Computing Optimal Attribute Weight Settings for Nearest Neighbor Algorithms

Charles X. Ling* Handong Wang
Department of Computer Science, M.C.
The University of Western Ontario
London, Ontario, Canada N6A 5B7
E-mail: {ling, hwang}@csd.uwo.ca

Abstract

Nearest neighbor learning algorithms, examples of the lazy learning paradigm, rely on a distance function to measure the similarity of testing examples with the stored training examples. Since certain attributes are more discriminative, while others can be less or totally irrelevant, attributes should be weighed differently in the distance function. Most previous studies on weight setting for NN learning algorithms are empirical. In this paper we describe our attempt on deciding theoretically optimal weights that minimize the predictive error for NN algorithms. Assuming a uniform distribution of examples in a 2-d continuous space, we first derive the average predictive error introduced by a linear classification boundary, and then determine the optimal weight setting for any polygonal classification region. Our theoretical results of optimal attribute weights can serve as a baseline or lower bound for comparing other empirical weight setting methods.

Key Words: Nearest neighbor learning algorithms, lazy learning, attribute weight setting, theoretical analyses.

*Contact author. Mailing address for 1996: Charles Ling, Department of Computer Science, University of Hong Kong, Hong Kong.

1 Introduction

Nearest neighbor learning algorithms, which are examples of the lazy learning paradigm, are some of the most popular and studied learning algorithms. They originated in the field of pattern recognition (Cover & Hart, 1967; Dasarathy, 1991), and have been well studied and applied in classification tasks in machine learning (e.g., (Stanfill & Waltz, 1986; Aha, Kibler, & Albert, 1991; Cost & Salzberg, 1993)). Although many variants exists, these algorithms, generally speaking, store all (or selected) training examples, and utilize a similarity (or distance) function to measure the similarity (or distance) of a testing example with the stored training examples when predicting the class of the testing example. That is, they do not induce concept description during the training process; instead, they delay the prediction to the testing period by relying on the training examples. A popular example is the *k-nearest neighbor* (k-NN) algorithm (Dasarathy, 1991). It determines the classification of a testing example by selecting a set of k closest training examples — normally the majority class of the k selected training examples. When $k = 1$, this algorithm is called the 1-nearest neighbor (1-NN) algorithm. Aha et al. (1991) describe a paradigm of lazy learning algorithms that they refer to as instance-based learning (IBL) algorithms, and their IB1 algorithm is an implementation of 1-NN.

The distance function of k-NN algorithms takes the form:

$$dist(x, y) = \sqrt{\sum (x_i - y_i)^2} \tag{1}$$

where x is the testing example, y is the stored training example, and x_i and y_i are the (normalized) values of the i^{th} attribute.[1] However, this distance function (Equation 1) treats every attribute with equal contribution, which for many cases is not ideal. Sometimes an attribute can be completely irrelevant to the classification, so it should not be included in the distance function. Therefore, a more realistic distance function is[2]

$$dist(x, y) = \sqrt{\sum w_i^2 \times (x_i - y_i)^2} \tag{2}$$

where w_i is a constant (to be determined) that weighs the role of the i^{th} attribute in the similarity function.

[1]If the attribute is symbolic, then $x_i - y_i = 0$ if $x_i = y_i$; 1 otherwise.

[2]We put a square on the w_i so the transformation along the axes is proportional to w_i instead of to $\sqrt{w_i}$. In addition, we assume that in this paper the attribute weight vector is the same in the whole instance space.

Setting the weights in the similarity function appropriately can dramatically improve the generalization of the k-NN algorithms, especially in domains with many irrelevant attributes (e.g., (Aha, 1989; Kelly & Davis, 1991; Cardie, 1993; Mohri & Tanaka, 1994; Ling, Parry, & Wang, 1994)). Intuitively, more important attributes should be assigned larger weights than less important attributes, while totally irrelevant attributes should be assigned 0 weight. Many researchers (see Section 2) have proposed algorithms for assigning w_i in Equation 2. However, most of them are empirical; little theoretical work on optimal weight setting is known. Even for a very simple case, where the classification region is an axis-parallel rectangle in a two-dimensional continuous space, it is unknown what the optimal weights w_1 and w_2 are as to minimize predictive error.

This paper reports our effort in deciding theoretically optimal weights that minimize the predictive error for 1-NN algorithms. Our results are still quite limited in scope since they apply only to domains of 2-d continuous space. That is, instances are described by two numeric-valued attributes. Also, it is assumed that the examples are uniformly distributed. We first review previous empirical and theoretical work on weight setting algorithms for k-NN in Section 2. In Section 3 we derive formulas for the average predictive error introduced by linear classification boundaries in 2-d continuous space when examples are uniformly distributed. We then use these results to design a method of deciding theoretically optimal weights for 1-NN in Section 4. Our theoretical results in Sections 3 and 4 have also been empirically verified, indicating that several assumptions and approximations in our derivation are reasonable. We discuss how to extend our work to k-NN ($k > 1$) and higher dimensions in Section 5.

Note that in our theoretical analyses of optimal attribute weights, we assume that the classification boundary (i.e., the underlying target concept) is known. Thus, our optimal results can serve as a useful baseline or lower bound for evaluating empirical attribute weight assigning methods on known concepts. We give some specific examples in Section 5. To apply our method in a more realistic setting where only training examples are given, we need to estimate the classification boundary using piecewise linear classifiers before applying our method. See Section 5 for more details.

