $k \cdot 2^0, k \cdot 2^1, \dots, k \cdot 2^i$ , until we reach some data set of size  $k \cdot 2^i$  ( $N \leq k \cdot 2^i < 2N$ ), will require only a constant factor more computation than simply applying the training method to the data set of  $N$  cases.

In situations where the sequence is adaptively selected, we can use points along the learning curve for the smallest data sets  $D_1, \dots, D_i$ , in conjunction with a model for the shape of the learning curve (e.g., Kadie, 1995), to estimate future points along the curve. We can then use these estimates to expand the number of alternatives in our sequential decision problem to include a choice about the number of cases in  $D_{i+1}$ . In this paper, we use the fixed geometric sequence and do not elaborate further on adaptive strategies.

The second ingredient is the utility or “goodness” of stopping with data set  $D_i$ . We decompose this utility into a benefit and cost. Let  $m_i$  denote the model learned with this data set. A natural measure of cost is proportional to the total time it takes to produce model  $m_i$ . A natural measure of benefit is proportional to the accuracy of  $m_i$  on the task to which that model will be applied. Of course, the measure of accuracy will depend on the task at hand. In the remainder of the paper, we shall consider a measure of accuracy that is used commonly in practice: the log-likelihood of the model on holdout data  $D_{ho}$ , denoted  $l(D_{ho}|m_i)$ . Note that the learning-curve sampling method is applied in situations where the full data set  $D$  is extremely large. Consequently, there will be ample data to hold out.

To combine these measures to produce an overall measure of utility, we need to scale benefit and cost appropriately. One possibility is to determine both scales on a problem-by-problem basis. In this paper, we consider an approach that can be applied across a wide range of problems. In particular, we measure the benefit of stopping with model  $m_i$  in terms of the accuracy of that model relative to a model trained with all the data ( $m_D$ ) and

a baseline model  $m_{\text{base}}$ :

$$\text{benefit}(m_i) = \frac{l(D_{ho}|m_i) - l(D_{ho}|m_{\text{base}})}{l(D_{ho}|m_D) - l(D_{ho}|m_{\text{base}})} \quad (1)$$

One simple choice for the baseline model is one in which all of the features are mutually independent and trained with a relatively small fraction of the data. We use this baseline model in our empirical study. The choice of this baseline model yields a measure of benefit that lies between 0 and 1 for a wide range of problems. (When the data set that produces  $m_i$  is extremely small, this measure may be negative. Such occurrences, however, are rare.) Note that this approach is appropriate for density-estimation models and can be extended to classification/regression models in a straightforward fashion. To scale cost, we simply write

$$\text{cost}(m_i) = \alpha \cdot \text{runtime}_i \quad (2)$$

where  $\text{runtime}_i$  is total time required to produce model  $m_i$ , and  $\alpha$  is the relative importance of benefit to run time. This quantity, which depends on the preferences of the *decision maker*—the person who is controlling the execution of the algorithm—should be assessed on a problem-by-problem basis. We discuss this assessment later in this section. Combining these two measures, we set

$$\text{utility}(m_i) = \text{benefit}(m_i) - \text{cost}(m_i) \quad (3)$$

The third ingredient for our decision problem are the uncertainties associated with the possible outcomes. These uncertainties are mostly problem specific, but commonly share an important observation. Namely, imagine we are at stage  $i$  of the decision, and want to decide whether to continue with larger data sets. As we look forward in time, the uncertainties associated with the benefits and costs of examining data set  $D_k$ ,  $k > i$ , increase with  $k$ . In particular, it will be extremely difficult to estimate these uncertainties for large  $k$ . This observation suggests that we may want to replace strict adherence to the MEU principle, which requires uncertainty estimates for all  $k$ , with an approximate procedure that uses only uncertainty estimates at the next stage.

Let us consider a procedure with this property. At stage  $i$  of this procedure, we incorrectly assume that there are only two alternatives: (1) stop now, and (2) learn model  $m_{i+1}$  and stop. We then choose the alternative among these two that maximizes our expected utility. If we choose alternative 1, we indeed stop, returning model  $m_i$  and its accuracy score. If we choose alternative 2, we evaluate another decision problem, now at stage  $i + 1$ . This strategy is often referred to as a “myopic” strategy, because it only looks one step into the future.

According to this myopic strategy, we stop at stage  $i$  if and only if the expected utility of stopping at stage  $i + 1$  is less than or equal to the expected utility of stopping at stage  $i$ —or, equivalently, if the expected increase in utility of moving from stage  $i$  to  $i + 1$  is less than zero. In particular, according to Equations 1 through 3, we stop if and only if

$$\frac{E_i(\text{benefit}(m_{i+1}) - \text{benefit}(m_i))}{E_i(\text{runtime}_{i+1} - \text{runtime}_i)} \leq \alpha \quad (4)$$

where  $E_i(\cdot)$  denotes expectation with respect to our uncertainties at stage  $i$ . The left-hand side of Equation 4 is the ratio of the expected incremental benefit to the expected

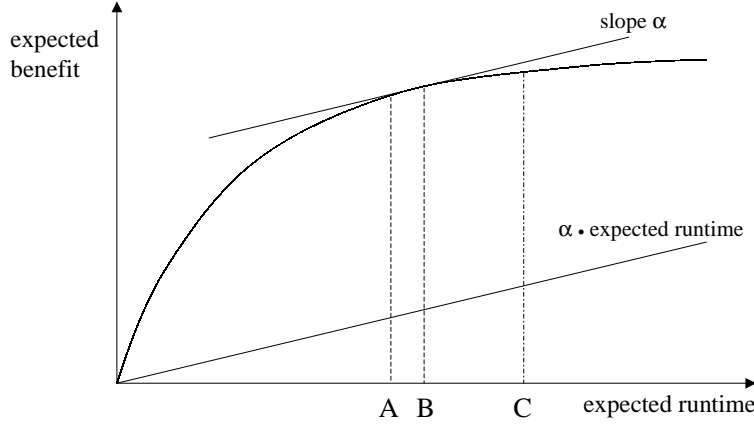


Figure 1: A hypothetical plot of expected benefit versus expected run time.

incremental cost of moving from stage  $i$  to stage  $i + 1$ . Thus,  $\alpha$  can be viewed as the value of this incremental-benefit-to-cost ratio (having units benefit per time), below which additional subsets of data should not be considered. We refer to  $\alpha$  as our *stopping threshold* and Equation 4 as our *stopping criterion*. Note that, with this understanding of  $\alpha$ , its assessment is straightforward. For example, a decision maker can be asked the question “How long would you be willing to wait to increase the relative accuracy of the learned model by one percent?” If the answer is (e.g.) one hour, then  $\alpha = 0.01$  benefit per hour.

