Learning Pattern Classification—A Survey

S. R. Kulkarni[*], G. Lugosi[†], and S. S. Venkatesh[‡]

January 30, 1998, Revised June 3, 1998

Abstract

Classical and recent results in statistical pattern recognition and learning theory are reviewed in a two-class pattern classification setting. This basic model best illustrates intuition and analysis techniques while still containing the most essential features and serving as a prototype for many applications. Topics discussed include nearest neighbor, kernel, and histogram methods, Vapnik-Chervonenkis theory, and neural networks. The presentation and the large (if non-exhaustive) list of references is geared to provide a useful overview of this field for both specialists and non-specialists.

Key Words: learning, classification, statistical pattern recognition, survey, review

1 Introduction

The goal of learning theory is to provide answers to basic questions such as:

- What problems can and cannot be learned?
- How much data are required?
- What are good algorithms for learning from examples?

Hence learning theory attempts to delineate the fundamental limitations on learning in much the same way as information theory does for communication, and the theory of computation does for computing.

Broadly speaking, by "learning" we think of an agent (the learner) immersed in an environment. The learner interacts with the environment, thereby gathering data. Using this data, together with any prior knowledge or assumptions about the environment, the learner forms some internal representation or model of the environment that is used for various tasks such as prediction, planning some future action, etc. Of course, to get concrete results one needs to specify in detail the different aspects of the model. Through specific choices on the prior assumptions, data, and success criterion, one can get a wide variety of topics generally associated with learning such as language identification, density and regression estimation, pattern classification, stochastic control, reinforcement learning, clustering, etc.

Work on these areas spans a number of fields and many years. In the last several decades, there was work in the 1940's and 1950's in areas such as statistics, information theory, cybernetics, and early work on neural networks that led to tremendous progress and in fact established several new fields of activity. Continued work in these areas, slightly later work on systems and control theory, pattern recognition, and optimization, and more recently the explosion of work on neural networks and other topics such as computational learning theory, are all part of this general area.

In this paper we focus on a very specific subset of this work dealing with two-class pattern classification. This problem, defined below, serves as a prototype for many real-life learning problems, while the mathematical simplicity of the model allows us to gain insight into most of the difficulties arising in learning problems. However, this model by no means covers all aspects of learning. For example, the assumption of having only two classes hides many of the basic difficulties of some practical problems with a huge number of classes, such as continuous speech recognition. Also, by assuming independence of the training samples we exclude important applications where the dependence of the data is an essential feature. We do not discuss problems of feature selection (i.e., determining which measurements should serve as components of the feature vector), which, as anyone who has ever tried to build a pattern classifier fully appreciates, is one of the most important and difficult problem-specific elements of learning. We also do not include in this review problems of active learning and sequential learning. Still, the simple two-class classification model we review here is sufficiently rich and general so that one can gain useful intuition for other, perhaps more complex, learning problems.

The pattern classification problem is generally formulated as follows.[1] There are two classes of objects (or states of nature) of interest, which we will call class 0 and 1, respectively. Our information about an object is summarized by a finite number, say d, of real-valued measurements called *features*. Together, these measurements comprise a *feature vector* $x \in \Re^d$. To model the uncertainty about which class objects we encounter belong, we assume that there are *a priori* probabilities P_0 and P_1 for the two classes. To model the relationship between the class to which an object belongs and the feature vector (including uncertainty or noise in the measurement process), we assume that an object in class $y \in \{0,1\}$ engenders a random feature vector with class-conditional distribution function $F_y(x)$. Random feature vectors X (the "observables" in this process) are generated according to the following two-stage process: a random class $Y \in \{0,1\}$ is first selected according to the *a priori* probabilities $\{P_0, P_1\}$; the observed feature vector X is then selected according to the class-conditional distribution F_Y. Given a realization of the measured feature vector $X = x$ the problem facing the classifier is to decide whether the unknown object engendering

[*]Department of Electrical Engineering, Princeton University, Princeton, NJ 08544, USA; kulkarni@ee.princeton.edu. Supported in part by the National Science Foundation under NYI grant IRI-9457645.

[†]Department of Economics, Pompeu Fabra University, Ramon Trias Fargas 25-27, 08005 Barcelona, Spain; lugosi@upf.es. Supported by DGES grant PB96-0300.

[‡]Department of Electrical Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA; venkatesh@ee.upenn.edu. Supported in part by the Air Force Office of Scientific Research under grants F49620-93-1-0120 and F49620-92-J-0344.

[1]Our notation is generally consistent with the mathematical statistics literature and follows [90].

the feature vector x belongs to class 0 or 1. Thus, a classifier or decision rule in this case is simply a map $g: \Re^d \to \{0,1\}$ which indicates the class $g(x)$ to which an observed feature vector $X = x$ should be assigned. Given a classifier g, the performance of g can be measured by the *probability of error*, given by

$$L(g) = \mathbf{P}\{g(X) \neq Y\}.$$

If the *a priori* probabilities and conditional distributions are known, then it is well known that the optimal decision rule in the sense of minimum probability of error (or, more generally, minimum risk if different costs are assigned to different types of errors) is the *Bayes decision rule*, denoted g^*. This decision rule simply uses the known distributions and the observation $X = x$ to compute the *a posteriori* probabilities $\eta_0(x) = \mathbf{P}\{Y = 0 \mid X = x\}$ and $\eta_1(x) = \mathbf{P}\{Y = 1 \mid X = x\}$ of the two classes, and selects the class with the larger *a posteriori* probability (or smaller risk), i.e.,

$$g^*(x) = \arg \min_{y \in \{0,1\}} \eta_y(x).$$

The performance of the Bayes decision rule, denoted L^*, is then given by

$$L^* = L(g^*) = \mathbf{E}\left[\min\{\eta_0(X), \eta_1(X)\}\right].$$

Of course, in many applications these distributions may be unknown or only partially known. In this case, it is generally assumed that in addition to the observed feature vector X, one has previous labeled observations

$$D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\},$$

where $Y_k \in \{0,1\}$ corresponds to the class of the objects and is assumed to form an i.i.d. sequence of labels drawn according to the unknown probability rule $\{P_0, P-1\}$, and X_k is a feature vector drawn according to the class-conditional distribution $F_{Y_k}(x)$. Thus, the (X_k, Y_k) pairs are assumed to be independent and identically distributed according to the (unknown) distributions P_y and $F_y(x)$ characterizing the problem. Intuitively, the data D_n provide some partial information about the unknown distributions, and we are interested in using this data to find good classifiers. Formally, a classifier (or classification rule) is a function $g_n(x) = g_n(x, D_n)$, which, based on the training data D_n, assigns a label (0 or 1) to any input point $X \in \Re^d$ to be classified. For fixed training data D_n, the conditional probability of error of such a classifier is

$$L(g_n) = \mathbf{P}\left\{g_n(X) \neq Y \mid D_n\right\},$$

where the pair (X, Y) is independent of D_n, and is drawn according to the same distribution as the one generating the training samples. The probability of error $L(g_n)$ is a random variable as it depends on the (random) training data D_n. The expected probability of error $\overline{L(g_n)} = \mathbf{E}L(g_n) = \mathbf{P}\left\{g_n(X) \neq Y\right\}$ tells us the average behavior of such a classifier (where the expectation is taken with respect to the random training data). In evaluating

the performance of a classifier, we use as our benchmark the Bayes error rate L^* which is the best we could do even if we knew the distributions completely.

In selecting a classifier using a finite amount of random data, it is natural to expect that the error rate of our classifier can only be close to the optimal Bayes error rate in some probabilistic sense. One goal to aim for might be to design a so called "*consistent rule*" such that as we get more and more training data ($n \to \infty$), we have $L(g_n) \to L^*$ in probability (which is equivalent to $\mathbf{E}L(g_n) \to L^*$ as $L(g_n)$ is dominated by 1). If instead we have $L(g_n) \to L^*$ with probability one then the rule is called *strongly consistent*. In general, given a rule the behavior of $L(g_n)$ will depend on the underlying (unknown) distribution of (X, Y). If a rule satisfies $\lim_{n\to\infty} L(g_n) = L^*$ in probability (respectively, with probability one) for *every* distribution of (X, Y) the rule is said to be *universally consistent* (respectively, *strongly universally consistent*). Of course, since it may be unrealistic to make assumptions or impossible to verify conditions on the distribution of (X, Y), if at all possible we would like to design universally consistent rules. A milestone in the theory of pattern recognition is the seminal paper of Stone [267] who first showed the existence of universally consistent rules. This gives the user the important guarantee that if a sufficient amount of data is collected, his classifier will perform almost as well as the optimal Bayes classifier, regardless of the underlying (unknown) distribution. However, this is not the end of the story. For all classification rules, the convergence to L^* may be arbitrarily slow, and, for any finite sample size n, the gap between L^* and the actual probability of error may be arbitrarily close to $1/2$ (see Cover [67], Devroye [83]). These facts show that designing good classifiers is a highly nontrivial task, and many different points of view may be adopted.

An initial approach one might take in designing good rules is to assume the distributions are of some known simple form, and only a small number of parameters are unknown. Such "parametric methods" have been widely studied, and are useful when in fact the problem under consideration is sufficiently well-understood to warrant the parametric assumptions on the distributions. However, in many cases one may have very little knowledge about the distributions and parametric assumptions may be quite unrealistic. Hence, the study of "nonparametric methods," and universally consistent rules, in particular, has also received a great deal of attention. The characterization of attainable performance, the search for good decision rules, and the analysis of their performance, for nonparametric statistical pattern classification is the focus of this paper. Both "classical" and recent results are reviewed, with an emphasis on universally consistent rules. Several excellent books have been published containing much of the classical work and/or recent work. There are also many books focusing on various subtopics. A partial list includes Anthony and Biggs [13], Breiman, Friedman, Olshen, and Stone [50], Devijver and Kittler [78], Devroye, Györfi, and Lugosi [90], Duda and Hart [97], Fukunaga [116], Kearns and Vazirani [164], McLachlan [193], Natarajan [205], Vapnik [274, 275], Vapnik and Chervonenkis [277], and Vidyasagar [285]. A huge number of papers have also been written on the subjects discussed here. In the sections discussing the various subtopics we have cited a number of these papers, but we certainly have not attempted to provide an exhaustive bibliography, and we apologize for any omissions. Rather, our aim is to provide a presentation and some pointers to the literature that will be a useful overview or entry into this field for both specialists and non-specialists.

In Section 2, we discuss nearest neighbor classifiers, which conceptually are perhaps the simplest methods for pattern classification. The closely related kernel classifiers are presented in Section 3, and histogram and classification trees are the subject of Section 4. These three techniques share many conceptual similarities, although the details in analysis and implementation are often quite distinct. A very different and more recent approach to the subject that has seen a great deal of recent activity is Vapnik-Chervonenkis theory, which is discussed in Section 5. This approach has, among other things, provided a theoretical basis for the analysis of neural networks, which are discussed in Section 6. Work on neural networks has been pursued for the last several decades, but there has been an explosion of work in this area since the early 1980's. In the final two sections we discuss some recent work on improving and evaluating the performance of some of the more basic techniques. Specifically, Section 7 describes large-margin classifiers and support vector machines, while Section 8 discusses automatic parameter selection and error estimation.

2 Nearest Neighbor Classifiers

In its original form (Fix and Hodges [111, 112]), the nearest neighbor classifier is perhaps the simplest pattern classification algorithm yet devised. In its classical manifestation, given a reference sample $D_n = \{(X_i, Y_i), 1 \le i \le n\}$, the classifier assigns any input feature vector to the class indicated by the label of the nearest vector in the reference sample. More generally, the k-nearest neighbor classifier maps any feature vector X to the pattern class that appears most frequently amongst the k-nearest neighbors. (If no single class appears with greatest frequency, then an auxiliary procedure can be invoked to handle ties.) Despite its simplicity, versions of this nonparametric algorithm discussed below are asymptotically consistent with a Bayes classifier, and are competitive with other popular classifiers in practical settings.

2.1 Formulation

Given a metric $d(X, X')$ on $\Re^d$ and a positive integer k, the k-nearest neighbor classifier generates a map from $\Re^d$ into $\{0,1\}$ as a function of the reference sample D_n wherein each point $X \in \Re^d$ is mapped into one of the two classes according to the majority of the labels of its k-nearest neighbors in the reference sample. More particularly, fix any $X \in \Re^d$. We may suppose, without loss of generality, that the indices of the labeled feature vectors in D_n are permuted to satisfy

$$\begin{aligned} d(X,X_1) \le d(X,X_2) \le \ldots \le d(X,X_k), \\ d(X,X_j) \ge d(X,X_k) \qquad \text{for } j = k+1, \ldots, n. \end{aligned} \tag{1}$$

The k-nearest neighbors of X are identified as the labeled subset of feature vectors

$$\{(X_1,Y_1),\ldots,(X_k,Y_k)\}$$

of the reference sample. The k-nearest neighbor classifier then assigns X to the class

$$Y' = \mathrm{maj}(Y_1,\ldots,Y_k), \tag{2}$$

viz., the most frequent class label exhibited by the k-nearest neighbors of X.

If k is even it is necessary to define an auxiliary procedure to handle ties in (2). (Although the equalities in (1) can also be problematic, they occur with zero probability if the class-conditional distributions are absolutely continuous.) A simple method is to break the tie according to a deterministic rule; for example, in the event that more than one class occurs with greatest frequency in the subset of k-nearest neighbors, then the input pattern could be assigned to the most prolific class in the subset with the smallest class label. A convenient way to describe deterministic tie-breaking rules such as this is to construct an *assignment partition* $(\mathcal{L}_0, \mathcal{L}_1)$ of the space $\{0,1\}^k$ that describes the action of the k-nearest neighbor classifier for every possible ordered k-tuple of class labels. Here, $\mathcal{L}_i$ contains every ordered k-tuple, $\mathbf{Y} = (Y_1,\ldots,Y_k)$, representing the respective class labels of $X_1, \ldots, X_k$, for which an assignment to class i occurs. For example, if $k = 2$, then (2) with the smallest-class-label tie-breaking algorithm induces the assignment partition, $\mathcal{L}_0 = \{(0,0),(0,1),(1,0)\}$ and $\mathcal{L}_1 = \{(1,1)\}$. By introducing an extra element $\mathcal{L}_r$ into the partition, one can represent a k-nearest neighbor classifier that rejects certain input patterns (Hellman [146]). A k-nearest neighbor classifier that can be described by an assignment partition is called *deterministic*.

Random tie-breaking algorithms can be represented by a stochastic process in which an assignment partition is selected from an ensemble of partitions, according to a discrete probability distribution, prior to each assignment.

2.2 Classifier Risk

Write $L_n(k)$ for the probability of error of a k-nearest neighbor classifier conditioned on the random sample D_n, and let $\overline{L}_n(k) = \mathbf{E}(L_n(k))$ denote the expected risk. Let $L_\infty(k)$ and $\overline{L}_\infty(k)$ denote the corresponding values in the limit of an infinite sample. Some examples may help fix the notions.

EXAMPLE 1: *Nonoverlapping distributions* (Cover and Hart [69]).

Consider a two-class problem in the d-dimensional feature space $\Re^d$. Suppose each class label $y \in \{0,1\}$ comes equipped with a conditional density f_y which has probability-one support in a compact set A_y in $\Re^d$. Suppose further that the distance between the sets A_0 and A_1 is larger than the diameter of either set, i.e., $d(A_0,A_1) > \max\{\mathrm{diam}(A_0), \mathrm{diam}(A_1)\}$. As the two classes have nonoverlapping probability-one supports, it is clear that $\overline{L}_\infty(k) = L^* = 0$ for every positive integer k. Moreover, the finite-sample risk $\overline{L}_n(k)$ of the k-nearest neighbor classifier approaches its infinite-sample limit $\overline{L}_\infty(k)$ exponentially fast for fixed k as n increases. Indeed, for definiteness suppose that the two classes have equal *a priori* probabilities, $P_0 = P_1 = 1/2$, and that k is an odd integer. The classifier will make a mistake on a given class if, and only if, there are fewer than $k/2$ examples of that class in the reference sample, whence

$$\overline{L}_n(k) = 2^{-n} \sum_{i=0}^{(k-1)/2} \binom{n}{i} = O(n^{(k-1)/2} 2^{-n}) \qquad (n \to \infty).$$

Note that the exponentially fast rate of convergence of $\overline{L}_n(k)$ to 0 is independent of the feature space dimension d.

EXAMPLE 2: *Normal Distributions.*

Consider the classification problem described by the two multivariate normal class-conditional densities,

$$\begin{aligned} f_0(x) &= \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left\{-\frac{1}{2\sigma^2}\left((x_1-\mu)^2 + \sum_{j=2}^{d} x_j^2\right)\right\}, \\ f_1(x) &= \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left\{-\frac{1}{2\sigma^2}\left((x_1+\mu)^2 + \sum_{j=2}^{d} x_j^2\right)\right\}, \end{aligned}$$

and *a priori* probabilities, $P_0 = P_1 = 1/2$. Here, again, the feature vectors $x = (x_1, \dots, x_d)$ range over d-dimensional feature space $\Re^d$. A direct calculation now shows that the risk of the 1-nearest neighbor classifier tends to

$$\overline{L}_\infty(1) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\mu^2/2\sigma^2} \int_0^\infty e^{-x^2/2\sigma^2} \operatorname{sech}\left(\frac{\mu x}{\sigma^2}\right) dx.$$

For $\mu = \sigma = 1$, a numerical integration yields $\overline{L}_\infty(1) \approx 0.22480$, which may be compared with the Bayes risk, $L^* = (1/2)\mathrm{erf}(1/\sqrt{2}) \approx 0.15865$.

2.2.1 Limiting Results: Infinite Sample

The enduring popularity of the nearest neighbor classifier stems in part from its extreme simplicity and in part from its near-optimal asymptotic behavior.

As before, write $\eta_y(x) = \mathbf{P}\{Y = y \mid X = x\}$ for the posterior probability of class y given feature vector x. In a classical paper, Cover and Hart [69] showed that,

$$\overline{L}_n(k) \to \overline{L}_\infty(k) = \sum_{y\in\{0,1\}} \sum_{\mathbf{Y}\notin\mathcal{L}_y} \mathbf{E}\left[\eta_y(X)\eta_{y_1}(X)\dots\eta_{y_k}(X)\right] \qquad (n\to\infty)$$

for *any* choice of metric d. In particular, for k odd, they obtain

$$\overline{L}_\infty(k) = \sum_{y\in\{0,1\}} \sum_{j=0}^{(k-1)/2} \binom{k}{j} \mathbf{E}\left[\eta_y(X)^j \left(1-\eta_y(X)\right)^{k-j}\right]. \tag{3}$$

While Cover and Hart proved the above asymptotic formulas under some weak conditions on the class-conditional densities, Devroye [80] showed subsequently that in fact these results are true for *all distributions* of (X, Y). The fundamental result established by Cover and Hart now relates the infinite sample risk $\overline{L}_\infty(k)$ to the minimum achievable risk L^* of the Bayes classifier through the two-sided inequalities

$$L^* \le \overline{L}_\infty(k) \le \dots \le \overline{L}_\infty(5) \le \overline{L}_\infty(3) \le \overline{L}_\infty(1) \le 2L^*(1-L^*),$$

where the upper and lower bounds are as tight as possible, in general. Thus, when the Bayes risk is small, as is the case in several important pattern classification problems such

as character recognition, the nearest-neighbor classifier exhibits an asymptotically optimal character.

The first explicit bounds on $\overline{L}_\infty(k)$ in terms of L^* were established by Györfi and Györfi [138] and Devijer [77] who showed

$$\overline{L}_\infty(k) \le L^* + \overline{L}_\infty(1)\sqrt{\frac{2}{\pi\lceil k/2\rceil}}.$$

Consequently, the rate of convergence (in k) of $\overline{L}_\infty(k)$ to L^* is at least of the order of $k^{-1/2}$. A survey and tighter bounds may be found in [90, Chapter 5].

2.2.2 Finite Sample

These encouraging infinite sample results notwithstanding, the utility of this non-parametric approach as a *practical* classifier is, however, tempered by how rapidly the finite-sample risk $\overline{L}_n(k)$ converges to $\overline{L}_\infty(k)$. The previous examples demonstrate indeed that the rate of convergence of $\overline{L}_n(k)$ to $\overline{L}_\infty(k)$ depends critically on the underlying classification problem, and, moreover, can vary over a considerable range.

For problems with two pattern classes and a one-dimensional feature space, Cover [66] has shown that the infinite-sample limit is approached as rapidly as $\overline{L}_n(1) = \overline{L}_\infty(1) + O(n^{-2})$ $(n\to\infty)$ if the probability distributions that define the classification problem are sufficiently smooth with densities that have compact probability-one support.