2 Review of Previous Weight Setting Methods

Quite a few researchers have investigated empirical work on the weight setting of k-NN algorithms. Several studies showed that one can set attribute weights using another popular learning algorithm — the decision tree learning algorithms ID3 (Quinlan, 1986) or C4.5 (Quinlan, 1993), both use information gain (ratio) to select nodes in decision trees. Kibler and Aha (1987) first presented a simple approach of combining C4.5 and k-NN. Their method uses the presence and absence of attributes in the decision tree built by C4.5 on the same set of training examples to determine the weights (1 and 0 respectively) in the similarity function (Equation 2). The underlying assumption of their method is that if the attributes are not used in the decision tree, they are regarded as irrelevant. Cardie (1993) independently rediscovered the same method. Daelemans, Gillis, Durieux, and van den Bosch (1993) and Wettschereck and Dietterich (1995) presented an approach of assigning continuous weights of all attributes in k-NN algorithms simply by their information gain values. Ling et al. (1994) presented several weight-setting strategies that consider positions of attributes in the decision tree, and showed empirically that these methods improve predictive accuracy dramatically on artificially generated data sets and some real-world data sets.

Other empirical methods on attribute weight setting have been published. Stanfill and Waltz (1986) used statistical information from the stored data to compute weights and applied the method to several tasks including English spelling-to-speech conversion. Aha (1989) changed weights in accordance with success or failure of classification of the training examples. A few methods on assigning binary weights (which is also called feature selection, a special case of numerical weight setting) have also been recently proposed (Aha & Bankert, 1994; Caruana & Freitag, 1994; Skalak, 1994; Moore & Lee, 1994). Wettschereck and Aha (1995) reviewed several empirical weight setting methods and explored their comparative empirical behavior.

A few theoretical investigations have focussed on the calculation of predictive error rate of k-NN algorithms with equal attribute weights in the distance function (thus the work does not deal with weight setting). Cover and Hart (1967) and Cover (1968) gave analyses on the asymptotic behaviour of k-NN algorithms and compared the predictive error rate of 1-NN algorithms with the Bayes decision rule. More recently, Albert and Aha (1991) presented PAC-learning analyses for 1-NN algorithms for both symbolic and numeric-domain tasks. Langley and Iba (1993) and Okamoto and Satoh (1995) presented an average-case analysis on 1-NN predictive accuracy.

The only theoretical work that we are aware of on optimal weight setting for k-NN algorithms is by Mohri and Tanaka (1994) and Satoh and Okamoto (1994). Mohri and Tanaka (1994) proposed a statistical technique on calculating attribute weights, and showed that such weights are optimal in the sense that they maximize the ratio of variance between classes to variance of all cases. However, it is not obvious that this criterion is the same as the optimality of predictive accuracy in 1-NN algorithms. Satoh and Okamoto (1994)'s work assumes that there is *target* weight vector (known to a teacher but unknown to the learner) that classifies pairs of training examples to a qualitative distance, *FAR* and *NEAR*. They showed that if this qualitative distance information is available to the learner, there is a polynomial PAC algorithm that learns (or reconstructs) the target weight vector. However, their work does not aim at choosing weight vectors that minimize the predictive error in classification. Further, the basic assumption of Satoh and Okamoto's work is different from ours; since we presume that the optimal weights for the minimal predictive error is unknown (to the teacher or learner), the teacher cannot provide the qualitative distance information.

3 1-NN's Predictive Error with a Dividing Line

In this section, we study a very simple problem: what is the predictive error of 1-NN with various attribute weights[3] introduced by an axis-parallel linear classification boundary in a 2-d continuous space? Figure 1 demonstrates such a case (i.e., a vertical line $x = 2$ divides a 10×10 space into two; left side is negative, right side is positive). Clearly, one attribute (x_2) is irrelevant; therefore, w_2 should be set to zero, or the ratio $\theta = w_1/w_2$ infinitely large. What exactly are the theoretical values of the predictive error for different θ values?

Before we derive formulas of the predictive accuracy with various θ, we can first see such an effect graphically. Figure 1 shows the regions of errors of 1-NN with various θ values. The error region of each figure is plotted with the misclassified testing examples

[3]Since what really matters are the weight ratios, rather than the weights themselves, we will use "weight ratio" and "weight" interchangeably in the rest of the paper.

out of a total of 40,000 testing examples after training with 500 examples, both uniformly distributed in the space. The predictive error is simply the ratio of the misclassified area over the total area. We can see clearly that the smaller the ratio $\theta = w_1/w_2$, the more the irrelevant attribute x_2 is taken into consideration, and the larger the predictive error. In the extreme case when $\theta = 0$, x_1 (the deciding attribute) is completely ignored, and the classification of testing examples is determined solely by the irrelevant attribute, x_2. We can in fact calculate the predictive error in this case: For any testing example, it has an 80% of chance of falling into the positive region, a 20% of chance of falling into the negative region. However, its class is determined only by the vertical distance from the closest training example. Since the training examples are uniformly distributed in the space, there is an 80% of chance that the vertically closest training example is a positive example, and a 20% of chance that it is a negative one. Therefore, for any positive example, there is a 20% of chance of being misclassified as negative; and for any negative example, there is an 80% of chance of being misclassified as positive. The overall predicative error is thus $80\% \times 20\% + 20\% \times 80\% = 32\%$. Our experiment with $\theta = 10^{-5}$ yields a similar error (30.6%). see Figure 1.

This graphical demonstration also gives us an intuitive understanding of the optimal weight ratio for an axis-parallel rectangular classification region. For the two vertical edges, as in Figure 1, the weight ratio θ should be set as large as possible to reduce predictive errors introduced along those edges. However, the two horizontal edges can be regarded just as the vertical edges with the weight ratio $1/\theta$ and, therefore, they require θ to be as small as possible to reduce the predictive errors introduced there. Therefore, there should be an optimal weight ratio that compromises the two sides so that the predictive error would be minimal.