Although this strategy is myopic, it will be optimal in many situations. Assuming incremental benefits decrease and incremental costs increase as we progress through the stages of our decision problem, it makes sense to stop when the ratio of these two quantities falls below  $\alpha$ , the relative importance of benefit to cost. To understand this observation more precisely, consider Figure 1, which shows a hypothetical plot of the expected benefit  $E_i(\text{benefit}(m_{i+1}))$  versus the expected run time  $E_i(\text{runtime}_{i+1})$  as a function of stage  $i$ . For purposes of argument, the curve is shown to be continuous under the (temporary) assumption that the sample sizes of successive stages are closely spaced. The more important aspect of the curve is that it is concave down and non-decreasing so as to represent the progressive decrease in incremental-benefit-to-incremental-cost ratio. In general, the curve will have this shape when (1) the “expected learning curve”—the plot of expected benefit versus sample size—is concave down and non-decreasing, and (2) the rate of change of expected run time with respect to sample size is non-decreasing in sample size. Both conditions are satisfied often in practice, and are satisfied (roughly) for the specific problems we consider in the remainder of the paper.

Now, consider the difference between the height of the curve and the height of a line with slope  $\alpha$  through the origin (also shown in the figure) as a function of expected run time. This difference represents expected utility as a function of expected run time. Because the expected-benefit-versus-expected-run-time curve is concave down and non-decreasing, this difference will be a maximum when the tangent to the curve has slope  $\alpha$ . Thus, according to the MEU principle, we should stop at a stage corresponding to point  $B$  in the figure. Just

beyond this point, the ratio of expected incremental benefit to expected incremental run time falls below  $\alpha$ —precisely the inequality in Equation 4. (Arguments similar to this one are common in the discipline of *cost-benefit analysis*—for example, Pearce, 1983.) When we relax our assumption that the sample sizes are closely spaced, we find that the criterion expressed by Equation 4 may be non-optimal. For example, suppose points  $A$  and  $C$  in the figure correspond to stages  $D_i$  and  $D_{i+1}$ . In this case, although our criterion yields a stopping point at stage  $i + 1$ , the expected utility of stopping at stage  $i$  is slightly greater. In general, we may stop one step late, but never early—that is, our stopping criterion is *conservative*.

We consider this approximate stopping criterion throughout the remainder of the paper. As we shall see, it can be computed efficiently and is amenable to improvement.

### 3. Model-Based Clustering

In this section, we describe the model-based approach to clustering using finite mixture models and how one can use the expectation maximization (EM) algorithm to learn such models. We focus on details important for the application of our learning-curve sampling method to model-based clustering described in Sections 4 and 5.

#### 3.1 Mixture Models

Let  $\mathbf{X} = \{X_1, \dots, X_L\}$  be a multivariate random variable taking on values corresponding to observations of individual objects or cases. The goal of clustering is to form groups of objects that share similar values for  $\mathbf{X}$ . In a model-based approach to clustering, we assume that our data is generated in the following fashion:

1. An object is assigned to one of  $K$  (hidden) clusters with some probability, and
2. Given that an object is in a cluster, its value for  $\mathbf{X}$  is generated from some statistical model specific to that cluster.

More formally, let  $C$  be a discrete-valued variable taking on values  $c_1, \dots, c_K$ . The value of  $C$  corresponds to the unknown cluster assignment for an object. Then, we have

$$\begin{aligned} p(\mathbf{x}|\theta) &= \sum_{k=1}^K p(c_k|\theta) p_k(\mathbf{X} = \mathbf{x}|c_k, \theta_k) \\ &= \sum_{k=1}^K \pi_k p_k(\mathbf{X} = \mathbf{x}|c_k, \theta_k) \end{aligned}$$

where  $\pi_k = p(c_k|\theta)$  is the marginal probability of the  $k^{th}$  cluster ( $\sum_k \pi_k = 1$ ),  $p_k(\mathbf{x}|c_k, \theta_k)$  is the statistical model describing the distribution over the variables for an object in the  $k^{th}$  cluster, and  $\theta = \{\pi_1, \dots, \pi_K, \theta_1, \dots, \theta_K\}$  are the *parameters* of the model. The statistical model  $p(\mathbf{x}|\theta)$  is called a *finite mixture model*. For additional information on finite mixture models see (e.g.) Titterton, Smith & Makov (1985) and McLachlan & Basford (1988).

In model-based clustering, one specifies the form of the cluster-specific parametric models  $p_k(\mathbf{X} = \mathbf{x}|c_k, \theta_k)$ , and then uses a data set to learn the specific cluster probabilities

and the cluster-specific model parameters. The approach is called model-based clustering because a separate statistical model is used for each cluster.

In our experiments, we concentrate on a simple but widely used class of finite mixture models where each component is described by a *product of multinomials model*:

$$p_k(\mathbf{x}|c_k, \theta_k) = \prod_{i=1}^L p(x_i|\theta_k^i),$$

where  $p(x_i|\theta_k^i)$  is a multinomial distribution over the values for variable  $X_i$  and  $\theta_k^i$  is the set of parameters. Here, we assume that, in each component, the variables  $X_1, \dots, X_L$  are mutually independent—that is, we assume that the statistical model for each component is a log-linear model with only main effects. This mixture model can alternatively be viewed as naive-Bayes models with a hidden class variable, also known as AutoClass models (Cheeseman & Stutz, 1995). We further restrict our attention to mixture models with a fixed, pre-specified number of components.

Given the parameters of a mixture model, we can assign an object (or case) to a cluster as follows. For an object with  $\mathbf{X} = \mathbf{x}$ , we use Bayes' rule to compute the probability distribution over the hidden variable  $C$ :

$$p(c_k|\mathbf{x}, \theta) = \frac{\pi_k p_k(\mathbf{x}|c_k, \theta_k)}{\sum_{j=1}^K \pi_j p_j(\mathbf{x}|c_j, \theta_j)} \quad (5)$$

The probabilities  $p(c_k|\mathbf{x}, \theta)$  are sometimes called *membership probabilities* and correspond to the object's (fractional) cluster assignment. Once we have computed these probabilities, we can either assign the object to the cluster with highest probability—a *hard* assignment—or assign the object fractionally to the set of clusters according to this distribution—a *soft* assignment.

### 3.2 Learning Mixture Models from Data

Now we describe how to learn the parameters of a finite mixture model with known number of components  $K$ , given training data  $D = (\mathbf{x}^1, \dots, \mathbf{x}^N)$ . One possible criterion for doing so is to identify those parameter values for  $\theta$  that maximize the likelihood of the training data:

$$\theta^{ML} = \operatorname{argmax}_{\theta} p(D|\theta) = \operatorname{argmax}_{\theta} \prod_{i=1}^N p(\mathbf{x}^i|\theta)$$

This value is often referred to as the *maximum likelihood* (ML) estimate and  $p(D|\theta)$  is the likelihood of the training data. Alternatively, one may have prior knowledge about the domain and can encode this information in the form of a *prior probability distribution* over the parameters, denoted  $p(\theta)$ . In this situation, a criterion for learning the parameters is to identify the parameter value of  $\theta$  that maximize the posterior probability of  $\theta$  given our training data:

$$\theta^{MAP} = \operatorname{argmax}_{\theta} p(\theta|D) = \operatorname{argmax}_{\theta} p(D|\theta) p(\theta)/p(D)$$

where  $p(\theta|D)$  is the posterior on the parameters and the second identity follows by Bayes' rule. This value is often referred to as the *maximum a posteriori* (MAP) estimate. When used in conjunction with vague or non-informative priors, MAP estimates are smoothed (i.e., less extreme) versions of ML estimates. In the work described in this paper we learn MAP estimates for the parameters  $\theta$  using the diffuse Dirichlet prior described in Cooper & Herskovits (1992).