More generally, a result analogous to (3) may be obtained in the finite sample case by conditioning on the sample. Suppose the class-conditional distributions F_0 and F_1 are absolutely continuous with corresponding class-conditional densities f_0 and f_1, respectively. Let $f = P_0 f_0 + P_1 f_1$ denote the mixture density. Introduce the notation

$$B(\rho, X) \stackrel{\text{def}}{=} \{\, X' \in \Re^d : d(X', X) \le \rho \,\}.$$

for the closed ball of radius ρ at X, and write

$$S_j \stackrel{\text{def}}{=} B(d(X_j, X), X) \cap S \qquad (1 \le j \le k)$$

for points in the probability-one support S of the mixture density f at distance no more than $d(X_j, X)$ from X. The indexing convention (1) can hence be written equivalently in the form

$$X_{j-1} \in S_j \qquad (2 \le j \le k).$$

For $\rho \ge 0$ and $x \in \Re^d$, let

$$\psi(\rho, x) \stackrel{\text{def}}{=} \int_{B(\rho,x)} f(x')\,dx'$$

represent the probability that a feature vector drawn from the mixture density f falls within the ball of radius ρ at x and introduce notation for the "falling factorial"

$$(m)_k \stackrel{\text{def}}{=} m(m-1)\dots(m-k+1).$$

Simple conditioning arguments now show that the finite sample risk $\overline{L}_n(k)$ can be expressed in the exact integral form (cf. Snapp and Venkatesh [258])

$$\overline{L}_n(k) = (m)_k \int_S \int_S \int_{S_k} \cdots \int_{S_3} \int_{S_2} G(x, x_1, \ldots, x_k)\, e^{-nh(d(x,x_k),x)}\, dx_1\, dx_2 \ldots dx_{k-1}\, dx_k\, dx, \qquad (4)$$

where

$$G(x, x_1, \ldots, x_k) \stackrel{\text{def}}{=} \frac{\sum_{y\in\{0,1\}} P_y f_y(x) \sum_{\mathbf{Y}\notin\mathcal{L}_y} \prod_{j=1}^k P_{y_j} f_{y_j}(x_j)}{(1-\psi(d(x,x_k),x))^k},$$

and

$$h(\rho, x) \stackrel{\text{def}}{=} -\ln(1-\psi(\rho,x)).$$

The form of the integral representation (4) for the finite sample risk is suggestive of a Laplace integral [108], and indeed, for sufficiently smooth class-conditional densities f_y with compact support, a complete asymptotic series expansion for the finite sample risk is obtainable in the form [119, 225, 259]

$$\overline{L}_n(k) \sim \overline{L}_\infty(k) + \sum_{j=2}^{\infty} c_j n^{-j/d} \qquad (n \to \infty).$$

The expansion coefficients c_j depend in general upon k, the choice of metric, and the chosen procedure for handling ties, in addition to the probability distributions that describe the classification problem under consideration, but are independent of the sample size n. The leading coefficient $\overline{L}_\infty(k)$ is just the infinite sample risk derived by Cover and Hart [69]; while $\overline{L}_\infty(k)$ depends on k and the underlying distributions, it is independent of the choice of metric as noted earlier. In particular, the rate of approach of $\overline{L}_n(k)$ to $\overline{L}_\infty(k)$ is given by

$$\overline{L}_n(k) = \overline{L}_\infty(k) + O(n^{-2/d}) \qquad (n \to \infty)$$

in accordance with Cover's result in one dimension. Note, however, that the result provides a vivid illustration of Bellman's curse of dimensionality: the finite sample nearest neighbor risk $\overline{L}_n(k)$ approaches its infinite sample limit $\overline{L}_\infty(k)$ only as slowly as the order of $n^{-2/d}$. Conversely, this indicates that the sample complexity demanded by the nearest neighbor algorithm to achieve acceptable levels of performance grows exponentially with the dimension d for a typical smooth classification problem. In particular, the sample complexity n needed to achieve a finite sample risk which is ϵ-close to the infinite sample risk is asymptotically $n \sim (c_2/\epsilon)^{d/2}$.

The conditional risk $L_n(k)$, i.e., the probability of error of a k-nearest neighbor classifier for a given random sample D_n, has also been shown to converge *in probability* to $\overline{L}_\infty(k)$ under a variety of conditions [115, 136, 287]. Indeed typical results for smooth distributions with compact, convex supports, for instance, show that

$$\mathbf{P}\{|L_n(k) - \overline{L}_\infty(k)| \geq \epsilon\} \leq Ae^{-Bn^\alpha},$$

where $A = A(\epsilon)$ and $B = B(\epsilon)$ are positive and depend on ϵ and $0 < \alpha < 1$ is independent of ϵ.

2.3 Refinements

2.3.1 Consistency and the choice of k

All the asymptotic results mentioned so far concern the k-nearest neighbor rule with a fixed k and increasing sample size n. In such cases the expected probability of error converges to a constant $\overline{L}_\infty(k)$. Also, as k increases, this limiting constant tends to L^*. A natural question arising immediately is if there is a way of increasing the value of k as the sample size grows such that the resulting k_n nearest neighbor rule is consistent. This question was answered in a groundbreaking paper of Stone [267] who proved that the probability of error $L(g_n)$ of the k_n nearest neighbor rule (with an appropriate tie-breaking strategy) satisfies

$$\lim_{n\to\infty} \mathbf{E}L(g_n) = L^*$$

for all distributions provided only that the sequence $\{k_n\}$ satisfies $k_n \to \infty$ and $k_n/n \to 0$ as $n \to \infty$. In other words, the k_n-nearest neighbor rule is universally consistent. Indeed, Stone's paper was the first to establish the existence of universally consistent rules—a startling result in 1977. In Devroye, Györfi, Krzyżak, and Lugosi [89] it is shown that the k_n-nearest neighbor rule is also *strongly* universally consistent under the same conditions on the sequence $\{k_n\}$. A partial list of related work includes Beck [39], Györfi and Györfi [137], Devroye [81, 82], Collomb [61, 62, 63], Bickel and Breiman [42], Mack [189], Stute [268], Devroye and Györfi [88], Bhattacharya and Mack [41], and Zhao [301].

Stone's consistency theorem suggests that the number of neighbors k considered in the decision should grow with the sample size n but not too rapidly. However, this theorem gives little guidance for the choice of k in a practical problem with (say) $n = 1500$. What makes the problem difficult is that it is impossible to determine k solely as a function of the sample size without erring grossly in most situations. As it turns out, the appropriate choice of k depends heavily on the actual distribution of (X, Y).

As Cover and Hart [69] observed, for some distributions $k = 1$ is optimal for all n! Thus, the 1-nearest neighbor rule is *admissible*, i.e., there is no $k > 1$ for which the expected risk of the k-nearest neighbor classifier dominates that of the 1-nearest neighbor classifier for all distributions. This is seen most clearly in the case of non-overlapping distributions as we saw earlier. Indeed, Devroye, Györfi, and Lugosi [90, Section 5.8] argue further that, when L^* is small, there is little advantage in choosing k larger than 3.

However, in most practical problems the data-dependent choice of k is inevitable. We refer to Section 8 for a discussion and survey of such methods.

One way of smoothing the k-nearest neighbor decision rule is to introduce different weights to the different neighbors of the input point X, typically decreasing with the distance from X. For example, Royall [237] proposed the following classifier:

$$g_n(X) = \begin{cases} 1 & \text{if } \sum_{i:Y_{(i)}(x)=1}^k w_i > \sum_{i:Y_{(i)}(x)=0}^k w_i \\ 0 & \text{otherwise,} \end{cases}$$

where $Y_1, \ldots, Y_k$ denote, as before, the labels of the (permuted) k nearest neighbors of X (ordered according to their increasing distance from X) and $w_1, \ldots, w_k$ are the corresponding fixed weights. Bailey and Jain [19] proved the surprising result that if k is fixed, the asymptotic probability of error of a weighted k-nearest neighbor rule is minimal for all distributions if the weights are uniform (see [90, Section 5.5] for a simple proof), and therefore in this sense weighting is useless. However, weighting may be advantageous for finite sample sizes, and also if k and the weight vector are allowed to change with the sample size. Indeed, Stone [267] established conditions for the consistency of such rules. Data-dependent choices of the weights are also possible, see [90, Section 26.3].

2.3.2 Choice of metric

Another parameter to be set by the user of the k-nearest neighbor rule is the metric according to which distances are measured. This is often a nontrivial task as different components of the input vector may represent quantities rather different in nature and which may indeed be given in different units. Therefore, the choice of metric plays an intimate role in the performance of the nearest neighbor classifier for a given sample size n. An optimal local linear distance measure is derived by Short and Fukunaga [253], while a corresponding optimal, global weighted Euclidean metric is derived by Fukunaga and Flick [117]. Snapp and Venkatesh [259] argue the merits of a weighted L^2 metric and show that for a family of smooth problems, the weighted Euclidean metric is optimal over all weighted L^p metrics of the form $\|A(x-x')\|_p$ where A is a nonsingular linear transformation and, for $x=(x_1,\ldots,x_d)\in\Re^d$,

$$\|x\|_p \stackrel{\text{def}}{=} \begin{cases} \sqrt[p]{|x_1|^p+\ldots+|x_d|^p} & \text{if } 1\le p<\infty, \\ \max_{1\le i\le d}|x_i| & \text{if } p=\infty, \end{cases}$$

is the usual L^p-norm. For the consistency of data-dependent choices of the transformation matrix A we refer to [90, Section 26.5].

A serious problem of nearest neighbor methods with the above mentioned metrics is that they are not scale-invariant, that is, the decision is sensitive to monotone transformations of the coordinate axes. Olshen [210] and Devroye [79] introduced a scale-invariant way of measuring empirical distances. Such a metric is of unquestionable importance in situations when the different components of the input vector represent incomparable quantities. The universal consistency of the resulting k-nearest neighbor classifier is proved by Devroye [79].

A very large literature has burgeoned on the nearest neighbor classifier dealing with diverse issues ranging from algorithmic innovations, error estimation, imperfect samples, editing experiments, and computational concerns. For recent reviews the reader is directed to [75] and [90].

3 Kernel Classifiers

Just like nearest neighbor rules, kernel rules classify an input point $X\in\Re^d$ according to a majority vote among the labels of the training points X_i in the vicinity of X. However, while the k-nearest neighbor rule bases the classification on the k training points that are closest to X, the simplest kernel rule considers all X_i's that are closer to X than some number

$h>0$. This simple rule is the so-called *moving window* classifier, and is formally defined by

$$g_n(X) = \begin{cases} 0 & \text{if } \sum_{i=1}^n I_{\{Y_i=0, X_i\in B(h,X)\}} \ge \sum_{i=1}^n I_{\{Y_i=1, X_i\in B(h,X)\}} \\ 1 & \text{otherwise,} \end{cases}$$

where, as before, $B(h,X)$ denotes the closed ball of radius h centered at X. In other words, the label assigned to X is 1 if and only if, among the training points within distance h to X, there are more points labeled by 1 than those labeled by 0.

Common sense suggests that training points very close to X should have a larger weight than those which are farther away. The moving window rule gives uniform weight to all points within distance h and zero weight to all other points. Alternatively, a smoother transition might be desirable. This suggests the general definition of a *kernel classification rule*:

$$g_n(X) = \begin{cases} 0 & \text{if } \sum_{i=1}^n I_{\{Y_i=0\}} K\left(\frac{X-X_i}{h}\right) \ge \sum_{i=1}^n I_{\{Y_i=1\}} K\left(\frac{X-X_i}{h}\right) \\ 1 & \text{otherwise,} \end{cases}$$

where $K:\Re^d\to\Re$ is a *kernel* function which is usually nonnegative and monotonically decreasing along rays starting from the origin. The positive number h is called the *smoothing factor*, or *bandwidth*. This is the most important parameter of a kernel rule. It determines the amount of "smoothing." If h is small, the rule gives large relative weight to points near X, and the decision is very "local," while for a large h many more points are considered with fairly large weight, and the decision is more stable. In choosing a value for h, one confronts the same kind of problem as the bias/variance trade-off in statistical estimation problems or the approximation error/estimation error conflict we see in Section 5.

The kernel rule with the special choice $K(x)=I_{\{x\in B(1,0)\}}$ is just the moving window rule. Other popular choices are listed below.

$$\begin{aligned} \text{Gaussian kernel:} &\quad K(x)=e^{-\|x\|^2}. \\ \text{Cauchy kernel:} &\quad K(x)=1/(1+\|x\|^{d+1}) \\ \text{Epanechnikov kernel:} &\quad K(x)=(1-\|x\|^2)I_{\{\|x\|\le 1\}}. \end{aligned}$$

One may increase the flexibility of the kernel rule by allowing more than one adjustable parameter. For example, one may classify according to the sign of

$$\sum_{i=1}^m (2Y_i-1)\,K_\theta(X-X_i),$$

where K_θ is a product kernel of the form

$$K_\theta(x)=\prod_{j=1}^d K\left(\frac{x^{(j)}}{h^{(j)}}\right),$$

where $\theta=(h^{(1)},\ldots,h^{(d)})$ is a vector of smoothing factors, $x^{(j)}$ denotes the jth component of the vector x, and K is a fixed one-dimensional kernel. Here a different smoothing factor may be chosen along each coordinate.

The kernel classification rule has its origin in the kernel density estimates of Parzen [213], Rosenblatt [235], and Akaike [5], in the analogous regression estimators of Nadaraya [203, 204], and Watson [294], and in the *potential function rules* of Bashkirov, Braverman and Muchnik [33], Aizerman, Braverman and Rozonoer [1, 2, 3, 4], Braverman [46], and Braverman and Pyatniskii [47]. For a more extensive bibliography we refer to Devroye, Györfi, and Lugosi [90].

Kernel rules are at the basis of many popular classifiers.

EXAMPLE 3: *Radial Basis Function Classifiers.*

These classifiers have a "neural" flavor and base their decision upon the sign of functions of the form

$$\sum_{i=1}^{k} a_i K\left(\frac{X - x_i}{h_i}\right)$$

where K is a kernel function (such as $K(u) = e^{-\|u\|^2}$ or $K(u) = 1/(1+\|u\|^2)$), k is an integer, and the constants $a_1, \ldots, a_k$, $h_1, \ldots, h_k \in \Re$, and $x_1, \ldots, x_k \in \Re^d$ are determined based on the training data. Taking $k = n$, $a_i = 2Y_i - 1$, $h_i = h$, and $x_i = X_i$, we obtain the standard kernel classifier. However, typical radial basis function classifiers choose $k \ll n$, and tune the rest of the parameters according to some criterion. For example, the x_i's may be chosen as cluster centers after grouping the data points into k concentrated groups. These methods are closely related to *neural network classifiers* discussed in Section 6 (see, for example, Powell [224], Broomhead and Lowe [51], Moody and Darken [201], Poggio and Girosi [220], and Krzyżak, Linder, and Lugosi [172]). Sometimes an even more general function is used of the form

$$\sum_{i=1}^{k} c_i K\left((X - x_i)^T A_i (X - x_i)\right) + c_0,$$

where the A_i's are tunable $d \times d$ matrices, and $c_0, \ldots, c_k \in \Re$, and $x_1, \ldots, x_k \in \Re^d$ may also be set based on the data.

EXAMPLE 4: *Polynomial Discriminant Function Classifiers.*

The *polynomial discriminant functions* of Specht [261, 262] are also derived from kernel rules. Specht [261] suggests applying a polynomial expansion to the kernel $K\left(\frac{x-y}{h}\right)$. This leads to a classifier based on the sign of the discriminant function

$$\sum_{i=1}^{n}(2Y_i - 1)\left(\sum_{j=1}^{k} \varphi_j(X_i)\varphi_j(X)\right),$$

where $\varphi_1, \ldots, \varphi_k$ are fixed real-valued functions on $\Re^d$. When these functions are polynomials, the corresponding classifier g_n is called a *polynomial discriminant function*. The discriminant function obtained this way may be written in the simple form

$$\sum_{j=1}^{k} w_{n,j}\varphi_j(X),$$

where the coefficients $w_{n,j}$ are given by

$$w_{n,j} = \sum_{i=1}^{n}(2Y_i - 1)\varphi_j(X_i).$$

This rule has computational advantages over the standard kernel rule as in many applications the data may be processed before seeing the observation X to be classified. The coefficients $w_{n,j}$ depend on the data D_n only, so classifying X amounts to computing the values of $\varphi_1(X), \ldots, \varphi_k(X)$, which may be done quickly since typically $k \ll n$. Specht's idea of expanding K reappeared recently in Vapnik's [275] popular *support vector machines* (see also Section 7). Such general rules are also examined in some more detail in Section 6 where we revisit polynomial discriminant functions from a neural network vantage point.

The principal theoretical question of import for kernel rules is whether there is a choice of the kernel function K and the smoothing factor h that leads to a universally consistent classification rule. Indeed, such a consistency result may be deduced from Stone's [267] general theorem for local averaging regression estimates. For example, Devroye and Wagner [96] and Spiegelman and Sacks [263] prove that under certain regularity conditions on the kernel K, $\lim_{n\to\infty} \mathbf{E}L(g_n) = L^*$, whenever $h = h_n$ depends on the sample size n is such a way that

$$\lim_{n\to\infty} h_n = 0 \quad \text{and} \quad \lim_{n\to\infty} nh_n^d = \infty.$$

The first condition guarantees the local nature of the decision (i.e., small bias), while the second condition is necessary to control the statistical variation (small variance). For stronger versions, and related results, see Devroye and Krzyżak [91], Devroye and Györfi [88], Zhao [302], Krzyżak [170], Greblicki, Krzyżak, and Pawlak [133], Krzyżak and Pawlak [173], Greblicki and Pawlak [134], Wolverton and Wagner [299], and Devroye, Györfi, and Lugosi [90].

While the consistency of kernel rules is reassuring, for successful applications, much more is needed. For example, in high-dimensional spaces only very large values of h guarantee that sufficiently many data points are taken into account at the decision to compensate for statistical variation, but then the local nature of the decision is lost. Selecting a nearly optimal value of h in a data-dependent manner is a nontrivial task as the optimal smoothing factor is a complicated function of the unknown distribution and the sample size. This problem and its counterparts in density and regression estimation have driven much of the research in nonparametric statistics in the last two decades. (For a recent survey of the density estimation problem see Devroye [86].) Devroye, Györfi, and Lugosi [90, Chapter 25] argue that the smoothing problem of kernel classification is essentially different from its more thoroughly studied analogues in density estimation and regression, though it may be useful to adapt some heuristics from the latter problems to the classification problem. In [90, Chapter 25] the data-splitting method—discussed in Section 8—is explored. Its success depends on the, appropriately defined, *complexity* of the kernel function. For an extensive discussion of the problem we refer to [90]. We review some general principles of automatic parameter selection in Section 8.

4 Histograms and Classification Trees

4.1 Histogram Classifiers

Histogram classifiers partition the feature space, and the classification of each point is made according to a majority vote within the cell of the partition in which the point falls. This simple idea has led to a huge variety of classification methods, depending on the way the space is partitioned.

The simplest of these methods is the *regular histogram rule*, which, independently of the data, partitions $\Re^d$ into congruent cubes of size h. In spite of its simplicity, this rule already has some nice theoretical properties: if $h = h_n$ decreases with the sample size such that

$$h_n \to 0 \quad \text{and} \quad nh_n^d \to \infty \quad \text{as} \quad n \to \infty,$$

then the regular histogram rule is strongly universally consistent (see [87], [90]). The conditions on h express the simple requirements that (i) each cell of the partition should be small enough so that the decision is "local," that is, the optimal decision may be well-approximated, and (ii) the cells should be large enough so that they contain sufficiently many points to "average out" statistical variation.

However, it is clear that even for moderately large dimensions the regular histogram rule has little practical use. For example, when $d = 10$, and X is distributed in $[0,1]^{10}$, the choice $h = 1/4$ already generates over a million cells! Thus, for "practical" sample sizes the intuitive requirements (i) and (ii) are far from being satisfied. This is further evidence of the *curse of dimensionality*.

To make partitioning classifiers worth considering for practical use, it is essential to let the partition depend on the data. By looking at the data, one may determine the regions where larger cells may be effective, and those where a finer partitioning is called for. Such methods have been proposed and studied from the birth of pattern recognition. For early work see, for example, Sebestyen [247], Morgan and Sonquist [202], Anderson [11], Patrick [214], Patrick and Fisher [215], Quesenberry and Gessaman [226], Anderson and Benning [12], Beakley and Tuteur [38], Gessaman and Gessaman [126], Henrichon and Fu [145], Meisel and Michalopoulos [195] and Friedman [114].

One may obtain obtain significantly improved classifiers even by constructing partitions by looking only at the locations of the data points $X_1, \ldots, X_n$ and ignoring the labels $Y_1, \ldots, Y_n$. One such class of partitions is based on *statistically equivalent blocks*, where (about) the same number of data points are forced into each cell of the partition. See, for example, [11], [214], [215], [226], [125], [126], [74], [90].

Universal consistency has been proved under various conditions on the partition generated by the data. The first such results were derived by Gordon and Olshen [130], [131], [132], followed by Breiman, Friedman, Olshen, and Stone [50], Chen and Zhao [57], Zhao, Krishnaiah, and Chen [303], Lugosi and Nobel [183], Nobel [208, 209], and Devroye, Györfi, and Lugosi [90].

4.2 Classification Trees

The most important family of partitioning classifiers are the so-called *binary tree classifiers*. Here the partition is built recursively, starting from $\Re^d$, splitting at each step a cell of the current partition. Such partitions may be conveniently represented by binary trees, where each node represents a set in the space $\Re^d$ and has exactly two or zero children. If a node u represents the set A and its children u', u'' represent A' and A'', then we require that $A = A' \cup A''$ and $A' \cap A'' = \emptyset$. In other words, a node A is split into the two sets A', A'' which are represented by its children in the tree. The root represents $\Re^d$, and the leaves, taken together, form a partition of $\Re^d$.

Typically, every split should represent a simple question such as: "Is the ith component of X less than a?" Trees which are entirely built from splits of this type are called *ordinary binary classification trees*. Such trees have some apparent advantages:

- Once the tree is built, the classification of an input point can be calculated easily.
- The classifier is transparent and easy to interpret.
- The trees are usually invariant under monotone transformations of the coordinate axes, that is, for any transformation $T: \Re^d \to \Re^d$ which maps every coordinate separately by a strictly increasing function, the classification remains unchanged if all data points are transformed, that is,

$$g_n(T(X); T(X_1), Y_1, \ldots, T(X_n), Y_n) = g_n(X; X_1, Y_1, \ldots, X_n, Y_n).$$

Invariant classifiers are extremely important in practice as the user does not have to worry about measurement units: it does not make a difference whether the weight and the volume of an object are measured in kilograms and liters or in pounds and the logarithm of cubic inches.

Of course, splits perpendicular to the coordinate axes are not the only possibility. Another popular choice is to split cells by hyperplanes. Using computer science terminology, such trees may be called *binary space partition (BSP) trees*. BSP trees were recommended for use in pattern recognition by Henrichon and Fu [145], Mizoguchi, Kizawa, and Shimura [199] and Friedman [114]; see also Sklansky and Michelotti [257], Argentiero, Chin, and Beaudet [17], Qing-Yun and Fu [227], Breiman, Friedman, Olshen, and Stone [50], Loh and Vanichsetakul [181], and Park and Sklansky [211]. In the sequel we restrict our discussion to ordinary binary trees and refer the interested reader to [90] for a partial review of BSP tree classifiers.

The fundamental problem is, of course, how to "grow a tree," that is, how to determine the tree partition based on the training data. Consider partitions that depend only on the values $X_1, \ldots, X_n$, with the labels $Y_1, \ldots, Y_n$ used only to determine the majority vote in each cell. Begin with a tree in which we split every node perfectly, that is, if there are m points in a cell, we find the median according to one coordinate, and split the cell into two cells of sizes $\lfloor (m-1)/2 \rfloor$ and $\lceil (m-1)/2 \rceil$. Repeat this for k levels of nodes, at each level cutting along the next coordinate axis in a rotational manner. This leads to 2^k leaf regions, each having at least $n/2^k - k$ points and at most $n/2^k$ points. It is easy to see that the resulting tree is *balanced* in the sense that its height is k. In [90] such a tree is called a *median tree*, and its consistency is proved provided X has a density, and the number of levels $k = k_n$ satisfies $n/(k_n 2^{k_n}) \to \infty$ and $k_n \to \infty$ as $n \to \infty$.

