Average-Case Analysis of a Nearest Neighbor Algorithm

PAT LANGLEY
Learning Systems Department
Siemens Corporate Research
755 College Road East
Princeton, NJ 08540 USA

WAYNE IBA
AI Research Branch
Mail Stop 269-2
NASA Ames Research Center
Moffett Field, CA 94035 USA

Abstract

In this paper we present an average-case analysis of the nearest neighbor algorithm, a simple induction method that has been studied by many researchers. Our analysis assumes a conjunctive target concept, noise-free Boolean attributes, and a uniform distribution over the instance space. We calculate the probability that the algorithm will encounter a test instance that is distance d from the prototype of the concept, along with the probability that the nearest stored training case is distance e from this test instance. From this we compute the probability of correct classification as a function of the number of observed training cases, the number of relevant attributes, and the number of irrelevant attributes. We also explore the behavioral implications of the analysis by presenting predicted learning curves for artificial domains, and give experimental results on these domains as a check on our reasoning.

1. Nearest Neighbor Algorithms

Most learning methods form some abstraction from experience and store this structure in memory. The field has explored a wide range of such structures, including decision trees (Quinlan, 1986), multilayer networks (Rumelhart & McClelland, 1986), and probabilistic summaries (Fisher, 1987). However, in recent years there has been growing interest in methods that store instances or cases in memory, and that apply this specific knowledge directly to new situations. This approach goes by many names, including *instance-based learning* and *case-based reasoning*, and one can apply it to many different tasks.

The simplest and most widely studied class of techniques, often called *nearest neighbor* algorithms, originated in the field of pattern recognition (Cover & Hart, 1967; Dasarathy, 1991) and applies to classification tasks. In the basic method, learning appears almost trivial – one simply stores each training instance in memory. The power of the method comes from the retrieval process. Given a new test instance, one finds the stored training case that is nearest according to some distance measure, notes the class of the retrieved case, and predicts the new instance will have the same class.

Many variants exist on this basic algorithm. For instance, Stanfill and Waltz (1986) have studied a version that retrieves the k closest instances and bases predictions on a weighted vote, incorporating the distance of each stored instance from the test case; such techniques are often referred to as k-nearest neighbor algorithms. Others (Cover & Hart, 1967; Aha, Kibler, & Albert, 1991) have studied an alternative approach that stores cases in memory only upon making an error, thus reducing memory load and retrieval time with little reduction in accuracy.

We would like to understand the learning behavior of this intriguing class of methods under various conditions. Aha et al. present a PAC analysis of one such algorithm, but our aim is to obtain tighter bounds that we can directly relate to experimental results. To this end, we decided to pursue an average-case analysis along the lines developed by Hirschberg and Pazzani (1991) for logical induction methods and by Langley, Iba, and Thompson (1992) for probabilistic ones. For the sake of tractability, we focused our efforts on the most basic of the instance-based techniques, which stores all training cases and bases its prediction on the single nearest neighbor.

However, the simplicity of this method does not mean it lacks power. Aha et al. (1991) report the results of an experimental study that compared the algorithm (which they called *IB1*) to Quinlan's (1986) more sophisticated C4 algorithm for inducing decision trees. Table 1 contains the results on four natural domains, two of them ("Cleveland" and "Hungarian") involving prediction of heart disease from symptoms, another concerning the diagnosis of primary tumors, and a fourth involving prediction of party affiliations for members of Congress from their voting records. For each domain, Aha et al. trained the algorithms on approximately 80% of the cases and tested them on the remaining instances, averaging over 50 different partitions. On the Cleveland data, the two algorithms' performance was indistinguishable, and IB1's behavior on the tumor and voting records nearly reached C4's level. Although the basic nearest neighbor algorithm fared much worse on the Hungarian data set, simple modifications produce accuracy comparable to that for C4 (Aha, 1990), and its performance on the other domains argues that it deserves closer inspection in any case.

In the remainder of this paper, we report the initial results of our average-case analysis of the simple nearest neighbor method. We begin by presenting the as-

Table 1. Percentage accuracies for a nearest neighbor method (IB1) and a decision-tree algorithm (C4) on four classification domains, taken from Aha et al. (1991). Each column reports both average accuracy of classification and standard error.