In situations in which the a statistical model has an unobserved variable—as is the case with finite mixture models—one can use the well-known expectation maximization (EM) algorithm to obtain ML and MAP estimates (Dempster, Laird & Rubin 1977). The EM algorithm is given starting values for the parameters, and then iterates between an Expectation or E step and a Maximization or M step until successive parameter values are stable. In the E step of the algorithm, given a current value of the parameters  $\theta$ , we fractionally assign an object with  $\mathbf{X} = \mathbf{x}$  to cluster  $c_k$  using the membership probabilities given by Equation 5. In the M step of the algorithm, we pretend that these fractional assignments correspond to real data, and reassign  $\theta$  to be the MAP estimate given this fictitious data. By iteratively applying the E step and M step, we monotonically improve the estimates of the model parameters  $\theta$ , ensuring convergence (under fairly general conditions) to a local maximum of the posterior distribution (or a maximum likelihood estimate) for  $\theta$ .

In our experiments, we initialize the parameters of the EM algorithm as follows. We choose the parameters  $\pi_1, \dots, \pi_K$  to be equal, and we set the parameters of our component models  $\theta_k$  by estimating the parameters for a single-component cluster model and then randomly perturbing the parameter values by a small amount to obtain  $K$  sets of parameters. This approach is described in detail in Thiesson, Meek, Chickering & Heckerman, 1999. The convergence criterion that we use to terminate the EM algorithm is one that is commonly used. Namely, we converge when the relative improvement in log-posterior (or log-likelihood) of the training data between successive EM iterations relative to the total improvement in log-posterior (or log-likelihood) over the initial model is less than a convergence threshold  $\gamma$ .

## 4. A Learning-Curve Sampling Method for Clustering

In this section, we describe how to apply the learning curve sampling method described in Section 2 to the problem of model-based clustering. We call this method the *standard learning-curve sampling method*

In applying the learning curve approach to model-based clustering, we make several approximations. To help illustrate the approximate validity of the assumptions, we consider three real-world data sets. We shall also use these data sets in our experiments. We note that the only criterion we used to select these data sets was large sample size, and that our method was developed prior to the examination of any of these data sets.

The three data sets that we consider are the MSNBC, MS.COM, and USCensus1990 data sets. The MSNBC data set is derived from web logs for one day in 1998 for the msnbc.com website. It records which of the 303 most popular stories on that day were read by each of the visitors. The MS.COM data set is derived from the web logs for one day in 2000 for the microsoft.com web site. It records which of the 775 most popular areas or “vroots” of the site were visited by each person. In each of these data sets, cases correspond to people,



and variables correspond to possibly viewed items. Both of these data sets are *sparse* in the sense that—on average—a person only views a few items. The USCensus1990 data set, available from the UC Irvine KDD Archive, is derived from the one percent sample of the Public Use Microdata Samples (PUMS) person records from the full 1990 census sample (all fifty states and the District of Columbia but not including “PUMA Cross State Lines One Percent Persons Record”). Each case corresponds to a person and has 68 categorical variables. The MSNBC, MS.COM, and USCensus1990 data sets contain 597,971, 1,938,877, and 2,458,284 cases, respectively.

Now let us examine the expectations in our stopping criterion for stage  $i$ , Equation 4. An examination of the EM algorithm shows that  $E_i(\text{runtime}_{i+1})$  is given by

$$E_i(\text{runtime}_{i+1}) \cong \text{runtime}_i + c_1 \cdot E_i(I_{i+1}) \cdot |D_{i+1}| + c_2 \cdot E_i(I_{i+1}) + c_3$$

where  $c_1$ ,  $c_2$ , and  $c_3$  are constants, and  $E_i(I_{i+1})$  is the expected number of times the EM algorithm iterates (when applied to  $D_{i+1}$ ) before reaching convergence. The first term in the sum is simply the known time to reach stage  $i$ . The second and third terms correspond to the time spent by the EM algorithm in the E and M steps, respectively. The fourth term corresponds to the time spent evaluating the accuracy of the model on the holdout set. (The time to load the clustering algorithm and initialize data structures in memory is insignificant.) The constants  $c_1$ ,  $c_2$ , and  $c_3$  are known once the first data set in the sequence has been evaluated. Furthermore, in our experience, we have found that the number of EM iterations is roughly constant for a fixed convergence threshold  $\gamma$ . Figure 2 shows the number of iterations versus sample size for the MSNBC, MS.COM and USCensus1990 domains. Consequently, we use the approximation

$$E_i(I_{i+1}) \cong \frac{1}{i} \sum_{j=1}^i I_j \equiv \bar{I}_i \quad (6)$$

Note that, at stage  $i$ , the quantities  $I_1, \dots, I_i$  are known with certainty.

The remaining quantity that we need at stage  $i$  is the expected incremental benefit  $E_i(\text{benefit}(m_{i+1}) - \text{benefit}(m_i))$ . Using Equation 1 and an approximation in which we take expectations of the numerator and denominator separately, we obtain

$$E_i(\text{benefit}(m_{i+1}) - \text{benefit}(m_i)) \cong \frac{E_i(l(D_{ho}|m_{i+1}) - l(D_{ho}|m_i))}{E_i(l(D_{ho}|m_D)) - l(D_{ho}|m_{base})}. \quad (7)$$

In addition, we can approximate both the numerator and denominator in this expression. To illustrate the approximation for the numerator, consider the learning curves for the MSNBC, MS.COM, and USCensus1990 data sets shown in Figure 3. This figure provides two plots of the learning curves for each of the data sets; one where the x-axis is linear in the sample size and the other logarithmic. In particular, the learning curves are linear or slightly concave down when sample size is plotted on the logarithmic scale. Consequently, because we are using a sequence of data sets in which data set size increases geometrically, it follows that

$$E_i(l(D_{ho}|m_{i+1}) - l(D_{ho}|m_i)) \lesssim l(D_{ho}|m_i) - l(D_{ho}|m_{i-1}) \quad (8)$$