Nobel [209] considers a tree-growing procedure, where, in each step, a cell is split such that the sum of the empirical *distortions*

$$\sum_{i: X_i \in A} \|X_i - c_A\|^2$$

is minimized, where the sum is taken over all cells A, and c_A denotes the empirical *centroid* of a cell defined by

$$c_A = \frac{\sum_{i: X_i \in A} X_i}{\sum_{i: X_i \in A} (1/n)}.$$

Nobel [209] proves that such a greedy growing procedure, equipped with an appropriate stopping rule, leads to a consistent classification rule under mild conditions. Such trees are also useful in *tree structured vector quantization* (see the references therein).

Other trees which solely depend on the X_i values include various versions of d-dimensional binary search trees. For a survey and several consistency results we refer to [90].

An obvious criticism that may be levelled against trees of the above kind is that discarding label information during the tree building phase is inefficient. For example, there is no reason to split a large rectangle which contains only label 0 data points. In principle, one might search exhaustively among all binary trees with (say) k nodes and choose one which minimizes the empirical error. The performance of such a classifier may be estimated by bounding the VC growth function which is discussed in Section 5. Indeed, such bounds are easy to prove; see [90] for examples.

4.3 Splitting Criteria

Unfortunately, an exhaustive search over all trees is clearly computationally prohibitive. The popularity and importance of binary tree classifiers lies precisely in the fact that one can hold out hope to obtain good classifiers by computationally inexpensive search. The basic principle is *greedy growing*, which means that the tree is built in a recursive manner, at each stage determining the next split by optimizing some criterion based on the data.

The splitting criterion that first comes into mind is splitting to minimize the number of errors on the training data. This criterion has its root in the classical work of Stoller [265] who analyzed a single-split rule in a univariate setting. Trees built based on this criterion were studied by Payne and Meisel [216], Rounds [236], and Gordon and Olshen [131]. It is pointed out in the last paper (and emphasized again in [50]) that the minimum-error criterion has serious drawbacks and cannot possibly lead to a universally consistent classifier. Nevertheless, Gordon and Olshen gave modifications with some built-in safeguards which are sufficient for consistency. (For example, they force split sets to contain a certain minimal number of data points and they also force splits along each coordinate axis.) These modifications are somewhat unnatural and many other splitting criteria have since been introduced, with a predecessor being the so-called AID criterion of Morgan and Sonquist [202].

Breiman, Friedman, Olshen, and Stone [50] consider a class of splitting criteria based on *impurity functions*. These criteria may be described as follows: let the function $\phi: [0,1] \to [0,\infty)$ be symmetric around $1/2$ with $\phi(0) = \phi(1) = 0$. Also assume that ϕ is increasing in $[0, 1/2]$, and takes its maximum at $1/2$. Such a function ϕ is called an impurity function. The impurity of a cell A of a partition containing N_0 training points with label 0 and N_1 training points with label 1 is defined as

$$I_A = N\phi\left(\frac{N_0}{N}\right),$$

where $N = N_0 + N_1$ is the total number of training points in A. The total impurity of a partition is the sum of the impurities corresponding to all its cells. At every step of the tree-growing procedure, a cut is made to minimize the total impurity of the resulting partition. In other words, a cell A and its cut into A' and A'' are searched so that the difference $I_A - (I_{A'} + I_{A''})$ is maximal. Popular examples of such impurity functions include:

(1) The *binary entropy function* $\phi(p) = -p \log p - (1-p)\log(1-p)$;

(2) The *Gini function* $\phi(p) = 2p(1-p)$, leading to the Gini index of diversity advocated in [50];

(3) The *probability of misclassification* $\phi(p, 1-p) = \min(p, 1-p)$. In this case the splitting criterion is just the minimum-error criterion discussed earlier.

The number of different versions is endless, depending on the choice of the impurity function, and the method chosen to terminate the growing procedure.

Trees built by minimizing such criteria have been thoroughly studied, see, Breiman, Friedman, Olshen, and Stone [50], Gelfand and Delp [122], Guo and Gelfand [135] Gelfand, Ravishankar, and Delp [123], [124], and Ciampi [59]), Michel-Briand and Milhaud [196], Talmon [270], Sethi and Sarvarayudu [249], Wang and Suen [292], Goodman and Smyth [129], Chou [58], Li and Dubes [179], Quinlan [228], Sethi [248], Burshtein, Della Pietra, Kanevsky, and Nádas [55], among others.

To understand the danger of impurity-based growing algorithms, consider the Gini criterion, which determines a split by minimizing the sum of

$$\frac{N_0 N_1}{N_0 + N_1}$$

for every cell A. Since this quantity only depends on the *ratio* of the label 0 and label 1 points in each cell: cells containing very few points will be preferentially split as improvement is much easier to achieve there. To remedy this anomaly, in their procedure CART, Breiman, Friedman, Olshen, and Stone [50] suggest a *growing-and-pruning* algorithm, where they let the tree grow until every cell contains just a few points, and then they use a clever algorithm to "prune" it to determine the best subtree.

In Devroye, Györfi, and Lugosi [90, Chapter 20] it is shown that with an easy modification of the Gini criterion one can get good performance even without additional pruning. If, instead of the Gini criterion, the sum of the N_0N_1 is minimized (i.e., the normalization by the total number of points in a cell is dropped), then all large inhomogeneous cells will be split. Indeed, they show that if the tree is based on this criterion such that splits are rotated along all coordinate axes, a simple stopping criterion suffices to achieve universal consistency of the resulting tree classifier.

Still, all the above methods are somewhat intuitively unappealing as the impurity functions (except, of course, the probability of misclassification) are not directly connected to the empirical probability of error, and artificial devices such as pruning and forced rotated splits have to be added.

There is no reason for pessimism, however. An "almost" greedy minimization of misclassification training error still leads to a good classifier. All one has to do is to "look ahead" $2d$ steps in the search (recall that d is the dimensionality of the feature space) and find, at each step, the cell with the best $2d$ splits which minimizes the training error of the resulting classifier. As shown in [90, Theorem 20.9], if tree growth is stopped after $k = k_n$ splits, then the tree classifier is universally consistent provided that $k_n \to \infty$ and $k_n = o\left(\sqrt{n/\log n}\right)$.

5 Vapnik-Chervonenkis Theory

One common approach in pattern classification is to restrict the form of the classifier to belong to some class, $\mathcal{C}$, of decision rules. Such a restriction might be used as a result of prior knowledge about a problem domain, introduced for purposes of efficiency and simplicity, or might arise implicitly as a result of the learning algorithm and architecture selected. For example, in neural networks, fixing the architecture and size of the network imposes a restriction on the class of decision rules that can be implemented by the network—namely, those rules computed by the network over all choices of the weights. A basic question of interest is to understand how learning performance depends on the class of rules $\mathcal{C}$.

Of course, with a restriction on the classifier, in general we cannot hope to perform as well as the optimal Bayes decision rule, even with a very large number of samples. Hence, we should attempt only to try to find the best rule from within the class $\mathcal{C}$. Moreover, with a finite amount of random data, we cannot hope to always find the optimal rule from C. Therefore, for finite sample sizes it is natural to strive only to find some near-optimal classifier, and only require that we succeed in some probabilistic sense. Much early and fundamental work related to this approach to pattern classification was done in the probability and statistics literature—c.f. Vapnik and Chervonenkis [276, 278, 274], Dudley [99], and Pollard [221]. The paper of Valiant [273] spurred recent work in the computer science community in this area as well as introduced the terminology PAC learning, the acronym standing for *probably-approximately-correct learning* (c.f. Haussler [140] and references therein). Details on various aspects of this approach can be found in a number of books; c.f. [13, 164, 90, 274, 275, 285].

5.1 Formulation

Let $\mathcal{C}$ be a collection of mappings $\phi: \Re^d \to \{0,1\}$ and let $L^*_{\mathcal{C}} = \inf_{\phi \in \mathcal{C}} L(\phi)$. Then, given the data $D_n = ((X_1,Y_1),\ldots,(X_n,Y_n))$, we seek a rule $\phi \in C$ such that $L(\phi) \approx L^*_{\mathcal{C}}$. Unfortunately, the performance $L(\phi)$ of a rule ϕ is unknown, in general, and the only information available to assess the performance of ϕ is the data D_n. Thus, a natural approach is to select a rule from the class $\mathcal{C}$ that looks best on the data. Namely, for a given ϕ, let the *empirical error probability* of ϕ be defined as

$$\hat{L}_n(\phi) = \frac{1}{n}\sum_{i=1}^{n} I_{\{\phi(X_i) \neq Y_i\}},$$

i.e., $\hat{L}_n(\phi)$ is the fraction of examples labelled incorrectly by the *hypothesis*[2] $\phi \in \mathcal{C}$, or, more succinctly, the relative frequency of errors by ϕ on the sample D_n. Let

$$\phi^*_n = \arg\min_{\phi \in \mathcal{C}} \hat{L}_n(\phi).$$

Then ϕ^*_n is a rule in $\mathcal{C}$ that minimizes the "empirical risk," and the hope is that such a rule will in fact also be close to minimizing the actual risk.

To gain some insight into this problem, recall that the Bayes error L^* is the absolute best one could hope for, while $L^*_{\mathcal{C}}$ is the best one could hope for using rules from $\mathcal{C}$. We can write how much worse ϕ^*_n is than the Bayes classifier as follows:

$$L(\phi^*_n) - L^* = (L(\phi^*_n) - L^*_{\mathcal{C}}) + (L^*_{\mathcal{C}} - L^*). \tag{5}$$

The second term $L^*_{\mathcal{C}} - L^*$ (often called the *approximation error*) represents how much we lose in performance by restricting our classifier to come from $\mathcal{C}$. That is, we have no reason to expect that $\mathcal{C}$ will contain the optimal Bayes classifier, and $L^*_{\mathcal{C}} - L^*$ represents the closest we can possibly approximate the performance of the optimal Bayes classifier using rules from $\mathcal{C}$. The first term $(L(\phi^*_n) - L^*_{\mathcal{C}})$ (often called the *estimation error*) represents how much we lose in performance by our lack of knowledge of the exact performance of the various rules from $\mathcal{C}$. That is, the only way we have to judge the performance of the various rules from $\mathcal{C}$ is to use the data, and $(L(\phi^*_n) - L^*_{\mathcal{C}})$ is the loss we incur by estimating the actual performance by the empirical performance.

There is an inherent tradeoff in the two error terms. For a very "rich" class of rules $\mathcal{C}$, we would expect the second term to be small, since we would have the representational power to closely approximate Bayes rule (whatever it might actually be). On the other hand, this approximation ability comes at the price of reducing our confidence in conclusions drawn from limited data. With a very rich collection of rules, there is a greater danger that some rule from $\mathcal{C}$ just happens to fit very well the particular data observed so far, but may be unlikely to fit new data.

[2]The terminology is from computational learning theory.

5.2 VC Dimension and Empirical Risk Minimization

Vapnik and Chervonenkis [276] introduced the following combinatorial measure of the "richness" of a class $\mathcal{C}$ that characterizes the behavior of the estimation error. First, observe that any classifier in $\mathcal{C}$ can also be identified with the subset of $\Re^d$ which the classifier maps to class 1. Thus, we may think of the class $\mathcal{C}$ as a collection of subsets of $\Re^d$. This interpretation of classifiers in $\mathcal{C}$ is used in the following definition.

Definition 1 Given a set of points $\Sigma = \{x_1, \ldots, x_n\} \subset \Re^d$, let $\Delta_{\mathcal{C}}(\Sigma)$ denote the number of subsets of Σ generated by intersection with $\mathcal{C}$, that is, the number of distinct sets of the form $\Sigma \cap \phi$ for $\phi \in \mathcal{C}$. The *n-th shatter coefficient* of $\mathcal{C}$ is defined as

$$s(\mathcal{C}, n) = \max_{\Sigma=\{x_1,\ldots,x_n\}\subset\Re^d} \Delta_{\mathcal{C}}(\Sigma).$$

The set $\Sigma = \{x_1, \ldots, x_n\}$ is said to be *shattered* by the class $\mathcal{C}$ if $\Delta_{\mathcal{C}}(\Sigma) = 2^n$. The *Vapnik-Chervonenkis (*VC*) dimension* of the class $\mathcal{C}$, denoted $V_{\mathcal{C}}$, is the largest integer n such that there exists a set of cardinality n that is shattered by $\mathcal{C}$. If there exist sets of arbitrarily large (integer) cardinality that are shattered by $\mathcal{C}$, then $\mathcal{C}$ is said to have infinite VC dimension.

The main result states that as long as the classifiers in $\mathcal{C}$ satisfy some mild measurability conditions, empirical risk minimization works in the sense that the estimation error $(L(\phi_n^*) - L_{\mathcal{C}}^*)$ (the first term in (5)) converges to zero uniformly in probability if and only if, $V_{\mathcal{C}} < \infty$. Moreover, rates of convergence can be obtained thereby giving bounds on the amount of data required for a given level of performance.

The key idea behind this result is a symmetrization argument used to prove a bound on the deviations of empirical probabilities of events from their true probabilities. Namely, given a class of events $\mathcal{A}$, Vapnik and Chervonenkis [276] showed in their landmark paper that the following explicit bound

$$\mathbf{P}\left\{\sup_{A\in\mathcal{A}} |\hat{P}_n(A) - P(A)| > \epsilon\right\} \le 8s(\mathcal{A}, n)e^{-n\epsilon^2/8} \tag{6}$$

holds for *every* distribution P, where $\hat{P}_n(A)$ is the empirical probability of event A on an n-sample drawn by independent sampling from the probability measure P. This type of result is also closely related to so-called "uniform laws of large numbers." Of course, the above inequality is useful only if the shatter coefficient grows at a subexponential rate. At first glance it is not clear how one can obtain manageable conditions for such subexponential growth. Luckily, a beautiful combinatorial result, known as the Vapnik-Chervonenkis lemma (also called Sauer's lemma in the computational learning literature), gives an easy characterization. The result, proved independently by Vapnik and Chervonenkis [276] and Sauer [242], states that for any class of sets $\mathcal{A}$,

$$s(\mathcal{A}, n) \le \sum_{j=0}^{V_{\mathcal{A}}} \binom{n}{j}. \tag{7}$$

It is easy to see that the right-hand side is bounded above by $1 + n^{V_{\mathcal{A}}}$ for all n and $V_{\mathcal{A}}$, while for $n \ge V_{\mathcal{A}}$ the sharper bound $s(\mathcal{A}, n) \le (ne/V_{\mathcal{A}})^{V_{\mathcal{A}}}$ can be derived with just a modicum of effort. It follows that the nth shatter coefficient $s(\mathcal{A}, n)$ has subexponential (polynomial) growth whenever $V_{\mathcal{A}} < \infty$ (while, of course, $s(\mathcal{A}, n) = 2^n$ for all n when $V_{\mathcal{A}} = \infty$). It follows that the right-hand side of (6) tends to zero as $n \to \infty$ for all distributions provided only that $V_{\mathcal{A}} < \infty$.

This elegant "distribution-free" result can be applied directly to the pattern recognition problem to get uniform bounds on the deviations of empirical error rates of a collection of classifiers from their true error rates, that is, bounds on $\sup_{\phi\in\mathcal{C}} |\hat{L}_n(\phi) - L(\phi)|$, which in turn imply the success of empirical risk minimization for a class $\mathcal{C}$ with finite VC dimension. To see how results on uniform convergence of the type (6) may be used in pattern recognition, define the *error of* ϕ to be the set $A_\phi = \{(x, y) : \phi(x) \ne y\}$, and identify the class of events $\mathcal{A}$ with the class of error events: $\mathcal{A} = \{A_\phi : \phi \in \mathcal{C}\}$. It is easy to argue that $V_{\mathcal{A}} = V_{\mathcal{C}}$ so that the Vapnik-Chervonenkis bound takes the form

$$\mathbf{P}\left\{\sup_{\phi\in\mathcal{C}} |\hat{L}_n(\phi) - L(\phi)| > \epsilon\right\} \le 8\left(\frac{ne}{V_{\mathcal{C}}}\right)^{V_{\mathcal{C}}} e^{-n\epsilon^2/8} \qquad (n \ge V_{\mathcal{C}}). \tag{8}$$

A little reflection now shows that

$$L(\phi_n^*) - L_{\mathcal{C}}^* \le 2\sup_{\phi\in\mathcal{C}} |\hat{L}_n(\phi) - L(\phi)|. \tag{9}$$

This simple inequality points out that if the empirical error $\hat{L}_n(\phi)$ is a good estimate of the true probability of error $L(\phi)$ *uniformly* for every $\phi \in \mathcal{C}$, then the probability of error of any classifier selected by minimizing the empirical error is close to the best possible error in the class. Now the Vapnik-Chervonenkis inequality (7′) may be used directly to obtain upper bounds for the estimation error $L(\phi_n^*) - L_{\mathcal{C}}^*$. For example, given any error parameter $\epsilon > 0$ and confidence parameter $\delta > 0$, no more than $(c'/\epsilon^2)(V_{\mathcal{C}} \log(1/\epsilon) + \log(1/\delta))$ training samples are required (for some positive constant c') to ensure that with probability at least $1 - \delta$ the estimation error will be less than ϵ. This formulation is popular in the PAC learning literature.

An essentially equivalent, somewhat simpler formulation results from a consideration of the expected probability of error $\mathbf{E}L(\phi_n^*)$ and in the sequel we focus on this measure. From (7′) and (8), one may easily deduce that for some universal constant c,

$$\mathbf{E}L(\phi_n^*) - L_{\mathcal{C}}^* \le c\sqrt{\frac{V_{\mathcal{C}} \log n}{n}}.$$

Thus, one of the main messages of the Vapnik-Chervonenkis inequality is that if the VC dimension of the class $\mathcal{C}$ is finite, then the estimation error converges to zero at the rate $O\left(\sqrt{V_{\mathcal{C}} \log n/n}\right)$ *for all distributions* of (X, Y). The beauty and power of this result lies in its distribution-free nature and in the fact that the properties of the class are reflected through the simple combinatorial parameter $V_{\mathcal{C}}$.

A question arising immediately relates to the tightness of the inequality (6). An immediate improvement may be effected via another inequality of Vapnik and Chervonenkis [277], who improved (6) to

$$\mathbf{P}\left\{\sup_{A\in\mathcal{A}}\frac{P(A)-\hat{P}_n(A)}{\sqrt{P(A)}}>\epsilon\right\}\leq 4s(\mathcal{A},2n)e^{-n\epsilon^2/4}. \tag{10}$$

(See Anthony and Shawe-Taylor [14] for a beautiful short proof.) The above inequality implies

$$\mathbf{E}L(\phi_n^*)-L_{\mathcal{C}}^*\leq c\max\left(\sqrt{\frac{L_{\mathcal{C}}^*V_{\mathcal{C}}\log n}{n}},\frac{V_{\mathcal{C}}\log n}{n}\right),$$

where c is a universal constant. This result points out an important phenomenon: if the smallest achievable error in the class $L_{\mathcal{C}}^*$ happens to be small, much smaller estimation errors are achievable than that predicted by the basic Vapnik-Chervonenkis inequality (6). For the special case $L_{\mathcal{C}}^*=0$ (which is a common assumption in the PAC learning literature) one may even improve the order of magnitude of the estimate (i.e., one obtains $O(V_{\mathcal{C}}\log n/n)$ instead of $O(\sqrt{V_{\mathcal{C}}\log n/n}\,)$). The tightness of the above upper bound is reflected by corresponding lower bounds. For example, Devroye and Lugosi [92] prove that for *any* classification rule g_n and for any class $\mathcal{C}$ there exists a distribution of (X,Y) with $\inf_{\phi\in\mathcal{C}}L(\phi)=L_{\mathcal{C}}^*$ such that

$$\mathbf{E}L(g_n)-L_{\mathcal{C}}^*\geq c'\max\left\{\sqrt{\frac{L_{\mathcal{C}}^*V_{\mathcal{C}}}{n}},\frac{V_{\mathcal{C}}}{n}\right\},$$

where c' is another universal constant. For related lower bounds see also Vapnik and Chervonenkis [277], Blumer, Ehrenfeucht, Haussler and Warmuth [44], Haussler, Littlestone, and Warmuth [141], Ehrenfeucht, Haussler, Kearns, and Valiant [107], Simon [255], [256], Schuurmans [246], Antos and Lugosi [15], and Devroye, Györfi, and Lugosi [90].

Thus, the upper bound achieved by (9) is optimal up to a logarithmic factor. Quite a lot of effort has been invested in closing this gap. It follows from a now classical result of Dudley that for some constant c,

$$\mathbf{E}L(\phi_n^*)-L_{\mathcal{C}}^*\leq c\sqrt{\frac{V_{\mathcal{C}}}{n}}$$

(see, e.g., Pollard [222], [223], Ledoux and Talagrand [177]). For the special case $L_{\mathcal{C}}^*=0$, a beautiful result of Haussler, Littlestone, and Warmuth [141] states that, even though in this case the $\log n$ factor is necessary if one considers an arbitrary ϕ_n^* minimizing the empirical risk, there exists a classifier g_n such that

$$\mathbf{E}L(g_n)\leq\frac{V_{\mathcal{C}}}{n}.$$

The inequalities (6) and (9) are hence extremely important from both theoretical and practical points of view. Several versions of the original inequalities have been derived since,

mostly improving on the constants. For lack of space, here we only give a list of references for such improvements: Devroye [84], Alexander [7], Massart [190], Shawe-Taylor, Anthony, and Biggs [250], Parrondo and Van den Broeck [212], Lugosi [182], and Talagrand [269].

Many interesting examples of classes with finite VC dimension are known. (See for example [65, 276, 95, 254, 264, 295] among others). Below, in Section 6, we discuss some important examples. Also some general results on VC dimension bounds have been obtained. Among the most interesting of such bounds are those that relate the VC dimension of a class $\mathcal{C}$ to bounds on the covering numbers of $\mathcal{C}$ with respect to metrics induced by probability measures. Covering numbers (and the closely related notion of metric entropy) were introduced by Kolmogorov and Tihomirov [167] and arise naturally in a number of problems in approximation, statistics, and information theory. In the present context, given a probability measure P on $\Re^d$, a pseudometric on $\mathcal{C}$ (and, actually, on the set of all measurable subsets of $\Re^d$) can be defined by $d_P(\phi_1,\phi_2)=\mathbf{E}|\phi_1(X)-\phi_2(X)|$. The covering number $N(\epsilon,\mathcal{C},d_P)$ is defined as the smallest number of ϵ-balls required to cover $\mathcal{C}$. The known results typically provide upper and lower bounds on $\sup_P N(\epsilon,\mathcal{C},d_P)$ in terms of the VC dimension of $\mathcal{C}$ (e.g., some known results are summarized in [174]). The upper bounds are the deeper results and the fundamental result along this line was obtained by Dudley [99], which was subsequently refined by Pollard [221] and more recently by Haussler [140]. These bounds imply that under some weak measurability conditions, $\sup_P N(\epsilon,\mathcal{C},d_P)<\infty$ for all $\epsilon>0$ if and only if $V_{\mathcal{C}}<\infty$ where the sup is taken over all distributions of X. Covering numbers enter naturally in learning problems when one replaces an infinite class of rules $\mathcal{C}$ by a carefully selected finite subset of rules, and then attempts to use empirical risk minimization (or other learning rule) over this finite sub-collection. Work along these lines can be found in [40, 53, 102, 155, 175, 274, 285].

