Seventh IEEE International Conference on Data Mining

Bandit-Based Algorithms for Budgeted Learning

Kun Deng Chris Bourke Stephen Scott Julie Sunderman
Yaling Zheng
Department of Computer Science & Engineering
University of Nebraska–Lincoln
Lincoln, NE 68588-0115 USA
Email: {kdeng, cbourke, sscott, jsunderm, yzheng}@cse.unl.edu

Abstract

We explore the problem of budgeted machine learning, in which the learning algorithm has free access to the training examples' labels but has to pay for each attribute that is specified. This learning model is appropriate in many areas, including medical applications. We present new algorithms for choosing which attributes to purchase of which examples in the budgeted learning model based on algorithms for the multi-armed bandit problem. All of our approaches outperformed the current state of the art. Furthermore, we present a new means for selecting an example to purchase after the attribute is selected, instead of selecting an example uniformly at random, which is typically done. Our new example selection method improved performance of all the algorithms we tested, both ours and those in the literature.

1. Introduction

Approaches to typical machine learning applications usually operate under the assumption that data are freely available. That is, it is usually taken for granted that an abundance of fully specified instances along with their labels are available to build a classifier. However, in many real-world applications, this assumption is far from realistic. Instead, collecting and specifying data may be very time-consuming and costly.

A recent area in machine learning has attempted to address the problem of learning in the absence of labels, so-called *active learning*. Rather than a passive learner that simply builds a hypothesis based on the available attribute/label pairs, an active learner must also choose which instances it wants some oracle to label. The efficacy of a learning algorithm is measured by its label complexity—how well it can learn with the least number of label requests.

More recently, a new line of research called *budgeted learning* [6, 11, 12] has focused on what may be considered a dual of active learning. In budgeted learning, a learner considers instances in which the labels are specified, but the attributes are not. Instead, the learner purchases attributes at some fixed cost subject to an overall budget. The difficulty for the learner is to decide which attributes of which instances will provide the best model from which to learn.

The original motivation for the budgeted learning model came from medical applications where the outcome of a treatment, drug trial, or control group (labels of an instance) is known and the features (results of running medical tests) are available for a price. Further, the costs of such tests can vary with some being quite expensive.

We present new algorithms for choosing which attributes of which examples to purchase in the budgeted learning model. Our algorithms are based on results in the "multi-armed bandit" model. In this model, we have n slot machines that we may play for some fixed number of rounds. Each round, one must decide which single slot machine to play in order to maximize total reward over all rounds. Our first two algorithms are based on the algorithm Exp3 of Auer et al. [2] originally designed for the multi-armed bandit problem, and our third is based on the "follow the perturbed leader" approach of Kalai & Vempala [8]. On the six data sets that we tested, we found that all three (especially Exp3CR and FEL) overall outperformed the current state of the art, Randomized Single Feature Lookahead (RSFL) [11].

Most budgeted learning algorithms focus only on selecting attributes to purchase and then choose uniformly at random the example whose attribute is to be purchased. In other words, such algorithms consider one example to be as good as any other given the selected attribute. We present a new method for selecting examples to purchase. This approach is based on a simple yet effective active learning algorithm (where the goal is to choose fully-specified examples to be labeled): choose the example whose classification is most uncertain in the current model. We measure uncertainty based on the Kullback-Liebler divergence between an

1550-4786/07 $25.00 © 2007 IEEE
DOI 10.1109/ICDM.2007.91

instance's class probability vector and the uniform distribution, choosing the instance whose class probability vector is closest to uniform. We found that such an instance selector overall improved performance of all algorithms that we tested, including those from the literature.

For simplicity, we focus on a unit-cost model of budgeted learning, i.e. all attributes cost one monetary unit. Future work is to generalize our algorithms to handle nonuniform cost models.

