A Sequential Sampling Algorithm for a General Class of Utility Criteria

Tobias Scheffer and Stefan Wrobel
University of Magdeburg, FIN/IWS
P.O. Box 4120
39016 Magdeburg, Germany
scheffer, wrobel@iws.cs.uni-magdeburg.de

ABSTRACT

Many discovery problems, *e.g.,* subgroup or association rule discovery, can naturally be cast as n-best hypothesis problems where the goal is to find the n hypotheses from a given hypothesis space that score best according to a given utility function. We present a sampling algorithm that solves this problem by issuing a small number of database queries while guaranteeing precise bounds on confidence and quality of solutions. Known sampling algorithms assume that the utility be the average (over the examples) of some function, which is not the case for many frequently used utility functions. We show that our algorithm works for all utilities that can be estimated with bounded error. We provide such error bounds and resulting worst-case sample bounds for some of the most frequently used utilities, and prove that there is no sampling algorithm for another popular class of utility functions. The algorithm is sequential in the sense that it starts to return (or discard) hypotheses that already seem to be particularly good (or bad) after a few examples. Thus, the algorithm is often even faster than its worst-case bounds.

1. INTRODUCTION

Even with discovery algorithms optimized for very large data sets, for many application problems it is infeasible to process all of the given data. In this case, an obvious strategy is to use only a randomly drawn *sample* of the data. Clearly, if parts of the data are not looked at, it is impossible, in general, to guarantee that the results produced by the discovery algorithm will be identical to the results returned on the complete dataset. If the use of sampling is to be more than a practitioner's "hack", sampling must be combined with discovery algorithms in a fashion that allows us to give the user *guarantees* about how far the obtained results differ from the optimal (non-sampling based) results. The goal of a sampling discovery algorithm then is to guarantee this quality using the minimum amount of examples.

Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD), 2000.

Existing research has concentrated primarily on discovery problems where the goal is to select from a space of possible hypotheses H one of the elements with maximal value of an *instance-averaging* utility function f, or all elements with an f-value above a user-given threshold (*e.g.,* all association rules with sufficient support) [5, 8]. With instance-averaging utility functions, the quality of a hypothesis h is the average across all instances in a dataset D of an instance utility function f_{inst}.

Many discovery problems, however, cannot easily be cast in this framework. Firstly, it is often more natural for a user to ask for the n best solutions instead of the single best or all hypotheses above a threshold (see *e.g.,* [20]). Secondly, many popular utility measures cannot be expressed as an averaging utility function. This is the case, *e.g.,* for all functions that combine coverage and distributional properties of a hypothesis, as popular in subgroup discovery. The task of subgroup discovery [12] is to find maximally general subsets of database transactions within which the distribution of a focused feature differs maximally from the default probability of that feature in the whole database. As an example, consider the problem of finding groups of customers who are particularly likely (or unlikely) to buy a certain product.

In this paper, we present a general sampling algorithm for the n-best hypotheses problem that works for any utility functions that can be estimated with bounded error. To this end, in Section 2, we first define the n-best hypotheses problem more precisely and identify appropriate quality guarantees. Section 3 then presents the generic algorithm. Our algorithm is a *sequential* sampling algorithm [19], in the sense that it does not wait for a fixed number of examples that can be guaranteed to suffice even in the worst case before starting the analysis. It starts to return (or discard) hypotheses that already seem to be particularly good (or bad) after a few examples. Thus, the algorithm is often faster than its worst-case bounds. In Section 4, we prove that many of the popular utility functions that have been used in KDD indeed can be estimated with bounded error, giving detailed bounds. For one popular class of functions that cannot be used by our algorithm, we prove that there cannot be a sampling algorithm at all. Our results thus also give an indication as to which of the large numbers of popular utility functions are preferable with respect to sampling. In Section 5, we evaluate our results and discuss their relation to previous work.