5.3 Complexity Regularization

We end this section with a discussion of a subject that goes by various names such as structural risk minimization [274, 275] and complexity regularization [22, 185]. The basic idea is to select a classifier from a collection of classifiers (or *models*) by trading off the misfit of the classifier on the observed data with the complexity of the classifier measured in an appropriate way. Such approaches are also closely related to ideas such as Ockham's razor, Rissanen's Minimum Description Length (MDL) principle [232], Akaike's information criterion (AIC) [6], and other complexity-based methods (e.g., as studied by Barron and Cover [22, 30]). To understand the motivation of these methods, consider the approximation error/estimation error tradeoff in (5) again: as discussed earlier, a rich class $\mathcal{C}$ favors the approximation error term $L_{\mathcal{C}}^*-L^*$ at the expense of the estimation error term $L(\phi_n^*)-L_{\mathcal{C}}^*$, while for a fixed class $\mathcal{C}$, the approximation error may be unsatisfactorily large (say close to 1/2). If we are not content with this situation, an alternative might be to consider a richer class of rules to improve the approximation error as long as the estimation error remains satisfactory.

One approach to carry out this tradeoff is to consider a hierarchy of classes $\mathcal{C}^{(1)},\mathcal{C}^{(2)},\ldots$ such that $V_{\mathcal{C}^{(j)}}<\infty$ for each j, but with $V_{\mathcal{C}^{(j)}}\to\infty$ as $j\to\infty$. Then we can select a rule from one of the classes $\mathcal{C}^{(j)}$ where the choice of j can, in general, depend on both how much data we have as well as the specific observations D_n. The idea is roughly that with only a small amount of data we should restrict ourselves to selecting a "simple" rule (i.e., one from a class with small j), but when there is a wealth of data we may be justified in selecting a rule from a richer class (i.e., one with large j). Now, if the classes $\mathcal{C}^{(j)}$ are chosen so that for any distribution of (X, Y), the corresponding Bayes decision rule can be approximated arbitrarily well using rules from the classes $\mathcal{C}^{(j)}$ (i.e., so that $\lim_{i\to\infty} L^*_{\mathcal{C}^{(i)}} = L^*$), then one might hope to be able to get consistency via the tradeoff discussed above.

It turns out that this can be done rather simply as follows. Assume, as before, that $V_{\mathcal{C}^{(j)}} < \infty$ for each j, and that for any distribution of (X, Y) we have $\lim_{i\to\infty} L^*_{\mathcal{C}^{(i)}} = L^*$. For each n, we select an integer k_n, and select a classifier ϕ^*_n that minimizes empirical risk over the class $\mathcal{C}^{(k_n)}$. Then, if the sequence $\{k_n\}$ satisfies the properties $k_n \to \infty$ and $V_{\mathcal{C}^{(k_n)}} \log(n)/n \to 0$ as $n \to \infty$, the classifier ϕ^*_n is strongly universally consistent.

Thus, k_n selects the complexity of the class from which we select the decision rule. The conditions on k_n guarantee that both the approximation and estimation errors will converge to zero. Once the classes $\mathcal{C}^{(j)}$ are fixed (and, hence, $V_{\mathcal{C}^{(j)}}$ is known for each j) then a suitable sequence $\{k_n\}$ can be fixed independently of the data so as to guarantee strong consistency. However, as one might expect, fixing the sequence $\{k_n\}$ in advance may not give the best tradeoff of the estimation and approximation errors for all distributions. It is possible to obtain much better results in general through a data-dependent choice of k_n. Such an approach was studied by Vapnik [274] using the term "structural risk minimization." A version using the term "complexity regularization" was recently provided by Lugosi and Zeger [185] and can be summarized as follows. Given the data D_n, for each j select a decision rule $\phi_{n,j}$ from the class $\mathcal{C}^{(j)}$ that minimizes the empirical risk and define the complexity penalty

$$r(n, j) = \sqrt{\frac{32}{n} V_{\mathcal{C}^{(j)}} \log(en)}.$$

Let ϕ^*_n be that classifier among the $\phi_{n,j}$ that minimizes the sum

$$\tilde{L}_n(\phi_{n,j}) = \hat{L}_n(\phi_{n,j}) + r(n, j).$$

For this decision rule and with the previous conditions on the $\mathcal{C}^{(j)}$, it can be shown (e.g., see [185]) that for any distribution of (X, Y) we have

$$\mathbf{E}L(\phi^*_n) - L^* \le \inf_{j\ge 1} \left(c\sqrt{\frac{V_{\mathcal{C}^{(j)}} \log n}{n}} + (L^*_{\mathcal{C}^{(j)}} - L^*) \right),$$

where c is some universal constant. In other words, the method of structural risk minimization automatically finds the optimal balance between the approximation error $L^*_{\mathcal{C}^{(j)}} - L^*$ and a tight upper bound of the estimation error. Thus, even though the optimal value of j depends heavily on the unknown distribution, it can be learned from the data. As mentioned above, and as can be seen by the form of $\tilde{L}_n(\phi_{n,j})$, this approach is closely related to other complexity-based methods all of which utilize a misfit versus complexity trade-off (the first

term of $\tilde{L}_n(\phi_{n,j})$ is the misfit of the hypothesis with the data, while the second term is the complexity of the hypothesis). Related results and refinements may be found in the work of Barron, Birgé and Massart [28], Kearns et al [162], Krzyżak and Linder [171], Meir [194], Modha and Masry [200], Shawe-Taylor et al [252], and Yang and Barron [300].

6 Neural Networks

The vast literature on linear discriminant functions in pattern classification begins with a classical paper of Fisher [110]. These functions have seen more attention (perhaps undeservedly) than any other largely because they are so amenable to analysis. However, they are severely limited in the class of problems to which they can be applied. The extension of the linear discriminant ideas to more flexible, complex, non-linear "neural" forms had its antecedents in a groundbreaking article in 1943 [191] in which McCulloch and Pitts published the first mathematical codification of aspects of brain function. In their article they outlined how simple mathematical assemblies of formal neurons, modelled very crudely on biological cells and assemblies, could be used to define a basis for a logical calculus of computation. A rich variety of "neural" computational paradigms which have proved effective in a variety of computation and decision problems have evolved in the half-century following McCulloch and Pitts' paper. (For a non-exhaustive chronology and surveys see [8, 10, 16, 29, 27, 142, 147, 165, 198, 230, 231, 234, 240] and the references contained therein.) In the standard setting, an (artificial) neural network is an assembly of formal computational elements operating collectively on a discrete time scale[3] and interconnected in various ways with the outputs of some elements functioning as inputs to others. If there is no feedback (i.e., the network is *acyclic*), given a set of external inputs, computation can be viewed as flowing through the network resulting ultimately in the computation of a given function of the inputs at those computational elements tagged as "output" units. The nature of the computation that is performed is dictated by the elementary computational unit that is used in the network and the graph of interconnections that describes the network architecture.

6.1 Neuron and Network Models

6.1.1 Formal Neurons

In the classical model of McCulloch and Pitts a formal neuron is a *linear threshold element*, i.e., a nonlinear computational element which computes a binary decision based on the sign of a linear form of its inputs. More specifically, a generic linear threshold element accepts d real inputs $x_1, \ldots, x_d$ and produces as output a binary value (which we hereafter identify

[3] Neurodynamical system models with system states evolving over continuous time have also been investigated in diverse settings where it is important to track the temporal evolution of a system. See, for instance, [9, 60, 148, 152, 217, 218].

as ± 1 for definiteness) according to the threshold rule

$$y = \mathrm{sgn}\Big(\sum_{i=1}^{d} w_i x_i - w_0\Big) = \begin{cases} -1 & \text{if } \sum_{i=1}^{d} w_i x_i < w_0, \\ +1 & \text{if } \sum_{i=1}^{d} w_i x_i \geq w_0. \end{cases}$$

The linear threshold element is completely specified by the real parameters $w_1, \ldots, w_d$

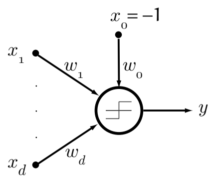

Figure 1: Linear threshold element.

called the *weights*, and the real value w_0 called the *threshold*. Observe that the threshold can be subsumed within the sum by the simple expedient of adding a fixed extra input, say $x_0 = -1$, as indicated schematically in Figure 1. It will be occasionally notationally convenient to hence pretend that the threshold is identically zero, whence $y = \mathrm{sgn}(\sum_i w_i x_i)$.

While this simple computational model has much to commend it, as we will see in the denouement, for the moment note just that the model incorporates a linear accumulation of inputs that allows a systematic integration of input information and a nonlinear comparison or thresholding operation which provides the critical nonlinear decision making or logic capability. Observe that, viewed as a pattern classification device, a linear threshold element dichotomizes d-dimensional feature space $\Re^d$ into two half-spaces separated by the hyperplane $\sum_i w_i x_i = 0$. The class of linear threshold functions (or *linear discriminant functions*) is exactly the class of decision functions from $\Re^d$ into $\{-1, +1\}$ that can be computed by such a device by varying the real weights w_i.

A natural extension of the basic linear threshold computational element computes the sign of a polynomial form of the inputs. More specifically, let

$$\mathcal{I}_D = \{\, I = (i_1, \ldots, i_D) : i_j \in \{0, 1, \ldots, d\}, j = 1, \ldots, D \,\}$$

denote the family of ordered multi-sets of cardinality D of the set $\{0, 1, \ldots, d\}$. Observe that $|\mathcal{I}_D| = (d+1)^D$. For each multi-set $I = (i_1, \ldots, i_D)$ in $\mathcal{I}_D$, define the polynomial map $\varphi_I \colon \Re^d \to \Re$ by

$$\varphi_I(x) \stackrel{\text{def}}{=} x_I = \prod_{j=1}^{D} x_{i_j} \qquad x = (x_0, x_1, \ldots, x_d) \in \{-1\} \times \Re^d.$$

A *polynomial threshold element of degree D* is specified by a set of weights w_I ($I \in \mathcal{I}_D$) and maps $\Re^d$ into $\{-1, +1\}$ according to the rule

$$y = \mathrm{sgn}\Big(\sum_{I \in \mathcal{I}_D} w_I x_I\Big) = \mathrm{sgn}\Big(\sum_{I \in \mathcal{I}_D} w_I \varphi_I(x)\Big).$$

For instance, when $D = 1$ we obtain hyperplane separating surfaces (recall $x_0 = -1$ subsumes the threshold) and when $D = 2$ we obtain quadrics (hyperhyperboloids) as separating surfaces. For general D, the separating surfaces are Dth-order rational varieties (cf. [52, 65, 283]).

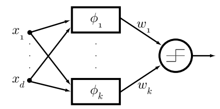

Figure 2: Canonical threshold element.

More generally, let $\varphi_1, \ldots, \varphi_k$ be a fixed set of measurement functions mapping $\Re^d$ into $\Re$ and write $\boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_k)$ for the corresponding vector of measurement functions. A canonical neuron (or *canonical threshold element*) is specified by a set of k weights w_j ($1 \leq j \leq k$) and the set of measurement function $\boldsymbol{\varphi}$ and computes the $\boldsymbol{\varphi}$-threshold function from $\Re^d$ into $\{-1, +1\}$ specified by

$$y = \mathrm{sgn}\Big\{\sum_{j=1}^{k} w_j \varphi_j(x_1, \ldots, x_d)\Big\}.$$

Such functions are also called $\boldsymbol{\varphi}$-functions in the literature (cf. [65, 207]).

An extension of the neural model in another direction is obtained by replacing the threshold function by some fixed *activation function* $\sigma \colon \Re \to \Re$. The formal neuron thus obtained computes a mapping from $\Re^d$ into $\Re$ of the form

$$y = \sigma\Big(\sum_{i=0}^{d} w_i x_i\Big).$$

The most popular activation functions in practice are the *sigmoids* which are measurable functions defined most generally by the property

$$\sigma(t) \to \begin{cases} -1 & \text{if } t \to -\infty, \\ +1 & \text{if } t \to +\infty. \end{cases}$$

(The limiting values ± 1 are not critical—0 and 1, for instance, would do just as well—; we select these limits to keep consistency with the definition of the sgn function.) The following are admissible choices for sigmoids, encountered most frequently in practice.

Threshold activation: $\sigma(t) = \mathrm{sgn}(t)$.

Piecewise linear activation:
$$\sigma(t) = \begin{cases} -1 & \text{if } t \leq -1, \\ t & \text{if } -1 < t < +1, \\ +1 & \text{if } t \geq +1. \end{cases} \tag{11}$$

Logistic activation:
$$\sigma(t) = \tanh\left(\tfrac{t}{2}\right) = \frac{1 - \exp(-t)}{1 + \exp(-t)}. \tag{12}$$

For any sigmoid σ, observe that $\sigma(\frac{t}{T})$ converges pointwise to $\operatorname{sgn}(t)$ (except possibly at $t = 0$) as the "annealing factor" $T \to 0$. Sigmoid activation functions are hence the natural generalization of the threshold activation function. While there is some biological evidence in favor of sigmoid activations (cf. [103, 104, 218]), the popularity of sigmoid neurons as the basic computational element in neural networks devolves largely upon the existence of *ad hoc* learning algorithms for training a network of sigmoid neurons from examples when σ is chosen to be sufficiently smooth.

6.1.2 Networks

A neural network is a collection of formal neurons (linear threshold elements, polynomial or canonical threshold elements, or sigmoid neurons) interconnected by having the output of each neuron function as input to any sub-collection of neurons. In addition, a designated set of neurons receive external inputs, while another designated set of neurons is identified as a set of output elements. Formally, the architecture of a network is specified by a directed graph $G = \langle V, E \rangle$.

- *Vertex set* V: The set of vertices V is comprised of a set of *source nodes* (corresponding to the set of external inputs) together with a set of *computation nodes* (corresponding to the neurons in the network) some of which are designated as output nodes. For networks of sigmoid neurons, the node functionality of an output node is frequently distinguished from the rest of the network comprising, for instance, just a linear form of its inputs (in function approximation settings), or with an added threshold to produce a binary value (in pattern recognition or decision function settings).

- *Edge set* E: A directed edge (i, j) is present in the set of edges E if, and only if, i is a source node, j is a computation node, and i is connected to j, or i and j are computation nodes with the output of i serving as an input to j.

Neural network architectures may be partitioned into two families based on whether the underlying graph has cycles.

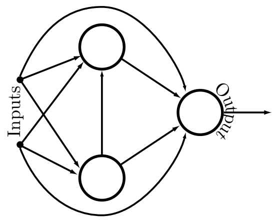

Figure 3: Acyclic network.

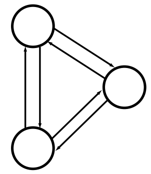

Figure 4: Recurrent network.

A network is *acyclic* if its architectural graph has no cycles. Network functionality is not affected by the choice of a particular mode of operation[4] for an acyclic network—with external inputs fixed, the network will settle into a stable steady state (independent of the mode of operation) in a finite number of steps at which point the outputs can be read out. The number of steps needed to reach the steady state is determined by the *depth* of the network, i.e., the number of edges in its longest path. Figure 3 shows an architectural graph of a depth-three, acyclic network.

Layered acyclic networks are of particular interest because of their very regular structure. Formally, an L*-layered feedforward* neural network is comprised of L ordered sub-collections of neurons called *layers* (layers 1 through $L-1$ are sometimes called *hidden layers*); the inputs to the first layer are the network inputs; for $l = 1, \ldots, L-1$, the inputs to layer $l+1$ are obtained from the outputs of layer l; the outputs of the Lth layer are the network outputs. The natural graph associated with a feedforward network is an L-partite graph where the depth of the network is identically the number of layers L. In such systems, starting from the external inputs, information flows sequentially from layer to layer until the output is obtained.

For definiteness, we consider acyclic neural networks with a single output node (with an added threshold in the case of sigmoid networks) in two-class pattern recognition contexts. Any such given network dichotomizes feature space $\Re^d$ into two classes identified naturally by the binary outputs ± 1 of the network. A rich, distinct family of classifiers is engendered by each such architecture by varying the weights of the formal neurons. Thus, for instance, a single linear threshold element engenders the family of hyperplane classifiers, while hyperspherical and hyperconical classifiers are obtained by suitable selection of quadric threshold element. Such facile geometrical descriptions of classifier families are unavailable, however, for even the simplest depth-two structures where we have to resort to purely abstract characterizations in terms of the underlying graph.

A neural network is said to be *recurrent* if the associated graph has cycles. In most cases in practice, for stability reasons, the network weights are symmetric with respect to permutation of indices. In such situations, the architectural graph may be simplified to an undirected graph $G = \langle V, E \rangle$, where connections between computation nodes are assumed

[4]In *synchronous* operation, the network has a clock and all neurons operate on the same (synchronized) time scale. In *asynchronous* operation, all neurons operate on different time scales.

to be two-way.

Figure 4 illustrates a fully recurrent network. Feedback creates potential instabilities and, indeed, the dynamics of state evolution are now strongly dependent on the mode of network operation. In the classical recurrent neural network setting it is desired to not just recognize, but retrieve prototypical patterns from distorted versions. This is the setting of associative (or content-addressable) memory (cf. [151, 165]). While the associative memory setting provides another pattern recognition (and retrieval) paradigm, the setting and issues have a somewhat different flavor from the Bayesian setting considered hitherto. We accordingly eschew considerations of recurrent networks in this paper and refer the reader to the following sampling of the more technical papers for details [10, 18, 151, 152, 165, 168, 169, 192, 206, 280, 282, 283, 284].

6.2 Universal Classification

The wide-ranging application of neural network models in pattern recognition rests upon the following fundamental result: *Formal neurons form a universal basis for computation.* The result is quite general so that virtually any kind of formal neuron is admissible as an elementary building block in a network approach to a classification problem.

6.2.1 Finite Problems

It has been long known that the working of any finite-state machine can be mimicked by a network of linear threshold elements [16, 191].

Identify the binary values -1 and $+1$ with the logical truth values 0 and 1, respectively. With this convention, a two-input NAND gate computes the truth function

$$\mathrm{NAND}(x_1, x_2) = \begin{cases} -1 & \text{if } x_1 = x_2 = 1, \\ +1 & \text{otherwise.} \end{cases}$$

The literals x_1 and x_2 in the above expression take values ± 1 only. Observe that we may represent the logical NAND also as a linear threshold function

$$\mathrm{NAND}(x_1, x_2) = \mathrm{sgn}(-x_1 - x_2 + 1), \qquad x_1, x_2 \in \{-1, +1\}.$$

As NAND gates form a universal basis for Boolean functions, so do linear threshold elements. In particular, any Boolean function of d literals can be computed by a depth-two network of linear threshold elements of size no more than $2^{d-1}+1$.[5] As a corollary, it follows immediately that the Bayes classifier can be implemented by a network of linear threshold elements if the mixture distribution of features has probability-one support over a finite set of points in feature space $\Re^d$.

[5] Very efficient neural circuit implementations have been designed in a variety of practical computational settings by allowing unbounded fan-in to exploit more fully the computational power latent in a linear threshold element (cf. [238], for instance).

6.2.2 Continuous Problems

An almost as simple Pythagorean argument shows that a corresponding universality result holds for continuous classification problems in $\Re^d$.

Let $E = [0,1]$ denote the unit interval and consider the class $\mathcal{C} = \mathcal{C}(E^d)$ of continuous, bounded real-valued functions on the unit cube E^d. The class $\mathcal{C}$ forms a real vector space which we may equip with the natural inner product

$$\langle f, g \rangle = \int_{E^d} f(x) g(x)\, dx,$$

and induced L^2-norm $\|f\|_2 = \sqrt{\langle f, f\rangle}$. Recall that the L^2-closure of $\mathcal{C}$ is just the Hilbert space $\mathcal{L}^2 = \mathcal{L}^2(E^d)$ of square-integrable functions on E^d.

Now, for any continuous sigmoid σ, let $\mathcal{N}_\sigma$ denote the family of continuous functions defined on the unit cube E^d of the form

$$g(x_1, \dots, x_d) = \sum_{j=1}^{N} c_j \sigma\Bigl(\sum_{i=1}^{d} w_{ij} x_i - w_{i0}\Bigr),$$

where N runs through the positive integers and c_j and w_{ij} are real weights. Observe that $\mathcal{N}_\sigma$ is just the family of functions on E^d computable by depth-two, arbitrary size networks of formal neurons with common activation function σ in the first layer and a linear accumulation at the output.

Let $\overline{\mathcal{N}_\sigma}$ denote the closure of $\mathcal{N}_\sigma$ under the L^2-norm. Clearly, $\overline{\mathcal{N}_\sigma}$ is a closed subspace of $\mathcal{L}^2$. Suppose $\overline{\mathcal{N}_\sigma} \neq \mathcal{L}^2$. Then there exists a non-zero $h \in \mathcal{L}^2$ which is orthogonal to the closed subspace $\overline{\mathcal{N}_\sigma}$. Now consider any function of the form

$$g(x_1, \dots, x_d) = \sigma\Bigl(\sum_{i=1}^{d} w_i x_i - w_0\Bigr).$$

As g is in $\overline{\mathcal{N}_\sigma}$, it follows that g is orthogonal to h. Consequently,

$$0 = \langle g, h \rangle = \int_{E^d} \sigma\Bigl(\sum_{i=1}^{d} w_i x_i - w_0\Bigr) h(x_1, \dots, x_d)\, dx_1 \dots dx_d$$

for all choices of weights w_i. It follows that $h \equiv 0$, contradicting the hypothesis. In consequence: *The subspace $\mathcal{N}_\sigma$ is dense in $\mathcal{L}^2$ (with respect to L^2-norm).*

By an elegant extension of this line of argument using the Hahn-Banach theorem, Cybenko [71] has shown the sharper result that $\mathcal{N}_\sigma$ is *uniformly dense* in $\mathcal{C}$ for a very wide range of choices for the sigmoid σ. Related approximation results and approximation error rates were derived by Barron [21], Barron [36], Chen, Chen, and Liu [56], Darken et al [73], Funahashi [120], Hornik, et al [153], Hornik [154], Jones [157], Leshno et al [178].