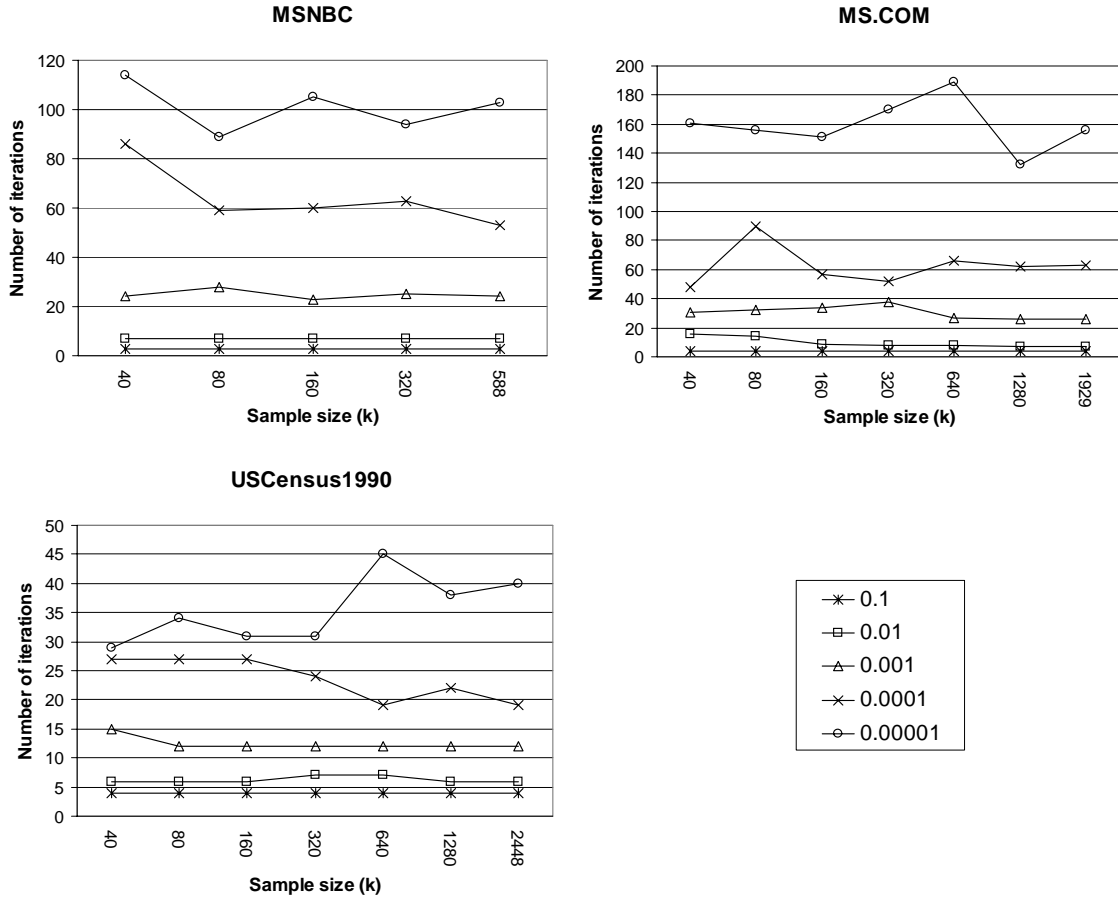


Figure 2: Number of iterations to convergence as a function of convergence threshold  $\gamma$  and sample size.

In this work, we shall use the right-hand side of Equation 8 as an approximation for the left-hand side. The approximation is conservative in that it will tend to over-estimate the expected change in benefit, and thus bias the stopping decision toward late stopping. Also, because we are computing differences, this approximation requires that we stop no sooner than stage  $i = 2$ .

For the denominator of Equation 7, we use the simple approximation

$$E_i(l(D_{ho}|m_D)) \cong l(D_{ho}|m_i) \quad (9)$$

If the learning curve is noisy, a better approximation is the maximum over  $l(D_{ho}|m_1), \dots, l(D_{ho}|m_i)$ . In our experiments, however, the better approximation is not needed. In either case, the approximation is conservative.

Given the approximations associated with Equations 4 through 9, we can re-write the stopping criterion for stage  $i$ , given in Equation 4, as

$$\frac{l(D_{ho}|m_i) - l(D_{ho}|m_{i-1})}{l(D_{ho}|m_i) - l(D_{ho}|m_{base})} / (c_1 \cdot \bar{I}_i \cdot |D_{i+1}| + c_2 \cdot \bar{I}_i + c_3) \leq \alpha \quad (10)$$

We determine  $l(D_{ho}|m_{base})$  before considering any data sets in the sequence. Consequently, at stage  $i$ , all the quantities in this expression are known.

Finally, recall the conditions from Section 2 needed for the myopic stopping criterion Equation 4 to be near optimal: (1) the “expected learning curve”—the plot of expected benefit versus sample size—should be concave down and non-decreasing, and (2) the rate of change of expected run time with respect to sample size should be non-decreasing in sample size. We note that, in our approach using EM for model-based clustering, both conditions are roughly satisfied. In particular, with a few exceptions due to small amounts of noise, the plots on the left-hand-side of Figure 3 are concave down and non-decreasing. Furthermore, assuming  $E_i(I_{i+1})$  does not decrease substantially with  $i$ , expected run time (Equation 4) satisfies the second condition.

## 5. Speeding up the Learning-Curve Sampling Method for Clustering

In this section, we consider an extension that obtains higher utilities than the standard learning-curve sampling method applied to model-based clustering. The extension substantially reduces training time without significantly affecting model accuracy.

To understand this extension, consider again the learning curves in Figure 3. Here, we see that learning curves for different EM convergence levels ( $\gamma$ ) are roughly parallel for each of the data sets. We have found that this is a common property for data sets. This property suggests the following extension. When determining the expected benefit for a given data set  $D_i$ , use an *abbreviated training method*—EM run to a high convergence threshold or run for only a relatively few iterations—to obtain an approximate value. Then, make the decision to stop or continue based on this approximation. Finally, once the decision has been made to stop, learn a model on the selected data set by running EM to full convergence. In the following section, we show that this extension can increase the utilities of the methods—that is, increase computational efficiency without sacrificing much accuracy. In the remainder of this section, we examine the details of the approach.