2. Background & Related Work

Budgeted learning is related to conventional machine learning techniques in that one is given a set D of labeled training data and infers a classifier (hypothesis, or model) to label new examples. The key difference is that in budgeted learning, the learner starts by only knowing the labels of the available examples and has to "buy" the attributes (features), spending at most some fixed budget B. This is in contrast with active learning (AL), in which one knows the features but has to pay for labels. In the budgeted learning model, one can think of D as a table where each row is an instance (example), each column is an attribute (with an additional column for the class label), and all entries in the table except for the class label are initialized with a value of "?" for "unknown." Buying an attribute/label pair means purchasing a specific attribute from any single instance with a given label. Budgeted learning falls under a general framework of problems that represent a trade-off between *exploration versus exploitation* in an online decision process [2].

One of the original applications of budgeted learning is in the medical domain. In this application, the examples are patients, and the (known) label of patient x is a label $y \in \{-1, +1\}$, indicating whether or not x responded to a particular treatment. The (initially unknown) attributes of x are the results of tests performed on tissue samples gathered from patient x, e.g. a blood test for the presence of a particular antibody. In this case, any attribute of any example can be determined. However, each costs time and money, and there is a finite budget limiting the attributes that one can buy. Further, each attribute can cost a different amount of money, e.g. a blood test may cost less than a liver biopsy.

A second existing application of budgeted learning is in customer modeling [19] where a company has significant data (attributes) on its own customers but may have the option to pay for other information on the same customers from other companies. Other work [16] has focused on budgeted learning with the goal of detecting irrelevant features rather than to build a classifier. Related to budgeted learning is the learning of "bounded active classifiers," in which one has fully-specified training examples with their labels, but the final hypothesis h must pay for attributes of new examples when predicting a label, spending at most B_h. Such classifiers can be learned in the budgeted learning model [6, 9, 10].

Most theoretical studies of budgeted learning have considered related problems within the exploration versus exploitation framework, such as the *multi-armed bandit problem* (Section 2.2) and the *coins problem* [12] which is a special case of budgeted learning. The coins problem is hard even under special conditions [12], but a 4-approximation algorithm was recently demonstrated [7].

2.1. Algorithms

One of the simplest early budgeted learning algorithms was Biased Robin (BR) [11, 10]. BR is similar to a Round Robin approach except that attribute i is repeatedly purchased until such purchases are no longer "successful," and then the algorithm moves on to attribute $i+1$. Here, success is measured by a feedback function, examples of which are described below.

Another method, Single Feature Lookahead (SFL) [11] introduces a lookahead into the state space S of all possible purchases without explicitly expanding the entire state space. At any point in the algorithm's execution, one has an *allocation* α: a description of how many times a particular value of a particular feature has been purchased from examples of a specified label. Given a current allocation, SFL calculates the expected loss of performing all possible single-purchase actions (purchasing an attribute/label pair which results in a specific attribute value) and chooses the action that minimizes the expected loss of the resulting allocation. Here, the expected loss is the sum of the probability of each possible allocation α' resulting from a single purchase multiplied with its loss with respect to the naïve Bayes model. Several heuristics have been considered for the loss function, including the GINI index [11] and expected classification error [10]. In our experiments, we found that Randomized SFL and Biased Robin performed their best when using conditional entropy (as used by Kapoor & Greiner [10]):

$$\mathsf{CE}(i, y) = -\sum_{k} P(a_i = k) \sum_{y} Pr(\cdot) \log_2 \left(Pr(\cdot)\right), \quad (1)$$

where $Pr(\cdot) = P(y \mid a_i = k)$ is the probability of an instance having label y given that its ith attribute has a value k as specified by the naïve Bayes model. Conditional entropy essentially measures the uncertainty of the class label given the value of an attribute.

We compared our algorithms against a randomized version of SFL called RSFL [10]. In RSFL, the conditional entropy is used to define a Gibbs distribution from which an attribute-label pair is chosen. RSFL can be characterized as the current state of the art within budgeted learning.

