Generalization Error Estimates and Training Data Valuation

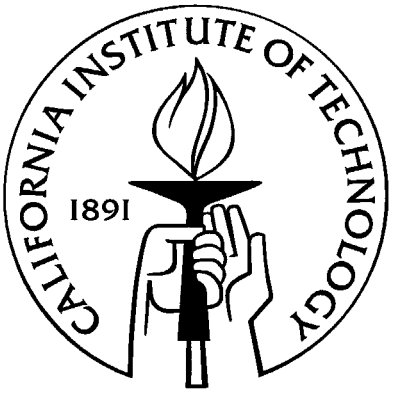
Thesis by

Alexander Nicholson

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy



California Institute of Technology

Pasadena, California

2002

(Submitted May 16, 2002)

Acknowledgements

I owe a great debt of gratitude to many people who have helped make this work possible. First and foremost I thank my advisor, Yaser Abu-Mostafa, who has provided me with a great deal of support, guidance and instruction throughout my time at Caltech.

The work on the bin model, which underlies this entire thesis, is built largely upon work done by Xubo Song, and was done with much help and input from Malik Magdon-Ismail.

My fellow members of the Learning Systems Group have provided me with many valuable discussions and helpful suggestions for my work. In addition to those already mentioned, my thanks go to Joe Sill, Ling Li, Zehra Cataltepe, Amir Atiya, Amrit Pratap, Genti Buzi and Dustin Boswell.

Finally, I thank my parents and family for their continuing love and support.

Abstract

This thesis addresses several problems related to generalization in machine learning systems. We introduce a theoretical framework for studying learning and generalization. Within this framework, a closed form is derived for the expected generalization error that estimates the out-of-sample performance in terms of the in-sample performance. We consider the problem of overfitting and show that, using a simple exhaustive learning algorithm, overfitting does not occur. These results do not assume a particular form of the target function, input distribution or learning model, and hold even with noisy data sets. We apply our analysis to practical learning systems, illustrate how it may be used to estimate out-of-sample errors in practice, and demonstrate that the resulting estimates improve upon errors estimated with a validation set for real world problems.

Based on this study of generalization, we develop a technique for quantitative valuation of training data. We demonstrate that this valuation may be used to select training sets that improve generalization performance. With a reasonable prior over target functions, it further allows us to estimate the level of noise in a data set and provides for detection and correction of noise in individual examples. Finally, this data valuation can be used to classify new examples, yielding a new learning algorithm that is shown to be relatively robust to noise.

Contents

Acknowledgements iii

Abstract iv

1 Introduction 1

1.1 Learning Systems 1

1.2 Generalization 3

1.2.1 Overfitting 3

1.2.2 Generalization Theory 5

1.3 Contributions 7

2 The Bin Model 8

2.1 Introduction to the Bin Model 8

2.2 Exhaustive Learning 9

2.3 Terminology 10

2.4 Generalization 11

2.5 Noise in the Bin Model 15

2.5.1 Uniform Noise 15

2.5.2 Input-Dependent Noise 18

2.5.3 The Effect of Noise on Generalization 20

2.6 Extensions of the Model 23

2.7 Discussion 24

3 Generalization in Practical Learning Systems 25

3.1 Linear Models 25
3.2 Neural Networks 28
3.2.1 Model Symmetry 30
3.2.2 Model Decomposition 31
3.2.3 Levels of Degeneracy 33
3.3 Two-Stage Learning 34
3.3.1 Practical Implementation 37
3.4 Experiments 38
3.4.1 Artificial Data 38
3.4.2 Ionosphere Radar Data 40
3.5 Training Set Dependence 43
3.6 Discussion 44

4 Data Valuation **45**
4.1 Overfitting 45
4.2 Error Correlations 48
4.3 Data Set Selection 50
4.3.1 ρ Estimation 51
4.3.2 Experimental Results 54
4.4 Noise 56
4.4.1 Noise Estimation 58
4.5 Application to Image Denoising 61
4.6 Linear Models and Financial Data 69
4.6.1 ρ-Distributions for Linear Models 69
4.6.2 Financial Time Series 75
4.6.3 Experiments 76
4.7 Discussion 78

5 ρ Learning **80**
5.1 The ρ Learning Idea 80
5.2 Edge Detection Example 81

5.3 Financial Time Series Prediction 82
5.4 Discussion 83

6 Conclusion **84**

List of Figures

1.1 The learning process. 3
1.2 Overfitting by polynomials. 4
1.3 Overtraining a neural network. 5

2.1 The learning problem modelled as a set of bins. 9
2.2 An example π-distribution and its generalization curve. 13
2.3 Generalization curves for a problem with uniform π-distribution. . . . 14
2.4 A binary symmetric channel. 16
2.5 Effects of noise on the π-distribution and generalization curve. 17
2.6 $\Delta_N(\tilde{\pi}, \nu)$ for varying noise levels. 18

3.1 π-distributions for a linear learning model and target. 27
3.2 Generalization curve for 20-dimensional linear model. 28
3.3 Empirical π-distributions. 29
3.4 Degeneracy of neural network models. 33
3.5 The two-stage learning scenario. 35
3.6 Example of two-stage learning. 36
3.7 Estimated π-distribution for a toy problem. 39
3.8 Average errors for a two-stage learning process. 39
3.9 π-distribution estimate for the ionosphere radar problem. 41
3.10 Average errors for the ionosphere radar problem. 42

4.1 ρ values for a simple problem. 50
4.2 Generalization error improvement using ρ based data selection. 55
4.3 Error improvements with data rejection and reclassification. 56

4.4 Noiseless and noisy ρ-distributions. 59
4.5 Black-and-white image row target function. 62
4.6 Natural image row learning model. 63
4.7 Clean and noisy targets for image denoising experiment. 64
4.8 Noiseless ρ distribution. 65
4.9 ρ_t as a function of noise level. 65
4.10 Estimation of the level of noise in the image. 66
4.11 Noise estimates as a function of actual noise level. 67
4.12 Images restored by ρ cleaning and median filtering. 68
4.13 SNR improvement for image denoising techniques. 69
4.14 Evaluation of $\rho(x)$ in two dimensions. 70
4.15 Evaluation of $\rho(x)$ in d dimensions. 72
4.16 ρ-distributions for a linear model. 74
4.17 Empirical distribution of $\widehat{\rho}$ for foreign exchange data. 77

5.1 Performance comparison for an edge detection task. 82

List of Tables

3.1 Average errors for the ionosphere radar classification problem. 42

4.1 Comparison of errors for ρ-based data selection. 78

5.1 Possible scenarios for ρ learning. 81

Chapter 1

Introduction

Machine learning systems are now quite commonly used to solve practical problems. Systems that learn from examples can be used to deal with problems for which solutions are unknown or are not mathematically well defined. Learning can also give efficient solutions (or approximate solutions) when known algorithms are computationally inefficient. The theoretical analysis of learning systems is inherently probabilistic and statistical [White 1989], and a great deal of the literature deals with *statistical learning theory* [Devroye *et al.* 1996; Vapnik 1998]. We draw on this statistical approach to address problems related to the generalization performance of learning systems.

1.1 Learning Systems

In order to study the nature of machine learning, we must first have a precise definition of what constitutes a learning system. We begin with a description of the learning process and introduce the notation that will be used throughout this thesis.

The goal of a learning system is to discover a function $f : \mathcal{X} \to \mathcal{Y}$. The *input space* $\mathcal{X}$ and *output space* $\mathcal{Y}$ are known, but f is by definition unknown and is referred to as the *target function.* For example, $\mathcal{X}$ and $\mathcal{Y}$ may be spaces of real numbers, with the target function being a mathematical relation like $f(x) = x^2 - 1$. Or elements of $\mathcal{X}$ may represent sets of symptoms, with $f(x)$ indicating the presence or absence of a certain disease.

The information available to the learning system is encapsulated in pairs of inputs and outputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$. For each pair, $y = f(x) + \eta$, where η is some random variable and is referred to as *noise*. These pairs are called *examples*, as they are (possibly noisy) examples of what the target function does. In general, learning systems may receive examples sequentially (this model is common in reinforcement learning [Sutton and Barto 1998]) or may be able to request an example with a specified value chosen from $\mathcal{X}$ (as in active learning [Cohn *et al.* 1995] or query based learning [Angluin 1987]). We restrict our discussion to the supervised learning model, in which a fixed finite set of examples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ is provided to the learning system. $\mathcal{D}$ is referred to as the *training set*, and it is assumed that the x_i are drawn randomly and independently from some *input distribution* p_X over $\mathcal{X}$.

Given a training set, the learning system must select a *hypothesis* g as a guess for the target function. The set $\mathcal{G}$ of candidate hypotheses is referred to as the *learning model*. In general, the learning model may be finite or infinite, countable or uncountable and may or may not contain the target function f. For example, if $\mathcal{Y} = \mathbf{R}$, then we might choose the set $\{g(x) \equiv c | c \in [0, 1]\}$ of constant functions as our learning model. Learning model selection is ad hoc, and normally is done so that g has a simple parametric representation.