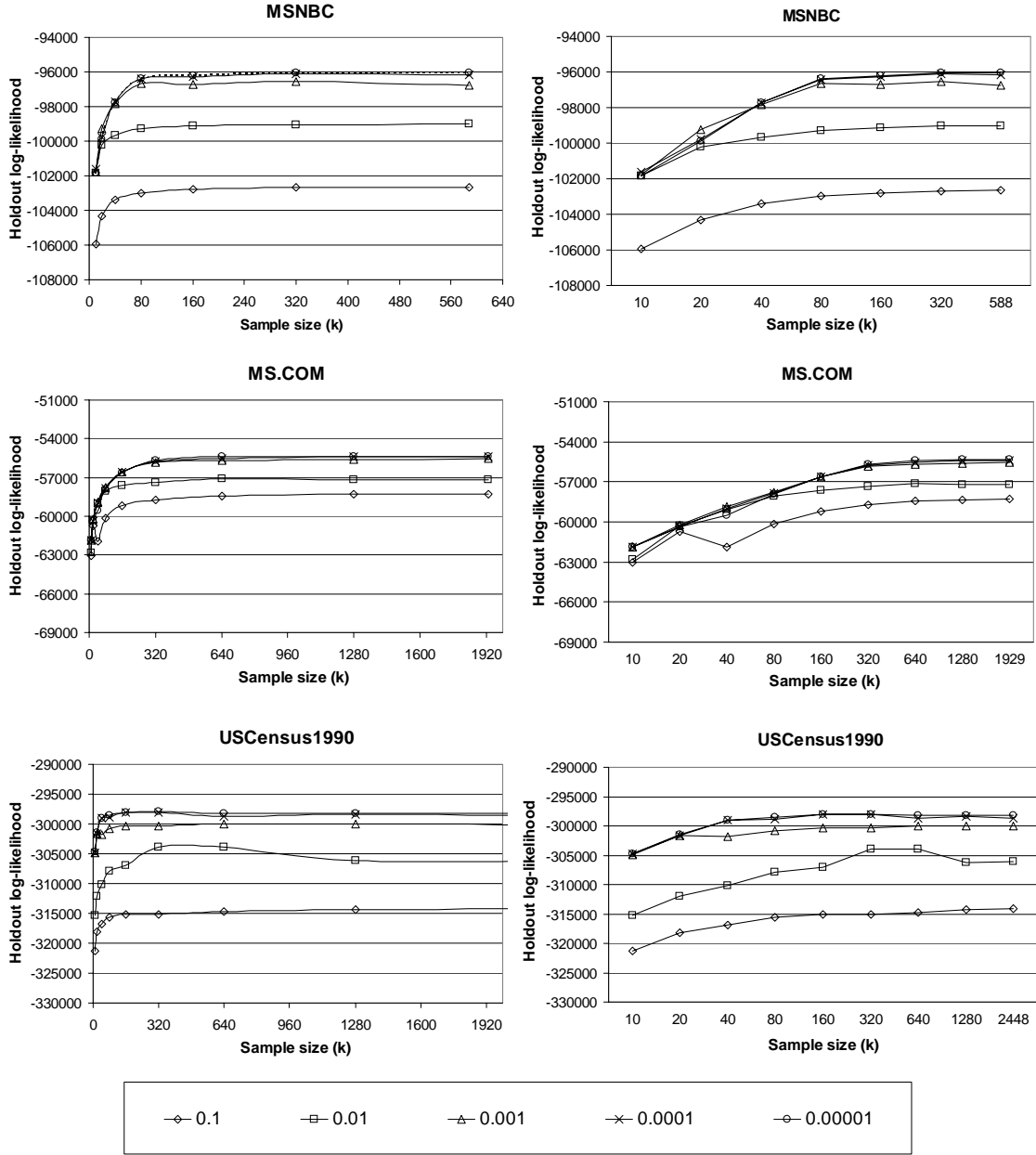


Figure 3: Learning curves for clustering. The x-axes are linear and logarithmic in the left and right graphs, respectively. The different curves in each graph correspond to different EM convergence thresholds  $\gamma$ .

Due to the use of an abbreviated training method, we need to alter our estimates for both the benefit and cost components of our stopping criterion (Equation 10). First, consider the determination of benefit at stage  $i$ —in particular, the estimation of the likelihood  $l(D_{ho}|m_i)$ . Given that the learning curves for the full and an abbreviated training method are roughly parallel, a natural approximation for the likelihoods obtained by the full (non-abbreviated) method using the likelihoods of the abbreviated method is

$$l(D_i|m_i) \cong l(D_i|m_i^a) + \delta \quad (11)$$

where  $\delta$  is an offset and  $m_i^a$  is the model learned by an abbreviated training method applied to data set  $D_i$ . A straightforward and fairly efficient way to approximate the offset  $\delta$  is to apply both the full and abbreviated training methods to  $D_1$ , the first (and smallest) data set in the sequence. We shall use this approach in our experiments described in the next section. Note that, when Equation 11 is substituted into Equation 10, the offset cancels in the numerator and thus only affects the estimation of  $l(D_i|m_i)$  in the denominator.

With regards to the determination of the cost to move from stage  $i$  to stage  $i + 1$ , there are now three components: (1) the time to run EM to full convergence on  $D_i$ , (2) the time to run the abbreviated training method on  $D_{i+1}$ , and (3) the time to run EM to full convergence on  $D_{i+1}$ . The first component is associated with the decision to stop at stage  $i$ , whereas the second and third components are associated with the decision to continue to stage  $i + 1$ . All three components can be estimated using Equation 4 with the following caveats. One, if the abbreviated training method uses a fixed number of EM iterations, there is no need to estimate  $I_{i+1}$ —this value is independent of  $i$  and known in advance. Two, if the abbreviated training method uses a convergence threshold,  $I_{i+1}$  can be estimated using Equation 6. Three, the time required to run EM to full convergence is approximated using

$$I_{i+1} \cong I_1 \quad (12)$$

The use of this approximation requires that we run EM to full convergence on only the first data set  $D_1$ —the same run used to determine the offset  $\delta$ . We thus avoid computationally expensive runs on larger data sets.

In closing this discussion, we note that our stopping criterion is potentially sensitive to local variations in the learning curve due to—for example—convergence to different local maxima during successive runs of EM. In our experience, we have found that learning curves for model-based clustering methods are usually smooth. In situations where the learning curves are less smooth, techniques that use additional samples to assess the shape of the learning curve (e.g., Provost, et al., 1999) may be useful and the use of an abbreviated training method can reduce the cost of doing so.

## 6. Empirical Study

In this section, we describe an evaluation of our learning-curve sampling method and extensions that used the MSNBC, MS.COM and USCensus1990 data. Our goals are to verify that (1) our standard learning-curve sampling method outperforms standard EM, and (2) that the abbreviated methods outperforms the standard learning-curve sampling method.

The primary measure of algorithm performance that we used was the algorithm’s actual (as opposed to expected) utility as given by Equations 1 through 3. The holdout data set

used to measure utility was the same as that used during the evaluation of the stopping criterion. In our experiments, we used a holdout data set of 10,000 randomly chosen cases. (Larger data sets produced similar results.) For each experimental condition, in addition to utility, we measured the components of utility (relative benefit and run time), the selected sample size ( $N_{lc}$ ), the *speedup factor*—the total run time of the EM-full algorithm divided by the run time of the evaluated algorithm—and the *overhead ratio*—the total run time of the evaluated algorithm divided by the run time for EM to run to full convergence on the selected sample size  $N_{lc}$ .

In our study, we compared the standard learning-curve sampling method described in Section 4 using an EM convergence threshold of  $10^{-5}$  with the standard EM algorithm run on the full data set using a threshold of  $10^{-5}$ . We call these methods *Standard LCS* and *EM-full*, respectively. In addition, we evaluated abbreviated training methods that used 1, 3, 5, and 10 EM steps as well as EM convergence thresholds of  $10^{-1}$ ,  $10^{-2}$ ,  $10^{-3}$ , and  $10^{-4}$ . In all conditions, we used a final EM run with a convergence threshold of  $10^{-5}$ . These methods are denoted *fixed-1*, *thresh-0.1*, and so on. Also, we considered a wide range of stopping thresholds:  $\alpha = 1$  benefit per hour,  $\alpha = 0.2$  benefit per hour, and  $\alpha = 0.04$  benefit per hour (1 benefit per day).

In every run, we used the geometrically increasing sequence  $D_1 = 40,000$ ,  $D_2 = 80,000$ , and so on. We selected  $D_1 = 40,000$  because the use of extremely small initial data sets produced poor estimates of  $I_i$ , and the application of EM to data sets of size 40,000 were fast enough to be unobtrusive—less than two minutes for each of the data sets. We used 10,000 cases to train the baseline model. (Again, larger data sets produced similar results.) In addition, as we varied the abbreviated training method, we held other experimental conditions fixed including the parameter initialization for EM. In every condition, after selecting the appropriate sample size, we used the parameters obtained by the abbreviated training method as the initial parameters for the final run of EM to full convergence. Finally, in preliminary experiments, we examined mixture models with 25, 50, and 100 components. The results were similar, and so we report results for the 25-component mixture model only.

All runs in our study were performed on a Pentium III (Family 6 Model 8 Stepping 3) 800 MHz Processor running the Windows 2000 Professional operating system. We used enough amount of random access memory—2 gigabytes—so that each of the data sets fit into memory. This eliminated the need for the operating system to perform disk accesses (swapping) when learning. We note that results using a system with less memory would have shown an even more striking improvement than those described below.