DOMAIN	IB1	C4
CLEVELAND	75.7 ± 0.8	75.5 ± 0.7
HUNGARIAN	58.7 ± 1.5	78.2 ± 0.9
PRIMARY TUMOR	34.7 ± 0.8	37.8 ± 0.9
VOTING RECORDS	91.8 ± 0.4	95.5 ± 0.3

sumptions of the analysis, followed by our derivation of the equations for predicting accuracy as a function of three variables. After this, we examine the implications of the analysis for the algorithm's behavior, comparing predicted learning curves for artificial domains of differing difficulty. Finally, we note some limitations of the analysis and suggest directions for future research.

2. An Average-Case Analysis

Recall that the algorithm under study stores all training cases in memory. Upon encountering a test instance, it retrieves the nearest observed case and predicts the same class as that for the stored instance. If a tie occurs, the version we will examine selects one of the nearest cases at random. We would like to compute A_n, the probability of correct classification (i.e., the predictive accuracy) after n training instances as a function of characteristics of the domain.

Our analysis will assume that there exist two classes, C and $\bar{C}$, defined over r relevant Boolean attributes A_j and i irrelevant ones. We will also assume that the probability of occurrence $P(A_j) = 1/2$ for each such attribute A_j, generating a uniform distribution over the instance space, and that the target concept is conjunctive, giving $P(C) = P(A)^r = 1/2^r$. Finally, we will measure the distance between two cases as the number of attributes on which they differ. Because there are $r+i$ attributes, there are exactly $r+i+1$ distinct distances, ranging from zero to $r+i$, that can occur.

We will use I_d to denote an arbitrary test case that is distance d from the prototype P for the positive class. We will also refer to J_e as an arbitrary training case that is distance e from test instance I_d. We will often treat groups of test cases as a class based on their distance d from the prototype; similarly, we will consider groups of training instances based on their distance e from a given class of test cases.

Our strategy considers positive and negative test instances separately. We will use $A(C)_n$ to refer to the probability of correct classification given that the algorithm encounters a positive test case after n training cases, and we will use $A(\bar{C})_n$ for the predictive accuracy given a negative test instance. The overall accuracy of the nearest neighbor algorithm after n training cases is

$$A_n = P(C)A(C)_n + P(\bar{C})A(\bar{C})_n \quad , \tag{1}$$

where $P(C)$ is the probability of a positive test case and $P(\bar{C}) = 1 - P(C)$ is the probability of a negative one.

Let us deal with the accuracy on negative test cases first. We can view this term as a weighted sum of the accuracies for different types of test cases I_d, where d is the distance from the prototype.[1] We must sum over all possible distances from the prototype at which a test case can occur (except for $d = 0$), multiplying the probability of each type d by its accuracy $B(\bar{C})_{d,n}$ according to the possible contents of memory after n training instances:

$$A(\bar{C})_n = \sum_{d=1}^{r+i} \frac{\binom{r+i}{d} - \binom{i}{d}}{2^{r+i} - 2^i} B(\bar{C})_{d,n} \quad , \tag{2}$$

where the accuracy component

$$B(\bar{C})_{d,n} = N(J_{d-i-1})_n + T(J_{d-i,d+i})_n^- + F(J_{d+i+1})_n \tag{3}$$

can be further divided into three terms, which are related to the regions shown in Figure 1. These correspond, respectively, to situations in which the closest stored training case is *near* enough to I_d to ensure correct classification, in which *ties* can occur because the closest case may belong to either class, and in which the closest instance is *far* enough from I_d to ensure correct prediction.