The procedure by which the learning system selects a hypothesis from the learning model is referred to as the *learning algorithm*. Figure 1.1 illustrates the entire learning process. A learning algorithm $\mathcal{A}$ takes the training set $\mathcal{D}$ and the learning model $\mathcal{G}$ and outputs a hypothesis $g \in \mathcal{G}$, usually based on some performance criterion on the training set. For example, the learning algorithm may attempt to find the hypothesis in $\mathcal{G}$ that minimizes the mean squared error $\langle (g(x) - y)^2 \rangle_{\mathcal{D}}$.[1]

In the end, the performance of a hypothesis is measured by a pointwise *error function* $e : \mathcal{Y} \times \mathcal{Y} \to \mathbf{R}$, however, the exact form of e may or may not be available to the learning algorithm. For a hypothesis g the mean error on the training set

[1] We use $p_X(\cdot)$ to denote the probability distribution for the random variable X. We write $\Pr[\cdot]$ for the probabilities of discrete events. $\mathrm{E}_X[\cdot]$ denotes the expectation of a quantity with respect to the random variable X. When a sample is available, we will denote by $\langle \cdot \rangle_S$ the sample mean taken over the set S.

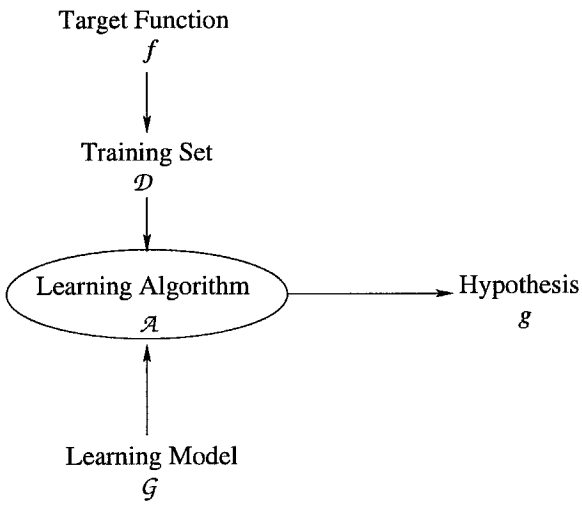


Figure 1.1: The learning process. The information about f available to the learning algorithm is only what is contained in the training set. Given the training set and a learning model, the learning algorithm outputs a hypothesis function g.

$\nu(g) = \langle e(g(x_i), y_i) \rangle_{\mathcal{D}}$ is called the *in-sample error* or *training error*. The expected performance on the entire input space $\pi(g) = E_x[e(g(x), f(x))]$ is called the *out-of-sample error* or *generalization error*, and is the real quantity of interest.

1.2 Generalization

Once we have selected a hypothesis g from our learning model, we are interested in how it will perform on new data, that is, how it will generalize. The generalization error cannot be determined exactly without knowing the target function, but we would like to have an estimate or bound. Unfortunately, it may seem that the only information available to us is what is contained in the training set, and a hypothesis that has a low in-sample error is not guaranteed to perform well out-of-sample.

1.2.1 Overfitting

A common problem observed in practice is that of *overfitting* the data. With a sufficiently complex learning model, we will be able to fit any data set without necessarily learning anything about the underlying target. This idea is illustrated in Figure 1.2.

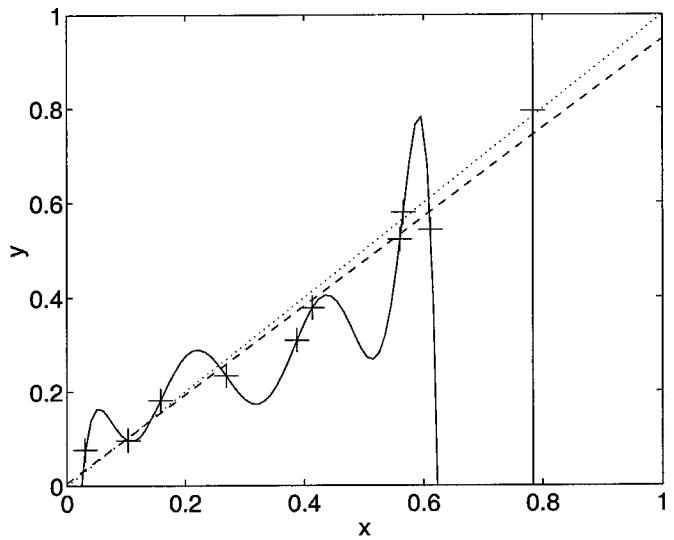


Figure 1.2: Overfitting by polynomials. Noisy examples (marked by +) are generated from the function $f(x) = x$ (shown as a dotted line). The data are fit with linear (dashed line) and twentieth order (solid curve) models.

The training set (indicated by + in the figure) is generated as noisy examples from the function $f(x) = x$. With a linear learning model, the best fit (in terms of mean squared error) is shown by the dashed line. The error does not go to zero, and the resulting line does not pass through any of the training data. With a learning model of twentieth-order polynomials, however, we can fit the data perfectly and can find a hypothesis that gives zero in-sample error (the solid curve in Figure 1.2). Even without knowledge of the generating function, we would tend to prefer the linear fit for its simplicity, and suspect that the twentieth order fit cannot generalize to points not in the training set. This preference for a simple explanation is embodied in Occam's Razor, and learning algorithms explicitly based on Occam's Razor [Blumer *et al.* 1987] and information theoretic simplicity [Rissanen 1978] have been developed.

A slightly different but related idea is that of *overtraining*. For a powerful learning model, a learning algorithm that does a sequential optimization ("training") may perform best if we limit how long it runs. Figure 1.3 shows error curves for training a neural network on the noisy data from Figure 1.2. In essence, the learning model is capable of overfitting the data, but to do so a sufficiently powerful algorithm is

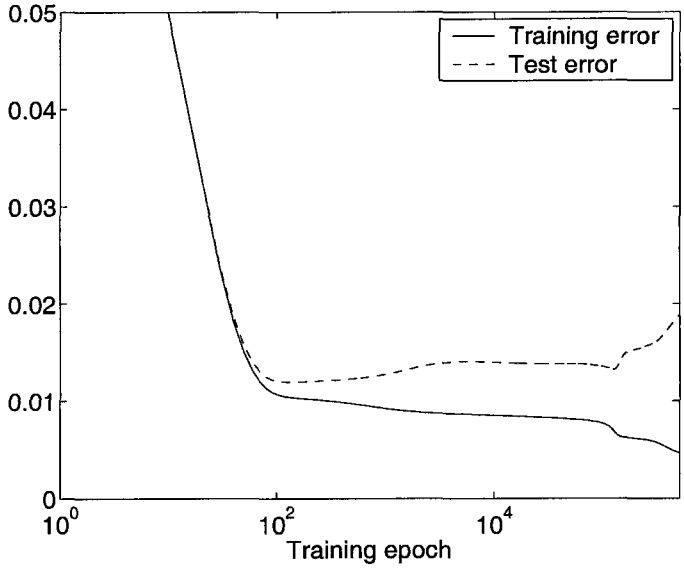


Figure 1.3: Overtraining a neural network. The in-sample and out-of-sample errors decrease at a similar rate in the early epochs of training. As training progresses, the in-sample error continues to decrease, while the out-of-sample error increases.

required.

Overfitting and overtraining are difficulties inherent in the machine learning problem. Expending more effort in the training process may degrade performance. The in-sample error alone is not necessarily a good indicator of future performance, so for a learning system to be of practical interest, we must have some further way of characterizing the generalization ability.

1.2.2 Generalization Theory

There are many heuristics aimed at improving generalization and many theoretical results relating in-sample and out-of-sample errors.

The worst-case performance analysis of Vapnik and Chervonenkis yields some important and widely applicable results [Vapnik and Chervonenkis 1971; Abu-Mostafa 1989; Parrondo and Van den Broeck 1993; Vapnik 1995]. This so-called VC theory provides assurance that the largest deviation between in-sample and out-of-sample

error will be small with high probability, that is,

$$\Pr[\sup_{g \in \mathcal{G}} |\pi(g) - \nu(g)| > \varepsilon] < \delta(\varepsilon, N, d_{\mathrm{VC}}), \tag{1.1}$$

where d_{VC} is a parameter related to model complexity (called the VC-dimension), and $\delta(\varepsilon, N, d_{\mathrm{VC}})$ goes rapidly to zero as N goes to infinity. Because it is a worst-case analysis, the bound can be applied universally, and for any learning model with finite VC-dimension, the probability of large deviations can be made arbitrarily small given enough data. On the other hand, the exact determination of d_{VC} for a given learning model can be difficult, and the amount of data required to get nontrivial bounds is very large for even modest d_{VC} (more than 13000 examples when $\epsilon = 0.05$ and $d_{\mathrm{VC}} = 100$ using the bound from [Parrondo and Van den Broeck 1993]).[2]

Furthermore, bounding the probability of large deviation might not allow us to infer low out-of-sample error from low in-sample error. In his "no free lunch" results, Wolpert showed that, in the absence of any assumptions about the target function, we can make no a priori distinctions between two learning algorithms when we consider zero-one loss and off-training-set error[3] [Wolpert 1996b; Wolpert 1996a]. As a consequence, for certain prior distributions over targets, random hypotheses will perform as well as those selected by a given learning algorithm, so no generalization can be expected.