Figure 4 contains a graphical summary of the utilities obtained for the MSNBC, MS.COM and USCensus1990 domains as a function of training method. The important observations are that (1) the Standard LCS obtains higher utilities than EM-full, (2) the abbreviated methods usually obtain higher utilities than Standard LCS, and (3) the fixed and thresh methods yielded similar utilities with the exception of the lower-convergence-threshold thresh methods, which did not perform as well.

Tables 1, 2 and 3 show the detailed results for the MSNBC, MS.COM and USCensus1990 domains, respectively. These tables contain the utilities presented in Figure 4 as well as speedup factors, overhead ratios, run times, benefits and selected sample sizes for the experiments.

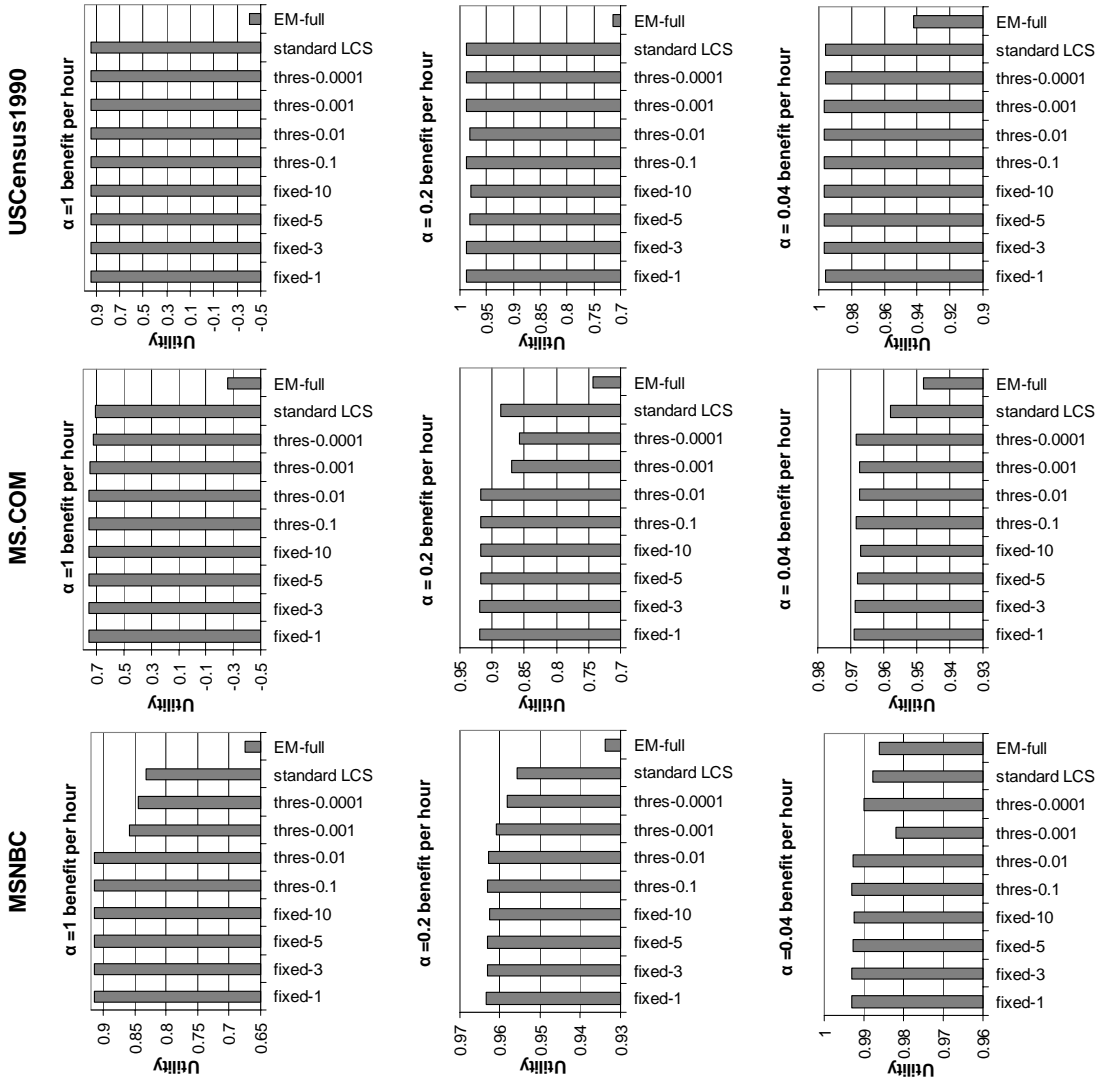


Figure 4: Utilities as a function of training method and  $\alpha$ .

These results help us discriminate the performance of the abbreviated methods. In all cases but one, fixed-1 does best. In particular, this abbreviated training method usually chooses the same sample size as the other training methods, but has the advantage of running the fastest. This speedup is demonstrated by the low overhead ratio for the fixed-1 method. In addition, the speedup factors are dramatic. For larger values of  $\alpha$ , the run times of the fixed-1 learning-curve sampling method are an order-of-magnitude smaller than that for EM run on the full data sets.

Note that the values for  $N_{lc}$  increase as the stopping threshold  $\alpha$  decreases. This is expected as  $\alpha$  corresponds to the relative importance of benefit to cost. Also note that the speedup factor increases with the size of the data set. Naturally, when the learning-curve sampling methods choose smaller sample size it leads to large speedup factors. For instance, on the USCensus1990 data set the speedup factor for an  $\alpha = 1$  benefit per hour, the speedup factor is 25. For massive data sets, speedups can be arbitrarily large. This is especially true for larger data sets where the data set cannot fit into random access memory.

In addition to the results described above, we evaluated the sensitivity of our results to the choice of data sequence. In particular, we evaluated all methods using  $\alpha = 1, 0.2, 0.04$  benefit per hour on ten different data sequences randomly generated from the MS.COM data set. In all runs except for three runs at  $\alpha = 0.04$  benefit per hour, Standard LCS yielded higher utilities than full-EM. Also, in all runs, fixed-1 yielded higher utilities than both Standard LCS and full-EM. Furthermore, in all but one run, fixed-1 yielded the highest utilities among abbreviated methods.

Finally, let us evaluate our approximations for expected values described in Equations 4, 6, 8, 9, 11, and 12. To do so, we can compare the sample sizes ( $N_{lc}$ ) chosen by our algorithm, to the sample sizes ( $N_{oracle}$ ) chosen by the same algorithm but where the future benefits and costs are known with certainty. Differences between these sample sizes reflect the quality of our approximations. Table 4 shows this comparison. As seen in the table, the largest differences in sample size correspond to differences of two stages. In addition,  $N_{oracle} \leq N_{lc}$  in all experimental conditions, providing evidence for the conservative nature of the approximations.

## 7. Related Work

In this section, we discuss related work on choosing the number of samples when applying a learning method.

Domingos & Hulten (2001) describe a method called *VFKM* that chooses a sample size for K-means clustering. The sample sizes they consider are adaptively chosen using Hoeffding bounds on the accuracy of the output of K-means as a function of sample size.