2.2. The Multi-Armed Bandit Problem

There are close connections between budgeted learning and the so-called "multi-armed bandit problem" first studied by Robbins [14], in which a gambler repeatedly chooses a slot machine to play, each with a different payoff. The gambler's goal is to maximize the total payoff over all pulls of all machines. The key difference between budgeted learning and the multi-armed bandit problem is that in the latter, one tries to maximize the sum of the rewards over all pulls, whereas with budgeted learning, one simply wants to maximize the accuracy of the resulting learner.

In the context of solving a multi-armed bandit problem, we apply results from Auer et al. [2]. Their most basic algorithm (Exp3) maintains a weight w_k (initialized to 1) for each arm. At each trial, it plays machine k with probability

$$P(k) = \frac{\gamma}{n} + (1-\gamma)\frac{w_k}{\sum_{j=1}^{n} w_j} \quad , \tag{2}$$

where γ is a parameter governing the mixture between the weight-based distribution (controlling exploitation) and the uniform distribution (allowing for exploration). After playing the chosen machine (call it k), reward r is returned, which is used to update weight w_k by multiplying it by $\exp(\gamma\, r/(P(k)n))$ (all other weights are unchanged).

Auer et al. proved that under appropriate conditions, the expected total reward of Exp3 will not differ from the best possible by more than $2.63\sqrt{gn\ln n}$, where g is an upper bound on the total reward of the best sequence of choices.

3. Our Algorithms

The theoretical guarantees in the work Auer et al. give us good motivation for using similar approaches on the budgeted learning problem. In particular, since Auer et al. make no assumptions about the underlying distribution of slot machines, we can plug in our choice of reward function for the slot machines (say, conditional entropy) and their bounds automatically translate into guarantees in the budgeted learning context. However, these bounds only apply to the regret with respect to the best sequence of arm pulls. It remains open whether under some conditions, we could bound the overall error with respect to the best set of purchases or overall error rate.

3.1. Exp3-Based Algorithms

Our first two algorithms are based on the multi-armed bandit algorithm Exp3 of Auer et al. [2] described in Section 2.2. Our first algorithm (Exp3C, for "Exp3-Column") treats each attribute (column) as an arm. Each column has a weight that is initialized to 1. Each purchasing round, Exp3C chooses a column to buy based on the weights. Specifically, Exp3C purchases column k according to the probability in Equation (2). After the purchase, we build a new naïve Bayes model on the training set and Exp3C gets as a reward the classification accuracy of the naïve Bayes model evaluated on the partially specified training set.[1]

After choosing a column to purchase, a row must also be selected. After choosing a column i, Exp3C selects a row uniformly at random from all rows that do not yet have column i filled in (ultimately, we improve performance by using KL divergence to select rows; see Section 3.3). Since we are assuming a naïve Bayes model, all attributes (columns) are conditionally independent of each other given the label. Thus, it doesn't matter which example (row) is chosen. This approach is common among current budgeted learning algorithms [11].

In our second algorithm (Exp3CR, for "Exp3-Column-Row"), we relax this assumption by defining a distribution over the rows as well as the columns, i.e. we now have two weight vectors instead of one. After choosing a column according to its distribution (which is done the same way as in Exp3C), our algorithm then chooses a row according to the row distribution. Once reinforcement is received, both weight vectors are updated independently of each other. Thus we replace the naïve Bayes assumption with a product distribution over the (column, row) pairs, and the most informative examples and most informative attributes will eventually be weighted more heavily.

3.2. Perturbed Leader

Our third algorithm is a variation of the "follow the perturbed leader" type algorithms due to Kalai & Vempala [8]. Though originally designed as an online expert-based algorithm, it is equally applicable as an algorithm for the multi-armed bandit problem. As with the previous two algorithms, we treat each attribute as an arm. The idea is to simply select the most informative attribute by selecting the best attribute so far, thus "follow the leader." At each time step, a certain cost (or equivalently reward, as they are inversely related) is counted toward the attribute selected. At time t, the accumulative cost of each attribute can be calculated, and the attribute that has incurred the least cost is chosen as the next purchase. Without randomization, an adversary can easily trick such deterministic algorithms into wrong decisions. To address this problem, a random perturbation is added to the cost of each attribute before making the decision, thus the name "follow the perturbed leader."