It is therefore necessary to make some assumptions about the nature of possible target functions. Under fairly general smoothness constraints, asymptotic (as the number of examples $N \to \infty$) relationships can be found describing the relationship between in-sample and out-of-sample error in terms of a measure of learning model complexity [Akaike 1970; Moody 1992; Murata and Amari 1999]. By restricting the target to a specific class of functions, computational learning theory provides learning algorithms that result in out-of-sample error going to zero with probability

[2]Lee et al. [Lee *et al.* 1995; Lee *et al.* 1997] showed that when $\mathcal{X} = \mathbf{R}^n$, a two-layer neural network with k sigmoid hidden units has $d_{\mathrm{VC}} \geq (k-1)(n-1)$. Thus, learning models used for practical problems could quite reasonably have $d_{\mathrm{VC}} \gg 100$.

[3]Off-training-set error is the expected error over all input points not in the training set.

approaching one [Valiant 1984; Kearns and Vazirani 1994].

Depending on the assumptions made about the target function, a variety of results can be obtained. It should be borne in mind that *some* form of prior information about the target function is necessary for any useful theory of generalization.

1.3 Contributions

This thesis is concerned with the theory of generalization and the generalization behavior of learning systems in practice, primarily in the context of classification problems.

A theoretical framework for the study of generalization is presented, building upon the model introduced in [Abu-Mostafa and Song 1996] and [Song 1999]. Within this framework, the full generalization behavior of a learning problem is encapsulated in a single probability distribution. Using a simple learning algorithm, overfitting is ruled out in the expectation, even for noisy data sets.

A two-stage learning procedure is introduced to apply the theoretical results to practical problems using standard learning models and learning algorithms. This procedure is shown to result in improved generalization error estimates.

The use of a fixed training set under this model is shown to lead to overfitting in some cases. Further investigation of this phenomenon leads to a method for valuing individual training examples. We show that data selection based on this valuation technique can be used to create training sets that result in better generalization. Furthermore, an analysis of the values of individual examples can be used to estimate noise levels and identify outliers.

Finally, a variation of the data valuation technique is applied as a new nonparametric learning algorithm. This algorithm is shown to be effective in practice and quite robust to noise.

Chapter 2

The Bin Model

In this chapter we present a framework for studying the generalization behavior of a general learning problem. We provide an estimate for the expected out-of-sample error in terms of the in-sample error without restricting the class of learning models or input distributions.

2.1 Introduction to the Bin Model

We begin the study of generalization by looking at a simple abstraction of the learning problem. Consider a collection of bins, each bin containing black and white marbles (see Figure 2.1). From each bin we are allowed to take a (random) handful of marbles and observe their colors. We would then like to select a bin that has a large fraction of white marbles.

A preliminary study of this "bin model" as it applies to the learning problem is given in [Abu-Mostafa and Song 1996]. The bins represent hypotheses in the learning model, and the marbles represent points in the input space. We associate white marbles with points for which the error $e(g(x), f(x))$ is 0, and black marbles with points on which the error is 1 (we consider only classification problems with zero-one loss). We can characterize the generalization behavior of the learning system by examining the relationship between the observed in-sample error (observed number of black marbles) and the out-of-sample error (overall fraction of black marbles in a selected bin).

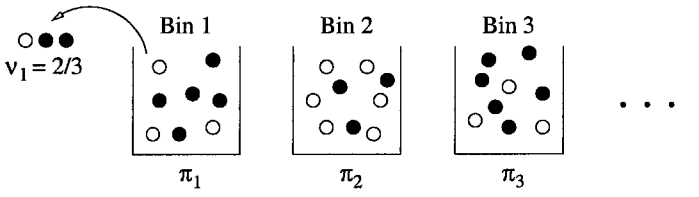


Figure 2.1: The learning problem modelled as a set of bins. The fraction of black marbles in a bin is denoted by π. We try to select a bin with small π observing only ν, the fraction of black marbles in a finite sample.

In order to study a general learning problem, we must move beyond the discretized bins and marbles, but we have already made an important abstraction. Explicit knowledge of the identity of the target function is not necessary—the target plays a role only through its errors with the hypotheses (in fact, only through the average out-of-sample errors). We extend this simple model to further specify what information about the target function is important, and to investigate the qualitative and quantitative aspects of generalization.

2.2 Exhaustive Learning

In many commonly used learning algorithms, a starting point is chosen and a small set of hypotheses are explored according to some sequential rule. Gradient based methods like back-propagation [Rumelhart *et al.* 1986], for example, typically explore only a few hypotheses that lie on a path of descent in parameter space. This can result in a great speed advantage, but is susceptible to getting stuck in local minima. Global optimization techniques like simulated annealing [Kirkpatrick *et al.* 1983] are less efficient, but explore a much greater number of hypotheses. The error spaces corresponding to popular parameterized learning models like neural networks tend to have many symmetries [Bishop 1995], so there remains the question of selecting from hypotheses with the same in-sample performance.

For the formulation of the bin model, we consider hypotheses to be selected from the learning model randomly according to a prior distribution p_G. We refer to this as the *exhaustive learning algorithm*, and it is exhaustive in the sense that any hypothesis

in the learning model may be selected.[1] The term 'exhaustive learning' is due to Schwartz et al., and was used to describe a framework for studying generalization that they developed that relied on random hypothesis selection [Schwartz *et al.* 1990]. Their approach is similar to the one we use here but considers only zero in-sample errors, and their generalization results can be considered a special case of those given in section 2.4.

2.3 Terminology

For a specified target function and learning model, the probability of selecting a hypothesis with a given out-of-sample error is implied by p_G. We use p_π to denote the resulting probability density, and we refer to this as the π-distribution. Formally, the π-distribution satisfies

$$\int_0^{\pi_0} p_\pi(s)ds = \Pr[\pi(g) \leq \pi_0] \tag{2.1}$$

for g selected according to p_G.

Certain classes of learning models are of particular interest. For each hypothesis g we define its complement $\overline{g}(x) = 1 - g(x)$ for every x. We call the learning model $\mathcal{G}$ *set-symmetric* if $g \in \mathcal{G} \Leftrightarrow \overline{g} \in \mathcal{G}$. We say that $\mathcal{G}$ *is symmetric under* p_G if $\mathcal{G}$ is set-symmetric and $p_G(g) = p_G(\overline{g})$ for all $g \in \mathcal{G}$. When p_G is implied, we will often simply refer to $\mathcal{G}$ as a symmetric learning model.

A model $\mathcal{G}$ is called *degenerate* if there exists a $g \in \mathcal{G}$ with $\Pr[g] > 0$, that is, if there is a hypothesis that has a positive prior probability. The learning models $\mathcal{G} \subset \{f, \overline{f}\}$ are called *strictly degenerate* and are not of practical interest. In order to simplify the exposition, we will assume throughout that the learning models used are not strictly degenerate.

[1] In reality, only functions in the support of p_G can be selected. For the purposes of exhaustive learning, if $p_G(g) = 0$ then we consider g not to be in the learning model.

2.4 Generalization

We are now prepared to analyze the generalization behavior of an arbitrary learning system under exhaustive learning. Specifically, we look at the expected out-of-sample error having observed the in-sample error on a random training set.

The input values of the examples in the training set are assumed to be i.i.d. random samples from the input distribution p_X. Thus the probability of error on each input is

$$\Pr[e(g(x), f(x)) = 1] = E_g[e(g(x), f(x))] = \pi(g). \tag{2.2}$$

The errors on our training set can be considered independent Bernoulli trials, and hence $\nu_{\mathcal{D}}(g)$ is a binomial random variable depending on $\pi(g)$ with the distribution

$$\Pr[\nu_{\mathcal{D}}(g) = \nu_0 | \pi(g) = \pi_0] = \binom{N}{N\nu_0} \pi_0^{N\nu_0} (1 - \pi_0)^{N(1-\nu_0)} \tag{2.3}$$

when $|\mathcal{D}| = N$. For any given hypothesis g, the expected distribution of the in-sample error ν depends only on $\pi(g)$. Equivalently, in the conceptual model of section 2.1, the statistics of the observed colors in a random handful of marbles are completely determined by the fraction of black marbles in the bin.

A straightforward application of Bayes' Rule allows us to find the expected generalization error for an observed in-sample performance, that is $\mathrm{E}_{g,\mathcal{D}}[\pi(g)|\nu_{\mathcal{D}}(g)]$. We write

$$\pi(\nu_0) = \mathrm{E}_{g,\mathcal{D}}[\pi(g)|\nu_{\mathcal{D}}(g) = \nu_0] \tag{2.4}$$

$$= \int_0^1 s p_{\pi|\nu}(s|\nu_0) ds \tag{2.5}$$

$$= \frac{\int_0^1 s p_\pi(s) \Pr[\nu = \nu_0 | \pi = s] ds}{\Pr[\nu = \nu_0]} \tag{2.6}$$

$$= \frac{\int_0^1 s p_\pi(s) s^{N\nu_0} (1-s)^{N(1-\nu_0)} ds}{\int_0^1 p_\pi(s) s^{N\nu_0} (1-s)^{N(1-\nu_0)} ds}. \tag{2.7}$$

We refer to the function $\pi(\nu)$ described by (2.7) as the *generalization curve.* Note that the expectation (2.4) is taken over data sets of fixed size N, and there will be a different generalization curve for each possible N.