John & Langley (1996) describe a method called *dynamic sampling* in which they consider a sequence of data sets whose sizes increase by a fixed increment. Their method, which they apply to classification problems, stops at data set  $D_i$  if  $\widehat{acc(D)} - acc(D_i) > \epsilon$  where  $\widehat{acc(D)}$  is an estimate of the accuracy on the entire data set and  $acc(D_i)$  is the accuracy on the current data set. Their method estimates the quantity  $\widehat{acc(D)}$  by extrapolating the shape of the learning curve.

Provost et al. (1999) describe a method called *progressive sampling*, which they apply to classification problems. They consider a sequence of data sets whose sizes increase



Table 1: Selected sample sizes ( $N_{lc}$ ), holdout scores, run times, utilities, speedups, and overheads as a function of training method and stopping threshold  $\alpha$  for the MSNBC domain.

MSNBC

	$\alpha = 1$ benefit per hour					
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	80000	0.978	0.063	0.915	5.12	1.65
fixed-3	80000	0.978	0.063	0.915	5.12	1.65
fixed-5	80000	0.978	0.063	0.915	5.12	1.65
fixed-10	80000	0.978	0.063	0.915	5.12	1.65
thres-0.1	80000	0.978	0.063	0.915	5.12	1.65
thres-0.01	80000	0.978	0.063	0.915	5.12	1.65
thres-0.001	160000	0.987	0.128	0.859	2.53	1.41
thres-0.0001	160000	0.987	0.142	0.845	2.29	1.56
standard LCS	160000	0.987	0.155	0.833	2.10	1.70
EM-full	587971	1.000	0.325	0.675	1.00	1.00
	$\alpha = 0.2$ benefit per hour					
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	160000	0.987	0.117	0.963	2.79	1.28
fixed-3	160000	0.987	0.117	0.963	2.76	1.29
fixed-5	160000	0.987	0.118	0.963	2.75	1.30
fixed-10	160000	0.987	0.120	0.963	2.70	1.32
thres-0.1	160000	0.987	0.117	0.963	2.77	1.29
thres-0.01	160000	0.987	0.119	0.963	2.72	1.31
thres-0.001	160000	0.987	0.128	0.961	2.53	1.41
thres-0.0001	160000	0.987	0.142	0.958	2.29	1.56
Standard LCS	160000	0.987	0.155	0.956	2.10	1.70
EM-full	587971	1.000	0.325	0.934	1.00	1.00
	$\alpha = 0.04$ benefit per hour					
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	320000	1.001	0.188	0.993	1.72	1.16
fixed-3	320000	1.001	0.191	0.993	1.70	1.18
fixed-5	320000	1.001	0.194	0.993	1.68	1.19
fixed-10	320000	1.001	0.200	0.993	1.62	1.24
thres-0.1	320000	1.001	0.191	0.993	1.70	1.18
thres-0.01	320000	1.001	0.196	0.993	1.65	1.21
thres-0.001	160000	0.987	0.128	0.982	2.53	1.41
thres-0.0001	320000	1.001	0.265	0.990	1.22	1.64
standard LCS	320000	1.001	0.317	0.988	1.03	1.95
EM-full	587971	1.000	0.325	0.986	1.00	1.00

Table 2: Selected sample sizes ( $N_{lc}$ ), holdout scores, run times, utilities, speedups, and overheads as a function of training method and stopping threshold  $\alpha$  for the MS.COM domain.

MS.COM

$\alpha = 1$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	160000	0.892	0.132	0.760	9.54	1.28
fixed-3	160000	0.892	0.133	0.760	9.50	1.28
fixed-5	160000	0.892	0.134	0.759	9.43	1.29
fixed-10	160000	0.892	0.135	0.757	9.33	1.31
thres-0.1	160000	0.892	0.133	0.759	9.46	1.29
thres-0.01	160000	0.892	0.136	0.756	9.23	1.32
thres-0.001	160000	0.892	0.143	0.750	8.83	1.38
thres-0.0001	160000	0.892	0.162	0.730	7.76	1.57
standard LCS	160000	0.892	0.185	0.707	6.81	1.79
EM-full	1928877	1.000	1.259	-0.259	1.00	1.00
$\alpha = 0.2$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	320000	0.972	0.261	0.919	4.82	1.13
fixed-3	320000	0.972	0.263	0.918	4.78	1.13
fixed-5	320000	0.972	0.266	0.918	4.74	1.15
fixed-10	320000	0.972	0.271	0.917	4.65	1.17
thres-0.1	320000	0.972	0.265	0.918	4.76	1.14
thres-0.01	320000	0.972	0.271	0.917	4.64	1.17
thres-0.001	640000	0.997	0.625	0.869	2.02	1.22
thres-0.0001	640000	0.997	0.680	0.858	1.85	1.33
standard LCS	320000	0.972	0.417	0.887	3.02	1.80
EM-full	1928877	1.000	1.259	0.743	1.00	1.00
$\alpha = 0.04$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	1280000	1.000	0.744	0.969	1.69	1.05
fixed-3	1280000	1.000	0.754	0.969	1.67	1.06
fixed-5	1280000	1.000	0.765	0.968	1.65	1.08
fixed-10	1280000	1.000	0.790	0.967	1.59	1.11
thres-0.1	1280000	1.000	0.760	0.968	1.66	1.07
thres-0.01	1280000	1.000	0.783	0.967	1.61	1.10
thres-0.001	1280000	1.000	0.787	0.967	1.60	1.11
thres-0.0001	640000	0.997	0.680	0.968	1.85	1.33
standard LCS	640000	0.997	0.928	0.958	1.36	1.82
EM-full	1928877	1.000	1.259	0.948	1.00	1.00

Table 3: Selected sample sizes ( $N_{lc}$ ), holdout scores, run times, utilities, speedups, and overheads as a function of training method and stopping threshold  $\alpha$  for the US-Census1990 domain.

$\alpha = 1$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	80000	0.998	0.056	0.942	25.05	1.43
fixed-3	80000	0.998	0.056	0.942	25.05	1.43
fixed-5	80000	0.998	0.056	0.942	25.05	1.43
fixed-10	80000	0.998	0.056	0.942	25.05	1.43
thres-0.1	80000	0.998	0.056	0.942	25.05	1.43
thres-0.01	80000	0.998	0.056	0.942	25.05	1.43
thres-0.001	80000	0.998	0.056	0.942	25.05	1.43
thres-0.0001	80000	0.998	0.056	0.942	25.05	1.43
standard LCS	80000	0.998	0.056	0.942	25.05	1.43
EM-full	2448248	1.000	1.403	-0.403	1.00	1.00
$\alpha = 0.2$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	80000	0.998	0.056	0.987	25.20	1.42
fixed-3	80000	0.998	0.056	0.987	25.09	1.43
fixed-5	160000	1.001	0.094	0.980	14.85	1.32
fixed-10	160000	1.001	0.101	0.980	13.93	1.41
thres-0.1	80000	0.998	0.056	0.987	24.94	1.43
thres-0.01	160000	1.001	0.096	0.981	14.62	1.34
thres-0.001	80000	0.998	0.057	0.987	24.78	1.44
thres-0.0001	80000	0.998	0.056	0.987	25.12	1.43
standard LCS	80000	0.998	0.056	0.987	25.19	1.42
EM-full	2448248	1.000	1.403	0.714	1.00	1.00
$\alpha = 0.04$ benefit per hour						
Training method	$N_{lc}$	Benefit	Run time (hours)	Utility	Speedup factor	Overhead ratio
fixed-1	80000	0.998	0.056	0.996	25.05	1.43
fixed-3	160000	1.001	0.092	0.997	15.25	1.29
fixed-5	160000	1.001	0.094	0.997	14.85	1.32
fixed-10	160000	1.001	0.101	0.997	13.93	1.41
thres-0.1	160000	1.001	0.093	0.997	15.01	1.31
thres-0.01	160000	1.001	0.096	0.997	14.62	1.34
thres-0.001	160000	1.001	0.104	0.996	13.46	1.46
thres-0.0001	80000	0.998	0.056	0.996	25.05	1.43
standard LCS	80000	0.998	0.056	0.996	25.05	1.43
EM-full	2448248	1.000	1.403	0.942	1.00	1.00