Similar to the results of Auer et al. [2], it can be shown that, under appropriate conditions, the regret of the perturbed leader algorithm is small relative to the best sequence of choices. Let $x_i(t)$ be the cost of the ith attribute at time step t. At each time step t, FPL computes the sum of all costs of each arm, $c_i = \sum_{j=1}^{t-1} x_i(j)$ and adds a perturbation factor (or noise) p_i generated uniformly at random from

[1] We tried several reward functions: GINI index, expected classification error, and conditional entropy. Classification error on the training set tended to work best for our algorithms.

$[0, \frac{1}{\epsilon}]$ where ϵ is a parameter. FPL then choses to play the arm $\text{argmin}_{1 \le i \le n}\{c_i + p_i\}$.

The framework for FPL assumes that we have access to the costs $x_i(t)$ for all arms at every time step (had they been chosen). However, this assumption is not reasonable in the context of budgeted learning. For this reason, our implementation is a slight variation of the standard FPL called FEL ("follow the *expected* leader") [8]. First, we assume that $x_i(t)$ is zero if the arm was not played (the attribute was not chosen as a purchase). Next, let $\#x_i(t)$ be the number of times attribute i was chosen up to time step t. Now, instead of accumulated cost, we use the average of the perturbed cost (over the trials that an attribute is actually purchased) as selection criteria. Just as with Exp3C and Exp3CR, we measure the cost as the training error on the partially specified training set. Again, we considered other measures, but ultimately chose this.

3.3. Improving Algorithms via KL Divergence

Most budgeted learning algorithms (except Exp3CR) focus only on selecting columns for purchase, implicitly assuming that given a column, any instance (or any instance with a given label) is equally good. Thus rows are selected uniformly at random. However, it is not always the case that two instances are equally informative given an attribute. Thus, we refine these algorithms (Exp3C and FEL in addition to BR and RSFL) by defining a criterion for choosing specific instances from which to purchase an attribute.

Intuitively, given a selected column j, we seek a row i such that the purchase of (i, j) gives the most information possible. Thus our algorithm chooses an instance whose classification is least certain under the current model. That is, we choose an example nearest to the current model's decision boundary. This technique has been very successful in active learning [3, 15, 17]. For naïve Bayes this means choosing the instance whose posterior class probability distribution is closest to uniform over the classes. To measure this, we use the Kullback-Liebler (KL) divergence between the instance's posterior class probability vector P and the uniform distribution. This is equivalent to maximizing the entropy of the posterior: $-\sum_{i=1}^{n} p_i \log_2 p_i$. In the event of ties, we choose a row uniformly at random among instances that have attribute j unpurchased. In the case of RSFL, the algorithm is making an informed choice about the label, so we only consider the KL divergence of instances with the appropriate label.

4. Experimental Results

In this section we present our experimental results on several UCI data sets [5]. To compare against RSFL, we chose only data sets that had nominal attributes or could easily be made nominal. For the few data sets with missing attributes, the mode was used to fill in that attribute value.

Table 1. Summary Statistics.

(a) Target Budget and (Data Utilization Ratios).

Dataset	Exp3C	Exp3CR	FEL	RSFL	BR
cancer	169.0	38.0	12.0	250.0	250.0
	(0.73)	(0.16)	(0.05)	(1.08)	(1.08)
colic	241.0	229.0	250.0	244.0	250.0
	(1.09)	(1.04)	(1.13)	(1.1)	(1.13)
kr-vs-kp	250.0	250.0	250.0	250.0	250.0
	(1.09)	(1.09)	(1.09)	(1.09)	(1.09)
mushroom	250.0	179.0	250.0	136.0	178.0
	(1.08)	(0.77)	(1.08)	(0.59)	(0.77)
vote	142.0	52.0	110.0	79.0	20.0
	(3.23)	(1.18)	(2.5)	(1.8)	(0.45)
zoo	114.0	195.0	118.0	128.0	155.0
	(0.67)	(1.15)	(0.69)	(0.75)	(0.91)
Mean TB	194.33	157.17	165	181.17	183.83
Mean DUR	1.31	0.9	1.09	1.07	0.91

(b) Area Above the Learning Curve Below Random.