The initial term, $N(J_{d-i-1})_n$, represents the contribution to the accuracy that results from the first of these situations, when the nearest stored training case J_e is closer to I_d than the latter is to the nearest positive instance (i.e., $0 \leq e < d - i$). When this occurs, the algorithm will correctly classify I_d as negative. To see this, consider the innermost region in Figure 1, and note that the algorithm must observe only one training case within the region for this to transpire. The probability that any given training instance will fall within e or fewer steps of the test case is

$$W(J_e) = \frac{1}{2^{r+i}} \sum_{j=0}^{e} \binom{r+i}{j} \quad , \tag{4}$$

giving the probability $1 - W(J_e)$ that this will *not* occur. Thus, the probability that (after n training instances) the algorithm will have seen at least one such case is

$$N(J_e)_n = 1 - [1 - W(J_e)]^n \quad , \tag{5}$$

which gives the first term in the definition of $B(\bar{C})_{d,n}$ in equation (3).

1. When some features are irrelevant, there are multiple positive instances, any of which we can select as the prototype without loss of generality.

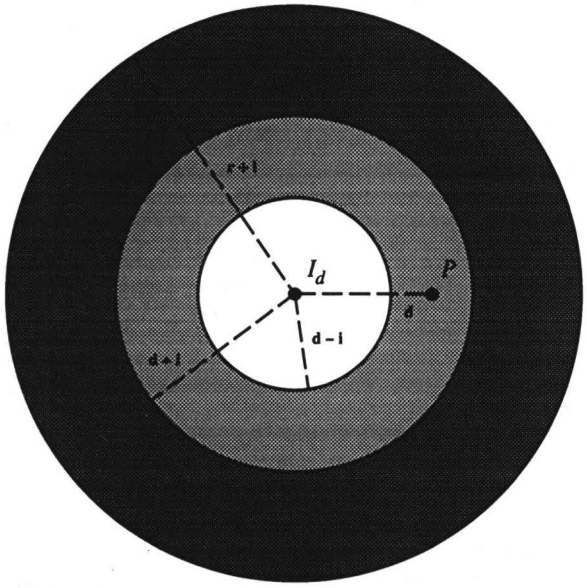

Figure 1. Regions of interest for computing the probability of correctly classifying a negative test case I_d, distance d from prototype P, using the stored training case J_e, which is distance e from I_d. The inner region depicts the probability that $e < d - i$, the outer region shows the probability that $e > d + i$, and the central region indicates the probability that $d - i \leq e \leq d + i$.

The second term from this equation, $T(J_{d-i,d+i})_n^-$, is the contribution to the accuracy when the distance to the nearest stored training instance J_e is between the distances to the nearest possible positive instance and the farthest one (i.e., $d-i \leq e \leq d+i$). This corresponds to the central region in the figure. In this situation, conflicts can occur during the classification process, in that the algorithm may retrieve both positive and negative training cases at distance e from the test case.

Ties are possible in this region because, given i irrelevant attributes, there are 2^i positive instances that can be located i steps or less away from the prototype. For any given test case I_d that is distance d from the prototype, the nearest stored positive case may be i steps away from the prototype in the direction toward I_d, i steps away from the prototype in the direction away from I_d, or somewhere between these two extremes. Negative instances can also occur anywhere within this region, making the entire middle band in the figure open to the possibility of ties.

To handle all possible ties, we must sum over all distances e between $d - i$ and $d + i$, then sum over the possible numbers k of nearest instances (positive or negative) that have been stored at each such distance. In each case, we must multiply the probability $M(J_e)_{n,k}$ of that number occurring by the accuracy $E(J_e)_k^-$ that results from such a tie. We can state this formally as

$$T(J_{a,b})_n^- = \sum_{e=a}^{b} \sum_{k=1}^{n} M(J_e)_{n,k} E(J_e)_k^- \quad . \tag{6}$$

We can further decompose the first term in the product into the probability that exactly k of the n training cases are distance e from the test case and that the remaining $n - k$ are at some greater distance (since the k cases are the nearest ones), giving

$$M(J_e)_{n,k} = \binom{n}{k} \left[\frac{\binom{r+i}{e}}{2^{r+i}} \right]^k [1 - W(J_e)]^{n-k} \tag{7}$$

as the formal expression. To compute the accuracy given a tie among k stored cases, we must sum over the possible numbers j of negative instances, in each case multiplying the subaccuracy by the probability of that occurrence. This gives