3.1 1-NN with Equal Weights

We first derive the predictive error rate of 1-NN with an equal weight ratio (i.e., $\theta = 1$) in a 2-d continuous space. Assume that a vertical line divides the instance space into two equal halves with S as the area for both the left half and the right half. See Figure 2(a). For a (positive) testing example O falling into the right half, we can calculate the probability of O being classified as a negative example (i.e., predictive error). For this to happen, there will be a constant-distance boundary centered at O (a circle here since $w_1 = w_2$) that goes over the vertical line, such that the shaded area (A_1) includes at least one (negative) training example while the area A_2 does not include any (positive) training examples. In this case, the positive example O is misclassified as negative. Assume λ is the density of the uniformly distributed training examples (per unit area), p_1 is the probability of area A_1 having at least one training example, and p_2 is the probability of area A_2 having no training example, then we have (assuming S is very large):

$$p_1 = 1 - (1 - \frac{A_1}{S})^{S\lambda} \simeq 1 - e^{-\lambda A_1} \tag{3}$$

$$p_2 = (1 - \frac{A_2}{S})^{S\lambda} \simeq e^{-\lambda A_2} \tag{4}$$

Therefore, the error rate at the point O is an integration of $p_1 p_2$ over d_1 (see Figure 2(a)):

$$E_O = \int_0^\infty p_1 p_2 \, dd_1$$

and the error introduced by a unit-length boundary is an integration of E_O over d_2:[4]

$$E = 2 \times \int_0^\infty E_O \, dd_2. \tag{5}$$

Equations 3, 4 and 5 are suitable for calculating error rates in other situations with corresponding A_1 and A_2. In the case of straight line boundary and $w_1 = w_2$, the areas of A_1 and A_2 are:

$$A_1 = \cos^{-1}(\frac{d_2}{d_1 + d_2})(d_1 + d_2)^2 - d_2\sqrt{(d_1 + d_2)^2 - d_2^2}$$

$$A_2 = \pi(d_1 + d_2)^2 - A_1$$

Unfortunately, there seems to be no closed formula[5] for the double integrations of the predictive error E, so we have to calculate E numerically. As an example, the numerical estimation of E is 1.03 % (see Appendix A for more details) when we assume the height of the vertical line[6] is 10, and the density λ of the training examples is 10. To see if

[4]We multiply the result by 2 to include the misclassified negative testing examples.

[5]We have tried Maple V (Char, Geddes, Gonnet, Leong, Monagan, & S.M.Watt, 1992), and it could not find a closed formula.

[6]When the circle is too close to the top or bottom boundaries, it goes over these boundaries, so the calculation of A_1 and A_2 would be different. To reduce error in E in the top and bottom boundaries, the height is chosen to be large.

several assumptions (such as in Equations 3 and 4) and approximations (such as in the numerical estimation in Appendix A) in calculating E are reasonable, we performed 10 runs of experiments using 1-NN with $w_1 = w_2$, $\lambda = 10$, and 10,000 testing examples. The average testing error of 10 runs was 0.95% with the standard error 0.058%. Clearly, the empirical and theoretical results match closely, indicating that these assumptions and approximations in deriving and estimating E are reasonable.

3.2 1-NN with Unequal Weights

Assume $\theta = w_1/w_2 \neq 1$, then the constant-distance line formed by the distance equation becomes an ellipse. The vertical radius b of the ellipse is therefore simply equal to $\theta(d_1 + d_2)$. See Figure 2(b). We can use the same method to calculate E, except that E contains a variable θ, and that the areas A_1 and A_2 are defined for ellipses in general. They can be calculated quite easily with Maple V (Char et al., 1992):

$$A_1 = \frac{2b}{a} \int_{d_2}^{d_1+d_2} \sqrt{a^2 - x^2} dx$$

$$A_2 = \pi ab - A_1 = \pi(d_1 + d_2)^2 \theta$$

Thus, by using the above formulas for A_1 and A_2, and that $a = d_1 + d_2$, $b = \theta(d_1 + d_2)$, we obtain the error rate E (Equation 5) when $\theta = w_1/w_2 \neq 1$. Clearly, it is a function of θ, and we call it $E(\theta)$.

To test our theoretical result $E(\theta)$ (which again is derived with several approximations and assumptions), we conducted an experiment with 1-NN with $\theta = 2$ for 10 runs. The average error rate of the 10 runs is 0.511%, while the theoretical error rate $E(\theta)$ is 0.514%. Again, the empirical and theoretical results match closely.

4 Optimal Weight Setting for 1-NN

We have obtained the error rate $E(\theta)$ for a unit length of a dividing line parallel to an axis. We can now calculate the optimal θ for 1-NN algorithms if the classification region is an axis-parallel rectangle. We then extend our method to any polygonal classification regions.

4.1 Axis-parallel Rectangular Classification Regions

Clearly, if the width and length of the rectangle are w and l respectively, then the total predictive error introduced by the four sides of the rectangle is[7]

$$E_{rec}(\theta) = 2lE(\theta) + 2wE(1/\theta). \tag{6}$$

One would choose θ such that $E_{rec}(\theta)$ can be minimized for a particular w and l. This could be done by solving the equation $E'_{rec}(\theta) = 0$; i.e., the first derivative with respect to θ is 0.

However, since $E(\theta)$ is a complicated double integration, we could attain only a numeric solution for θ from $E'_{rec}(\theta) = 0$ at best. Alternatively, we use a "short-cut" table look-up method: we compute values of $E(\theta)$ for $\theta = 0.05, 0.1, 0.15, 0.2, ..., 1.0$, and then the corresponding values of $E(1/\theta)$ (i.e., $E(20)$, $E(10)$ and so on). Table 1 lists the values for $E(\theta)$ at multiples of 0.1, and Figure 3 shows such a curve.[8] For a particular w and l, we can just check the table to see which value of θ minimizes $lE(\theta) + wE(1/\theta)$.