Dataset	Exp3C	Exp3CR	FEL	RSFL	BR
cancer	7.45	11.53	12.14	-3.35	1.4
colic	-1.21	1.35	-1.49	-10.04	-1.58
kr-vs-kp	-2.84	-2.75	-3.09	-3.77	-20.06
mushroom	3.32	6.36	3.91	10.67	9.13
vote	0.91	0.01	0.27	1.45	2.87
zoo	11.61	4.17	10.06	9.98	7.58
Mean	3.21	3.45	3.63	0.82	-0.11

All algorithms were written in Java within the Weka machine learning framework [18] and used its naïve Bayes as a base learner.

We used 10-fold cross validation in our tests. In addition to our algorithms (Exp3C, Exp3CR, FEL) and those in the literature described earlier (RSFL and BR), as a control, we also considered a "random shopper" that uniformly at random selects an unpurchased instance/attribute pair. For our algorithms, we used the classification accuracy on the partially specified training set as a reward function. For RSFL and BR, we used conditional entropy. Again, we tried various reward functions (GINI index, expected classification error) with each algorithm, but chose the reward functions that worked best for each algorithm. When applicable, we ran each algorithm with uniform selection of rows as well as with the KL divergence-based approach of Section 3.3. For simplicity we assumed uniform costs over all attributes. For the algorithms Exp3C and Exp3CR, we tuned the γ parameter by taking the best overall value from $\{0.01, 0.05, 0.10, 0.15, 0.20, 0.25\}$. For Exp3C this was $\gamma = 0.15$ and for Exp3CR this was $\gamma = 0.20$. Similarly, for FEL we set $\epsilon = 0.10$.

Each algorithm (on each fold) was run up to a budget of $B = 250$, after which almost all the algorithms (with the ex-

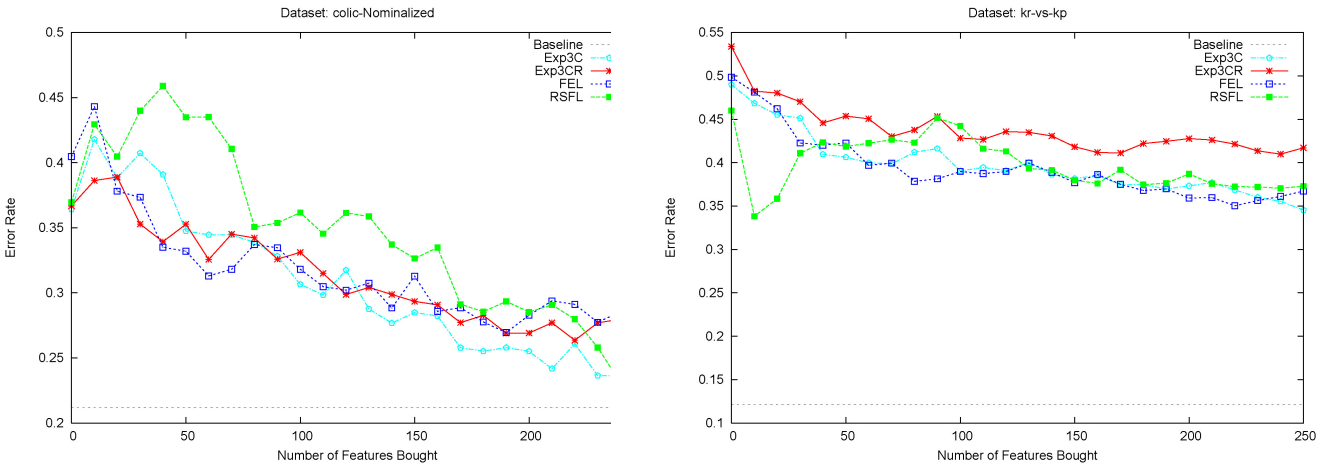

Figure 1. Learning curves for selected data sets.