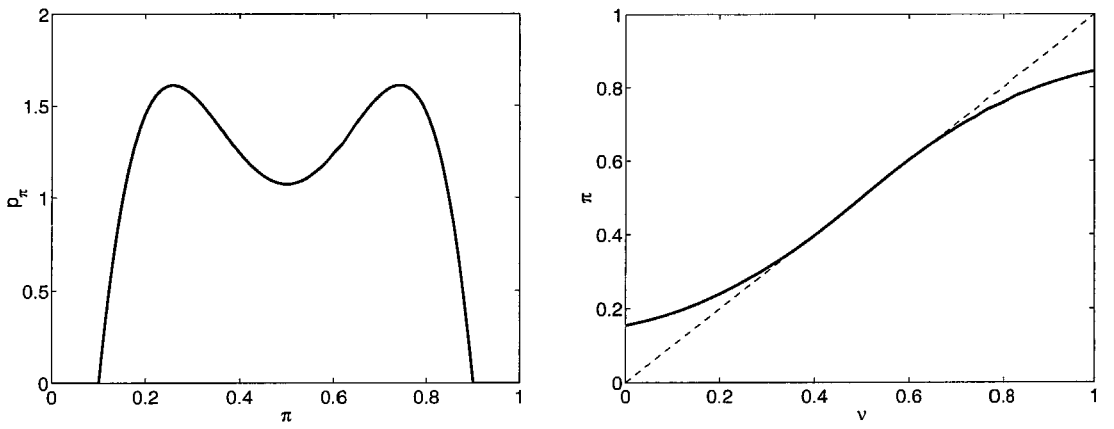
Heuristically, a "good" generalization curve is one for which $\pi(\nu) \simeq \nu$, which implies that the observed in-sample error is a good indicator of the expected out-of-sample error. In fact, we will say that *ideal generalization* corresponds to the case $\pi(\nu) = \nu$ for all $\nu \in [0, 1]$. We quantify "goodness" in this sense by measuring the expected squared difference between ν and $\pi(\nu)$

$$\Delta_N(\pi, \nu) = \mathrm{E}_{g,\mathcal{D}_N}[(\nu_{\mathcal{D}_N}(g) - \pi(g))^2] \tag{2.8}$$

$$= \sum_{i=0}^{N} \Pr\left[\nu_{\mathcal{D}_N}(g) = \frac{i}{N}\right] \left(\frac{i}{N} - \pi\left(\frac{i}{N}\right)\right)^2. \tag{2.9}$$

Note that we have made a distinction here between good generalization behavior and low generalization (out-of-sample) error. What is considered a good level of out-of-sample error is highly variable and depends on the specific problem. The practical value of a learning system ultimately depends on a confident estimate of the expected out-of-sample error. Good generalization in the sense of low $\Delta_N(\pi, \nu)$ indicates that the observed in-sample error for a selected hypothesis provides such an estimate.

Figure 2.2(a) shows a hypothetical π-distribution and Figure 2.2(b) shows the corresponding generalization curve for data sets of $N = 25$ examples. According to this model, when the training error ν is zero, the expected out-of-sample error is actually about 0.17, indicating that this in-sample error is an optimistic estimation of the corresponding out-of-sample error. The generalization curve is very close to the ideal generalization curve $\pi(\nu) = \nu$ (shown as a dashed line) for most ν, and has $\Delta_N(\pi, \nu) = 1.5 \times 10^{-3}$. To put this value into perspective, we can consider a hopeless learning problem in which every hypothesis has $\pi = \frac{1}{2}$ (a completely random target or a learning model of random functions will result in this π-distribution). In this



(a) (b)

Figure 2.2: An example π-distribution and its generalization curve. The π-distribution is symmetric about $\pi = 1/2$ and has no hypotheses with $\pi < 0.1$. The resulting generalization curve for 25 examples shows that an observed $\nu = 0$ has an expected $\pi(\nu) \simeq 0.17$.

case,

$$\Delta_N(\pi, \nu) = \sum_{i=0}^{N} \binom{N}{i} 2^{-n} \left(i - \frac{1}{2}\right)^2 = \frac{1}{4N}. \tag{2.10}$$

For certain π-distributions, we can obtain $\pi(\nu)$ explicitly. As an illustration, we consider a polynomial distributions $p_\pi(\pi_0) = (d+1)\pi_0^d$ for nonnegative integers d. The resulting generalization curve is given by

$$\pi(\nu) = \frac{N\nu + d + 1}{N + d + 2}. \tag{2.11}$$

Setting $d = 0$ results in uniform distribution $p_\pi(\pi_0) = 1$ for all $\pi_0 \in [0, 1]$. In this case, a randomly selected hypothesis will have any value of π with equal probability. The resulting $\pi(\nu)$ is linear in ν and lies somewhere between $\pi(\nu) \equiv \frac{1}{2}$ (no generalization, $\Delta_N(\pi, \nu) = 1/4N$) and $\pi(\nu) = \nu$ (ideal generalization, $\Delta_N(\pi, \nu) = 0$) depending on the amount of available data (see Figure 2.3).

The preceding examples illustrate one of the key points of the bin model framework: the generalization behavior is completely characterized by the π-distribution.

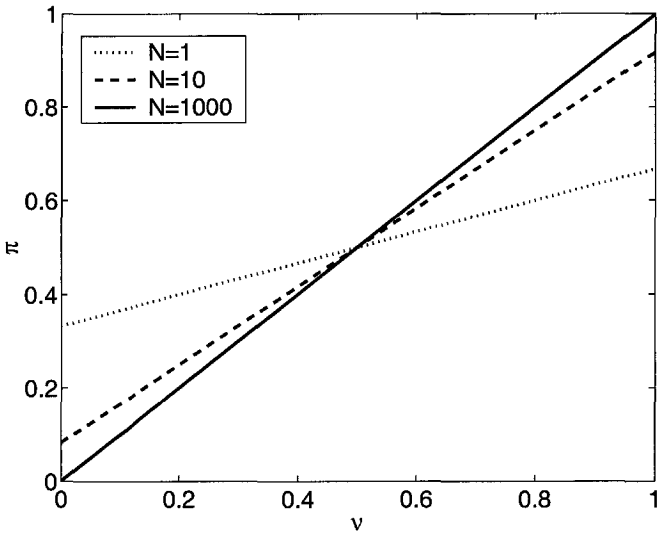


Figure 2.3: Generalization curves for a problem with uniform π-distribution. With only one example ($N = 1$), $\pi(\nu) = (\nu + 1)/3$. As the amount of data increases, $\pi(\nu)$ approaches the ideal generalization curve.

The identity of the target function and the nature of the learning model only affect the generalization curve through their influence on p_π. In principle p_π cannot be determined without knowledge of the target function, but this abstraction allows us to analyze the learning problem without explicit dependence on f.

The overfitting phenomenon corresponds to a generalization curve which has a minimum for some $\nu > 0$. That is, the best out-of-sample error is reached for some nonzero in-sample error, and seeking a lower training error will cause the out-of-sample error to increase. We demonstrated such a case in section 1.2.1, and classification problems are not generally immune to overfitting. Under the exhaustive learning algorithm, however, overfitting cannot occur. This is made precise by the following theorem.

Theorem 2.4.1 *The expected test error $\pi(\nu)$ is monotonically nondecreasing in the empirical error ν.*[2]

The proof is deferred to section 2.5.2, where we show this to be a special case of a

[2]A function $F(x)$ is *monotonically nondecreasing (nonincreasing) in* x iff $x_1 \leq x_2 \Rightarrow F(x_1) \leq F(x_2)$ ($x_1 \leq x_2 \Rightarrow F(x_1) \leq F(x_2)$).

more general result. Theorem 2.4.1 implies that, with exhaustive learning, further decreasing the in-sample error always improves the expected out-of-sample error. Overfitting is commonly ascribed to learning the idiosyncrasies of the training set, and so is expected to be worse with noisy data. We will discuss the effects of noise on this result and the bin model in general.

2.5 Noise in the Bin Model

In nearly all learning problems, the available data contain some form of noise. Inexact or erroneous data may arise for a variety of reasons. The target function may be inherently stochastic or even undefined on parts of the input space. Data collected from physical measurements have some variability related to the measurement device and conditions. If a set of examples is provided by an expert (a human transcribing speech or a credit agency reporting a credit score, for example), the expert cannot be considered infallible, and the data will reflect the expert's best guess and not a ground truth. Errors and omissions may be further introduced by transmission, transcription or other processing of a data set.

The effects of noise on a learning system are thus of great interest to theorists and practitioners alike. We consider the effects of noise on generalization in the context of the bin model analysis.