Table 4: Sample sizes ( $N_{lc}$ ) selected by learning-curve sampling methods and sample sizes ( $N_{oracle}$ ) selected by these methods when given the true incremental benefit and cost at each stage.

## MSNBC

	$\alpha = 1$ benefit per hour		$\alpha = 0.2$ benefit per hour		$\alpha = 0.04$ benefit per hour	
Training method	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$
fixed-1	80000	80000	160000	8000	320000	320000
fixed-3	80000	80000	160000	8000	320000	320000
fixed-5	80000	80000	160000	8000	320000	320000
fixed-10	80000	80000	160000	8000	320000	320000
thres-0.1	80000	80000	160000	8000	320000	320000
thres-0.01	80000	80000	160000	8000	320000	320000
thres-0.001	160000	80000	160000	8000	160000	320000
thres-0.0001	160000	80000	160000	8000	320000	320000
standard LCS	160000	80000	160000	8000	320000	320000

## MS.COM

	$\alpha = 1$ benefit per hour		$\alpha = 0.2$ benefit per hour		$\alpha = 0.04$ benefit per hour	
Training method	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$
fixed-1	160000	160000	320000	320000	1280000	640000
fixed-3	160000	160000	320000	320000	1280000	640000
fixed-5	160000	160000	320000	320000	1280000	640000
fixed-10	160000	160000	320000	320000	1280000	640000
thres-0.1	160000	160000	320000	320000	1280000	640000
thres-0.01	160000	160000	320000	320000	1280000	640000
thres-0.001	160000	160000	640000	320000	640000	640000
thres-0.0001	160000	160000	640000	320000	640000	640000
standard LCS	160000	160000	320000	320000	640000	640000

## USCensus1990

	$\alpha = 1$ benefit per hour		$\alpha = 0.2$ benefit per hour		$\alpha = 0.04$ benefit per hour	
Training method	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$	$N_{lc}$	$N_{oracle}$
fixed-1	80000	40000	80000	40000	80000	80000
fixed-3	80000	40000	80000	40000	160000	80000
fixed-5	80000	40000	160000	40000	160000	80000
fixed-10	80000	40000	160000	40000	160000	80000
thres-0.1	80000	40000	80000	40000	160000	80000
thres-0.01	80000	40000	160000	40000	160000	80000
thres-0.001	80000	40000	80000	40000	160000	80000
thres-0.0001	80000	40000	80000	40000	80000	80000
standard LCS	80000	40000	80000	40000	80000	80000

geometrically, and stop at data set  $D_i$  with  $|D_i|$  cases if the slope of the learning curve at  $|D_i|$  is below some threshold  $\delta$ . The slope of the learning curve at a particular data set is determined by linear regression from a set of points in the neighborhood of  $|D_i|$ . As the authors indicate, this can lead to an accurate estimate of the slope of the learning curve but at a significant cost to the overall run-time. As mentioned in Section 5, the use of an abbreviated training method can significantly reduce this cost.

When viewed from a decision-theoretic perspective, the methods of Domingos & Hulten (2001) and John & Langley (1996) do not consider the costs of computation. Instead, they seek to achieve a result whose performance differs from that obtained using the full data set by at most a fixed amount. Thus, from our perspective, these algorithms may be performing too little or too much computation, depending on how easy it is to obtain the performance guarantee. Also, whereas Provost et al. (1999) do consider a tradeoff, they consider one between benefit and sample size rather than between benefit and cost.

Finally, we note that we chose the name *learning-curve sampling method* because we felt that the name was more descriptive than either dynamic or progressive sampling. Both of these alternative names capture the notion that one is choosing larger and larger samples of examples, but fail to indicate the method by which these choices are being made.

## 8. Summary and Future Work

Learning-curve sampling methods are a natural way to apply a learning algorithm to large data sets. In this paper, we have formalized the cost-benefit tradeoff in the learning-curve sampling method in terms of decision theory. In addition, we have applied the learning-curve sampling method to the task of model-based clustering, and have shown that the approach yields higher utilities—dramatically increasing computational efficiency while sacrificing little accuracy. Finally, we have shown the use of one-step EM to identify sample size yields even higher utility.

There are many areas for future investigation. For example, one can consider (1) benefits and costs that are non-linear in model-score and run time, respectively, (2) methods for choosing the size of  $D_1$ , (3) and on-the-fly selection of sample size and training method. As another example, our methods can be extended to include the simultaneous selection of sample size and number of clusters. Here, an interesting challenge arises because the optimal number of clusters may increase with the size of the data set. Finally, our approach of using computationally efficient abbreviated training methods for determining the appropriate number of training cases can be—in principle—applied to various iterative training methods such as stochastic gradient descent and Newton-Raphson, and to alternative statistical models including classification/regression models and finite mixture models having components without the mutual independence assumption. The performance of our approach on these alternative training methods and model classes should be investigated.

## Acknowledgments

We would like to thank Steven White for the MSNBC data set and Lolan Song for the MS.COM data set.

## References

- P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI Press, Menlo Park, CA, 1995.
- G. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.
- A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B 39:1–38, 1977.
- P. Domingos and G. Hulten. A general method for scaling up machine learning algorithms and its application to clustering. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 106–113. Morgan Kaufmann, San Mateo, CA, 2001.
- R.A. Howard. Decision analysis: Applied decision theory. In D.B. Hertz and J. Melese, editors, *Proceedings of the Fourth International Conference on Operational Research*, Cambridge, MA, pages 55–71. International Federation of Operational Research Societies, 1966.
- G. John and P. Langley. Static versus dynamic sampling for data mining. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 367–370. AAAI Press, 1996.
- C. Kadie. *Seer: Maximum Likelihood Regression for Learning-Speed Curves*. PhD thesis, Department of Computer Science, University of Illinois, Urbana, IL, August 1995.
- G. McLachlan and K. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988.
- D. Pearce. *Cost-benefit analysis*. Macmillan, New York, 1983.
- F. Provost, D. Jensen, and T. Oates. Efficient progressive sampling. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 23–32. ACM, 1999.
- 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.
- D.M. Titterton, A.F.M. Smith, and U.E. Makov. *Statistical analysis of finite mixture distributions*. John Wiley and Sons, 1985.