Table 2. KL Divergence Versus Random Row Selection. The numbers indicate the area above the learning curve using KL divergence and below the learning curve using Random Row Selection. A positive value indicates using KL divergence outperformed Random Row Selection.

Dataset	Exp3C	FEL	RSFL	BR
`cancer`	4.76	0.18	23.15	4.12
`colic`	3.12	6.47	-1.92	-8.26
`kr-vs-kp`	11.38	-3.29	4.67	11.46
`mushroom`	-4.31	0.46	-4.76	-4.55
`vote`	1.56	-3.68	5.50	0.56
`zoo`	0.75	-3.11	3.76	2.19
Mean	2.88	-0.50	5.07	0.92

ception of the `kr-vs-kp` data set) were able to reasonably converge to the baseline (the mean error rate of naïve Bayes with fully-specified attributes). To evaluate the overall behavior of each of the algorithms, we constructed learning curves that reflect the performance of a heuristic as more attributes are purchased with respect to the mean error over the ten folds. Learning curves for two of the UCI data sets can be found in Figure 1. For greater clarity we have only sampled every tenth data point and have omitted learning curves for the random shopper and Biased Robin as their performance was typically worse than RSFL. To evaluate the results more rigorously, we provide summary statistics (for which all data points are considered). The methodology of these statistics is summarized in the following section.

4.1. Summary Statistics

Ultimately, the goal in budgeted learning is to reduce the number of attributes one must purchase in order to effectively learn. Thus to summarize and compare learning curves, we use summary statistics. For the first such statistic, we define the *target budget* for a given data set as the minimum budget needed by an algorithm to be competitive with our random shopper. Let the *target mean* be the mean of the error rates of the final 20% of the total budget achieved by the random shopper. For a given algorithm A and trial t (representing the Bayesian model after t purchases), we compute the mean error rate of the last 5% of purchases for A. Then the target budget is the smallest t for which the target mean is achieved by A. We use a window size of 5% to reduce the influence of outliers as the learning curves can have high variance early on. If an algorithm fails to achieve the target mean, its target budget is simply the entire budget B.

We also report the *data utilization ratio*, which is defined as an algorithm's target budget divided by the target budget of the random shopper. Thus, a lower data utilization ratio reflects that the algorithm was able to make more useful purchases overall while excluding large changes in performance as the budget is exhausted. This metric is similar to one used in the context of active learning [1, 13, 4]. To summarize over all data sets, we also report the mean data utilization ratio for each heuristic.

The target budget and data utilization ratio are good measures for how fast a heuristic is able to form an effective learning model. However, it doesn't quantify how an algorithm may performs overall. For this purpose we define another statistic. For a given algorithm A, we measure the area above a learning curve for A but below the learning curve for the random shopper. As we are computing mean error, for this statistic, a *larger* area indicates a better performance with respect to the random shopper.

4.2. Summary of Results

The results of the statistics outlined in the previous subsection can be found Tables 1(a) and 1(b). In addition, Table 2 summarizes the improvements made to each algorithm by using KL divergence as a row selection criterion versus random row selection as detailed in Section 3.3.

We first note that using KL divergence substantially improved the performance Exp3C and RSFL, which saw an improvement in classification error of about 1-2% on average for each round. Note that since Exp3CR defines a distribution across both columns and rows, it already gives a criterion for choosing a specific attribute-instance pair. Thus, using KL divergence on Exp3CR is unnecessary.