2.5.1 Uniform Noise

We first consider the case of noise that affects data points independently. For classification problems ($\mathcal{Y} = \{0, 1\}$), uniform noise in the output corresponds to a random flip of the classification with some fixed probability. This is the same noise model as that of communication across a binary symmetric channel (BSC) [Cover and Thomas 1991] illustrated in Figure 2.4. If the level of noise in the classification problem is ε,

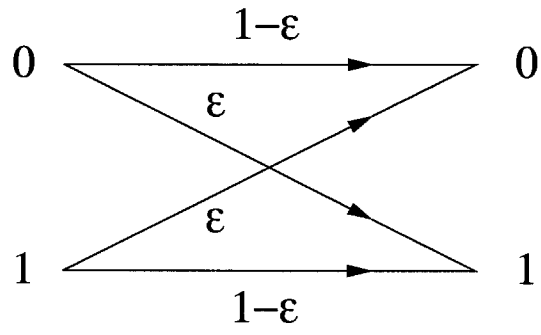


Figure 2.4: A binary symmetric channel. Each bit transmitted across the channel is received correctly with probability $1-\varepsilon$ and incorrectly with probability ε.

then we can define a noisy realization $\tilde{f}$ of the target function

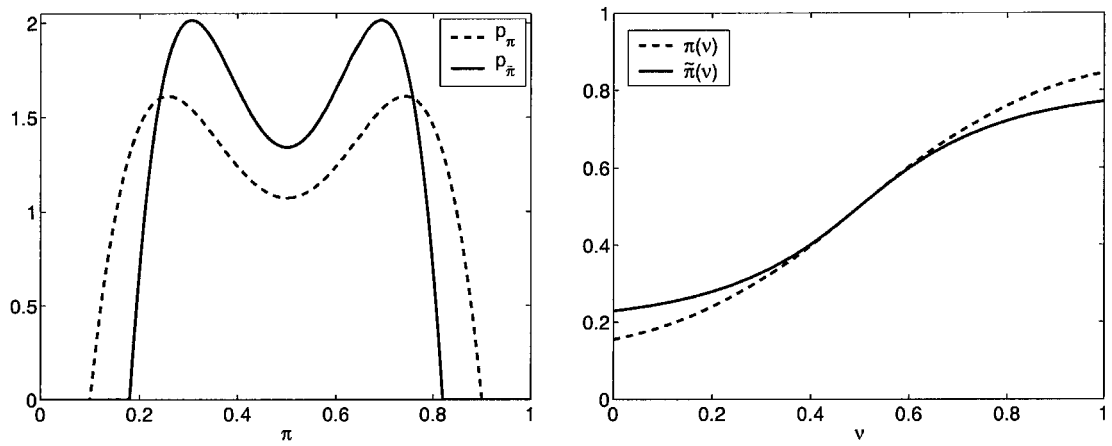
$$\tilde{f}(x) = \begin{cases} f(x) & \text{with probability } (1-\varepsilon) \\ 1-f(x) & \text{with probability } \varepsilon \end{cases} . \tag{2.12}$$

The underlying target function f is the same, but our available data are noisy, that is, for an example (x, y), we now have $y = \tilde{f}(x)$.

This uniform noise model can easily be incorporated into the bin model analysis. Let $\tilde{\pi}(g)$ denote the expected error of a hypothesis g with the noisy target $\tilde{f}$. We can write $\tilde{\pi}$ in terms of π.

$$\begin{aligned} \tilde{\pi}(g) &= \mathrm{E}_x[e(g(x), \tilde{f}(x))] && (2.13)\\ &= \Pr[g(x) \neq \tilde{f}(x)] && (2.14)\\ &= (1-\varepsilon)\Pr[g(x) \neq f(x)] + \varepsilon \Pr[g(x) = f(x)] && (2.15)\\ &= (1-\varepsilon)\pi(g) + \varepsilon(1-\pi(g)) && (2.16)\\ &= \pi(g)(1-2\varepsilon) + \varepsilon. && (2.17) \end{aligned}$$

Furthermore, if we know the distribution p_π, we can find the distribution of $\tilde{\pi}$ by a



(a) (b)

Figure 2.5: Effects of noise on the π-distribution and generalization curve. The solid curve in (a) shows the $p_{\tilde{\pi}}$ for the p_π shown by the dashed curve under BSC noise with $\varepsilon = 0.1$. The corresponding noiseless and noisy generalization curves are shown in (b).

change of variables.

$$p_{\tilde{\pi}}(\pi_0) = \begin{cases} \frac{p_\pi((\pi_0-\varepsilon)/(1-2\varepsilon))}{1-2\varepsilon} & \varepsilon \leq \pi_0 \leq 1-\varepsilon \\ 0 & \text{otherwise} \end{cases} . \tag{2.18}$$

Thus, the addition of uniform BSC noise to a classification problem results in a linear transformation of the expected out-of-sample errors and a contraction of the π-distribution. These effects are illustrated in Figure 2.5. The solid curve in Figure 2.5(a) gives an example of a π-distribution, and the dashed curve shows the resulting distribution of $\tilde{\pi}$ when we flip classifications with probability $\varepsilon = 0.1$. The transformation of (2.18) effectively squeezes the π-distribution around $\pi = \frac{1}{2}$. The result is a generalization curve with larger deviations when ν is close to 0 or 1 (Figure 2.5(b)).

We can quantify the effect of noise on generalization by investigating the change in $\Delta_N(\pi, \nu)$. We define an analogous measure

$$\Delta_N(\tilde{\pi}, \nu) = \mathrm{E}_{g,\mathcal{D}_N}[(\nu_{\mathcal{D}_N}(g) - \tilde{\pi}(g))^2]. \tag{2.19}$$

The noiseless generalization curve in Figure 2.5(b) has $\Delta_N(\pi, \nu) = 1.5 \times 10^{-3}$, the

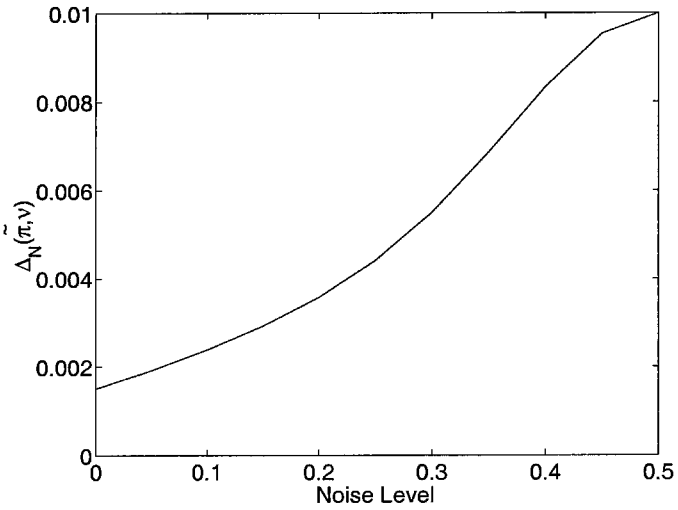


Figure 2.6: $\Delta_N(\tilde{\pi}, \nu)$ for varying noise levels. For the learning problem described by the noiseless π-distribution of Figure 2.5(a), the addition of uniform noise of varying levels results in $\Delta_N(\tilde{\pi}, \nu)$ values shown. As the noise goes to 50%, the value of Δ_N goes to $1/4N = 0.01$.

noisy version has $\Delta_N(\tilde{\pi}, \nu) = 2.4 \times 10^{-3}$. As the noise level $\varepsilon \to \frac{1}{2}$, the π-distribution goes to $p_\pi(\pi_0) = \delta(\pi_0 - \frac{1}{2})$, and from (2.10) we know that Δ_N goes to $1/4N$. The value of $\Delta_N(\tilde{\pi}, \nu)$ for varying noise levels for the example of figure 2.5 is shown in Figure 2.6. As is to be expected, increasing the noise results in worse generalization.

2.5.2 Input-Dependent Noise

In some problems, the noise in the data set may be input dependent. For example, on experimental measurements, the uncertainty can depend on the characteristics of the particular instrument used and may change for different measurement scales. We model this input dependence by writing the noise as a function $\varepsilon = \varepsilon(x)$. Once again we define the noisy target function

$$\tilde{f}(x) = \begin{cases} f(x) & \text{with probability } 1 - \varepsilon(x) \\ 1 - f(x) & \text{with probability } \varepsilon(x) \end{cases} . \tag{2.20}$$

On a specific point in the input space, now

$$\Pr[g(x) \neq \tilde{f}(x)] = e(g(x), f(x))(1 - \varepsilon(x)) + (1 - e(g(x), f(x)))\varepsilon(x) \tag{2.21}$$

and the noisy out-of-sample error becomes

$$\begin{aligned}
\tilde{\pi}(g) &= \Pr[g(x) \neq \tilde{f}(x)] && (2.22)\\
&= \mathrm{E}_x[e(g(x), f(x))(1 - \varepsilon(x)) + (1 - e(g(x), f(x)))\varepsilon(x)] && (2.23)\\
&= \mathrm{E}_x[e(g(x), f(x))] + \mathrm{E}_x[\varepsilon(x)] - 2\mathrm{E}_x[e(g(x), f(x))\varepsilon(x)] && (2.24)\\
&= \pi(g) + \mathrm{E}_x[\varepsilon(x)] - 2\mathrm{E}_x[e(g(x), f(x))\varepsilon(x)]. && (2.25)
\end{aligned}$$

Of particular interest are noise models which allow $\tilde{\pi}(g)$ to be determined given only $\pi(g)$.