2. APPROXIMATING N-BEST HYPOTHESES PROBLEMS

In many cases, it is more natural for a user to ask for the n best solutions instead of the single best or all hypotheses above a threshold. Such n-best hypotheses problems can be stated more precisely as follows (adapted from [20], where this formulation is used for subgroup discovery): Let D be a database of instances, H a set of possible hypotheses, f a quality or utility function on H mapping a hypothesis and a database to a nonnegative number, and n, $1 \leq n \leq |H|$ an integer, the number of desired solutions. The *n-best hypotheses problem* is to find a set $G \subseteq H$ of size n such that there is no $h' \in H$: $h' \notin G$ and $f(h', D) > f_{min}$, where $f_{min} := min_{h \in G} f(h, D)$.

Whenever we use sampling, the above optimality property cannot be guaranteed, so we must find appropriate alternative guarantees. Since for n-best problems, the exact quality and rank of hypotheses is often not central to the user, it is sufficient to guarantee that G indeed "approximately" contains the n best hypotheses. We can operationalize this by guaranteeing that there will never be a non-returned hypothesis that is "significantly" better than the worst hypothesis in our solution. More precisely, we will use the following problem formulated along the lines of PAC (probably approximately correct) learning:

Definition 1 (Approximate n-best hypotheses). *Let D, H, f and n as in the preceding definition. Then let δ, $0 < \delta \leq 1$, be a user-specified confidence, and $\varepsilon \in \mathbb{R}^+$ a user-specified maximal error. The* approximate n-best hypotheses problem *is to find a set $G \subseteq H$ of size n such that, with confidence $1 - \delta$, there is no $h' \in H$: $h' \notin G$ and $f(h', D) > f_{min} + \varepsilon$, where $f_{min} := min_{h \in G} f(h, D)$.*

In other words, we want to find a set of n hypotheses such that, with high confidence, no other hypothesis outperforms any one of them by more than ε, where f is an arbitrary performance measure. In order to design an algorithm for this problem, we need to make certain assumptions about the utility function f. Ideally, an algorithm should be capable of working (at least) with the kinds of utility functions that have already proven themselves useful in practical applications. If the problem is to *classify* database items (*i.e.,* to find a total function mapping database items to class labels), *accuracy* is often used as utility criterion. For the discovery of *association rules*, by contrast, one usually relies on *generality* as primary utility criterion [1]. Finally, for subgroup discovery, it is commonplace to combine both generality and *distributional unusualness*, resulting in relatively complex evaluation functions (see, *e.g.,* [13] for an overview).

In light of the large range of existing and possible future utility functions, in order to avoid unduly restricting our algorithm, we will not make syntactic assumptions about f. In particular, unlike [5], we will not assume that f is a single probability nor that it is based on averages of instance properties. Instead, we only assume that it is possible to determine a function Δ for a particular f that bounds the probability of errors when computing f based on a sample, and vanishes with increasing sample sizes. As we will show in Section 4 below, finding such Δ is relatively straightforward for classification accuracy, and is also possible for all but one of the popular utility functions from association rule and subgroup discovery. More precisely, we define an error probability bound function Δ for f as follows.

Definition 2 (Error probability bound). *Let f be a utility function, let $h_1 \in H$ and $h_2 \in H$ be two hypotheses. Let $f_1 := f(h_1, D)$ denote the true utility of h_1 on the entire dataset, $\hat{f}_1 := f(h_1, S)$ its estimated utility computed based on a sample $S \subseteq D$ of size m ($f_2, \hat{f}_2$ defined analogously). Then $\Delta : \mathbb{N} \times \mathbb{R} \times \mathbb{R} \to [0, 1]$ is an* error probability bound *for f iff for any $\varepsilon, \hat{\varepsilon}$*

$$Pr_S[\hat{f}_1 - \hat{f}_2 > \hat{\varepsilon} | f_1 - f_2 \leq \varepsilon] \leq \Delta(m, \varepsilon, \hat{\varepsilon}). \quad (1)$$