$$E(J_e)_k^- = \sum_{j=0}^{k} \frac{j}{k} \binom{k}{j} [1 - V(C)_{e,d}^-]^j [V(C)_{e,d}^-]^{k-j} \quad , \tag{8}$$

where j/k is the expected accuracy when one selects a training case at random from a set that contains j out of k negative instances. The term $V(C)_{e,d}^-$ represents the probability of a positive instance J_e given that the instance is e steps away from negative test case I_d, which is in turn d steps away from the prototype. We can expand this term to

$$V(C)_{e,d}^- = \sum_{k=1}^{min(r,d)} \frac{\binom{r}{k}\binom{i}{d-k}}{\binom{r+i}{d} - \binom{i}{d}} \cdot \frac{\binom{i}{e-k}}{\binom{r+i}{e}} \quad , \tag{9}$$

provided $k \leq e$, $k \leq r$, and $d - k \leq i$, and to zero otherwise. This expression sums over different ways in which distance d can occur, with some steps along relevant attributes and others along irrelevant ones, multiplied by the probability that an instance e steps away from the resulting test case will be positive.

The final term from equation (3), $F(J_{d+i+1})_n$, specifies the contribution to the accuracy when the nearest training instance J_e has distance greater than $d+i$ from test case I_d (i.e., $d+i < e \leq r+i$), and thus falls within the outermost region in Figure 1, which contains only negative instances. As with the $N(J_{d-i-1})_n$ term, the algorithm will correctly classify I_d as negative whenever this occurs. The chance that this situation will arise is

$$F(I_e)_n = \left[\frac{1}{2^{r+i}} \sum_{j=e}^{r+i} \binom{r+i}{j} \right]^n \quad , \tag{10}$$

which is the probability that any given training case will fall at distance e or greater from the test instance, taken to power n to generate the probability that every training instance seen so far satisfies this condition.

Now we can turn to $A(C)_n$, the accuracy on positive test cases after n training instances. The situation here is simpler than for negative test cases, but still nontrivial. The algorithm is guaranteed to classify a positive test case I_d correctly only when the nearest stored training instance is itself the test case (i.e., $e = 0$). Ties can occur anywhere in the range $1 \leq e \leq i$, giving the expression

$$A(C)_n = N(J_0)_n + T(J_{1,i})_n^+ \quad . \tag{11}$$

Because one can treat any positive instance as the prototype, there is no need to sum over different distances d here. Moreover, since no positive instance can be more than i steps away from any other, we can omit the third term of equation (3), $F(J_{i+1})_n$, which is always zero.

The term for handling ties is analogous to equation (6) for the negative situation, but we must revise the definition for $E(J_e)_k^-$ in equation (8) to

$$E(J_e)_k^+ = \sum_{j=0}^{k} \frac{k-j}{k} \binom{k}{j} [1 - V(C)_e^+]^j [V(C)_e^+]^{k-j} \ . \tag{12}$$

Note that here we must use the numerator $k - j$ rather than k, in that we are dealing with positive test cases. Moreover, we must take a different approach to computing $V(C)_e^+$, the probability of a positive instance J_e given that the instance is e steps away from positive test case I_d. In this case, we have

$$V(C)_e^+ = \frac{\binom{i}{e}}{\binom{r+i}{e}} \quad , \tag{13}$$

when $i \geq e$, but zero in other situations. Taken together, the definitions for $A(C)_n$, $A(\bar{C})_n$, and their component terms let us predict the overall accuracy A_n for the nearest neighbor algorithm as a function of the number of training instances n, the number of relevant attributes r, and the number of irrelevant attributes i.

3. Behavioral Implications of the Analysis

Although the equations in the previous sections provide a formal characterization of the nearest neighbor algorithm's behavior, their implications are not obvious. To better understand the effects of domain characteristics, we systematically varied certain domain parameters and examined the predicted results. In addition to computing theoretical predictions, we also collected experimental learning curves that summarized the algorithm's actual behavior. Each datum on these curves reports the classification accuracy averaged over 100 runs on randomly generated training sets, measured over the entire space of uniformly distributed noise-free instances. In each case, we bound the mean accuracy with 95% confidence intervals to show the degree to which our predicted learning curves fit the observed ones. These experimental results provide an important check on our reasoning, and they identified a number of problems during development of the analysis.