Definition 2.5.1 *We say a noise model $\varepsilon(x)$ is* **regular** *iff $\tilde{\pi}(g) = \tilde{\pi}(\pi(g))$.*

From (2.25) it is clear that $\varepsilon(x)$ is regular iff $\mathrm{E}_x[e(g(x), f(x))\varepsilon(x)]$ is a function of $\pi(g)$. In the case of input-independent noise $\varepsilon(x) \equiv \varepsilon_0$, and (2.25) reduces to (2.17).

We illustrate the input-dependent noise scenario with a simple example. Consider the learning model that consists of threshold functions, $g_w(x) = \mathrm{sgn}(x - w)$, $w \in [0, 1]$. Assume that the target function is in the learning model, $f(x) = g_\alpha(x)$, and for simplicity that $\frac{1}{2} < \alpha < 1$. Let the input distribution be uniform in $[0, 1]$ and assume that exhaustive learning is used with w chosen uniformly in $[0, 1]$.

For this simple scenario, an error occurs (that is, $e(g_w(x), f(x)) = 1$) when either $w < x < \alpha$ or $\alpha < x < w$. It follows that $\pi(g_w) = |\alpha - w|$. For any noise function $\varepsilon(x)$, we can compute the noisy out-of-sample error

$$\tilde{\pi}(g_w) = \begin{cases} \pi(g_w) + \int_0^1 \varepsilon(x)dx - 2\int_w^\alpha \varepsilon(x)dx & w < \alpha \\ \pi(g_w) + \int_0^1 \varepsilon(x)dx - 2\int_\alpha^w \varepsilon(x)dx & w > \alpha \end{cases}. \tag{2.26}$$

Note that when $p < (1 - \alpha)$ there are two hypotheses, $g_{\alpha-p}$ and $g_{\alpha+p}$, that have $\pi = p$. For $(1 - \alpha) < p < \alpha$, there is only one such hypothesis, and no hypothesis has $\pi > \alpha$.

Thus the noise model is regular if it satisfies $\tilde{\pi}(g_{\alpha-p}) = \tilde{\pi}(g_{\alpha+p})$ for all $p < (1-\alpha)$. If we assume the noise is regular, we can rewrite the noisy error $\tilde{\pi}$ as a function of the noiseless error π. Rewriting (2.26) for a regular noise function, we get

$$\tilde{\pi}(\pi = \pi_0) = \tilde{\pi}(g_{\alpha-\pi_0}) \tag{2.27}$$

$$= \pi(g_w) + \int_0^1 \varepsilon(x)dx - 2\int_{\alpha-\pi_0}^{\alpha} \varepsilon(x)dx. \tag{2.28}$$

2.5.3 The Effect of Noise on Generalization

For a general learning problem, we would like to characterize the generalization behavior for both regular and non-regular noise models. Given a training error ν_0 on N points, we can determine the noisy and noiseless expected out-of-sample errors, $\tilde{\pi}(\nu_0)$ and $\pi(\nu_0)$ respectively. The in-sample error is now a binomial random variable dependent on the mean noisy error $\tilde{\pi}$.

$$\Pr[\nu(g) = \nu_0 | g] = \Pr[\nu(g) = \nu_0 | \tilde{\pi}(g)] \tag{2.29}$$

$$= \binom{N}{N\nu_0} \tilde{\pi}(g)^{N\nu_0}(1-\tilde{\pi}(g))^{1-N\nu_0}, \tag{2.30}$$

and hence,

$$\tilde{\pi}(\nu_0) = \mathrm{E}_g[\tilde{\pi}(g)|\nu = \nu_0] \tag{2.31}$$

$$= \int_0^1 s\ p_{\tilde{\pi}|\nu}(s|\nu_0)ds \tag{2.32}$$

$$= \frac{\int_0^1 s\ \Pr[\nu(g) = \nu_0 | \tilde{\pi}(g) = s] p_{\tilde{\pi}}(s)ds}{\int_0^1 \Pr[\nu(g) = \nu_0 | \tilde{\pi}(g) = s] p_{\tilde{\pi}}(s)ds}, \tag{2.33}$$

where $p_{\tilde{\pi}}$ is the distribution of $\tilde{\pi}(g)$ induced by $p_G(g)$. If we are concerned only with the noisy out-of-sample error, the result of Theorem 2.4.1 generalizes immediately.

Theorem 2.5.1 *For any $\tilde{\pi}(g)$, $p_G(\cdot)$ and noise $\varepsilon(x)$, $\tilde{\pi}(\nu)$ is monotonically nondecreasing in ν.*

Proof of Theorem 2.5.1:

Let $0 \leq k < N$. We compare $\tilde{\pi}(\frac{k}{N})$ and $\tilde{\pi}(\frac{k+1}{N})$.

$$\tilde{\pi}\left(\frac{k+1}{N}\right) - \tilde{\pi}\left(\frac{k}{N}\right) \tag{2.34}$$

$$= \frac{\int_0^1 s \ \Pr_{\nu|\tilde{\pi}}[\frac{k+1}{N}|s] p_{\tilde{\pi}}(s) ds}{\int_0^1 \Pr_{\nu|\tilde{\pi}}[\frac{k+1}{N}|s] p_{\tilde{\pi}}(s) ds} - \frac{\int_0^1 t \ \Pr_{\nu|\tilde{\pi}}[\frac{k}{N}|t] p_{\tilde{\pi}}(t) dt}{\int_0^1 \Pr_{\nu|\tilde{\pi}}[\frac{k}{N}|t] p_{\tilde{\pi}}(t) dt} \tag{2.35}$$

$$= \frac{\int s^{k+2}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) ds}{\int s^{k+1}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) ds} - \frac{\int t^{k+1}(1-t)^{N-k} p_{\tilde{\pi}}(t) dt}{\int t^{k}(1-t)^{N-k} p_{\tilde{\pi}}(t) dt} \tag{2.36}$$

$$= A_0(k) \left[\iint s^{k+2}(1-s)^{N-k-1} t^k (1-t)^{N-k} p_{\tilde{\pi}}(t) p_{\tilde{\pi}}(s) ds \ dt \right.$$

$$\left. - \iint t^{k+1}(1-t)^{N-k} s^{k+1}(1-s)^{N-k-1} p_{\tilde{\pi}}(t) p_{\tilde{\pi}}(s) ds \ dt \right] \tag{2.37}$$

$$= A_0(k) \iint s(s-t)(1-t) s^k t^k (1-s)^{N-k-1}(1-t)^{N-k-1} p_{\tilde{\pi}}(t) p_{\tilde{\pi}}(s) ds \ dt \tag{2.38}$$

$$= A_0(k) \iint t(t-s)(1-s) t^k s^k (1-t)^{N-k-1}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) p_{\tilde{\pi}}(t) dt \ ds \tag{2.39}$$

$$= \frac{A_0(k)}{2} \iint \left((s(s-t)(1-t) + t(t-s)(1-s)) \times \right.$$

$$\left. t^k s^k (1-t)^{N-k-1}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) p_{\tilde{\pi}}(t) \right) dt \ ds \tag{2.40}$$

$$= \frac{A_0(k)}{2} \iint (s-t)^2 t^k s^k (1-t)^{N-k-1}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) p_{\tilde{\pi}}(t) dt \ ds \tag{2.41}$$

$$\geq 0 \tag{2.42}$$

We use the shorthand $\Pr_{\nu|\tilde{\pi}}[a|b]$ for the conditional probability $\Pr[\nu(g) = a|\tilde{\pi}(g) = b]$. In (2.37), the factor

$$A_0(k) = \left(\int s^{k+1}(1-s)^{N-k-1} p_{\tilde{\pi}}(s) ds \int t^k (1-t)^{N-k} p_{\tilde{\pi}}(t) dt \right)^{-1} \tag{2.43}$$

is positive (since the integrands are nonnegative) and finite (since $\mathcal{G}$ is not strictly degenerate, hence $\exists \ 0 < \tilde{\pi}_0 < 1$ with $p_{\tilde{\pi}}(\tilde{\pi}_0) > 0$) for all k. (2.39) is (2.38) rewritten with a change of variables, and (2.40) is obtained by summing (2.39) and (2.38). Thus $\tilde{\pi}(0) \leq \tilde{\pi}(\frac{1}{N}) \leq \tilde{\pi}(\frac{2}{N}) \leq \ldots \leq \tilde{\pi}(1)$ completing the proof.

■

We can now use this result to prove Theorem 2.4.1 for the noiseless case.

Proof of Theorem 2.4.1:

This is a special case of Theorem 2.5.1 with $\tilde{\pi}(g) = \pi(g) \forall g$.

■

Thus, for any noise model, there is no overfitting of the noisy error $\tilde{\pi}$—reducing the in-sample error always leads to a reduction of the out-of-sample error observed under the same noise model.