The dense approximation property of depth-two networks immediately implies as a corollary the following universal approximation result: *The Bayes classifier in the feature space $\Re^d$ can be approximated arbitrarily well by networks of formal neurons.* Of course, for better approximation, more neurons are needed. This fact may tempt the unwary practitioner to increase the size of the network to be used for classification. However, one should proceed with care. A large network has many tunable parameters all of which must be learned from a fixed amount of training data. Thus, one may easily run into the problem of *overfitting*, that is, the resulting classifier will perform very well on the data but no generalization will take place. As we have seen in Section 5, it is the VC dimension of the class of classifiers defined by a given network architecture that determines the generalization ability. This is the same approximation/generalization trade-off we have already encountered. Therefore, to be able to determine the size of a neural network to be used, it is important to study its VC dimension. This is precisely what we do in the next section, starting with single-neuron classifiers.

6.3 Acyclic Networks

Consider acyclic neural networks computing maps from $\Re^d$ into $\{-1,+1\}$. As seen earlier, the Vapnik-Chervonenkis dimension of a given network architecture (more precisely, the VC dimension of the class of decision functions computable by the architecture) is the critical parameter determining the sample complexity for learning a function in the class in a distribution-free fashion. While exact results are available only for the simplest cases of a single computational element, these results can be parlayed into fairly tight upper and lower bounds for the VC dimension of general architectures.

6.3.1 Canonical Threshold Elements

Let $\varphi_1, \dots, \varphi_k$ be a fixed set of linearly independent measurement functions mapping $\Re^d$ into $\Re$ and suppose $\varphi = (\varphi_1, \dots, \varphi_k)$ is the corresponding vector of functions. Recall that the canonical form for a neuron computes a φ-threshold function

$$y = \operatorname{sgn}\Big\{\sum_{j=1}^{k} w_j \varphi_j(x_1, \dots, x_d)\Big\}.$$

Let $\Sigma = \{\, X_i, 1 \le i \le n \,\}$ be an n-set of points in $\Re^d$ and suppose (Σ_+, Σ_-) is a dichotomy of Σ. As the entire family of φ-threshold functions may be explored by varying the weights $w_1, \dots, w_k$, the notion of separability takes the following form: the dichotomy (Σ_+, Σ_-) is *φ-separable* (i.e., can be separated by a φ-threshold element) if, and only if, there exists a vector of weights $w = (w_1, \dots, w_k)$ such that

$$\operatorname{sgn}\langle w, \varphi(X)\rangle = \operatorname{sgn}\Big\{\sum_{j=1}^{k} w_j \varphi_j(X)\Big\} = \begin{cases} -1 & \text{if } X \in \Sigma_-, \\ +1 & \text{if } X \in \Sigma_+, \end{cases} \tag{13}$$

where $\langle \cdot, \cdot \rangle$ denotes the usual Euclidean inner-product.

Separability. Let $\Delta_\varphi(\Sigma)$ denote the number of dichotomies of Σ that are φ-separable and, as before, define the nth shatter coefficient $s_k(n) = \max \Delta_\varphi(\Sigma)$ where the maximization is

over all n-sets Σ. The key to the determination of the VC dimension of a φ-threshold element is *Schläffli's recurrence* [65, 245, 295]

$$s_k(n) = s_k(n-1) + s_{k-1}(n-1) \qquad (k \ge 2, n \ge 2) \tag{14}$$

with equality $\Delta_\varphi(\Sigma) = s_k(n)$ holding when Σ is in φ-general position.[6] This recurrence is fundamental to the basic Vapnik-Chervonenkis inequality and may be established in this setting by a pleasing geometrical argument: Each point X_i engenders a φ-surface

$$H_i = \Big\{\, w \in \Re^k : \sum_{j=1}^{k} w_j \varphi_j(X_i) = 0 \,\Big\}$$

which may be identified as a hyperplane in $\Re^k$. Consider the hyperplanes H_i generated by the $n-1$ points $\{\, X_i, 1 \le i \le n-1 \,\}$. These surfaces partition $\Re^k$ into components (i.e., maximally connected regions), the number of components being exactly the number of φ-separable dichotomies of these $n-1$ points. Suppose that the first $n-1$ points X_i engender the maximum number of φ-separable dichotomies. Then the number of components in $\Re^k$ carved out by the hyperplanes $\{\, H_i, 1 \le i \le n-1 \,\}$ is exactly $s_k(n-1)$ (by induction hypothesis). Suppose the nth point X_n is added in such a way as to retain the maximum number of φ-separable dichotomies. The $s_k(n-1)$ components may now be seen to fall into two categories: Q_1 type 1 components which are intersected by H_n, and Q_2 type 2 components which are not intersected by H_n. Observe that $s_k(n) = 2Q_1 + Q_2 = s_k(n-1) + Q_1$. To complete the argument, observe that each the projection of each type 1 component into the $(k-1)$-dimensional hyperplane H_n is itself a unique component in H_n. It follows that the number of type 1 components is exactly $s_{k-1}(n-1)$ (by induction hypothesis) completing the recurrence. It is clear that the recurrence holds for points in φ-general position as general position is preserved under projections so that the preceding induction carries through *in toto*.

The recurrence (13), together with the boundary conditions $s_k(1) = s_1(n) = 2$ immediately yield the solution

$$\Delta_\varphi(\Sigma) \le s_k(n) = 2\sum_{j=0}^{k-1} \binom{n-1}{j} \tag{15}$$

with equality when the n-set of points Σ is in φ-general position.[7] The following fundamental result follows immediately: *The VC dimension of the class of decision functions $\mathcal{C}_\varphi$ computable by a φ-threshold element, where $\varphi = (\varphi_1, \dots, \varphi_k)$ is any vector of linearly independent measurement functions, is given by $V(\mathcal{C}_\varphi) = k$.*

EXAMPLE 5: *Linear threshold function.*

If $k = d+1$, $\varphi_0(x) = -1$, and $\varphi_j(x) = x_j$ $(1 \le j \le d)$, then φ-general position coincides with the usual notion of general position—no set of $d+1$ points lies on a hyperplane in $\Re^d$.

[6] A set of points $\Sigma = \{X_1, \dots, X_n\}$ is in φ-general position if every m-element subset of the vectors $\{\varphi(X_1), \dots, \varphi(X_n)\}$ is linearly independent for all $m \le n$.

[7] Observe that (14) tightens the Vapnik-Chervonenkis bound (7) for the shatter coefficient in this particular case.

The separating surfaces are hyperplanes defined by equations of the form $\sum_{i=1}^{d} w_i x_i = w_0$. The VC dimension of the class of linear threshold functions is hence $V = d + 1$.

EXAMPLE 6: *Hyperspherical threshold function.*

A hyperspherical surface in $\Re^d$ is defined by an equation of the form $\|w - x\|^2 - r^2 = 0$, where, with $\langle w, x\rangle = \sum_{i=1}^{d} w_i x_i$ denoting the usual Euclidean inner-product, $\|w - x\| = \sqrt{\langle w - x, w - x\rangle}$ is the usual induced metric. Now note that we can write the corresponding hyperspherical threshold function in the form

$$y = \operatorname{sgn}(\|w - x\|^2 - a^2) = \operatorname{sgn}\left((\|w\|^2 - a^2) - 2\langle w, x\rangle + \|x\|^2\right).$$

It follows that this is a special case of a φ-threshold function with $k = d+2$ and the mapping from $\Re^d$ into $\Re^{d+2}$ determined by $x \overset{\varphi}{\mapsto} (1, x, \|x\|^2)$. Consequently, the VC dimension is not more than $d + 2$. (Actually, in this case the VC dimension equals $d + 1$, see Dudley [100].)

EXAMPLE 7: *Polynomial threshold function.*

The measurement functions comprising a quadric are monomials of degree 2 or less. In general, a polynomial threshold function of degree D is generated by monomials of degree D or less. In particular, if we set $x_0 = -1$ for simplicity, the vector of measurement functions takes the form

$$\varphi(x) = \left(x_0^{i_0} x_1^{i_1} \dots x_d^{i_d},\ i_0, i_1, \dots, i_d \geq 0,\ \sum_{j=0}^{d} i_j = D\right).$$

The separating surface for such a system is a Dth-order rational variety and the VC dimension is $\binom{d+D}{D}$.

A very sharp characterization of the breakdown of separability is suggested by the explicit form (14) for the growth function $s_k(n)$. Suppose Σ is a set of points in $(\varphi_1, \dots, \varphi_k)$-general position. Then the probability that a random dichotomy (Σ_+, Σ_-) (selected uniformly from the set of 2^n possible dichotomies of Σ) is $(\varphi_1, \dots, \varphi_k)$-separable is given by

$$P(k, n) = 2^{-n} s_k(n) = 2^{-(n-1)} \sum_{j=0}^{k-1} \binom{n-1}{j}.$$

The exponential decay of the binomial tail results in a sharp asymptotic concentration around $(n - 1)/2 + O(\sqrt{n})$ and a consequent remarkable asymptotic breakdown in separability: for every $\epsilon > 0$, as $k \to \infty$,

$$P(k, n) \to \begin{cases} 1 & \text{if } n = n_k \leq (1 - \epsilon)2k, \\ 0 & \text{if } n = n_k \geq (1 + \epsilon)2k. \end{cases}$$

This sharp threshold property suggests that we identify $2k$ as the "capacity" of a $(\varphi_1, \dots, \varphi_k)$-threshold element [65, 279, 283].

Learning Algorithms A wide variety of learning algorithms are available when the data are drawn from a φ-separable class. In this case, we are assured that there exists a *solution weight vector* $w = (w_1, \dots, w_k)$ satisfying (12). Classical off-line learning approaches for finding a solution weight vector treat the problem as an instance of linear programming which can be solved using the simplex algorithm [72], or (with guaranteed polynomial time convergence) by Karmarkar's algorithm [160]. Such off-line algebraic approaches, however, are not best suited for adaptive pattern recognition applications.

The classical on-line learning procedure for φ-separable classes is the Perceptron training procedure of Rosenblatt [234]. In its simplest variant, the algorithm is provided with a sequence of training data $\{X(t), t \geq 1\}$ generated from the sample X_i $(1 \leq i \leq n)$ in such a way that each X_i recurs infinitely often in the sequence (by cyclically running through the sample, for instance). The algorithm then generates an error-driven sequence of estimates of solution weight vectors $\{w(t), t \geq 1\}$, where $w(1)$ is an arbitrary initial guess of the solution vector, and at each epoch t, the succeeding weight vector estimate $w(t + 1)$ is generated on-line as a function of the current estimate $w(t)$ and the current datum $(X(t), Y(t))$ only. To simplify presentation, observe that by the simple expedient of replacing $\varphi(X_i)$ by $-\varphi(X_i)$ whenever $Y_i = -1$, we may, without loss of generality, suppose that $Y(t) = 1$ for all t. The fixed-increment perceptron training procedure now incrementally adapts the weight vector estimate as follows:

$$w(t + 1) = \begin{cases} w(t) & \text{if } \langle w(t), \varphi(X(t))\rangle \geq 0, \\ w(t) + \rho\varphi(X(t)) & \text{if } \langle w(t), \varphi(X(t))\rangle < 0. \end{cases}$$

The algorithm is on-line and error-driven. Geometrically, the algorithm has the pleasing intuitive interpretation that whenever there is a misclassification, weight vector updates are in the direction of the positive half-space of the vector $\varphi(X(t))$ corresponding to the current pattern $X(t)$. The algorithm provably converges in finite time[8] whenever (12) has a solution. See [198, 207] for proofs and considerations of algorithm behavior for nonseparable cases.

Various extensions of the basic fixed-increment perceptron training procedure exist: variable increment procedures allow a time-varying increment $\rho = \rho(t)$; relaxation procedures select $\rho = \rho(t, w(t), X(t))$ as a function of the current estimate and datum as well. See Duda and Hart [97] for the literature.

Perceptron-based procedures which focus attention on the misclassified examples have drawbacks when the sample is not separable. They are also not easily extended to general network architectures, in large part because it is not clear what classifications the intermediate (or hidden) neurons should be assigned. On the other hand, gradient-based procedures which minimize instead a continuous functional of the sample, such as the mean-squared error on the sample, are readily extendable to network situations (with the proviso that a continuously differentiable activation function is chosen) and have consequently become very popular.

In the single element formulation, a weight vector $w = (w_1, \dots, w_k)$ is sought which satisfies $\langle w, \varphi(X_i)\rangle = b_i$ $(1 \leq i \leq n)$ where the b_i are arbitrarily specified positive con-

[8] While the worst-case behavior of the algorithm is not well regulated, the algorithm converges rapidly in typical situations [35, 281].

stants. (Recall that we have set $Y_i = 1$ without loss of generality.) The least mean-square approach [298] seeks to minimize the squared-error

$$E(t) = [\langle w(t), \varphi(X(t))\rangle - b(t)]^2$$

by updating the current weight vector estimate $w(t)$ in the direction of the gradient. This results in a relaxation-type update rule

$$w(t+1) = w(t) + \rho(t)[b(t) - \langle w(t), \varphi(X(t))\rangle]\varphi(X(t)).$$

A proper choice of increment $\rho = \rho(t)$ will drive the algorithm to a limiting solution. Related procedures include stochastic approximation methods [293] and the family of Ho-Kashyap algorithms [149] which also adapt the margin vector $b = (b_1, \dots, b_n)$. However, while computationally attractive, such relaxation approaches have the disadvantage that it is not clear how the solutions they generate are related to the best classifier in the class from the probability of error point of view.

6.3.2 General Acyclic Networks

Computational Considerations The results for a single computational element may be parlayed into upper bounds for general acyclic architectures by a recursive greedy use of Schläffli's counting formula (14) (cf. [37, 67]). Consider an arbitrary acyclic architecture G comprised of canonical threshold elements with d source nodes, one output node, and K programmable parameters (weights). Suppose computational node j has associated with it k_j programmable weights. Then, from (14), node j can compute at most $n^{k_j+1} + 1$ different functions from any finite n-set of points Σ in $\Re^d$ into $\{-1, +1\}$. Thus, the architecture G can separate at most $\prod_j (n^{k_j+1} + 1) \leq n^{2K}$ dichotomies of Σ. As there exists a set of V points which is shattered, where $V = V(\mathcal{C}_G)$ is the VC dimension of the family of decision functions $\mathcal{C}_G$ computable in the architecture G, it follows that $2^V \leq V^{2K}$. We thus have the following upper bound on the VC dimension: *If the architecture G is comprised of canonical threshold elements with d source nodes, one output node, and K programmable parameters,*

$$V(\mathcal{C}_G) \leq cK \log K \tag{16}$$

for an absolute positive constant c.

In particular, depth-two linear threshold networks with d source nodes, k computational nodes in the first layer, and a single output node have VC dimension bounded by $V \leq c(dk+2k+1)\log(dk+2k+1)$. On the other hand, Baum [34] has shown constructively using a modified slice technique pioneered by Nilsson [207] that $V \geq 2\lfloor \frac{k}{2} \rfloor d$. Thus, $V = O(dk \log dk)$ and $V = \Omega(dk)$. More generally, in an L-layered, feedforward setting, the bounds suggest that the VC dimension increases faster by adding nodes to existing layers (i.e., by increasing the "width" of the network) than by adding layers (i.e., by increasing the "depth" of the network). This may be taken as a step in defense of a connectionist thesis: *Shallow networks (with dense interconnectivity) are computationally more efficient than deep networks [20].*

Intuition (perhaps bolstered by (14) and Baum's lower bound for depth-two threshold networks) suggests that the VC dimension of a network of threshold elements should not

exceed the sum of the VC dimensions of the individual threshold elements in the network. Surprisingly, however, constructions by Maass [187] and Sakurai [241] show that the upper bound (15) is indeed admissible (up to a multiplying constant): *There exists a sequence of linear threshold network architectures $\{G_K, K \geq 1,\}$, where G_K has K free programmable parameters, for which*

$$V(\mathcal{C}_{G_K}) \geq c' K \log K \qquad (K \geq 1) \tag{17}$$

for an absolute positive constant c'. This implies that in larger networks, an average programmable parameter can contribute more than a constant amount to the net VC dimension. In fact, an average parameter can contribute as much as the order of $\log K$ *vide* the lower bound (16), so that the contribution of a typical parameter to the VC dimension can actually *increase* with network size. This may be taken as mathematical support toward a connectionist manifesto: *A network of formal neurons is more than just the sum of its components*; (cf. [151] for another version of this sentiment—this time in a recurrent network setting).

The superlinear lower bound (16) continues to hold for networks with common logistic activation function (11) (with a threshold at the output node). However, good upper bounds are in abeyance and seem to demand a case-by-case exploitation of the specific properties of the activation function. Smoothness of the activation function is not sufficient: for instance, Sontag [260] shows that a simple depth-two network with two real-valued inputs, two first layer sigmoid elements with some common smooth sigmoid activation function and a single linear threshold element as output in the second layer has *infinite* VC dimension. The infinite VC dimension of two-layered networks remains true even for some monotone, continuous activation functions which are convex on $(-\infty, 0)$ and concave on $(0, \infty)$ (see [90, Section 30.4]). For the case of acyclic networks with the popular logistic activation function, however, MacIntyre and Sontag [188] have shown that the VC dimension is finite. The explicit role played by the number of programmable parameters is open. For networks using piecewise-linear activation functions (10), real inputs and thresholded outputs, Goldberg and Jerrum [128] show an explicit upper bound $O(K^2)$ for the VC dimension of the network in terms of the number of programmable parameters (weights) K in the network. For recent work on the VC dimension of neural networks, see Karpinsky and Macintyre [161], Koiran and Sontag [166], and Bartlett, Maiorov, and Meir [32].

Learning Algorithms In general, it is much harder to find algorithms for learning from examples with provable performance for a general network architecture. Indeed, the loading (or consistency) problem is computationally intractable[9] for all but the simplest network constructs. The first results along this direction were established by Judd (cf. [158]). Subsequently, the loading problem was shown to be intractable for several classes of neural networks of threshold elements, including: 0–1 halfspace [219]; union of two halfspaces [43]; two-layer network of linear threshold elements with two elements in the first layer and a single output element [43]; XOR of two halfspaces [43]; and two-cascade network [180]. For related results for learning models with equivalence and membership queries, see Hegedüs [144, 219].

[9] More formally, the loading problem is NP-complete. For a reference to this and other notions of time complexity in computation, we refer the reader to Garey and Johnson [121].

More recently, DasGupta, Siegelmann, and Sontag [76] have shown that the loading problem remains intractable for two-layer networks with two first layer neurons with piecewise-linear activation (and varying input dimension) and a single output threshold element. Shifting focus from the number of misclassifications as criterion function to a squared-error criterion (i.e., shifting from l^∞-norm to l^2-norm) does not improve matters: Vu [286] has shown very recently that training two-layer networks, where the first layer consists of linear threshold functions or sufficiently smooth sigmoids and the output element in the second layer simply forms a linear combination of the outputs of the first layer, so as to obtain small average squared-error is computationally infeasible.[10]

A noteworthy exception to these negative results when queries are permitted is a half-space learning algorithm due to Baum [36] which learns weights for a depth-two network of linear threshold elements via queries. The algorithm exploits the geometric interpretation of hyperplanes generated by the hidden (first) layer of linear threshold elements as partitioning input space $\Re^d$ into components (with hyperplanes as boundaries). Suppose examples are drawn at random from a function generated by a target depth-two network. Observe that all points in any given component generated by the first layer elements of the target network will be labelled either positive or negative. The task of learning such a network can now be viewed as determining the piecewise linear boundary between positively and negatively labelled components.

Baum's algorithm proceeds by first drawing a sufficient number of random examples and then using queries to determine the hyperplanes corresponding to the hidden first layer elements. A boundary point between a positive and a negative example is first found by binary search following which a hyperplane containing the boundary point is identified by exploring its neighborhood randomly. Once the hyperplanes are identified, the weight vector of the output unit is determined quickly by finding a hyperplane separating positive and negative labelled components.

While geometrically appealing, the algorithm is, however, provably efficient only for relatively small networks with fewer than five first-layer elements.

For more general acyclic structures, the learning heuristic of choice is based on gradient descent ideas. Consider a sigmoid network where the activation function σ is continuously differentiable. For a given exemplar pair (X, Y), let $\tilde{Y} = \tilde{Y}(W) = \tilde{Y}(W; X)$ denote the output of a network of given architecture with weights W. Writing $\hat{P}_n$ for the empirical probability measure which puts equal weight on each of the n examples in D_n, we can write the empirical mean-square error on the sample in the form

$$E = E(W) = \hat{P}_n(\tilde{Y} - Y)^2.$$

The generic gradient descent algorithm modifies each weight w in the system in a natural fashion by setting

$$w \mapsto w - \rho \frac{\partial E}{\partial w}.$$

For a layered network of sigmoids, the computation of the various gradients can be recursively generated starting from the output unit and working backwards one-layer at a time to

[10] To make the statement more precise, the problem is NP-hard. See Garey and Johnson [121].

the input layer. The procedure is sometimes hence called *back-error-propagation* or simply backpropagation [240]. Variations on this theme include conjugate gradient and sequential quadratic programming methods.

While the backpropagation heuristic is popular by virtue of its simplicity, analysis is difficult (cf. [288, 289, 290]) and performance guarantees hard to come by. The algorithm is beset by a variety of problems: it is notoriously inconsistent unless stringent conditions are placed upon the problem; the algorithm tends to get stuck in local minima which may abound and are very hard to characterize; algorithm convergence can be very slow; and, finally, even when it successfully attains close to a minimum of the criterion function, it is not clear how the resulting solution relates to the best classifier (in terms of the probability of error) in the class. Successful applications in practice tend to be predicated on the availability of sufficient side information to enable canny choices of network architectures and good initial conditions.