Table 1: The predictive error $E(\theta)$ for various θ values.

θ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$E(\theta)$	64.2	32.1	21.4	16.1	12.8	10.7	9.17	8.03	7.13	6.42
θ	1/0.1	1/0.2	1/0.3	1/0.4	1/0.5	1/0.6	1/0.7	1/0.8	1/0.9	1.0
$E(\theta)$	0.642	1.28	1.93	2.57	2.89	3.85	4.49	5.14	5.78	6.42

As an example, we tried two rectangles with sizes 8×2 and 8×6 respectively (centered in a 10×10 two-dimensional space). The results of $lE(\theta) + wE(1/\theta)$ using Table 1 are plotted in Figure 4(a) and (b). The minimal testing error rates are obtained when θ is equal to 0.5 and 0.9 respectively. An experiment was carried out to determine empirically the actual θ value that minimizes the predictive error. For each rectangle, the 1-NN

[7] As discussed in Section 3, a weight ratio of θ for a vertical boundary becomes a weight ratio of $1/\theta$ for the horizontal boundary. Also, there is an overlap in error regions at the corners introduced by the vertical and horizontal edges. Therefore, Equation 6 is an approximation.

[8] Note that values in Table 1 are all *relative* for the purpose of finding the optimal θ that minimizes the total predictive error. Therefore, they can be scaled without affecting the results on the optimal weight selection.

algorithm was run with randomly generated training and testing examples using different θ (with an interval of 0.05). The optimal θ values obtained empirically are 0.4 and 0.9 respectively, matching with the theoretical values closely. The predictive errors of the experiment are also plotted in Figure 4.

It is clear that the predictive error curves of experiments and the theoretical derivation are similar, and the actual optimal θ values are close to the theoretical optimal θ values as well. We can also see that when one side of the rectangle is much larger (or smaller) than the other side, the curve has a more distinctive minimal point; that is, the choice of θ affects the predictive error more dramatically. When the rectangle is close to a square, the optimal θ is close to 1, and the predictive errors are not as sensitive to the θ values close to the optimal value.[9]

Using the same method, we can calculate the optimal θ for rectangles with a length of 8 and various widths from 1 to 8. Again, we can first obtain the theoretically optimal values θ for each width using Table 1. We then ran experiments to see empirically what the optimal values of θ are with each width. The optimal values of θ by our theoretical derivation and from experiments with various $l : w$ ratios are listed in Table 2 for comparison. From Table 2, we can see clearly that they are quite close for most $l : w$ ratios. In addition, the optimal θ values are not proportional to $l : w$.

Table 2: The optimal θ values predicted by theory and from experiments.

$l : w$	8:1	8:2	8:3	8:4	8:5	8:6	8:7	8:8
$1/\theta$, theory	0.35	0.50	0.60	0.70	0.80	0.85	0.95	1.00
$1/\theta$, exp.	0.25	0.45	0.60	0.65	0.65	0.90	0.95	1.00

4.2 Arbitrary Polygonal Classification Regions

If the classification boundaries are not parallel to any axis, then we first need to calculate the predictive error rate (a function of θ) along such boundaries. We can then obtain

[9]In either case, however, the absolute difference in predictive errors when θ is near the optimal value is small. As indicated in (Ling et al., 1994), the advantage of weight setting methods for 1-NN is more evident in domains with a large number of irrelevant attributes.

the total predictive error of the whole region by integrating predictive errors along the circumference of the region. The total predictive error is again a function of θ. We could find the optimal θ value that minimizes the total error by function minimization techniques.

However, the method described above, though theoretically correct, is not practical. We have again designed a similar table look-up method for finding the optimal θ for polygonal classification regions. The predictive error introduced by a side (of a polygon) of a certain degree with the x axis can be calculated using the same equations (Equations 3, 4 and 5) in Section 3, except that now the areas A_1 and A_2 are more complicated. We constructed a table[10] containing (theoretical) predictive errors introduced by a unit length of a linear boundary at a certain degree (α) with varying θ values. The total predictive errors of the region is the sum of errors introduced by each boundary of the polygon. We can then vary θ to find the optimal value for the minimal predictive error of the polygonal region.

For example, if a classification region can be approximated by a polygon of n sides l_{α_1}, l_{α_2}, ..., l_{α_n}, and each side intersects with the x axis with angles α_1, α_2, ..., α_n respectively, then for a particular θ value, we can obtain the total predictive error introduced by the four edges as

$$l_{\alpha_1}E_{\alpha_1}(\theta) + l_{\alpha_2}E_{\alpha_2}(\theta) + \ldots + l_{\alpha_n}E_{\alpha_n}(\theta)$$

where $E_{\alpha_i}(\theta)$ is the table entry for the angle α_i and the weight ratio θ. By varying θ, we can then obtain the optimal weight ratio θ that minimizes the total predictive error rate above.

We ran an experiment with two simple rectangular shaped regions that are not parallel to any axis (Figure 5). Our experimental results show that the θ value that minimizes the predictive errors is between 0.7 to 0.9[11] for a 8×4 rectangle, and is between 0.7 to 0.8 for a 8×2 rectangle (again, this is obtained by averaging 10 runs). Our theoretical results of the optimal θ value using the table are 0.8 and 0.7, respectively; both fall in the ranges of the experimental results. This suggests that our method, based on theoretical analyses, works well.

[10] The table (20×10 in size) is too big to present here. It can be obtained on request.

[11] The predictive error within the range is within 1% of the minimal value.