Equation 1 says that Δ bounds the probability of drawing a sample S (when drawing m transactions independently and identically distibuted from D), such that the empirical difference between two utility values appears overly large. We will refer to $\hat{f}(h, S)$ as the *measured*, or *empirical* utility. If, in addition, for any $\delta, 0 < \delta \leq 1$ and any ε there is a number m such that $\Delta(m, \varepsilon, 0) \leq \delta$ we say that Δ *vanishes.* Note that it can be meaningful for ε to be negative; in this case, Δ bounds the chance that h_1 appears better on the sample ($\hat{f}_1 - \hat{f}_2$ positive) although h_2 really is better ($f_1 - f_2$ negative).

3. SAMPLING ALGORITHM

The general approach to designing a sampling algorithm is to use an appropriate error probability bound to determine the required number of examples for a desired level of confidence and accuracy. When estimating a single probability, Chernoff bounds that are used in PAC theory [10, 18] and many other areas of statistics and computer science can be used to determine appropriate sample bounds [17]. When such algorithms are implemented, the Chernoff bounds can be replaced by tighter normal or Student's t-distribution tables.

Unfortunately, the straightforward extension of such approaches to selection or comparison problems like the n-best hypotheses problem leads to unreasonably large bounds: to avoid errors in the worst case, we have to take very large samples to recognize small differences in utility, even if the actual differences between hypotheses to be compared are very large. This problem is addressed by *sequential* sampling methods [4, 19] (that have also been referred to as *adaptive* sampling methods [5]). The idea of sequential sampling is that when a difference between two frequencies is very large after only a few examples, then we can conclude that one of the probabilities is greater than the other with high confidence; we need not wait for the sample size specified by the Chernoff bound, which we have to when the frequencies are similar. Sequential sampling methods have been reported to reduce the required sample size by several orders of magnitude (*e.g.,* [7]).

In our algorithm (Table 1), we combine sequential sampling with the popular "loop reversal" technique found in many KDD algorithms. Instead of processing hypotheses one after another, and obtaining enough examples for each hypothesis to evaluate it sufficiently precisely, we keep obtaining examples (step 2b) and apply these to all remaining hypotheses simultaneously (step 2c). This strategy allows the algorithm to be easily implemented on top of database systems (assuming they are capable of drawing samples), *and* enables us to reach tighter bounds. After the statistics of each remaining hypothesis have been updated, the algorithm checks whether it has seen enough examples to distinguish all the remaining good hypotheses from the bad ones with sufficient confidence, in which case it can exit (step 2f). Otherwise, in step 2g it checks all remaining hypotheses and (i) outputs those where it can be sufficiently certain that the number of better hypotheses is no larger than the number of hypotheses still to be found (so they can all become solutions), or (ii) discards those hypotheses where it can be sufficiently certain that the number of better other hypotheses is at least the number of hypotheses still to be found (so it can be sure the current hypothesis does not need to be in the solutions). Indeed it can be shown that this strategy leads to a total error probability less than δ as required.

THEOREM 1. *The algorithm will output a group G of exactly n hypotheses such that, with confidence $1-\delta$, no other hypothesis in H has a utility which is more than ε higher than the utility of any hypothesis that has been returned:*

$$Pr[\exists h \in H \setminus G : f(h) > f_{min} + \varepsilon] \leq \delta \qquad (2)$$

where $f_{min} = \min_{h' \in G}\{f(h')\}$; assuming that $|H| \geq n$.

The idea of the proof is that we have to sum up the probabilities that either one of the n best hypotheses is discarded or any significantly worse hypothesis is returned over all steps i. This sum must be no more than δ. Due to lack of space, we leave the proof for the full paper.

Table 1: Sequential sampling algorithm for the n-best hypotheses problem

Input: *num* (number of desired hypotheses), ε and δ (approximation and confidence parameters). **Output:** *num* approximately best hypotheses (with confidence $1-\delta$).