6.4 Performance Bounds

Equipped with the approximation-theoretic results and VC dimension bounds, it is easy to study the behavior of the probability of error of certain neural network classifiers. As always, we are interested in the performance of the classifier compared to the Bayes risk. Let g_n denote the classifier obtained by data-based tuning of the parameters of a certain neural-network architecture with (say) k neurons. Denote the class of all possible such classifiers by $\mathcal{C}_k$. Then, as before, a typical analysis splits the error into two nonnegative terms:

$$L(g_n) - L^* = \left(L(g_n) - \inf_{g \in \mathcal{C}_k} L(g)\right) + \left(\inf_{g \in \mathcal{C}_k} L(g) - L^*\right).$$

Recall that the second term on the right-hand side represents the part of the error due to the limited approximation abilities of the class $\mathcal{C}_k$. The denseness results cited in Section 6.2 indicate that, in most cases, as $k \to \infty$, this term converges to zero for all possible distributions. Unfortunately, the speed of convergence depends on the unknown distribution, and may be arbitrarily slow. For certain classes of smooth distributions it is possible to derive uniform upper bounds for the approximation term, as in Barron [36]. The first term on the right-hand side represents, as noted before, the estimation error which is due to the finite sample size and the limitations of the learning algorithm. In an ideal situation, when g_n minimizes the empirical error over the class $\mathcal{C}_k$, the tools of Section 5 may be applied to derive distribution-free upper bounds for the estimation error. For example, for all distributions we have

$$\mathbf{E}L(g_n) - \inf_{g \in \mathcal{C}_k} L(g) \le c\sqrt{\frac{V_{\mathcal{C}_k}}{n}},$$

where $V_{\mathcal{C}_k}$ is the VC dimension of the class $\mathcal{C}_k$ and c is a universal constant. Therefore, universal consistency may be achieved simply by choosing $k = k_n$ as a function of the sample size n such that $k_n \to \infty$ and $V_{\mathcal{C}_{k_n}}/n \to 0$ as $n \to \infty$. For neural network architectures for which explicit upper bounds for $V_{\mathcal{C}_k}$ are available, this is a trivial matter.

Unfortunately, the above analysis has several weak points. First of all, the optimal network size cannot be determined solely as a function of the sample size as it heavily depends on the actual distribution of the data. Such a data-dependent choice might be based on the structural-risk-minimization ideas have discussed in Section 5.3.

A more serious problem is that minimizing the empirical risk is computationally not feasible even for small toy problems. The practical learning algorithms discussed above attempt to approximate the optimal solution but without universal performance guarantees. In fact, for some malicious distributions, all practical neural network classifiers may fail miserably. Therefore, in some sense, neural networks are inferior to other, more robust classifiers such as nearest-neighbor, kernel, or decision tree-based methods.

One should also note that instead of minimizing the empirical misclassification rate, most neural network learning algorithms attempt to minimize the empirical mean-square error, which is more appropriate for regression type problems. Even if an algorithm could minimize the empirical mean square error, the VC dimension bounds would not be applicable anymore. Nevertheless, universal consistency of such classifiers may be achieved even for some activation functions which induce classes of infinite VC dimension (see [184], for instance). Also, in some distribution-dependent setting, neural networks trained by minimizing the empirical squared error achieve surprisingly good performance as is shown in a remarkable paper of Bartlett [31]. In fact, Bartlett points out that, as opposed to what VC-dimension bounds suggest, the total weight of the parameters is more important in a certain sense than merely the number of tunable parameters. For consistency and related results we refer to Barron [24, 25, 26], Faragó and Lugosi [109], Haussler [140], Lugosi and Zeger [184], Mielniczuk and Tyrcha [197], Wang [288], Wang, Venkatesh, and Judd [289, 290, 291], and White [296, 297].

7 Large-Margin Classifiers

Consider empirical risk minimization over the class of perceptrons (i.e., linear classifiers), and assume first, for simplicity, that the two classes are linearly separable. This means that there exists a hyperplane cutting $\Re^d$ into two halves such that it separates label 0 training points from those with label 1. Then the Vapnik-Chervonenkis bounds guarantee that the probability of error of *any* such classifier will not be more than a constant multiple of $d \log n/n$. This bound is acceptable if $d \ll n/\log n$. However, sometimes in practice, the dimensionality of the feature space is very large, making the bound unsatisfactory. A question which arises immediately is whether, among all hyperplanes separating the data, some are better than others. In other words, can we gain substantially by fine-tuning the linear classifier and, in some way, selecting a "better" separating hyperplane? We have seen that the VC bounds are basically tight in a minimax sense, that is, for any classifier there is some distribution for which the upper bound is (almost) achieved, which also indicates that it does not really matter which separating hyperplane we choose. However, one may argue that the "bad" distributions are pathological in some sense, and for distributions appearing in "real-life" one may hope for much better performance. This might certainly be true, but to benefit from such "lucky" cases, one should be able to observe this from the training data at one's disposal.

The first significant step made in this direction was a beautiful observation of Vapnik and

Chervonenkis [277] who proved that if one *happens* to find a separating hyperplane which separates with a *large margin* (i.e., all data points are far away from the decision surface) then one may derive a much smaller upper bound than that given by the distribution-free analysis. The reason is that the "effective" VC dimension of linear classifiers with a large margin may be much smaller than the dimensionality of the feature space, and, indeed, may only depend on the size of the margin, measured in an appropriate way. (For the precise statement and more details we refer to Vapnik [275].) Thus, this bound gives a *distribution-dependent* performance guarantee, which, most importantly, can be assessed directly from the data. The same principle may be adapted to situations when the data are not necessarily separable by a hyperplane. Recently, this principle has been investigated more generally by Shawe-Taylor, Bartlett, Williamson, and Anthony [251, 252], and Bartlett [31].

The basic idea behind the *Support Vector Machines* (c.f. Vapnik [275], Boser, Guyon, and Vapnik [45], and Cortes and Vapnik [64]) is to transform the data into a high-dimensional space by some transformation fixed in advance, and find a large-margin separating hyperplane in the transformed space. Clever implementational tricks make this seemingly complicated principle feasible for practical applications, and, even though very little of its theoretical properties are known, it has been reported to achieve remarkable performance for large and complex pattern classification problems (c.f. [275], [45], [64]).

Closely related principles are the methods of *boosting*, developed by Freund [113] and Schapire [243], and the *bagging* and *arcing* methods of Breiman [48, 49], see also Drucker and Cortes [98], Quinlan [229]. These methods combine, by (weighted) majority voting, the decisions of several simple classifiers. For example, boosting generates a sequence of classifiers, all taken from a class of small VC dimension, in such a way that every classifier tries to classify those points well which were found to be "difficult" by the previous classifier. As argued by Schapire, Freund, Bartlett, and Lee [244], these methods attempt to maximize the "margin" in a general sense, making the decision more "stable.'

8 Automatic Parameter Selection

In earlier sections we have seen several different principles and methods for designing classifiers. From a certain point of view all of them may look appealing. The obvious questions a practitioner facing a classification problem may ask are:

(1) Which of the many methods should I use?

(2) Is there a classifier which is superior to all others?

The answer to the second question is "No." No single classifier is uniformly superior to all the others and, consequently, the user has to make a choice on a case-by-case basis. Since the only available information is the training data, such a selection should be based on the data. Even if one has decided to use (say) the k-nearest neighbor rule, the value of k and perhaps the metric might be selected in a data-based manner. In fact, to hope for any success in practical problems, one *must* let the data do the talking. This section deals with such data-dependent classifier selection.

The problem, in its most general form, may be formulated as follows: given a set of classifiers $\mathcal{C}_n = \{g_{n,\theta} : \theta \in \Theta\}$, choose one with a small probability of error. Here Θ is an abstract set of parameters. Interesting examples of Θ and the corresponding set of classifiers include:

- $\Theta = \{1, \ldots, n\}$ and $g_{n,k}$ is the k-nearest neighbor rule with the Euclidean distance.
- Θ is a set of positive definite $d \times d$ matrices and $g_{n,\theta}$ is the 1-nearest neighbor rule based on the distance $d(x,y) = ((x-y)^t\theta(x-y))^{1/2}$.
- Θ is the set of positive integers and $g_{n,\theta}$ is the kernel rule with a fixed kernel and smoothing parameter θ.
- Θ is the set of positive integers and $g_{n,\theta}$ is a neural network classifier with a fixed architecture, trained by θ iterations of back-propagation.

Essentially all problems of parameter selection may be cast in this general framework. In some sense, even the problem of feature extraction belongs here.

The most sensible way of making such a selection is to use the data to estimate the probability of error of each classifier $g_{n,\theta}$ in the class, and to choose a classifier minimizing the estimate. In fact, basically all methods (directly or indirectly) are based on this general principle. The basic difficulty is that the same data which defines the classifiers $g_{n,\theta}$ are to be used for estimating purposes as well, so the problem should be handled with extreme care.

The simplest case is when a separate set of labelled samples is available; call it

$$T_m = ((X_{n+1}, Y_{n+1}), \ldots, (X_{n+m}, Y_{n+m})).$$

This may be achieved by artificially holding out m samples from defining the classifiers $g_{n,\theta}$. Since in most applications data is very expensive to acquire, most designers don't like the idea of "wasting" valuable data for merely testing purposes. However, as we will see shortly, holding out just a few samples can be extremely rewarding. So, for the sake of simplicity, suppose now that a testing set T_m is available, and is independent of D_n. Then, for each classifier in $\mathcal{C}_n$, we may form the empirical error estimate

$$\hat{L}_m(g_{n,\theta}) = \frac{1}{m}\sum_{i=1}^{m} I_{\{g_{n,\theta}(X_{n+i}) \neq Y_{n+i}\}},$$

and choose a θ minimizing the estimate over all $\theta \in \Theta$. Call the obtained classifier g_{n+m}. (The subscript reflects the fact that the final classifier depends on $m+n$ samples.) Now it is easy to see that, due to the independence of the testing sub-sample T_m and the training sample D_m, the performance of g_{n+m} compared to the best classifier in $\mathcal{C}_n$, may be analyzed by techniques described in Section 5. This simple argument is at the basis of the important paper of Devroye [85] who surveys parameter selection based on minimizing the empirical risk measured on an independent test sample. For example, it follows by a simple application of the Vapnik-Chervonenkis inequality (7′) that

$$\mathbf{E}\left\{ L(g_{n+m}) - \inf_{g_{n,\theta}\in\mathcal{C}_n} L(g_{n,\theta}) \middle| D_n \right\} \leq c\sqrt{\frac{\log s(\mathcal{C}_n, m)}{m}}, \tag{18}$$

where c is a universal constant and $s(\mathcal{C}_n, m)$ is the mth shatter coefficient of the set of classifiers $\mathcal{C}_n$ defined by a particular realization of the training data D_n. Thus, the success of this parameter selection procedure is guaranteed if $\log s(\mathcal{C}_n, m)$ is small compared to the size m of the testing data. From this point on, the analysis is purely combinatorial: one has to derive upper bounds for the shatter coefficient. Since the class $\mathcal{C}_n$ is defined by the random data D_n, the shatter coefficient $s(\mathcal{C}_n, m)$ is also a random variable. However, in many interesting cases, it is possible to find upper bounds for $s(\mathcal{C}_n, m)$ that only depend on n and m, and not on the particular realization of D_n. Devroye [85] and Devroye, Györfi, and Lugosi [90] derive such bounds for several interesting classes. Here we merely mention two simple examples:

EXAMPLE 8: Let $\mathcal{C}_n$ be the class of all k-nearest neighbor rules with $k = 1, 2, \ldots, n$. In this case, the value of k is to be selected by the data. Here clearly $\mathcal{C}_n$ is finite with $|\mathcal{C}_n| = n$, and (17) becomes

$$\mathbf{E}\left\{ L(g_{n+m}) - \inf_{g_{n,\theta}\in\mathcal{C}_n} L(g_{n,\theta}) \middle| D_n \right\} \leq c\sqrt{\frac{\log n}{m}}.$$

Therefore, in order to make sure that the probability of error of the k-nearest neighbor rule based on the selected k is close to that with the best possible k, all we have to guarantee is that $m \gg \log n$. In other words, if one is willing to "sacrifice" a tiny fraction of the training sample for testing purposes, the reward is a nearly optimal value of k. Also, the rule obtained by this procedure is consistent under very mild assumptions on m. This follows from the fact that, as we have seen before, for *some* choice of k, the k-nearest neighbor rule is consistent. This implies that

$$\mathbf{E}\left\{ \inf_{g_{n,\theta}\in\mathcal{C}_n} L(g_{n,\theta}) \right\} \to L^* \quad \text{as} \quad n \to \infty,$$

and therefore

$$\mathbf{E}\left\{ L(g_{n+m}) \right\} \to L^*$$

whenever $n \to \infty$ and $m/\log n \to \infty$.

EXAMPLE 9: Consider the class $\mathcal{C}_n$ of all moving window classifiers (as defined in Section 3) with all possible values of the smoothing parameter $h > 0$. Thus, the goal is to select h in a nearly optimal way. Now $\mathcal{C}_n$ is an infinite class, so bounding $s(\mathcal{C}_n, m)$ is less trivial than in the previous example. Still, it is not difficult to find useful bounds. For example, it is easy to prove that

$$s(\mathcal{C}_n, m) \leq mn$$

(c.f. [90, Chapter 25]). Substituting this bound in (17) we get

$$\mathbf{E}\left\{ L(g_{n+m}) - \inf_{g_{n,\theta}\in\mathcal{C}_n} L(g_{n,\theta}) \middle| D_n \right\} \leq c\sqrt{\frac{\log n + \log m}{m}}.$$

Clearly, this bound is as useful as the one obtained in the previous example, and all conclusions mentioned there remain valid. Selection of h for more general kernels is also possible; the theory is summarized in [90, Chapter 25].

In some cases it is impossible to obtain useful bounds for $s(\mathcal{C}_n, m)$. Clearly, if the class $\mathcal{C}_n$ is too large, it may overfit the test sample T_m and minimization of the error estimate cannot work. This is the same phenomenon encountered in Section 5. Many other examples are worked out in Devroye [85] and Devroye, Györfi, and Lugosi [90].

The situation is much more complicated if no independent testing data is available, and parameter selection has to be made based on error estimates calculated on the very same data from which the classifier is defined. Taking one step back, we may formulate the problem of *error estimation* as follows: given the training data $D_n = ((X_1, Y_1), \ldots, (X_n, Y_n))$ and the classifier $g_n(x) = g_n(x, D_n)$, estimate the probability of error

$$L(g_n) = \mathbf{P}\{g_n(X, D_n) \neq Y \mid D_n\}.$$

Mathematically, an error estimate $\hat{L}(g_n)$ is simply a real-valued function of the data D_n. Being able to estimate well the probability of error for each g_n in a set $\mathcal{C}_n$ of classifiers does not necessarily guarantee that the classifier minimizing the error estimate has a nearly optimal probability of error. To have such a guarantee, one needs to assure that the estimates are *uniformly* close to the true probabilities of error over the whole class $\mathcal{C}_n$. In the case of estimation using an independent test sample, we were able to derive such a guarantee using the Vapnik-Chervonenkis inequality. The absence of independent test data makes the problem significantly more difficult, so first we focus on the error estimation problem for an individual classifier g_n. Error estimation has been one of the key topics in pattern recognition. For surveys we refer to Kanal [159], Cover and Wagner [70], Toussaint [271], Glick [127], Hand [139], Jain, Dubes, and Chen [156], McLachlan [193], and Devroye, Györfi, and Lugosi [90].

Perhaps the simplest and most general method for estimating the probability of error is the *resubstitution estimate* (also called *apparent error rate*). This estimate simply counts the number of errors committed by g_n on the training sequence. Formally,

$$\hat{L}^{(R)}(g_n) = \frac{1}{n}\sum_{i=1}^{n} I_{\{g_n(X_i)\neq Y_i\}}.$$

Since the estimate tests the classifier on the same data on which it was defined, there is an evident danger of being optimistically biased. In some cases this estimate is simply useless. The simplest example is the 1-nearest neighbor rule for which $\hat{L}^{(R)}(g_n) = 0$ whenever all X_i's are different, regardless of the true probability of error. Still, the resubstitution estimate has a surprisingly good performance in some cases. For example, Devroye and Wagner [93] observed that if g_n chooses a hypothesis from a fixed class of rules $\mathcal{C}$ in an arbitrary data-dependent way, then

$$\mathbf{E}\left\{\left|\hat{L}^{(R)}(g_n) - L(g_n)\right|\right\} \leq c\sqrt{\frac{V_{\mathcal{C}}\log n}{n}},$$

where c is a universal constant and $V_{\mathcal{C}}$ is the VC dimension of the class $\mathcal{C}$. Thus, for example, for any linear classification rule, the resubstitution estimate is guaranteed to be within $O(\sqrt{d\log n/n})$ of the true error. For additional examples, see [90].

To eliminate the bias of the resubstitution estimate, several authors have proposed the *deleted* estimate (also called the *leave-one-out* estimate or the *U-method*) — e.g., see Lunts and Brailovsky [186], Lachenbruch [176], Cover [68], and Stone [266]. In fact, the leave-one-out estimate is probably the one most frequently used among practitioners. To compute this estimate, one first deletes the first pair (X_1, Y_1) from the data, forms the classifier g_{n-1} based on the rest of the data, and tests whether g_{n-1} classifies X_1 correctly. Then the procedure is repeated n times, each time deleting a different pair (X_i, Y_i). Finally, the deleted estimate $\hat{L}^{(D)}$ is the average number of errors. Formally,

$$\hat{L}^{(D)}(g_{n-1}) = \frac{1}{n}\sum_{i=1}^{n} I_{\{g_{n-1}(X_i, D_{n,i})\neq Y_i\}},$$

where

$$D_{n,i} = ((X_1, Y_1), \ldots, (X_{i-1}, Y_{i-1}), (X_{i+1}, Y_{i+1}), \ldots, (X_n, Y_n)).$$

is the training set with (X_i, Y_i) deleted. Now clearly, $\hat{L}^{(D)}(g_{n-1})$ is an estimate of $L(g_{n-1})$ rather than of $L(g_n)$, but the intuition is that if n is sufficiently large and g_n is "stable" in some sense, then the difference between $L(g_{n-1})$ and $L(g_n)$ is negligible. In many cases, the analysis of the deleted estimate may be based on a general inequality of Rogers and Wagner [233] and Devroye and Wagner [94] who prove that if g_n is symmetric, then

$$\mathbf{E}\left\{\left(\hat{L}^{(D)}(g_{n-1}) - L(g_n)\right)^2\right\} \leq \frac{1}{n} + 6\mathbf{P}\left\{g_n(X, D_n) \neq g_{n-1}(X, D_{n-1})\right\}.$$

This inequality may be used to obtain useful performance bounds for the deleted estimate for nearest neighbor, kernel, and histogram rules (see [90, Chapter 24] for a survey). Recent studies by Holden [150] and Kearns and Ron [163] investigate the performance of the deleted estimate for classifiers choosing their hypotheses from fixed VC classes. In spite of the practical importance of this estimate, relatively little is known about its theoretical properties. The available theory is especially poor when it comes to analyzing parameter selection based on minimizing the deleted estimate.

The deleted estimated has been criticized for its relatively large variance and for the computational demand it imposes. Many other error estimates have been proposed and investigated in the literature and it is impossible to discuss all of them here. We merely mention some of the most popular ones:

- the smoothed error estimate of Glick [127];
- the *a posteriori* probability estimate of Fukunaga and Kessel [118];
- the rotation estimate of Toussaint and Donaldson [272];
- several versions of Efron's bootstrap estimate [105, 106];

and refer the reader to the above-mentioned survey texts on error estimation.

References

[1] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control*, vol. 25, pp. 917–936, 1964.

[2] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer, "The probability problem of pattern recognition learning and the method of potential functions," *Automation and Remote Control*, vol. 25, pp. 1307–1323, 1964.

[3] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer, "The method of potential functions for the problem of restoring the characteristic of a function converter from randomly observed points," *Automation and Remote Control*, vol. 25, pp. 1546–1556, 1964.

[4] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer, "Extrapolative problems in automatic control and the method of potential functions," *American Mathematical Society Translations*, vol. 87, pp. 281–303, 1970.

[5] H. Akaike, "An approximation to the density function," *Annals of the Institute of Statistical Mathematics*, vol. 6, pp. 127–132, 1954.

[6] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Automatic Control*, vol. AC-19, pp. 716-723, 1974.

[7] K. Alexander, "Probability inequalities for empirical processes and a law of the iterated logarithm," *Annals of Probability*, vol. 4, pp. 1041–1067, 1984.

[8] I. Aleksander and H. Morton, *An Introduction to Neural Computing*, London: Chapman & Hall, 1990.

[9] S. Amari, "Mathematical foundations of neurocomputing," *Proc. IEEE*, vol. 78, pp. 1443–1463, 1990.

[10] J. A. Anderson, *Introduction to Practical Neural Modeling*, Cambridge, MA: MIT Press, 1994.

[11] T.W. Anderson, "Some nonparametric multivariate procedures based on statistically equivalent blocks," in P.R. Krishnaiah, editor, *Multivariate Analysis*, pages 5–27, New York, Academic Press, 1966.

[12] M.W. Anderson and R.D. Benning, "A distribution-free discrimination procedure based on clustering," *IEEE Transactions on Information Theory*, vol. 16, pp. 541–548, 1970.

[13] M. Anthony and N. Biggs, *Computational Learning Theory*, Cambridge University Press, Cambridge, U.K., 1992.

[14] M. Anthony and J. Shawe-Taylor, "A result of Vapnik with applications," *Discrete Applied Mathematics*, vol. 47, pp. 207–217, 1993.

[15] A. Antos and G. Lugosi, "Strong minimax lower bounds for learning," *Machine Learning*, 1997.

[16] M. A. Arbib, *Brains, Machines, and Mathematics*, New York: Springer-Verlag, 1987.

[17] P. Argentiero, R. Chin, and P. Beaudet, "An automated approach to the design of decision tree classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 51–57, 1982.

[18] A. F. Atiya and Y. S. Abu-Mostafa, "An analogue feedback associative memory," *IEEE Trans. Neural Networks*, vol. 4, pp. 117–126, 1993.

[19] T. Bailey and A.K. Jain, "A note on distance-weighted k-nearest neighbor rules," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, pp. 311–313, 1978.

[20] P. Baldi and S. S. Venkatesh, "On properties of networks of neuron-like elements," in *Neural Information Processing Systems*, (ed. D. Anderson), New York: American Institute of Physics, 1988.

[21] A.R. Barron, "Statistical properties of artificial neural networks," in *Proceedings of the 28th Conference on Decision and Control*, pp. 280–285, Tampa, FL, 1989.

[22] A.R. Barron, "Complexity regularization with applications to artificial neural networks," in *Nonparametric Functional Estimation and Related Topics*, edt. by G. Roussas, pp. 561-576, NATO ASI Series, Kluwer Academic Publishers, 1991.

[23] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *Tech. Rep. # 58*, University of Illinois at Urbana-Champaign, Urbana, IL, 1991.

[24] A.R. Barron, "Complexity regularization with application to artificial neural networks," in G. Roussas, editor, *Nonparametric Functional Estimation and Related Topics*, pp. 561–576. NATO ASI Series, Kluwer Academic Publishers, Dordrecht, 1991.

[25] A.R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information Theory*, vol. 39, pp. 930–944, 1993.