5 Discussion

As we indicated in Section 1, our theoretical analyses assume that the classification boundary (i.e., the underlying target concept) is known. Thus, our optimal results can thus serve as a useful baseline or lower bound for evaluating empirical attribute weight assigning methods on known concepts. We choose two empirical weight setting methods to compare with our theoretical results on a simple target concept — three 2×8 axis-parallel rectangles centered in a 10×10 2-d continuous space as shown in Figure 6. As we discussed in Section 4.1 (also see Figure 4 (a)), the theoretically optimal θ value (w_1/w_2) for the minimal predictive error is 0.5.[12]

The first empirical method, as studied in (Daelemans et al., 1993; Wettschereck & Dietterich, 1995; Ling et al., 1994), is to set the attribute weights according to the information gain values of attributes. The second empirical method simply sets the weight (ratio) according to the ratio of the two lengths of a rectangle that is just large enough to enclose all positive examples. This simple heuristic uses one axis-parallel rectangle to approximate the classification boundary. In this case, the weight ratio θ is 1 (i.e., $w_1/w_2 = 1$). Note that when applying empirical methods in (Kibler & Aha, 1987) and (Cardie, 1993), the weight ratio θ would also be set to 1. We perform 20 runs for the comparison. For each run, 200 training examples and 40,000 testing examples are randomly drawn, and the same training and testing examples are used to evaluate the performance of the 1-NN algorithm with three weight settings: theoretical weight (always 0.5), information-gain based weight (determined by the training examples), and the rectangle ratio (around $8/8 = 1$). The results of the comparison are presented in Table 3. From the table, we can see clearly that theoretically optimal weights from our derivation have lower (about 30% lower) predictive error than either of the two empirical methods. The difference is significant in a t-test (at the 0.001 level).

Our theoretical analyses of optimal weight setting can be extended easily to k-NN algorithms. The only difference is at the derivation of p_1 and p_2 (Equations 3 and 4) in Section 3.1. p_1 is now the probability of area A_1 having at least k' (an integer) training examples where $k' > k/2$, and p_2 is the probability of area A_2 having $k - k'$ (0 if negative) training examples. The formulas would be more complex but the rest of the analyses and

[12] Since the three rectangles are relatively far apart, the overlap of errors introduced from edges of adjacent rectangles can be ignored, the optimal θ value is the same as the one for one rectangle.

Table 3: Comparison of the theoretical weight setting method with two empirical methods.

	$\theta = w_1/w_2$	Predictive error
Theoretical	0.5	10.4 $\pm$ 0.26
Information-based	0.94 $\pm$ 0.06	13.01 $\pm$ 0.49
Rectangle ratio	1.0 $\pm$ 0.01	13.50 $\pm$ 0.30

calculations remain the same.

It should be possible, at least in theory, to extend our method to domains with higher dimensions. In a space with n dimensions, the boundary becomes an n-d hyperplane, and the predictive error introduced by the hyperplane can be calculated by the volumes (similar to areas A_1 and A_2 in Section 3) formed by the intersection of the hyperplane and the hyper-ellipse. The volumes can be obtained by n-dimensional integration, though only numeric approximations are easily attainable. Then an n-dimensional table (similar to Table 1) can be constructed. The $n-1$ dimensions are used to specify the slope of the hyperplane, and the other dimension is for the various θ values. After such a table is constructed, by looking-up the table (i.e., the same method as in Section 4.2) we can determine the optimal θ value that minimizes the predictive error for any polygonal classification region in the n-dimensional space. Clearly, such a method is not practically feasible. We plan to study how optimal weights can be obtained more directly and effectively in higher dimensional spaces.

Another interesting area for future research is optimal weight setting for discrete or Boolean functions. One might think that this problem can be solved by our method in continuous space if we determine a set of boundaries in the continuous space that separate positive and negative examples of the Boolean function. However, the choice of such boundaries critically affects predictive accuracy. Further, the testing error in discrete domains may be quite different from the error introduced along the boundary in the continuous space. The work by Langley and Iba (1993) might be extended to solve this problem.

Finally, we plan to apply our theoretical method to a more realistic learning setting where no concept boundary is known (i.e., the only available information is the set of training examples). With an unknown concept boundary, the techniques described in

Section 4.2 cannot be used directly. One feasible approach to this problem is to first estimate a set of linear classifiers that separate positive and negative examples. Several standard techniques of finding piecewise linear classifiers (Fukunaga, 1990, pages 169–173) can be employed. Once the linear classifiers are found, our method (Section 4.2) can be applied to find the optimal weights for the k-NN algorithm. The actual classification boundary formed by the k-NN algorithm with optimal weights is different from the one formed by the piecewise linear classifiers and, therefore, it may be worthwhile to use such a k-NN learning algorithm with optimal attribute weights.

6 Conclusion

Our theoretical work is a first attempt towards computing optimal attribute weight settings that minimize the predictive errors for 1-NN algorithms. Under the assumption of uniform distribution of training and testing examples in a 2-d continuous space, we first derived the average predictive error introduced by the linear classification boundaries, and used these results to determine the optimal weights in the 1-NN distance function. We verified our theoretical findings with empirical experiments, and found that they match closely, indicating that several assumptions and approximations in our theoretical derivation and numeric estimation are reasonable. Our theoretical results can serve as a baseline or lower bound when evaluating other empirical weight setting methods on known concepts. However, our method is still quite limited in its scope since it only applies to any polygonal classification region in a 2-d continuous space. Several extensions for future work are outlined.