1. **Let** $n = num$ (n counts the number of hypotheses that we still need to find) and **Let** $H_1 = H$ (the set of hypotheses that have, so far, neither been discarded nor accepted). **Let** $Q_1 = \emptyset$ (no sample drawn yet).
2. **For** $i = 1 \ldots \infty$
   (a) **Let** $H_{i+1} = H_i$.
   (b) Query a random item of the database q_i. **Let** $Q_i = Q_{i-1} \cup \{q_i\}$.
   (c) Update the empirical utility $\hat{f}$ of the hypotheses in H_i.
   (d) **Let** h_n be the hypothesis in H_i that achieves the nth highest empirical utility $\hat{f}$.
   (e) **Let** h_{n+1} be the hypothesis in H_i that achieves the $n+1$st highest empirical utility $\hat{f}$.
   (f) **If** $\Delta(i, -\varepsilon, 0) \leq \frac{2\delta}{3|H_i|^2}$ **Then Exit** (the for loop).
   (g) **For** $j = 1 \ldots |H_i|$
      i. **If** $\hat{f}(h_j, Q_i) > \hat{f}(h_{n+1}, Q_i)$ (h_j appears good) **and** $n > 0$ **and** $\Delta\left(i, -\varepsilon, \hat{f}(h_j, Q_i) - \hat{f}(h_{n+1}, Q_i)\right) \leq \frac{4\delta}{|H_i|^2 i^2 \pi^2}$ **Then Output** hypothesis h_j and then **Delete** h_j from H_{i+1} and decrement n.
      ii. **If** $\hat{f}(h_j, Q_i) < \hat{f}(h_n, Q_i)$ (h_j appears poor) **and** $|H_i| > n$ **and** $\Delta\left(i, 0, \hat{f}(h_n, Q_i) - \hat{f}(h_j, Q_i)\right) \leq \frac{4\delta}{|H_i|^2 i^2 \pi^2}$ **Then Delete** h_j from H_{i+1}.
   (h) **If** $n = 0$ **Or** $|H_{i+1}| = n$ **Then Exit** (the For loop).
3. **Output** the n hypotheses from H_i which have the highest empirical utility.

4. INSTANTIATIONS

In order to implement the algorithm for a given interestingness function we have to find a function $\Delta(m, \varepsilon, \hat{\varepsilon})$ that satisfies Equation 1 for that specific f. We will in the following present a list of Δ functions for the most commonly used interestingness functions. Table 2 summarizes our results and presents, for each studied utility function f, the error bound Δ and a corresponding worst-case bound on the required sample size. (Since the database is constant, we abbreviate $f(h, D)$ as $f(h)$.)

4.1 Instance-Averaging Functions

This simplest form of a utility function is the average, over all example queries, of some instance utility function $f_{inst}(h, q_i)$. The utility is then defined as $f(h) = \frac{1}{|D|}\sum_{i=1}^{D} f_{inst}(h, q_i)$ (the average over the whole database) and the estimated utility is $\hat{f}(h, Q_m) = \frac{1}{m}\sum_{i=1}^{m} f_{inst}(h, q_i)$ (average over the example queries). An easy example of an instance-averaging utility is the classifiation accuracy. Besides being potentially useful, this class of utility functions serves as an introducing example of how Δ functions can be derived. We assume that there is a lower bound $lb = \min_{q \in D, h \in H} f_{inst}(h, q_i)$ and an upper bound $ub = \max_{q \in D, h \in H} f_{inst}(h, q)$ for this function (*e.g.,* classification accuracy is bounded between 0 and 1) and we define $\Lambda = \max(f_{inst}(h_1, q) - f_{inst}(h_2, q)) - \min(f_{inst}(h'_1, q') - f_{inst}(h'_2, q'))$ as the range of possible values of measured performance differences.