Figure 2 shows the effects of the number of relevant attributes in the conjunctive target concept. For this study, we held the number of irrelevant attributes i constant at one, and we varied both the number of training instances and the number of relevant attributes r. As typical with learning curves, the accuracy starts low and gradually improves as the algorithm encounters more training instances. The effects of target complexity also make sense. Increasing the number of relevant features should increase the overall number of negative instances, giving higher accuracy early in the induction process; however, this factor also increases the total number of possible instances, requiring more training cases to reach asymptote and producing a crossover effect. The learning rate seems to degrade gracefully with increasing complexity, and the theoretical and actual learning curves are in close agreement, which lends confidence to the analysis.

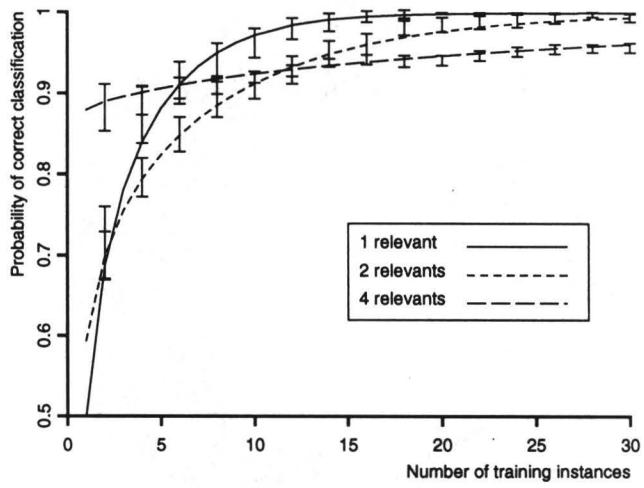

Figure 2. Predictive accuracy of the nearest neighbor algorithm on a conjunctive concept, assuming the presence of one irrelevant attribute, as a function of training instances and the number of relevant attributes. The lines represent theoretical learning curves, whereas the error bars indicate experimental results.

The sensitivity of the nearest neighbor algorithm to irrelevant attributes is more dramatic, as shown in Figure 3. This graph summarizes the results of a similar study of the interaction between the number of training instances n and the number of irrelevant attributes. Here we held the number of relevant attributes constant at two, and we examined three levels of the i parameter. As with the previous study, the degradation in learning rate is graceful, but the effect is somewhat greater. The difference between the two results appears more significant when one realizes that increasing i does not reduce the proportion of positive instances, as does increasing the number of relevant attributes. These observations are consistent with Aha's (1990) reports on the sensitivity of nearest neighbor methods to the number of irrelevant attributes.

We can also compare the behavior of the nearest neighbor algorithm to that of other induction methods for which average-case analyses exist. In particular, Pazzani and Sarrett (1992) have studied the WHOLIST algorithm, which initializes its concept description to the conjunction of features in the first positive training instance, then removes any feature that fails to occur in later positive instances. Similarly, Langley, Iba, and Thompson (1992) have analyzed the behavior of the Bayesian classifier, a simple probabilistic method that stores observed

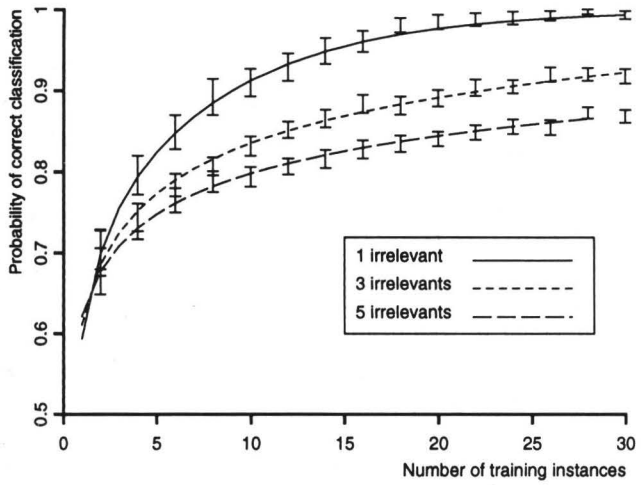

Figure 3. Predictive accuracy of the nearest neighbor algorithm on a conjunctive concept, assuming the presence of two relevant attributes, as a function of training instances and the number of irrelevant attributes. The lines represent theoretical learning curves, whereas the error bars indicate experimental results.