Appendix A

Since we have to calculate E numerically, we need to give finite bounds in the integration. We restrict O within $MaxR$ (some constant) from the boundary. This is because when O is too far away from the boundary, $p_1 p_2$ would be too small to be significant to E. Hence, the integration on d_2 ranges from 0 (when O is on the boundary) to $MaxR$ (when O is $MaxR$ from the boundary). The integration on d_1 could still go from 0 to infinity. We restrict the edge of the circle to expand to at most 2 $MaxR$ from the boundary. This is because we found that when the circle expands more, it does not contribute much to

E_O in integration. Therefore, the maximum diameter of the circle is $2MaxR - d_2$, and the maximum value of d_1 is $2MaxR - d_2 - d_2 = 2MaxR - 2d_2$. Thus, the bound on d_1 ranges from 0 (when the circle just touches the boundary) to $2MaxR - 2d_2$ (when the circle expands to 2 $MaxR$ from the boundary). We found that when $MaxR \geq 2$, E is very close to E with $MaxR = 2$. Thus, we set $MaxR = 2$ (unit length) in the calculation.

Acknowledgements

The authors thank reviewers for many helpful comments. Comments from David Aha were particularly useful in improving the paper.

Reference

Aha, D. W. (1989). Incremental, instance-based learning of independent and graded concept descriptions. In *Proceedings of the 1989 International Workshop on Machine Learning*, pp. 387 – 391. Irvine, CA: Morgan Kaufmann.

Aha, D. W., & Bankert, R. L. (1994). Feature selection for case-based classification of cloud types: An empirical comparison. In Aha, D. W. (Ed.), *Proceedings of the AAAI-94 Workshop on Case-Based Reasoning (Technical Report WS-94-01)*, pp. 106 – 112. Menlo Park, CA: AAAI Press.

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

Albert, M. K., & Aha, D. W. (1991). Analyses of instance-based learning algorithms. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pp. 553–558. Menlo Park, CA: AAAI.

Cardie, C. (1993). Using decision trees to improve case-based learning. In *Proceedings of the 1993 International Conference on Machine Learning*, pp. 25 – 32. Morgan Kaufmann, San Mateo, CA.

Caruana, R., & Freitag, D. (1994). Greedy attribute selection. In *Proceedings of the 1994 International Conference on Machine Learning*, pp. 28 – 36. Morgan Kaufmann, CA.

Char, B., Geddes, K., Gonnet, G., Leong, B., Monagan, M., & S.M.Watt (1992). *First Leaves: A Tutorial Introduction to Maple V.* Springer-Verlag and Waterloo Maple Publishing.

Cost, S., & Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, *10*, 57.

Cover, T. (1968). Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, *14*, 50–55.

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

Daelemans, W., Gillis, S., Durieux, G., & van den Bosch, A. (1993). Learnability and markedness in data-driven acquisition of stress. Tech. rep. ITK Research Report No. 43, Institute for Language Technology and AI (ITK), Tilburg University.

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

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition* (Second edition). Academic Press.

Kelly, J. D., & Davis, L. (1991). A hybrid algorithm for classification. In Mylopoulos, J., & Reite, R. (Eds.), *Proceedings of IJCAI-91 (Thirteenth International Conference on Artificial Intelligence)*, pp. 645–650. Morgan Kaufmann, San Mateo, CA.

Kibler, D., & Aha, D. W. (1987). Learning representative exemplars of concepts: An initial case study. In *Proceedings of the 1987 International Workshop on Machine Learning*, pp. 24–30. Irvine, CA: Morgan Kaufmann.

Langley, P., & Iba, W. (1993). Average-case analysis of a nearest neighbor algorithm. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pp. 889–894. Morgan Kaufmann: San Mateo, CA.

Ling, C. X., Parry, J. J., & Wang, H. (1994). Deciding weights for IBL using C4.5. Submitted.

Mohri, M., & Tanaka, H. (1994). An optimal weighting criterion of case indexing for both numeric and symbolic attributes. Tech. rep. WS-94-01, Case-Based Reasoning: Papers from the 1994 Workshop. Menlo Park, CA: AAAI Press.

Moore, A. W., & Lee, M. S. (1994). Efficient algorithms for minimizing cross validation error. In *Proceedings of the 1994 International Conference on Machine Learning*, pp. 190–198. Morgan Kaufmann, CA.

Okamoto, S., & Satoh, K. (1995). An average-case analysis of k-nearest neighbor classifier. In *Proceedings of the First International Conference on Case-Based Reasoning*, pp. 253–264. Sesimbra, Portugal: Springer-Verlag.

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

Quinlan, J. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann: San Mateo, CA.

Satoh, K., & Okamoto, S. (1994). Toward PAC-learning of weights from qualitative distance information. Tech. rep. WS-94-01, Case-Based Reasoning: Papers from the 1994 Workshop. Menlo Park, CA: AAAI Press.

Skalak, D. (1994). Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *Proceedings of the 1994 International Conference on Machine Learning*, pp. 293 – 301. Morgan Kaufmann, CA.

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

Wettschereck, D., & Aha, D. (1995). Weighting features. In *Proceedings of the First International Conference on Case-Based Reasoning*, pp. 347–358. Sesimbra, Portugal: Springer-Verlag.

Wettschereck, D., & Dietterich, T. (1995). An experimental comparison of the nearest neighbor and nearest hyperrectangle algorithms. *Machine Learning*, *19*, 5–28.