$\hat{f}(h_1, Q_i) - \hat{f}(h_2, Q_i)$ is a random variable with mean value $f(h_1) - f(h_2)$ and bounded range Λ. We can use the Hoeffding inequality [9] to bound the chance that an arbitrary (bounded) random variable takes a value which is far away from its mean value. When X is a random variable with expectation $E(X)$ and range at most Λ and the sample size is m, then the Hoeffding inequality guarantees that $Pr[X - E(X) > \epsilon] \leq \exp\{-2m\frac{\epsilon^2}{\Lambda^2}\}$. In our situation, this implies Equation 3

$$\begin{aligned} Pr[\hat{f}(h_1) - \hat{f}(h_2) > \hat{\varepsilon} | \bar{f}(h_1) - \bar{f}(h_2) \leq \varepsilon] \\ \leq \exp\left\{-2m\frac{(\hat{\varepsilon}-\varepsilon)^2}{\Lambda^2}\right\}. \end{aligned} \qquad (3)$$

Table 2: Summary of Instantiations

$f(h)$	$\Delta(m,\varepsilon,\hat{\varepsilon})$	w/c bound on m
instance-averaging	$\exp\left\{-2m\frac{(\hat{\varepsilon}-\varepsilon)^2}{\Lambda^2}\right\}$	$\frac{\Lambda^2}{\varepsilon^2}\log\frac{\sqrt{3}\lvert H_i\rvert}{\sqrt{2\delta}}$
$g(p-p_0)$, $g\lvert p-p_0\rvert$, $g\frac{1}{c}\sum_{i=1}^{c}\lvert p_i-p_{0_i}\rvert$	$4\exp\left\{-m\frac{(\hat{\varepsilon}-\varepsilon)^2}{8}\right\}$	$\frac{16}{\varepsilon^2}\log\frac{\sqrt{6}\lvert H_i\rvert}{\sqrt{\delta}}$
$g^2(p-p_0)$, $g^2\lvert p-p_0\rvert$, $g^2\frac{1}{c}\sum_{i=1}^{c}\lvert p_i-p_{0_i}\rvert$, $g^2(p-p_0)^2$, $g^2\frac{1}{c}\sum_{i=1}^{c}(p_i-p_{0_i})^2$	$4\exp\left\{-2m\left(\sqrt{1+\frac{\lvert\hat{\varepsilon}-\varepsilon\rvert}{4}}-1\right)^2\right\}$	$\frac{4}{\left(\sqrt{4+\varepsilon}-2\right)^2}\log\frac{\sqrt{6}\lvert H_i\rvert}{\sqrt{\delta}}$
$\sqrt{g}(p-p_0)$, $\sqrt{g}\lvert p-p_0\rvert$, $\sqrt{g}\frac{1}{c}\sum_{i=1}^{c}\lvert p_i-p_{0_i}\rvert$	$4\exp\left\{-m\frac{(\hat{\varepsilon}-\varepsilon)^4}{128}\right\}$	$\frac{256}{\varepsilon^4}\log\frac{\sqrt{6}\lvert H_i\rvert}{\sqrt{\delta}}$
$\frac{g}{1-g}(p-p_0)^2$, $\frac{g}{1-g}\frac{(p-p_0)^2}{p_0}$, $\frac{g}{1-g}\ldots$	1	∞

We can therefore define Δ as in Equation 4.

$$\Delta(m,\varepsilon,\hat{\varepsilon}) \quad = \quad \exp\left\{-2m\frac{(\hat{\varepsilon}-\varepsilon)^2}{\Lambda^2}\right\}. \qquad (4)$$

The algorithm exits the for loop (at latest) when $\Delta(m,-\varepsilon,0) \leq \frac{2\delta}{3|H_i|^2}$. This is the case with certainty when $m \geq \frac{\Lambda^2}{\varepsilon^2}\log\frac{\sqrt{3}|H_i|}{\sqrt{2\delta}}$ (the proof is left for the full paper). But note that our algorithm will generally terminate much earlier; firstly, because we use the t-distribution (for large m) rather than the Hoeffding bound and, secondly, our sequential sampling approach will terminate much earlier when the n best hypotheses differ considerably from many of the "bad" hypotheses. The worst case occurs only when all hypotheses in the hypothesis space are equally good which makes it much more difficult to identify the n best ones.