[26] A.R. Barron, "Approximation and estimation bounds for artificial neural networks," *Machine Learning*, vol. 14, pp. 115–133, 1994.

[27] A.R. Barron and R.L. Barron, "Statistical learning networks: a unifying view," in E.J. Wegman, D.T. Gantz, and J.J. Miller, editors, *Proceedings of the 20th Symposium on the Interface: Computing Science and Statistics*, pp. 192–203, AMS, Alexandria, VA, 1988.

[28] A.R. Barron, L. Birgé, and P. Massart, "Risk bounds for model selection via penalization," *Probability Theory and Related fields*, page to appear, 1996.

[29] R.L. Barron, "Learning networks improve computer-aided prediction and control," *Computer Design*, vol. 75, pp. 65–70, 1975.

[30] A.R. Barron and T. Cover, "Minimum complexity density estimation," *IEEE Transactions on Information Theory*, vol. 37, pp. 1034-1054, 1991.

[31] P.L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *IEEE Transactions on Information Theory*, to appear, 1997.

[32] P. Bartlett, V. Maiorov, and R. Meir, "Almost linear VC dimension bounds for piecewise polynomial networks," *Neural Computation*, to appear.

[33] O. Bashkirov, E.M. Braverman, and I.E. Muchnik, "Potential function algorithms for pattern recognition learning machines," *Automation and Remote Control*, vol. 25, pp. 692–695, 1964.

[34] E. B. Baum, "On capabilities of multi-layer perceptrons," *J. Complexity*, vol. 4, pp. 193–215, 1988.

[35] E. B. Baum, "The perceptron algorithm is fast for non-malicious distributions," *Neural Computation*, vol. 2, pp. 248–260, 1990.

[36] E. B. Baum, "Neural net algorithms that learn in polynomial time from examples and queries," *IEEE Trans. Neural Networks*, vol. 2, pp. 5–19, 1991.

[37] E. B. Baum and D. Haussler, "What size net gives valid generalization?" *Neural Computation*, vol. 1, pp. 151–160, 1989.

[38] G. W. Beakley and F. B. Tuteur, "Distribution-free pattern verification using statistically equivalent blocks," *IEEE Transactions on Computers*, vol. 21, pp. 1337–1347, 1972.

[39] J. Beck, "The exponential rate of convergence of error for k_n-NN nonparametric regression and decision," *Problems of Control and Information Theory*, vol. 8, pp. 303–311, 1979.

[40] G.M. Benedek and A. Itai, "Learnability with respect to fixed distributions," *Theoretical Computer Science*, 86(2), pp. 377-390, 1991.

[41] P.K. Bhattacharya and Y.P. Mack, Weak convergence of k–NN density and regression estimators with varying k and applications, *Annals of Statistics*, vol. 15, pp. 976–994, 1987.

[42] P.J. Bickel and L. Breiman, "Sums of functions of nearest neighbor distances, moment bounds, limit theorems and a goodness of fit test," *Annals of Probability*, vol. 11, pp. 185–214, 1983.

[43] A. Blum and R. L. Rivest, "Training a 3-node neural network is NP-complete," *Neural Networks*, vol. 5, pp. 117–127, 1992.

[44] A. Blumer, A. Ehrenfeucht, D. Haussler, and M.K. Warmuth, "Learnability and the Vapnik-Chervonenkis dimension," *Journal of the ACM*, vol. 36, pp. 929–965, 1989.

[45] B. Boser, I. Guyon, and V.N. Vapnik, "A training algorithm for optimal margin classifiers," In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pp. 144–152. Association for Computing Machinery, New York, NY, 1992.

[46] E.M. Braverman, "The method of potential functions," *Automation and Remote Control*, vol. 26, pp. 2130–2138, 1965.

[47] E.M. Braverman and E.S. Pyatniskii, "Estimation of the rate of convergence of algorithms based on the potential function method," *Automation and Remote Control*, vol. 27, pp. 80–100, 1966.

[48] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[49] L. Breiman, "Bias, variance, and arcing classifiers," Technical report, Department of Statistics, University of California at Berkeley, Report 460, 1996.

[50] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Pacific Grove, CA: Wadsworth & Brooks, 1984.

[51] D.S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks. *Complex Systems*, vol. 2, pp. 321–323, 1988.

[52] J. Bruck, "Harmonic analysis of polynomial threshold functions," *SIAM J. Discr. Math.*, vol. 3, no. 2, pp. 168–177, 1990.

[53] K.L. Buescher and P.R. Kumar, "Learning by canonical smooth estimation, part I: simultaneous estimation," *IEEE Trans. Automatic Control*, vol. 42, pp. 545-556, 1996.

[54] K.L. Buescher and P.R. Kumar, "Learning by canonical smooth estimation, part II: learning and choice of model complexity," *IEEE Trans. Automatic Control*, vol. 42, pp. 557-569, 1996.

[55] D. Burshtein, V. Della Pietra, D. Kanevsky, and A. Nádas, "Minimum impurity partitions," *Annals of Statistics*, vol. 20, pp. 1637–1646, 1992.

[56] T. Chen, H. Chen, and R. Liu, "A constructive proof and an extension of Cybenko's approximation theorem," in *Proceedings of the 22nd Symposium of the Interface: Computing Science and Statistics*, pp. 163–168, Alexandria, VA, 1990. American Statistical Association.

[57] X.R. Chen and L.C. Zhao, "Almost sure L_1-norm convergence for data-based histogram density estimates," *Journal of Multivariate Analysis*, vol. 21, pp. 179–188, 1987.

[58] P.A. Chou, "Optimal partitioning for classification and regression trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 340–354, 1991.

[59] A. Ciampi, "Generalized regression trees," *Computational Statistics and Data Analysis*, vol. 12, pp. 57–78, 1991.

[60] M. A. Cohen and S. Grossberg, "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Trans. Sys., Man, and Cybern.*, vol. SMC-13, pp. 815–826, 1983.

[61] G. Collomb, "Estimation de la regression par la méthode des k points les plus proches: propriétés de convergence ponctuelle," *Comptes Rendus de l'Académie des Sciences de Paris*, vol. 289, pp. 245–247, 1979.

[62] G. Collomb, *Estimation de la regression par la méthode des k points les plus proches avec noyau*, pp. 159–175, Lecture Notes in Mathematics #821, Springer-Verlag, Berlin, 1980.

[63] G. Collomb, "Estimation non parametrique de la regression: revue bibliographique," *International Statistical Review*, vol. 49, pp. 75–93, 1981.

[64] C. Cortes and V.N. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 1–25, 1995.

[65] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications to pattern recognition," *IEEE Trans. Elec. Computers*, vol. 14, pp. 326–334, 1965.

[66] T. M. Cover, "Rates of convergence of nearest neighbor decision procedures," *Proc. First Annual Hawaii Conf. on Systems Theory*, pp. 413–415, 1968.

[67] T. M. Cover, "Capacity problems for linear machines," in *Pattern Recognition*, (L. Kanal, ed.), pp. 283–289. Thompson Book Co., 1968.

[68] T.M. Cover, "Learning in pattern recognition," In S. Watanabe, editor, *Methodologies of Pattern Recognition*, pp. 111–132. Academic Press, New York, 1969.

[69] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. IT–13, pp. 21–27, 1967.

[70] T.M. Cover and T.J. Wagner, "Topics in statistical pattern recognition," *Communication and Cybernetics*, vol. 10, pp. 15–46, 1975.

[71] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control, Signals, Systems*, vol. 2, pp. 303–314, 1989.

[72] G. B. Dantzig, *Linear Programming and Extensions*, Princeton, NJ: Princeton University Press, 1963.

[73] C. Darken, M. Donahue, L. Gurvits, and E. Sontag, "Rate of approximation results motivated by robust neural network learning," in *Proceedings of the Sixth ACM Workshop on Computational Learning Theory*, pp. 303–309. Association for Computing Machinery, New York, 1993.

[74] S. Das Gupta, "Nonparametric classification rules," *Sankhya Series A*, vol. 26, pp. 25–30, 1964.

[75] B. V. Dasarathy (ed.), *Nearest Neighbor Pattern Classification Techniques*, Los Alamitos, CA: IEEE Computer Society Press, 1991.

[76] B. DasGupta, H. T. Siegelmann, and E. Sontag, "On the intractability of loading neural networks," in *Theoretical Advances in Neural Computation and Learning*, (eds. V. Roychowdhury, K-Y. Siu, and A. Orlitsky), pp. 357–390. Norwell, Massachusetts: Kluwer Academic Publishers, 1994.

[77] P. A. Devijer, "New error bounds with the nearest neighbor rule," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 749–753, 1979.

[78] P.A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[79] L. Devroye, "A universal k-nearest neighbor procedure in discrimination," in *Proceedings of the 1978 IEEE Computer Society Conference on Pattern Recognition and Image Processing*, pp. 142–147. IEEE Computer Society, Long Beach, CA, 1978.

[80] L. Devroye, "On the inequality of Cover and Hart in nearest neighbor discrimination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 75–78, 1981.

[81] L. Devroye, "On the almost everywhere convergence of nonparametric regression function estimates," *Annals of Statistics*, vol. 9, pp. 1310–1309, 1981.

[82] L. Devroye, "Necessary and sufficient conditions for the almost everywhere convergence of nearest neighbor regression function estimates," *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 61, pp. 467–481, 1982.

[83] L. Devroye, "Any discrimination rule can have an arbitrarily bad probability of error for finite sample size," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI–4, pp. 154–157, 1982.

[84] L. Devroye, "Bounds for the uniform deviation of empirical measures," *Journal of Multivariate Analysis*, vol. 12, pp. 72–79, 1982.

[85] L. Devroye, "Automatic pattern recognition: A study of the probability of error," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 530–543, 1988.

[86] L. Devroye, "Universal smoothing factor selection in density estimation: theory and practice," (with discussion), *Test*, vol. 6, pp. 223–320, 1997.

[87] L. Devroye and L. Györfi, "Distribution-free exponential bound on the L_1 error of partitioning estimates of a regression function," In F. Konecny, J. Mogyoródi, and W. Wertz, editors, *Proceedings of the Fourth Pannonian Symposium on Mathematical Statistics*, pp. 67–76, Akadémiai Kiadó, Budapest, Hungary, 1983.

[88] L. Devroye and L. Györfi, *Nonparametric Density Estimation: The L_1 View*, John Wiley, New York, 1985.

[89] L. Devroye, L. Györfi, A. Krzyżak, and G. Lugosi, "On the strong universal consistency of nearest neighbor regression function estimates," *Annals of Statistics*, vol. 22, pp. 1371–1385, 1994.

[90] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer-Verlag, New York, 1996.

[91] L. Devroye and A. Krzyżak, "An equivalence theorem for L_1 convergence of the kernel regression estimate," *Journal of Statistical Planning and Inference*, vol. 23, pp. 71–82, 1989.

[92] L. Devroye and G. Lugosi, "Lower bounds in pattern recognition and learning," *Pattern Recognition*, vol. 28, pp. 1011–1018, 1995.

[93] L. Devroye and T.J. Wagner, "A distribution-free performance bound in error estimation," *IEEE Transactions Information Theory*, vol. 22, pp. 586–587, 1976.

[94] L. Devroye and T.J. Wagner, "Nonparametric discrimination and density estimation," Technical Report 183, Electronics Research Center, University of Texas, 1976.

[95] L. Devroye and T. Wagner, "Distribution-free inequalities for the deleted and holdout error estimates," *IEEE Trans. Information Theory*, vol. 25, pp. 202-207, 1979.

[96] L. Devroye and T.J. Wagner, "Distribution-free consistency results in nonparametric discrimination and regression function estimation," *Annals of Statistics*, vol. 8, pp. 231–239, 1980.

[97] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[98] H. Drucker and C. Cortes, "Boosting decision trees," in *Advances in Neural Information Processing Systems 8*, pp. 148–156, 1996.

[99] R.M. Dudley, "Central limit theorems for empirical measures" *Ann. Probability*, vol.6, no. 6, pp. 899-929, 1978.

[100] R. M. Dudley, "Balls in R^k do not cut all subsets of $k+2$ points," *Advances in Mathematics*, vol. 31, no. 3, pp. 306–308, 1979.

[101] R.M. Dudley, *A Course on Empirical Processes*, in Lecture Notes in Mathematics, no. 1097, pp. 1-142, 1984.

[102] R.M. Dudley, S.R. Kulkarni, T.J. Richardson and O. Zeitouni, "A metric entropy bound is not sufficient for learnability," *IEEE Trans. Information Theory*, vol. 40, pp. 883-885, 1994.

[103] F. H. Eeckman, "The sigmoid nonlinearity in prepyriform cortex," in *Neural Information Processing Systems*, (D. Z. Anderson, ed.), pp. 242–248, New York: American Institute of Physics, 1988.

[104] F. H. Eeckman and W. J. Freeman, "The sigmoid nonlinearity in neural computation: An experimental approach," in *Neural Networks for Computing*, (J. S. Denker, ed.), pp. 135–145, New York: American Institute of Physics, 1986.

[105] B. Efron, "Bootstrap methods: another look at the jackknife," *Annals of Statistics*, vol. 7, pp. 1–26, 1979.

[106] B. Efron, "Estimating the error rate of a prediction rule: improvement on cross validation," *Journal of the American Statistical Association*, vol. 78, pp. 316–331, 1983.

[107] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant, "A general lower bound on the number of examples needed for learning," *Information and Computation*, vol. 82, pp. 247–261, 1989.

[108] A. Erdélyi, *Asymptotic Expansions*, New York, New York: Dover, 1956.

[109] A. Faragó and G. Lugosi, "Strong universal consistency of neural network classifiers," *IEEE Transactions on Information Theory*, vol. 39, pp. 1146–1151, 1993.

[110] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, Part II, pp. 179–188, 1936.

[111] E. Fix and J. L. Hodges, Jr., "Discriminatory Analysis—Nonparametric Discrimination: Consistency Properties," *Project 21–49–004, Report No. 4*, USAF School of Aviation Medicine, Randolf Field, TX, pp. 261–279, 1951.

[112] E. Fix and J. L. Hodges, Jr., "Discriminatory Analysis—Nonparametric Discrimination: Small Sample Performance," *Project 21–49–004, Report No. 11*, USAF School of Aviation Medicine, Randolf Field, TX, pp. 280–322, 1952.

[113] Y. Freund, "Boosting a weak learning algorithm by majority," *Information and Computation*, vol. 121, pp. 256–285, 1995.

[114] J.H. Friedman, "A recursive partitioning decision rule for nonparametric classification," *IEEE Transactions on Computers*, vol. 26, pp. 404–408, 1977.

[115] J. Fritz, "Distribution-free exponential error bound for nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. IT–21, pp. 552–557, 1975.

[116] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, New York: Academic Press, 1972.

[117] K. Fukunaga and T. E. Flick, "An optimal global nearest neighbor metric," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI–6, pp. 314–318, 1984.

[118] K. Fukunaga and D.L. Kessel, "Nonparametric Bayes error estimation using unclassified samples," *IEEE Transactions Information Theory*, vol. 19, pp. 434–440, 1973.

[119] K. Fukunaga and D. M. Hummels, "Bias of nearest neighbor estimates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI–9, pp. 103–112, 1987.

[120] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, pp. 183–192, 1989.

[121] M. R. Garey and D. J. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman, 1979.

[122] S.B. Gelfand and E.J. Delp, "On tree structured classifiers," in I.K. Sethi and A.K. Jain, editors, *Artificial Neural Networks and Statistical Pattern Recognition, Old and New Connections*, pp. 71–88. Elsevier Science Publishers, Amsterdam, 1991.

[123] S.B. Gelfand, C.S. Ravishankar, and E.J. Delp, "An iterative growing and pruning algorithm for classification tree design," in *Proceedings of the 1989 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 818–823. IEEE Press, Piscataway, NJ, 1989.

[124] S.B. Gelfand, C.S. Ravishankar, and E.J. Delp, "An iterative growing and pruning algorithm for classification tree design," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 163–174, 1991.

[125] M.P. Gessaman, "A consistent nonparametric multivariate density estimator based on statistically equivalent blocks," *Annals of Mathematical Statistics*, vol. 41, pp. 1344–1346, 1970.

[126] M.P. Gessaman and P.H. Gessaman, "A comparison of some multivariate discrimination procedures," *Journal of the American Statistical Association*, vol. 67, pp. 468–472, 1972.

[127] N. Glick, "Additive estimators for probabilities of correct classification," *Pattern Recognition*, vol. 10, pp. 211–222, 1978.

[128] P. Goldberg and M. Jerrum, "Bounding the Vapnik-Chervonenkis dimension of concept classes parametrized by real numbers," *Proc. 6th Annual ACM Conf. on Computational Learning Theory*, pp. 361–369, 1993.

[129] R.M. Goodman and P. Smyth, "Decision tree design from a communication theory viewpoint," *IEEE Transactions on Information Theory*, vol. 34, pp. 979–994, 1988.

[130] L. Gordon and R. Olshen, "Almost surely consistent nonparametric regression from recursive partitioning schemes," *Journal of Multivariate Analysis*, vol. 15, pp. 147–163, 1984.

[131] L. Gordon and R.A. Olshen, "Asymptotically efficient solutions to the classification problem," *Annals of Statistics*, vol. 6, pp. 515–533, 1978.

[132] L. Gordon and R.A. Olshen, "Consistent nonparametric regression from recursive partitioning schemes," *Journal of Multivariate Analysis*, vol. 10, pp. 611–627, 1980.

[133] W. Greblicki, A. Krzyżak, and M. Pawlak, "Distribution-free pointwise consistency of kernel regression estimate," *Annals of Statistics*, vol. 12, pp. 1570–1575, 1984.

[134] W. Greblicki and M. Pawlak, "Necessary and sufficient conditions for Bayes risk consistency of a recursive kernel classification rule," *IEEE Transactions on Information Theory*, vol. 33, pp. 408–412, 1987.

[135] H. Guo and S.B. Gelfand, "Classification trees with neural network feature extraction," *IEEE Transactions on Neural Networks*, vol. 3, pp. 923–933, 1992.

[136] L. Györfi, "On the rate of convergence of nearest neighbor rules," *IEEE Trans. Inform. Theory*, vol. IT–24, pp. 509–512, 1978.

[137] L. Györfi and Z. Györfi, "On the nonparametric estimate of a posteriori probabilities of simple statistical hypotheses," in *Colloquia Mathematica Societatis János Bolyai: Topics in Information Theory*, pp. 299–308, Keszthely, Hungary, 1975.

[138] L. Györfi and Z. Györfi, "An upperbound on the asymptotic error probability of the k-nearest neighbor rule," vol. IT-24, pp. 512–514, 1978.

[139] D.J. Hand, "Recent advances in error rate estimation," *Pattern Recognition Letters*, vol. 4, pp. 335–346, 1986.

[140] D. Haussler, "Decision theoretic generalizations of the PAC model for neural net and other learning applications," *Information and Computation*, vol. 100, pp. 78-150, 1992.

[141] D. Haussler, N. Littlestone, and M. Warmuth, "Predicting $\{0,1\}$ functions from randomly drawn points," in *Proceedings of the 29th IEEE Symposium on the Foundations of Computer Science*, pp. 100–109. IEEE Computer Society Press, Los Alamitos, CA, 1988.

[142] S. Haykin, *Neural Networks*, New York: Macmillan, 1994.

[143] D. O. Hebb, *The Organization of Behavior*, New York: Wiley, 1949.

[144] T. Hegedüs, "On training simple neural networks and small-weight neurons," in *Proc. First European Conf. Computational Learning Theory*, 1993.

[145] E.G. Henrichon and K.S. Fu, "A nonparametric partitioning procedure for pattern classification," *IEEE Transactions on Computers*, vol. 18, pp. 614–624, 1969.

[146] M. E. Hellman, "The nearest-neighbor classification rule with a reject option," *IEEE Transactions on Systems Science and Cybernetics*, vol. SSC–6, pp. 179–185, 1970.

[147] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Reading, MA: Addison-Wesley, 1991.

[148] M. W. Hirsch, "Convergent activation dynamics in continuous time networks," *Neural Networks*, vol. 2, pp. 331–349, 1989.

[149] Y-C. Ho and R. L. Kashyap, "A class of iterative procedures for linear inequalities," *J. SIAM Control*, vol. 4, pp. 112–115, 1966.

[150] S.B. Holden, "PAC-like upper bounds for the sample complexity of leave-one-out cross validation," in *Proceedings of the Nineth Annual ACM Workshop on Computational Learning Theory*, pp. 41–50, Association for Computing Machinery, New York, 1996.

[151] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational capabilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, pp. 2554–2558, 1982.

[152] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Nat. Acad. Sci. USA*, vol. 81, pp. 3088–3092, 1984.

[153] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.

[154] K. Hornik, "Some new results on neural network approximation," *Neural Networks*, vol. 6, pp. 1069–1072, 1993.

[155] M. Horváth and G. Lugosi, "A data-dependent skeleton estimate and a scale-sensitive dimension for classification," *Discrete Applied Mathematics*, Special Issue on the Vapnik-Chervonenkis dimension, to appear, 1997.

[156] A.K. Jain, R.C. Dubes, and C. Chen, "Bootstrap techniques for error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 628–633, 1987.

[157] L. Jones, "Constructive approximations for neural networks by sigmoidal functions," *Proc. IEEE*, vol. 78, no. 10, pp. 1586–1589, 1990.

[158] J. S. Judd, *Neural Network Design and the Complexity of Learning*, Cambridge, Massachusetts: MIT Press, 1990.

[159] L.N. Kanal, "Patterns in pattern recognition," *IEEE Transactions on Information Theory*, vol. 20, pp. 697–722, 1974.

[160] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[161] M. Karpinski and A. Macintyre, "Polynomial bounds for VC dimension of sigmoidal and general pfaffian neural networks," *Journal of Computer and System Science*, vol. 54, 1997.

[162] M. Kearns, Y. Mansour, A.Y. Ng, and D. Ron, "An experimental and theoretical comparison of model selection methods," in *Proceedings of the Eighth Annual ACM Workshop on Computational Learning Theory*, pp. 21–30. Association for Computing Machinery, New York, 1995.

[163] M. Kearns and D.Ron, "Algorithmic stability and sanity-check bounds for leave-one-out cross validation," in *Proceedings of the Tenth Annual ACM Workshop on Computational Learning Theory*, Association for Computing Machinery, New York, 1997.

[164] M. Kearns and U. Vazirani, *Introduction to Computational Learning Theory*, M.I.T. Press, Cambridge, MA, 1994.

[165] T. Kohonen, *Self-Organization and Associative Memory*, New York: Springer-Verlag, 1988.