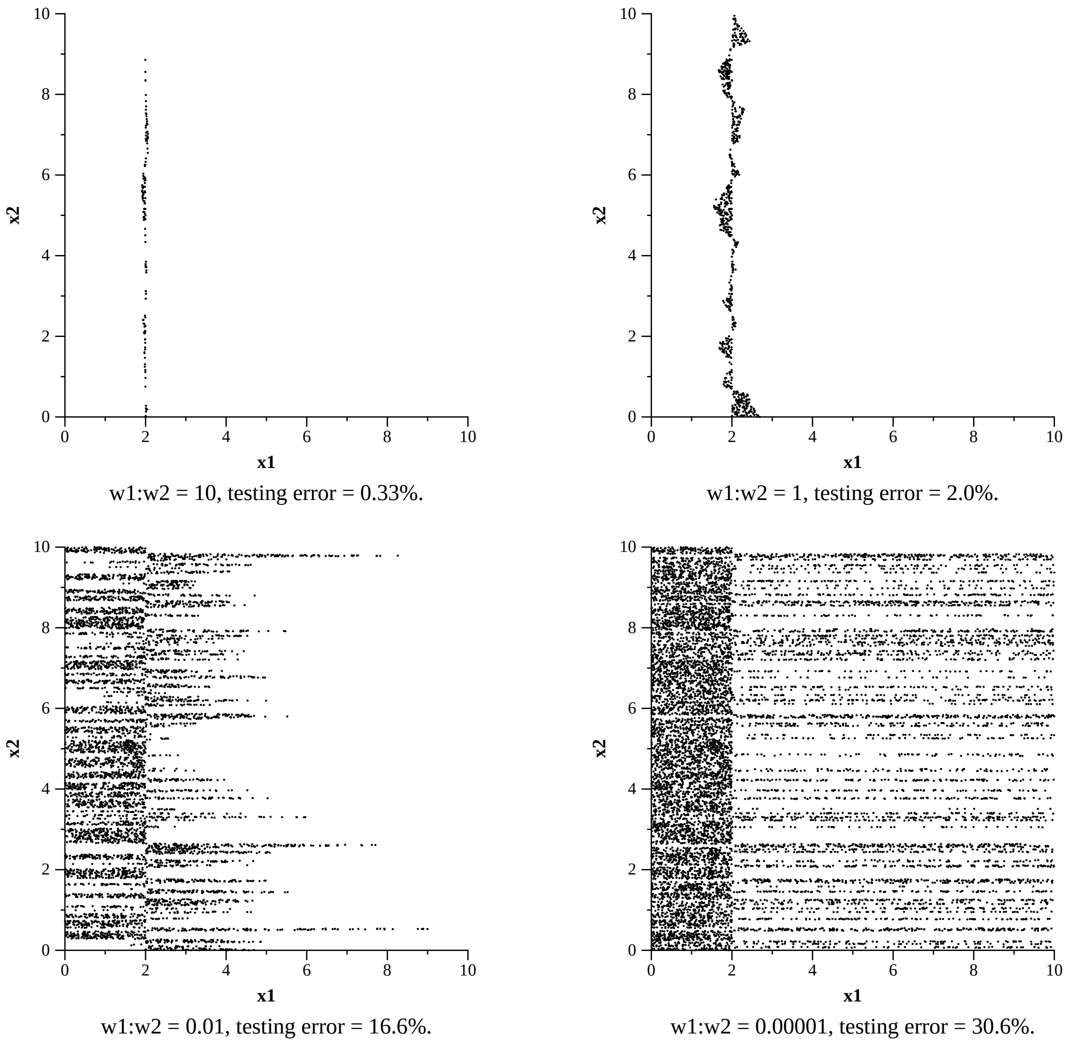

Figure 1: Misclassified testing examples with various weight ratios for the classification boundary $x = 2$ (left side negative, right side positive).

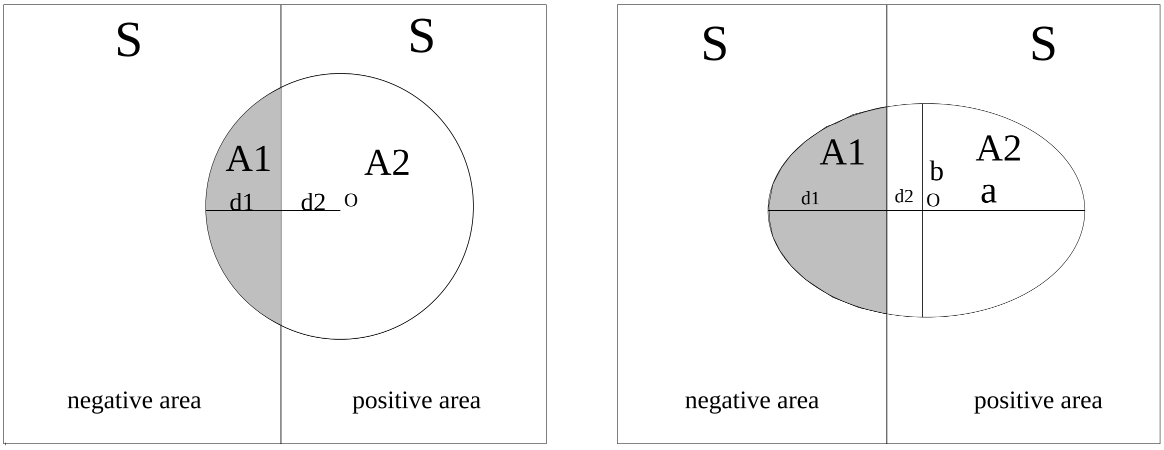

Figure 2: Situations where a positive example (at O) is classified as a negative example. (a): $w_1 = w_2$; (b): $w_1 \neq w_2$.