When the noise distribution is regular, we can write the noiseless and noisy expected errors in terms of the (noiseless) π-distribution p_π. Substituting $\tilde{\pi}(g) = \tilde{\pi}(\pi)$ into (2.30),

$$\Pr[\nu(g) = \nu_0 | \pi(g) = \pi_0] = \binom{N}{N\nu_0} \tilde{\pi}(\pi_0)^{N\nu_0} (1 - \tilde{\pi}_0(g))^{1-N\nu_0}. \tag{2.44}$$

This allows us to give conditions under which minimizing the noisy in-sample error cannot overfit even the noiseless out-of-sample error.

Theorem 2.5.2 *If $\varepsilon(x)$ is regular and $\tilde{\pi}(\pi)$ is monotonically nondecreasing (nonincreasing) in π then for any $p_\pi(\cdot)$, $\pi(\nu)$ is monotonically nondecreasing (nonincreasing) in ν.*

Proof of Theorem 2.5.2:

The proof is similar to that of Theorem 2.5.1. Let $0 \leq k < N$. We look at $\pi\left(\frac{k+1}{N}\right) - \pi\left(\frac{k}{N}\right)$.

$$\pi(\frac{k+1}{N}) - \pi(\frac{k}{N}) \tag{2.45}$$

$$= \frac{\int_0^1 s \ \Pr_{\nu|\pi}[\frac{k+1}{N}|s] p_\pi(s) ds}{\int_0^1 \Pr_{\nu|\pi}[\frac{k+1}{N}|s] p_\pi(s) ds} - \frac{\int_0^1 t \ \Pr_{\nu|\pi}[\frac{k}{N}|t] p_\pi(t) dt}{\int_0^1 \Pr_{\nu|\pi}[\frac{k}{N}|t] p_\pi(t) dt} \tag{2.46}$$

$$= \frac{\int s \ \tilde{\pi}(s)^{k+1} (1 - \tilde{\pi}(s))^{N-k-1} p_\pi(s) ds}{\int \tilde{\pi}(s)^{k+1} (1 - \tilde{\pi}(s))^{N-k-1} p_\pi(s) ds} - \frac{\int t \ \tilde{\pi}(t)^k (1 - \tilde{\pi}(t))^{N-k} p_\pi(t) dt}{\int \tilde{\pi}(t)^k (1 - \tilde{\pi}(t))^{N-k} p_\pi(t) dt} \tag{2.47}$$

$$= A_1(k) \left(\iint s \ \tilde{\pi}(s)^{k+1} (1 - \tilde{\pi}(s))^{N-k-1} \tilde{\pi}(t)^k (1 - \tilde{\pi}(t))^{N-k} p_\pi(t) p_\pi(s) ds \ dt \right.$$
$$\left. - \iint t \ \tilde{\pi}(t)^k (1 - \tilde{\pi}(t))^{N-k} \tilde{\pi}(s)^{k+1} (1 - \tilde{\pi}(s))^{N-k-1} p_\pi(t) p_\pi(s) ds \ dt \right) \tag{2.48}$$

$$= A_1(k) \iint \big((s-t)\tilde{\pi}(s)(1-\tilde{\pi}(t))\tilde{\pi}(s)^k\tilde{\pi}(t)^k \times$$
$$(1-\tilde{\pi}(s))^{N-k-1}(1-\tilde{\pi}(t))^{N-k-1}p_\pi(t)p_\pi(s)\big)\, ds\, dt \tag{2.49}$$

$$= A_1(k) \iint \big((t-s)\tilde{\pi}(t)(1-\tilde{\pi}(s))\tilde{\pi}(s)^k\tilde{\pi}(t)^k \times$$
$$(1-\tilde{\pi}(s))^{N-k-1}(1-\tilde{\pi}(t))^{N-k-1}p_\pi(t)p_\pi(s)\big)\, ds\, dt \tag{2.50}$$

$$= \frac{A_1(k)}{2} \iint \big((s-t)(\tilde{\pi}(s)-\tilde{\pi}(t))\tilde{\pi}(t)^k\tilde{\pi}(s)^k \times$$
$$(1-\tilde{\pi}(t))^{N-k-1}(1-\tilde{\pi}(s))^{N-k-1}p_\pi(s)p_\pi(t)\big)\, dt\, ds \tag{2.51}$$

In (2.48) the factor

$$A_1(k) = \left(\int \tilde{\pi}(s)^{k+1}(1-\tilde{\pi}(s))^{N-k-1}p_\pi(s)ds \int \tilde{\pi}(t)^k(1-\tilde{\pi}(t))^{N-k}p_\pi(t)dt\right)^{-1} \tag{2.52}$$

is positive and finite for all k, and the steps are essentially the same as those in the proof of Theorem 2.5.1.

If $\tilde{\pi}(\pi)$ is monotonically nondecreasing in π, then $(s-t)(\tilde{\pi}(s)-\tilde{\pi}(t)) \geq 0\ \forall s,t$, the integrand in (2.51) is always nonnegative, and hence $\pi(\nu)$ is monotonically nondecreasing in ν. Likewise, if $\tilde{\pi}(\pi)$ is monotonically nonincreasing in π, then the integrand in (2.51) is always nonpositive and $\pi(\nu)$ is monotonically nonincreasing in ν.

■

In particular, the uniform BSC noise model is regular and $\tilde{\pi}$ (given by (2.17)) is monotonically nondecreasing in π. Theorems 2.5.1 and 2.5.2 indicate that even with uniform noise we should not be concerned with overfitting under exhaustive learning—our expected noiseless out-of-sample error will be minimized by minimizing the noisy in-sample error.

2.6 Extensions of the Model

The bin model analysis presented in this chapter deals only with binary classification problems. The extension to multiclass and regression problems is nontrivial, since

the relationship between ν and π is generally not as nice as (2.3), and there is much more flexibility in the type of noise that may be present in the data.

Nevertheless, some of the important results may be extended to the more general learning problem. In particular, conditions for monotonicity of the expected out-of-sample error for real-valued error functions can be derived, and the effects of noise in regression problems have been studied. These results do not play an important role in this dissertation, so we only mention them here and refer the reader to [Nicholson 2000] for a more thorough discussion.

2.7 Discussion

The bin model framework allows us to analyze the generalization characteristics of a general classification problem. Only the π-distribution is needed to describe the generalization curve fully, eliminating the explicit dependence on the particular target function and learning model.

The main theoretical consequence of this analysis is that there is no overfitting in expectation when an exhaustive learning algorithm is used. This remains true in the presence of input-independent or regular noise.

The major practical shortcomings of the bin model are that the π-distribution cannot, in general, be known exactly without knowledge of the target function, and that it relies on the inefficient exhaustive learning algorithm. Even with knowledge of the π-distribution, the quantitative results cannot be applied directly when sophisticated learning algorithms are used. We will address these practical limitations in the next chapter.

Chapter 3

Generalization in Practical Learning Systems

In this chapter, we investigate the application of the theoretical results of the previous chapter to practical learning systems. We begin by looking at some specific learning models that lend themselves to a simple analysis. We then introduce a modified learning process that allows us to use some of the bin model results for learning systems that are not restricted to exhaustive learning.

3.1 Linear Models

We provide here an exact derivation of the π-distribution for a simple class of linear learning models. Consider classifiers $g : \mathbf{R}^d \to \{0, 1\}$ with a linear decision boundary that passes through the origin. This learning model can be written $\mathcal{G} = \{g_{\mathbf{w}}\}$ where $g_{\mathbf{w}}(x) = \text{sgn}(\mathbf{w} \cdot x)$ for $x, \mathbf{w} \in \mathbf{R}^d$. We assume that the inputs x and weights $\mathbf{w}$ are selected from distributions that are spherically symmetric about the origin. For target functions in the learning model we can calculate the π-distribution from which the expected generalization behavior can be determined.