4.2 Other Utility Functions

Often the task of data mining problems is to identify sets of transactions that are both frequent (*i.e.*, general) and statistically unusual. We define the generality g as the probability that a transaction lies within the support of a hypothesis (*i.e.*, the hypothesis applies to the transaction). A number of utility functions have been proposed that measure these two properties of hypotheses. We refer the reader to [11] for a discussion on the background of these utility functions.

One class of utility functions weights the generality g of a subgroup and the deviation of the probability p of a certain feature from the default probability p_0 equally [16]. Hence, these functions multiply generality and distributional unusualness of subgroups. Alternatively, we can use the absolute distance $|p-p_0|$ between probability p and default probability p_0. The multi-class version of this function is $g\frac{1}{c}\sum_c |p_i-p_{0_i}|$ where p_{0_i} is the default probability for class i. The appropriate definition of Δ as well as the resulting worst-case sample bounds can be found in Table 2.

Squared terms [20] are introduced to put more emphasis on either the generality or the difference between p and the default probability. The resulting utility functions are variations of $g^2(p-p_0)$.

The Binomial test heuristic [11] is based on elementary considerations. Suppose that the probability p is really equal to p_0 (*i.e.*, the corresponding subgroup is really uninteresting). How likely is it, that the subgroup with generality g displays a frequency of $\hat{p}$ on the sample Q with a greater difference $|\hat{p}-p_0|$? For large $|Q|\times g$, $(\hat{p}-p_0)$ is governed by the normal distribution with mean 0 and variance at most $\frac{1}{2\sqrt{m}}$. The probability density function of the normal distribution is monotonic, and so the criterion $\sqrt{m}(p-p_0)$ (which is $\sqrt{g}(p-p_0)$ times a constant factor) orders the hypotheses according to the probability that they are uninteresting. Several variants of this utility function have been used. See Table 2 for the results.

4.3 Negative Result

Several independent impurity criteria have led to utility functions which are equivalent (up to a constant factor) to $f(h) = \frac{g}{1-g}(p-p_0)^2$; *e.g.*, Gini diversity index, twoing criterion [3], and the chi-square test [16]. The order which this criterion imposes on hypotheses is also equal to the order imposed by the criterion of Inferrule [2]. Unfortunately, this utility function is not bounded and a few examples that have not been included in the sample can impose dramatic changes on the values of this function.

THEOREM 2. *There is no algorithm that satisfies Theorem 1 when $f(h) = \frac{g}{1-g}(p-p_0)^2$.*

The idea of the proof is that $(\hat{f}(h_1,Q)-\hat{f}(h_2,Q))-(f(h_1)-f(h_2))$ is unbounded for any finite m. This implies that, even after an arbitrarily large sample has been observed (that is smaller than the whole database), the utility of a hypothesis with respect to the sample can be arbitrarily far from the true utility. But one may argue that demanding $\hat{f}(h,Q)$ to be within an additive constant ε is overly restricted. However, the picture does not change when we require $\hat{f}(h, Q)$ only to be within a multiplicative constant, since $(\hat{f}(h_1, Q) - \hat{f}(h_2, Q))/(f(h_1) - f(h_2))$ is unbounded for any finite m as well.