[166] P. Koiran and E.D. Sontag, "Neural networks with quadratic VC dimension," *Journal of Computer and System Science*, vol. 54, 1997.

[167] A.N. Kolmogorov and V.M. Tihomirov, "ϵ-Entropy and ϵ-capacity of sets in functional spaces," *Amer. Math. Soc. Transl.*, vol. 17, pp. 277-364, 1961.

[168] J. Komlós and R. Paturi, "Convergence results in an associative memory model," *Neural Networks*, vol. 1, pp. 239–250, 1988.

[169] J. Komlós and R. Paturi, "Effect of connectivity in associative memory models," *Technical Report CS88–131*, University of California, San Diego, 1988; to appear *Journal of Computer and Systems Sciences*.

[170] A. Krzyżak, "On exponential bounds on the Bayes risk of the kernel classification rule," *IEEE Transactions on Information Theory*, vol. 37, pp. 490–499, 1991.

[171] A. Krzyżak and T. Linder, "Radial basis function networks and complexity regularization in function learning," *IEEE Transactions on Neural Networks*, to appear, 1998.

[172] A. Krzyżak, T. Linder, and G. Lugosi, "Nonparametric estimation and classification using radial basis function nets and empirical risk minimization," *IEEE Transactions on Neural Networks*, vol. 7, pp. 475–487, 1996.

[173] A. Krzyżak and M. Pawlak, "Almost everywhere convergence of recursive kernel regression function estimates," *IEEE Transactions on Information Theory*, vol. 31, pp. 91–93, 1984.

[174] S.R. Kulkarni, S.K. Mitter, and J.N. Tsitsiklis, "Active learning using arbitrary binary valued queries," *Machine Learning*, vol. 11, pp. 23-35, 1993.

[175] S.R. Kulkarni and M. Vidyasagar, "Learning classes of decision rules under a family of probability measures," *IEEE Trans. Information Theory*, vol. 43, pp. 154-166, 1997.

[176] P.A. Lachenbruch, "An almost unbiased method of obtaining confidence intervals for the probability of misclassification in discriminant analysis," *Biometrics*, vol. 23, pp. 639–645, 1967.

[177] M. Ledoux and M. Talagrand, *Probability in Banach Space*, Springer-Verlag, New York, 1991.

[178] M. Leshno, V. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a polynomial activation function can approximate any function. *Neural Networks*, vol. 6, pp. 861–867, 1993.

[179] X. Li and R.C. Dubes, "Tree classifier design with a permutation statistic," *Pattern Recognition*, vol. 19, pp. 229–235, 1986.

[180] J. H. Lin and J. S. Vitter, "Complexity results on learning by neural nets," *Machine Learning*, vol. 6, pp. 211–230, 1991.

[181] W.Y. Loh and N. Vanichsetakul, "Tree-structured classification via generalized discriminant analysis," *Journal of the American Statistical Association*, vol. 83, pp. 715–728, 1988.

[182] G. Lugosi, "Improved upper bounds for probabilities of uniform deviations," *Statistics and Probability Letters*, vol. 25, pp. 71–77, 1995.

[183] G. Lugosi and A. Nobel, "Consistency of data-driven histogram methods for density estimation and classification," *Annals of Statistics*, vol. 24, pp. 687–706, 1996.

[184] G. Lugosi and K. Zeger, "Nonparametric estimation via empirical risk minimization," *IEEE Transactions on Information Theory*, vol. 41, pp. 677–678, 1995.

[185] G. Lugosi and K. Zeger, "Concept Learning Using Complexity Regularization," *IEEE Transactions on Information Theory*, vol. 42, pp. 48-54, 1996.

[186] A.L. Lunts and V.L. Brailovsky, "Evaluation of attributes obtained in statistical decision rules," *Engineering Cybernetics*, vol. 3, pp. 98–109, 1967.

[187] W. Maass, "Bounds for the computational power and learning complexity of analog neural nets," *Proc. 25th Annual ACM Symp. on Theory of Computing*, pp. 335–344, 1993.

[188] M. MacIntyre and E. D. Sontag, "Finiteness results for sigmoidal neural networks," *Proc. 25th Annual ACM Symp. on Theory of Computing*, pp. 325–334, 1993.

[189] Y.P. Mack, "Local properties of k–nearest neighbor regression estimates," *SIAM Journal on Algebraic and Discrete Methods*, vol. 2, pp. 311–323, 1981.

[190] P. Massart, "The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality," *Annals of Probability*, vol. 18, pp. 1269–1283, 1990.

[191] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in neural activity," *Bull. Math. Biophys.*, vol. 5, pp. 115–133, 1943.

[192] R. J. McEliece, E. C. Posner, E. R. Rodemich, and S. S. Venkatesh, "The capacity of the Hopfield associative memory," *IEEE Transactions on Information Theory*, vol. IT–33, pp. 461–482, 1987.

[193] G.J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, John Wiley, New York, 1992.

[194] R. Meir, "Performance bounds for nonlinear time series prediction," in *Proceedings of the Tenth Annual ACM Workshop on Computational Learning Theory*, Association for Computing Machinery, New York, 1997.

[195] W.S. Meisel and D.A. Michalopoulos, "A partitioning algorithm with application in pattern classification and the optimization of decision tree," *IEEE Transactions on Computers*, vol. 22, pp. 93–103, 1973.

[196] C. Michel-Briand and X. Milhaud, "Asymptotic behavior of the AID method," Technical report, Université Montpellier 2, Montpellier, 1994.

[197] J. Mielniczuk and J. Tyrcha, "Consistency of multilayer perceptron regression estimators," *Neural Networks*, to appear, 1993.

[198] M. L. Minsky and S. A. Papert, *Perceptrons*, Cambridge, MA: MIT Press, 1988.

[199] R. Mizoguchi, M. Kizawa, and M. Shimura, "Piecewise linear discriminant functions in pattern recognition," *Systems-Computers-Controls*, vol. 8, pp. 114–121, 1977.

[200] D.S. Modha and E. Masrym "Minimum complexity regression estimation with weakly dependent observations," *IEEE Transactions on Information Theory*, to appear, 1996.

[201] J. Moody and J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.

[202] J.N. Morgan and J.A. Sonquist, "Problems in the analysis of survey data, and a proposal," *Journal of the American Statistical Association*, vol. 58, pp. 415–434, 1963.

[203] E.A. Nadaraya, "On estimating regression," *Theory of Probability and its Applications*, vol. 9, pp. 141–142, 1964.

[204] E.A. Nadaraya, "Remarks on nonparametric estimates for density functions and regression curves," *Theory of Probability and its Applications*, vol. 15, pp. 134–137, 1970.

[205] B.K. Natarajan, *Machine Learning: A Theoretical Approach*, Morgan Kaufmann, San Mateo, CA, 1991.

[206] C. Newman, "Memory capacity in neural network models: rigorous lower bounds," *Neural Networks*, vol. 1, pp. 223–238, 1988.

[207] N. J. Nilsson, *The Mathematical Foundations of Learning Machines*, San Mateo, California: Morgan-Kaufmann, 1990.

[208] A. Nobel, "Histogram regression using data-dependent partitions," *Annals of Statistics*, vol. 24, pp. 1084–1105, 1996.

[209] A. Nobel, "Recursive partitioning to reduce distortion," *IEEE Transactions on Information Theory*, vol. 43, pp. 1122–1133, 1997.

[210] R.A. Olshen, "Comments on a paper by C.J. Stone," *Annals of Statistics*, vol. 5, pp. 632–633, 1977.

[211] Y. Park and J. Sklansky, "Automated design of linear tree classifiers," *Pattern Recognition*, vol. 23, pp. 1393–1412, 1990.

[212] J.M. Parrondo and C. Van den Broeck, "Vapnik-Chervonenkis bounds for generalization," *J. Phys. A*, vol. 26, pp. 2211–2223, 1993.

[213] E. Parzen, "On the estimation of a probability density function and the mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[214] E.A. Patrick, "Distribution-free minimum conditional risk learning systems," Technical Report TR-EE-66-18, Purdue University, Lafayette, IN, 1966.

[215] E.A. Patrick and F.P. Fisher, "Introduction to the performance of distribution-free conditional risk learning systems," Technical Report TR-EE-67-12, Purdue University, Lafayette, IN, 1967.

[216] H.J. Payne and W.S. Meisel, "An algorithm for constructing optimal binary decision trees," *IEEE Transactions on Computers*, vol. 26, pp. 905–916, 1977.

[217] P. Peretto and J.-J. Niez, "Stochastic dynamics of neural networks," *IEEE Trans. Sys., Man, and Cybern.*, vol. SMC-16, pp. 73–83, 1986.

[218] F. J. Pineda, "Recurrent backpropagation and the dynamical approach to adaptive neural computation," *Neural Computation*, vol. 1, pp. 161–172, 1989.

[219] L. Pitt and L. G. Valiant, "Computational limitations of learning from examples," *J. ACM*, vol. 35, pp. 965–984, 1988.

[220] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, pp. 1481–1497, 1990.

[221] D. Pollard, *Convergence of Stochastic Processes*, Springer-Verlag, 1984.

[222] D. Pollard, "Asymptotics via empirical processes," *Statistical Science*, vol. 4, pp. 341–366, 1989.

[223] D. Pollard, *Empirical Processes: Theory and Applications*, NSF-CBMS Regional Conference Series in Probability and Statistics, Institute of Mathematical Statistics, Hayward, CA, 1990.

[224] M.J.D. Powell, "Radial basis functions for multivariable interpolation: a review," *Algorithms for Approximation*, Clarendon Press, Oxford, 1987.

[225] D. Psaltis, R. R. Snapp, and S. S. Venkatesh, "On the finite sample performance of the nearest neighbour classifier," *IEEE Trans. Inform. Theory*, vol. IT–40, pp. 820–837, 1994.

[226] C.P. Quesenberry and M.P. Gessaman, "Nonparametric discrimination using tolerance regions," *Annals of Mathematical Statistics*, vol. 39, pp. 664–673, 1968.

[227] S. Qing-Yun and K.S. Fu, "A method for the design of binary tree classifiers," *Pattern Recognition*, vol. 16, pp. 593–603, 1983.

[228] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, 1993.

[229] J.R. Quinlan, "Bagging, boosting, and C4.5," in *Proceedings of the 13th National Conference on Artificial Intelligence*, pp. 725–730, 1996.

[230] B.D. Ripley, "Statistical aspects of neural networks," in O.E. Barndorff-Nielsen, J.L. Jensen, and W.S. Kendall, editors, *Networks and Chaos—Statistical and Probabilistic Aspects*, pp. 40–123, London, U.K., Chapman and Hall, 1993.

[231] B.D. Ripley, "Neural networks and related methods for classification," *Journal of the Royal Statistical Society*, vol. 56, pp. 409–456, 1994.

[232] J. Rissanen, *Stochastic Complexity in Statistical Inquiry*, Series in Computer Science Vol. 15, World Scientific, 1989.

[233] W.H. Rogers and T.J. Wagner, "A finite sample distribution-free performance bound for local discrimination rules," *Annals of Statistics*, vol. 6, pp. 506–514, 1978.

[234] F. Rosenblatt, *Principles of Neurodynamics*, Washington, DC: Spartan Books, 1962.

[235] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *Annals of Mathematical Statistics*, vol. 27, pp. 832–837, 1956.

[236] E.M. Rounds, "A combined nonparametric approach to feature selection and binary decision tree design," *Pattern Recognition*, vol. 12, pp. 313–317, 1980.

[237] R.M. Royall, *A Class of Nonparametric Estimators of a Smooth Regression Function*, PhD thesis, Stanford University, Stanford, CA, 1966.

[238] V. Roychowdhury, K-Y. Siu, and A. Orlitsky, eds., *Theoretical Advances in Neural Computation and Learning*, Norwell, MA: Kluwer Academic Publishers, 1994.

[239] W. Rudin, *Real and Complex Analysis*, New York: McGraw-Hill, 1974.

[240] D. E. Rumelhart and J. L. McClelland, eds., *Parallel Distributed Processing*, vol. 1. Cambridge, MA: MIT Press, 1986.

[241] A. Sakurai, "Tighter bounds of the VC-dimension of three-layer networks," *Proc. WCNN*, vol. 3, pp. 540–543, 1993.

[242] N. Sauer, "On the density of families of sets," *Journal of Combinatorial Theory Series A*, vol. 13, pp. 145–147, 1972.

[243] R.E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, pp. 197–227, 1990.

[244] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," 1997.

[245] L. Schläffli, *Gesammelte Mathematische Abhandlungen I*, Basel, Switzerland: Verlag Birkhäuser, pp. 209–212, 1950.

[246] D. Schuurmans, "Characterizing rational versus exponential learning curves," in *Computational Learning Theory: Second European Conference, EuroCOLT'95*, pp. 272–286, Springer Verlag, 1995.

[247] G. Sebestyen, *Decision-Making Processes in Pattern Recognition*, Macmillan, New York, 1962.

[248] I.K. Sethi, "Decision tree performance enhancement using an artificial neural network interpretation," in I.K. Sethi and A.K. Jain, editors, *Artificial Neural Networks and Statistical Pattern Recognition, Old and New Connections*, pp. 71–88, Amsterdam, Elsevier Science Publishers, 1991.

[249] I.K. Sethi and G.P.R. Sarvarayudu, "Hierarchical classifier design using mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 441–445, 1982.

[250] J. Shawe-Taylor, M. Anthony, and N.L. Biggs, "Bounding sample size with the Vapnik-Chervonenkis dimension," *Discrete Applied Mathematics*, vol. 42, pp. 65–73, 1993.

[251] J. Shawe-Taylor, P.L. Bartlett, R.C. Williamson, and M. Anthony, "A framework for structural risk minimization," in *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pp. 68–76. Association of Computing Machinery, New York, 1996.

[252] J. Shawe-Taylor, P.L. Bartlett, R.C. Williamson, and M. Anthony, "Structural risk minimization over data-dependent hierarchies," *IEEE Transactions on Information Theory*, to appear, 1997.

[253] R. D. Short and K. Fukunaga, "The optimal distance measure for nearest neighbor classification," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 622–627, 1981.

[254] H.U. Simon, "The Vapnik-Chervonenkis dimension of decision trees with bounded rank," *Information Processing Letters*, vol. 39, pp. 137-141, 1991.

[255] H.U. Simon, "General lower bounds on the number of examples needed for learning probabilistic concepts," in *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, pp. 402–412. Association for Computing Machinery, New York, 1993.

[256] H.U. Simon, "Bounds on the number of examples needed for learning functions," *SIAM Journal on Computing*, vol. 26, pp. 751–763, 1997.

[257] J. Sklansky and Michelotti, "Locally trained piecewise linear classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, pp. 101–111, 1980.

[258] R. R. Snapp and S. S. Venkatesh, "Asymptotic derivation of the finite-sample risk of the k-nearest neighbor classifier," *Tech. Rep.*, TR-UVM-CS-1997-001, Dept. of Computer Science, University of Vermont, 1997.

[259] R. R. Snapp and S. S. Venkatesh,"Asymptotic expansions of the k nearest neighbour risk," *Ann. Stat.*, to appear, 1998.

[260] E. D. Sontag, "Feedforward nets for interpolation and classification," *J. Comp. Syst. Sci.*, vol. 45, pp. 20–48, 1992.

[261] D.F. Specht, "Generation of polynomial discriminant functions for pattern classification," *IEEE Transactions on Electronic Computers*, vol. 15, pp. 308–319, 1967.

[262] D.F. Specht, "Probabilistic neural networks and the polynomial Adaline as complementary techniques for classification," *IEEE Transactions on Neural Networks*, vol. 1, pp. 111–121, 1990.

[263] C. Spiegelman and J. Sacks, "Consistent window estimation in nonparametric regression," *Annals of Statistics*, vol. 8, pp. 240–246, 1980.

[264] G. Stengle and J. Yukich, "Some new Vapnik-Chervonenkis classes," *Annals of Statistics*, vol. 17, pp. 1441-1446, 1989.

[265] D.S. Stoller, "Univariate two-population distribution-free discrimination," *Journal of the American Statistical Association*, vol. 49, pp. 770–777, 1954.

[266] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society*, vol. 36, pp. 111–147, 1974.

[267] C. J. Stone, "Consistent nonparametric regression," *Ann. Stat.*, vol. 5, 1977, pp. 595–645.

[268] W. Stute, "Asymptotic normality of nearest neighbor regression function estimates," *Annals of Statistics*, vol. 12, pp. 917–926, 1984.

[269] M. Talagrand, "Sharper bounds for Gaussian and empirical processes," *Annals of Probability*, vol. 22, pp. 28–76, 1994.

[270] J.L. Talmon, "A multiclass nonparametric partitioning algorithm," in E.S. Gelsema and L.N. Kanal, editors, *Pattern Recognition in Practice II*, Elsevier Science Publishers, Amsterdam, 1986.

[271] G.T. Toussaint, "Bibliography on estimation of misclassification," *IEEE Transactions on Information Theory*, vol. 20, pp. 472–479, 1974.

[272] G.T. Toussaint and R.W. Donaldson, "Algorithms for recognizing contour-traced hand-printed characters," *IEEE Transactions on Computers*, vol. 19, pp. 541–546, 1970.

[273] L.G. Valiant, "A theory of the learnable," *Comm. ACM*, vol. 27, no. 11, pp. 1134-1142, 1984.

[274] V.N. Vapnik, *Estimation of Dependencies Based on Empirical Data*, Springer-Verlag, New York, 1982.

[275] V.N. Vapnik. *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1991.

[276] V.N. Vapnik and A.Ya. Chervonenkis, "On the uniform convergence of relative frequencies to their probabilities," *Theory of Prob. and its Appl.*, vol. 16, no. 2, pp. 264-280, 1971.

[277] V.N. Vapnik and A.Ya. Chervonenkis. *Theory of Pattern Recognition*, Nauka, Moscow, 1974, (in Russian); German translation: *Theorie der Zeichenerkennung*, Akademie Verlag, Berlin, 1979.

[278] V.N. Vapnik and A.Ya. Chervonenkis, "Necessary and sufficient conditions for the uniform convergence of means to their expectations," *Theory of Prob. and its Appl.*, vol. 26, no. 3, pp. 532-553, 1981.

[279] S. S. Venkatesh, "Computation and learning in the context of neural network capacity," in *Neural Networks for Perception*, vol. 2, (H. Wechsler, ed.). San Diego, CA: Academic Press, 1992.

[280] S. S. Venkatesh, "Robustness in neural computation: Random graphs and sparsity," *IEEE Transactions on Information Theory*, vol. 38, no. 3, pp. 1114–1118, 1992.

[281] S. S. Venkatesh, "Directed drift: A new linear threshold algorithm for learning binary weights on-line," *J. Comp. Sys. Sciences*, vol. 46, pp. 198–217, 1993.

[282] S. S. Venkatesh, "Connectivity versus capacity in the Hebb rule," in *Theoretical Advances in Neural Computation and Learning*, (V. Roychowdhury, K-Y. Siu, and A. Orlitsky, eds.). Norwell, MA: Kluwer Academic Publishers, 1994.

[283] S. S. Venkatesh and P. Baldi, "Programmed Interactions in Higher-Order Neural Networks: Maximal Capacity," *J. Complexity*, vol. 7, no. 3, pp. 316–337, 1991.

[284] S. S. Venkatesh and P. Baldi, "Programmed Interactions in Higher-Order Neural Networks: The Outer-Product Algorithm," *J. Complexity*, vol. 7, no. 4, pp. 443–479, 1991.

[285] M. Vidyasagar, *A Theory of Learning and Generalization*, Springer-Verlag, 1997.

[286] V. H. Vu, "On the infeasibility of training neural networks with small squared error," in *Proc. Conf. Neural Information Processing Systems*, Denver, Colorado, 1997.

[287] T. J. Wagner, "Convergence of the Nearest Neighbor Rule," *IEEE Trans. Inform. Theory*, vol. IT–17, pp. 566–571, 1971.

[288] C. Wang, *A Theory of Generalization in Learning Machines with Neural Network Applications*. Ph.D. Thesis, University of Pennsylvania, 1995.

[289] C. Wang and S. S. Venkatesh, "Machine size selection for optimal generalisation," *Workshop on Applications of Descriptional Complexity to Inductive, Statistical, and Visual Inference*, New Brunswick, New Jersey, July 1994.

[290] C. Wang, S. S. Venkatesh, and S. J. Judd, "When to stop: On optimal stopping and effective machine size in learning," *Conference on Neural Information Processing Systems*, Denver, Colorado, November 1993.

[291] C. Wang, S. S. Venkatesh, and S. J. Judd, "Optimal stopping and effective machine complexity in learning," *IEEE Trans. Inform. Theory*, to appear.

[292] Q.R. Wang and C.Y. Suen, "Analysis and design of decision tree based on entropy reduction and its application to large character set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 406–417, 1984.

[293] M. T. Wasan, *Stochastic Approximation*, New York: Cambridge University Press, 1969.

[294] G.S. Watson, "Smooth regression analysis," *Sankhya Series A*, vol. 26, pp. 359–372, 1964.

[295] R. S. Wenocur and R. M. Dudley, "Some special Vapnik-Chervonenkis classes," *Disc. Math.*, vol. 33, pp. 313–318, 1981.

[296] H. White, "Connectionist nonparametric regression: multilayer feedforward networks can learn arbitrary mappings," *Neural Networks*, vol. 3, pp. 535–549, 1990.

[297] H. White, "Nonparametric estimation of conditional quantiles using neural networks," In *Proceedings of the 23rd Symposium of the Interface: Computing Science and Statistics*, pp. 190–199, Alexandria, VA, 1991. American Statistical Association.

[298] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE Wescon Conv. Record*, Part 4, pp. 96–104, 1960.

[299] C.T. Wolverton and T.J. Wagner. Recursive estimates of probability densities. *IEEE Transactions on Systems, Science and Cybernetics*, vol. 5, p. 307, 1969.

[300] Y. Yang and A.R. Barron, "An asymptotic property of model selection criteria," *IEEE Transactions on Information Theory*, to appear, 1998.

[301] L.C. Zhao, "Exponential bounds of mean error for the nearest neighbor estimates of regression functions," *Journal of Multivariate Analysis*, vol. 21, pp. 168–178, 1987.

[302] L.C. Zhao, "Exponential bounds of mean error for the kernel estimates of regression functions," *Journal of Multivariate Analysis*, vol. 29, pp. 260–273, 1989.

[303] L.C. Zhao, P.R. Krishnaiah, and X.R. Chen, "Almost sure L_r-norm convergence for data-based histogram estimates," *Theory of Probability and its Applications*, vol. 35, pp. 396–403, 1990.