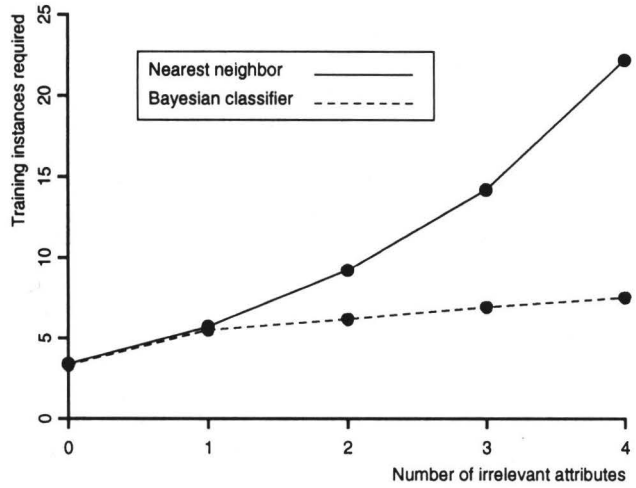

Figure 4. Theoretical number of training instances (interpolated) required to reach 90% accuracy by a nearest neighbor algorithm and a simple Bayesian classifier on a conjunctive concept, assuming the presence of one relevant attribute, as a function of the number of irrelevant attributes.

base rates and conditional probabilities. As Pazzani and Sarrett note, WHOLIST's learning rate is unaffected by the number of relevant attributes, so their algorithm clearly scales up better on this dimension than does the nearest neighbor technique. Comparison to the Bayesian classifier on this factor is more difficult, in that Langley et al.'s study examined equal probabilities for the two classes, whereas the current analysis assumes that they differ.

In some domains, effective learning relies more on the ability to handle many irrelevant features than many relevant ones. In this vein, we have shown analytically that the number of training instances required for WHOLIST to achieve a given level of accuracy increases only with the logarithm of the number of irrelevant attributes. Although we have not yet derived similar analytic relations for the nearest neighbor or probabilistic methods, we can use the existing analyses to estimate ability to scale on this dimension.

Figure 4 graphs the predicted number of training instances needed to achieve 90% accuracy for each algorithm as a function of the number of irrelevant attributes, assuming a target concept involving only one relevant feature and a uniform distribution of instances. The analyses do not provide these quantities directly, but one can interpolate them from the theoretical learning curves. The figure reveals that the Bayesian classifier scales well to increasing numbers of irrelevant attributes, with the dependent measure growing as an approximate linear function of this factor. In contrast, the number of training instances required by the nearest neighbor method grows much faster, although we cannot yet determine the precise superlinear relation. These results are also consistent with Aha's (1990) conclusions about the response of standard instance-based methods to many irrelevant features.

However, the above comparisons are not entirely fair. Neither the WHOLIST algorithm nor the Bayesian classifier are designed to handle disjunctive concepts, which present no obstacles to even the simplest nearest neighbor algorithm. Our focus on conjunctive concepts in the current analysis has obscured this strength. Also, Aha (1990) has developed a variant of the nearest neighbor algorithm that retains statistics on the usefulness of each attribute, and he has shown that this approach fares better in domains with many irrelevant terms. Nevertheless, the ability to make comparisons of the above type is one advantage of careful formal analyses, and they have provided insights about the relative strengths of the different learning algorithms.

4. General Discussion

In this paper we presented an average-case analysis of the most basic nearest neighbor algorithm. Our treatment assumes that the target concept is conjunctive, that instances are free of noise, that attributes are Boolean, and that instances are uniformly distributed. Given information about the number of relevant and irrelevant attributes, our equations let us compute the expected classification accuracy after a given number of training instances.