5. DISCUSSION

Learning algorithms which require a number of examples that can be guaranteed to suffice for finding a nearly optimal hypothesis even in the worst case have early on been criticized as being impractical. Maron, Moore, & Lee [14, 15] have introduced sequential sampling techniques [4, 19] into the machine learning context by proposing the "Hoeffding Race" algorithm that combines loop-reversal with adaptive Hoeffding bounds. A general scheme for sequential local search has been proposed by Greiner [8]. Sequential sampling can often reduce the required sample sizes in cases by considerable factors [7].

Sampling techniques are particularly needed in the context of knowledge discovery in databases where often much more data are available than can be processed. A non-sequential sampling algorithm for KDD has been presented by Toivonen [17]; a sequential algorithm (that imposes further restrictions on f and possesses an additional parameter) by Domingo *et al.* [5, 6].

So far, all sampling algorithms have been restricted to instance-averaging utility functions (such as error probabilities), and to finding a single approximately best hypothesis. For the subgroup discovery problem, utility functions are used that combine generality and a distributional property of the hypothesis; this cannot be expressed as an instance-averaging function. Also, users might often be interested in the n best hypotheses. We presented an algorithm that works for a wide range of utility functions. For the only widely used function for which our algorithm does not work $(g/(1-g)\dots)$ we proved that there exists no sampling algorithm at all.

By giving worst-case bounds on the sample size (and proving that there is no sampling algorithm for some utility functions) our results give an indication as to which of the many utility functions appear preferable from a sampling point of view.

Acknowledgement

We wish to thank Frank Schulz for carefully proof-reading the paper and giving us helpful comments.

6. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Conference on Management of Data*, pages 207–216, 1993.

[2] R. Uthurusamy an U. Fayyad and S. Spangler. Learning useful rules from inconclusive data. In *Knowledge Discovery in Databases*, pages 141–158, 1991.

[3] L. Breiman, J Friedman, R. Ohlsen, and C. Stone. *Classification and Regression Trees.* Pacific Grove, 1984.

[4] H. Dodge and H. Romig. A method of sampling inspection. *The Bell System Technical Journal*, 8:613–631, 1929.

[5] C. Domingo, R. Gavelda, and O. Watanabe. Practical algorithms for on-line selection. In *Proc. International Conference on Discovery Science*, pages 150–161, 1998.

[6] C. Domingo, R. Gavelda, and O. Watanabe. Adaptive sampling methods for scaling up knowledge discovery algorithms. Technical Report TR-C131, Dept. de LSI, Politecnica de Catalunya, 1999.

[7] R. Greiner and R. Isukapalli. Learning to select useful landmarks. *IEEE Transactions on Systems, Man, and Cybernetics*, Part B:473–449, 1996.

[8] Russell Greiner. PALO: A probabilistic hill-climbing algorithm. *Artificial Intelligence*, 83(1–2), July 1996.

[9] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[10] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, 1994.

[11] W. Klösgen. Problems in knowledge discovery in databases and their treatment in the statistics interpreter explora. *Journal of Intelligent Systems*, 7:649–673, 1992.

[12] W. Klösgen. Assistant for knowledge discovery in data. In P. Hoschka, editor, *Assisting Computer: A New Generation of Support Systems*, 1995.

[13] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In Fayyad et al., editor, *Advances in Knowledge Discovery and Data Mining*, pages 249–271. AAAI, 1996.

[14] O. Maron and A. Moore. Hoeffding races: Accelerating model selection search for classification and function approximating. In *Advances in Neural Information processing Systems*, pages 59–66, 1994.

[15] A. Moore and M. Lee. Efficient algorithms for minimizing cross validation error. In *ICML-94*, pages 190–198, 1994.

[16] G. Piatetski-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248, 1991.

[17] H. Toivonen. Sampling large databases for association rules. In *Proc. VLDB Conference*, 1996.

[18] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1996.

[19] A. Wald. *Sequential Analysis.* Wiley, 1947.

[20] Stefan Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proc. First European Symposion on Principles of Data Mining and Knowledge Discovery (PKDD-97)*, pages 78–87, Berlin, 1997.