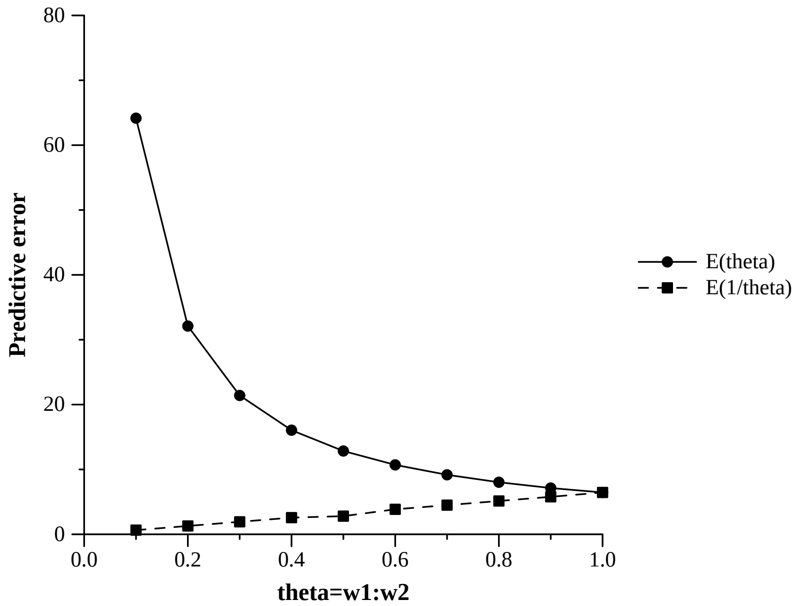

Figure 3: The predictive error $E(\theta)$ with various weights in 1-NN.

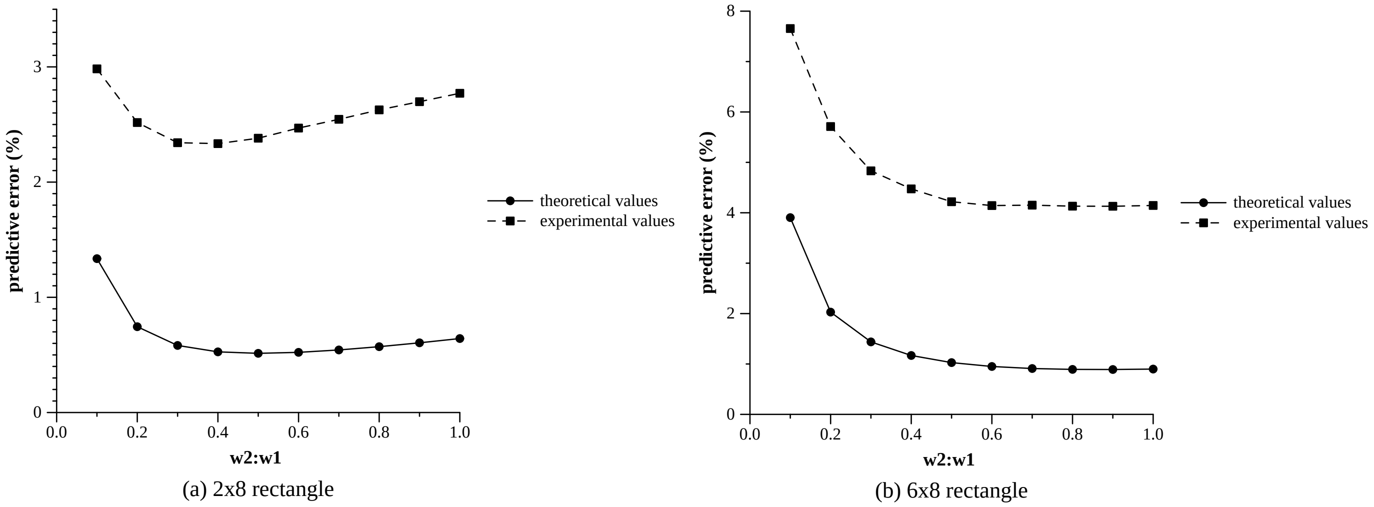

Figure 4: The predictive error with various θ values. The theoretical and experimental curves do not overlap since the theoretically optimal weights are relative, and irrelevant to the actual number of training examples.

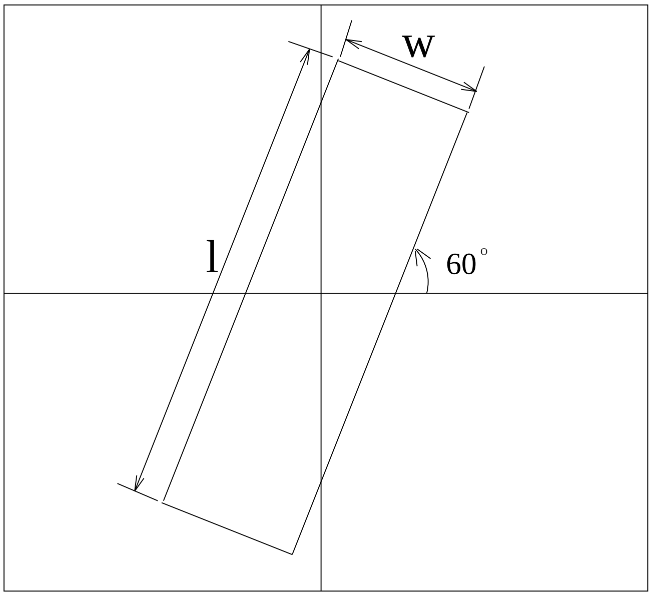

Figure 5: An arbitrary rectangular classification region.

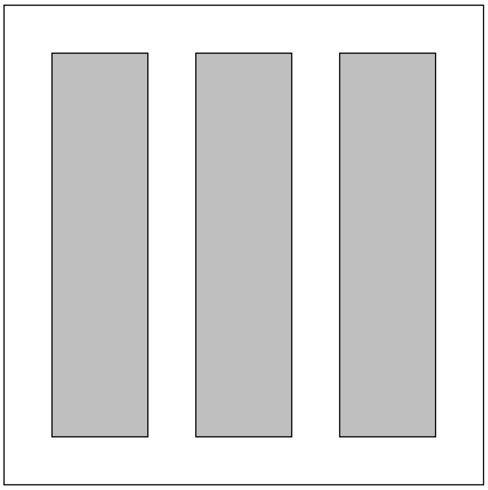

Figure 6: A simple concept (shaded areas represent positive examples) for comparing theoretical and empirical weight setting methods.