Figure 1 shows the learning curves for the data sets `colic` and `kr-vs-kp`. The curves for all data sets (not shown) indicate that all algorithms tended to converge to the baseline with the limited budget of $B = 250$. For `cancer` and `colic`, our algorithms showed a substantial improvement over RSFL. However, RSFL showed a generally better performance on `kr-vs-kp` and `mushroom`, though the difference was far less significant. The performances for the final two data sets were more competitive.

Regarding target budget (Table 1(a)), all algorithms were competitive over all data sets, with Exp3CR having a slight advantage with respect to data utilization ratio. However, according to the learning curve for the data set `vote`, it is a reasonably easy set to learn on. Thus, the numbers will tend to be a bit skewed. If we were to exclude this set, then our algorithms show a clear advantage over RSFL and BR.

This conclusion is reinforced when we consider the areas below the learning curve of the random shopper (Table 1(b)). On average, across all the data sets, BR was no better than our random shopper. Our algorithms were more than three times as effective than RSFL in outperforming the random shopper. On balance, it is reasonable to conclude that our algorithms performed better than the naïve BR and RSFL. This is particularly the case in the context of smaller budgets and more "difficult" data sets.

5. Conclusions & Future Work

We presented three new algorithms for the budgeted learning problem, each showing improvement over the state of the art. We also presented a means to improve row (example) selection in budgeted learning algorithms by selecting the row with most uncertainty in the current model.

There are several directions one can head in future work. In particular, since there are no learning-theoretic results for the general budgeted learning problem, an area of interest would be deriving learning-theoretic results (e.g. PAC-style results). Another area would include extending our algorithms to the problem of budgeted learning of bounded active classifiers (BACs) [10, 9].

Acknowledgments

We thank Peter Auer, Russell Greiner, Aloak Kapoor, Omid Madani, Robert Schapire and several anonymous reviewers for useful discussions and feedback. Chris Bourke was supported by NSF grant CCF-0430991. Julie Sunderman was supported by an Undergraduate Creative Activities and Research Experiences (UCARE) grant.

References

[1] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In *ICML '98*, pages 1–10, 1998.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The non-stocastic multi-armed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[3] C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *ICML'00*, pages 111–118, 2000.

[4] M. Culver, D. Kun, and S. Scott. Active learning to maximize area under the ROC curve. In *ICDM'06*, pages 149–158, 2006.

[5] C. B. D.J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases, 1998.

[6] R. Greiner, A. J. Grove, and D. Roth. Learning cost-sensitive active classifiers. *Artificial Intelligence*, 139(2):137–174, 2002.

[7] S. Guha and K. Munagala. Approximation algorithms for budgeted learning problems. In *STOC'07*, pages 104–113, 2007.

[8] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71:291–307, 2005.

[9] A. Kapoor and R. Greiner. Budgeted learning of bounded active classifiers. In *Proceedings of the ACM SIGKDD Workshop on Utility-Based Data Mining*, 2005.

[10] A. Kapoor and R. Greiner. Learning and classifying under hard budgets. In *ECML'05*, 2005.

[11] D. J. Lizotte, O. Madani, and R. Greiner. Budgeted learning of naïve-bayes classifiers. In *UAI'03*, pages 378–385, 2003.

[12] O. Madani, D. J. Lizotte, and R. Greiner. Active model selection. In *UAI'04*, 2004.

[13] P. Melville and R. J. Mooney. Diverse ensembles for active learning. In *ICML'04*, pages 584–591, 2004.

[14] H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 55:527–535, 1952.

[15] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *ICML'00*, pages 839–846, 2000.

[16] P. A. Sriharsha Veeramachaneni, Emanuele Olivetti. Active sampling for detecting irrelevant features. In *ICML'06*, pages 961–968, 2006.

[17] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2(Nov):45–66, 2001.

[18] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[19] Z. Zheng and B. Padmanabhan. On active learning for data aquisition. In *ICDM'02*, pages 562–570, 2002.