To explore the implications of the analysis, we plotted the predicted behavior of the algorithm as a function of these three factors, finding graceful degradation as the number of relevants r and irrelevants i increased, but finding a stronger effect for the second. As a check on our analysis, we ran the algorithm on artificial domains with the same characteristics. The predicted behavior closely fit that found in the experiments, but only after correcting several errors in our reasoning that the empirical studies revealed.

These results begin to account for the wide range of performance observed for the algorithm by Aha and others on natural domains. However, a full explanation will require several extensions to the analysis. In particular, we must incorporate the influence of both class and attribute noise, as we have done in earlier analyses (Iba & Langley, 1992; Langley et al., 1992). We must also handle situations in which each attribute follows a separate probability distribution, following the approach taken by Hirschberg and Pazzani (1991).

Even more important, we must extend the framework to handle broader classes of target concepts. Nearest neighbor methods are well suited for *M of N* concepts, in which any M of the N features in the prototype are sufficient for membership in the class. Since distance from the prototype plays a central role in the current analysis, we believe extending it to handle such concepts will be quite feasible. Similarly, because the algorithm stores many training instances in memory, it can easily acquire disjunctive concepts that require multiple prototypes. Again, we hope that simple extensions to the existing framework will handle this situation. We should also generalize the analysis to include k-nearest neighbor methods, following the lead recently provided by Turney (in press).

Another direction for future work would attempt to map the extended analysis onto natural domains in which there already exist experimental results with the method. Given information about the distributions of attributes (which are available in the data), along with estimates of the noise levels and target concepts (which require informed guesses), we can compare learning curves predicted by the theory with those observed in experimental runs. This approach would extend the applicability of our average-case model beyond the artificial domains to which we have limited our tests to date.

In summary, we believe that our initial analysis has provided some useful insights about the behavior of the basic nearest neighbor algorithm. These begin to explain why the algorithm compares favorably with more complex induction methods on some domains but not others, and our results are consistent with intuitions about the algorithm's sensitivity to irrelevant attributes. We also believe the existing theoretical framework can be extended to handle more challenging target concepts and other factors that complicate the learning task, thus providing a solid base on which to carry out further studies of instance-based learning.

Acknowledgements

We would like to thank Stephanie Sage and David Aha for discussions that helped clarify our ideas, David Aha and three reviewers for useful comments, and Nils Nilsson for his support and encouragement. The first author contributed to this work while he was at the Center for the Study of Language and Information, Stanford University, and the Institute for the Study of Learning and Expertise.

References

Aha, D. W. (1990). *A study of instance-based algorithms for supervised learning tasks: Mathematical, empirical, and psychological evaluations*. Doctoral dissertation, Department of Information & Computer Science, University of California, Irvine.

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*, 37–66.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Dasarathy, B. V. (Ed.). (1991). *Nearest neighbor (NN) norms: NN pattern classification techniques*. Los Alamitos, CA: IEEE Computer Society Press.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, *2*, 139–172.

Hirschberg, D. S., & Pazzani, M. J. (1991). *Average-case analysis of a k-CNF learning algorithm* (Technical Report 91–50). Irvine: University of California, Department of Information & Computer Science.

Iba, W., & Langley, P. (1992). Induction of one-level decision trees. *Proceedings of the Ninth International Conference on Machine Learning* (pp. 233–240). Aberdeen, Scotland: Morgan Kaufmann.

Langley, P., Iba, W., & Thompson, K. (1992). An analysis of Bayesian classifiers. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 223–228). San Jose, CA: AAAI Press.

Pazzani, M. J., & Sarrett, W. (1992). A framework for the average case analysis of conjunctive learning algorithms. *Machine Learning*, *9*, 349–372.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106.

Rumelhart, D. E., & McClelland, J. L. (1986). On learning the past tenses of English verbs. In J. L. McClelland & D. E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 2). Cambridge, MA: MIT Press.

Stanfill, C., & Waltz, D. (1986). Toward memory-based reasoning. *Communications of the ACM*, *29*, 1213–1228.

Turney, P. (in press). Voting in instance-based learning: A theoretical analysis. *Journal of Experimental and Theoretical Artificial Intelligence*.