Let the target function be $f = \text{sgn}(\mathbf{w}_f \cdot x)$. By symmetry, we can say w.l.o.g. that $\mathbf{w}_f = (1, 0, \ldots, 0)$. Appealing to the rotational symmetry of the input distribution,

we have

$$\pi(g_{\mathbf{w}}) = \frac{\phi(\mathbf{w})}{\boldsymbol{\pi}} \tag{3.1}$$

where $\phi(\mathbf{w})$ denotes the measure of the interior solid angle ($0 \leq \phi \leq \boldsymbol{\pi}$) between $\mathbf{w}$ and $\mathbf{w}_f$.[1] If we write $\mathbf{w} = (w_1, w_2, \ldots, w_d)$, then $\phi(\mathbf{w}) = \arccos(w_1)$. The region of weight space for which $\pi(g) < s$ is the area of the spherical cap of the unit d-ball in which $\phi(\mathbf{w}) < s\boldsymbol{\pi}$. Thus, we can find the cumulative distribution of π by calculating this area.

$$\Pr[\pi(g) < s] = \frac{1}{\mathcal{S}(d)} \int_{\phi=0}^{s\boldsymbol{\pi}} \int_{\Theta_1=0}^{\boldsymbol{\pi}} \cdots \int_{\Theta_{d-2}=0}^{2\boldsymbol{\pi}} \mathbf{J} d\Theta_{d-2} \cdots d\Theta_1 d\phi \tag{3.2}$$

$$= \frac{\mathcal{S}(d-1)}{\mathcal{S}(d)} \int_{\phi=0}^{s\boldsymbol{\pi}} \sin^{d-2}\phi \; d\phi, \tag{3.3}$$

where

$$\mathbf{J} = \sin^{d-2}\phi \prod_{i=1}^{d-3} \sin^{d-2-i}\Theta_i \tag{3.4}$$

is the Jacobean transforming Cartesian coordinates to polar coordinates $(\phi, \Theta_1, \Theta_2, \ldots, \Theta_{d-2})$ on the surface of the unit d-ball, and

$$\mathcal{S}(k) = \frac{2\boldsymbol{\pi}^{k/2}}{\Gamma(k/2)} \tag{3.5}$$

is the surface area of the unit k-ball. Hence

$$p_\pi(\pi_0) = \left. \frac{d \Pr[\pi < s]}{ds} \right|_{s=\pi_0} \tag{3.6}$$

$$= \frac{\mathcal{S}(d-1)}{\mathcal{S}(d)} \boldsymbol{\pi} \sin^{d-2}(\pi_0 \boldsymbol{\pi}). \tag{3.7}$$

These π-distributions are illustrated in Figure 3.1 for varying d. For $d = 2$, the π-distribution is uniform, resulting in linear generalization curves as in Figure 2.3. For

[1] We use boldface $\boldsymbol{\pi}$ to denote the constant 3.14159....

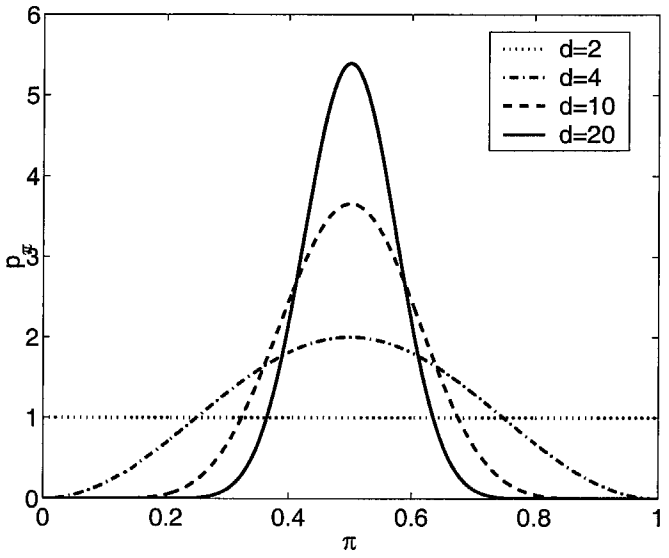


Figure 3.1: π-distributions for a linear learning model and target. p_π is shown for $d = 2, 4, 10, 20$. The distribution becomes more and more peaked around $\pi = 1/2$ as the dimension increases.

$d \geq 3$ the generalization curve can be calculated from (3.7), but cannot be expressed simply. A numerical evaluation of the generalization curve for $d = 20$ is shown in Figure 3.2. The generalization is quite poor due to the narrow peak of the distribution. In this case $\Delta_N(\pi, \nu) = 6.3 \times 10^{-3}$, which is worse than the 30% noise case in the example of Figure 2.6 ($N = 25$ for both examples).

In this section we have demonstrated a class of problems for which the π-distribution can be computed exactly, allowing a direct application of the bin model analysis. The results only describe the generalization for uniformly randomly selected hypotheses and only apply when the target is also in the model, and therefore seem to be of very little use. We will return to this analysis in Chapter 4, though, where we show an extension of this analysis to be valuable for data selection and discuss methods for dealing with noise.

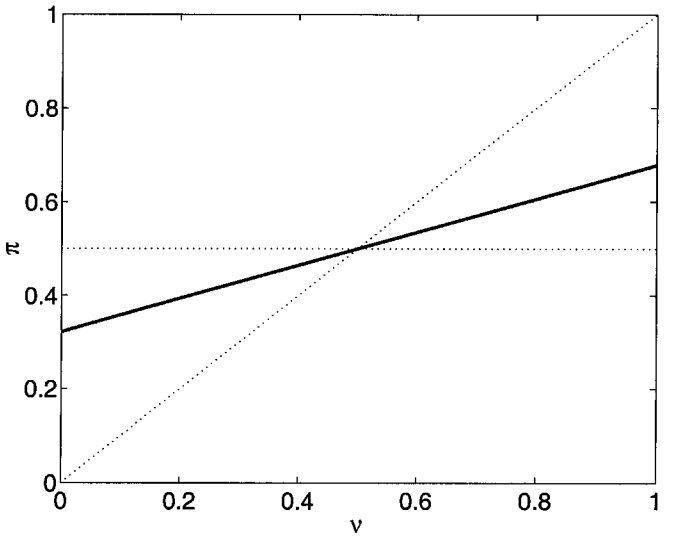


Figure 3.2: Generalization curve for 20-dimensional linear model. $\pi(\nu)$ is shown for the case $d = 20$ and $N = 25$. The ideal generalization curve and $\pi = 1/2$ are also shown for reference. The generalization is quite poor and has $\Delta_N(\pi, \nu) = 6.3 \times 10^{-3}$.

3.2 Neural Networks

Perhaps the most widely used and studied learning models are layered neural networks (feed-forward neural networks, multilayer perceptrons and variations thereof, see, for example, [Haykin 1994; White 1992]). Except for the most trivial cases, the complex functional form of these networks prohibits us from calculating exact π-distributions as we did for linear models in the previous section, even given the target function. Nevertheless, we are able to gain some insight into the generalization behavior of these models by looking at the qualitative characteristics of the learning model.

Consider learning models of the form $g(x) = \text{sgn}(\mathbf{w} \cdot \mathbf{v}(x) - t)$. This is the general form of a neural network with a linear threshold unit at the output. This model includes perceptrons (linear threshold functions), multilayer perceptrons and certain radial basis function networks. Since

$$g(x) = \text{sgn}(\mathbf{w} \cdot \mathbf{v}(x) - t) \tag{3.8}$$

$$\overline{g}(x) = \text{sgn}((-\mathbf{w}) \cdot \mathbf{v}(x) - (-t)), \tag{3.9}$$

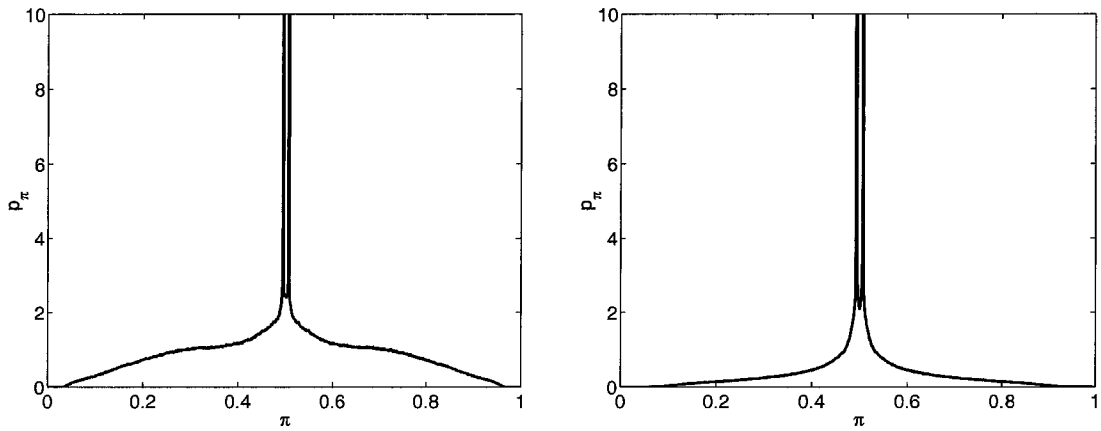


Figure 3.3: Empirical π-distributions. The figures show histograms of the π-distribution estimated from 10^7 samples from neural network models. In (a) the model used is a feed-forward neural network, and the model in (b) is a radial basis function network. In both cases the model is symmetric and a degeneracy is evident near $\pi = 1/2$.

these models are set-symmetric when the allowed values of the weight vector $\mathbf{w}$ are symmetric about 0. They may also be degenerate, since $t > |\mathbf{w}| \cdot |\mathbf{v}(x)| \Rightarrow g(x) \equiv 0$. Therefore, if $|\mathbf{v}(x)|$ is bounded above and $|\mathbf{w}|$ is sufficiently small with positive probability, then the zero hypothesis $g(x) \equiv 0$ (and by symmetry the one hypothesis $g(x) \equiv 1$) will have positive probability under exhaustive learning. Although we cannot carry out an exact analysis, we can make general statements about the generalization curve for symmetric models, and we can analyze the effects of degeneracy on generalization by a decomposition of the learning model.

Some empirical examples of π-distributions for neural network models are illustrated in Figure 3.3. The distributions are for a model of feed-forward neural networks with tanh hidden units and a model of radial basis function networks with Gaussian basis functions. These were computed by Monte Carlo simulations with known target functions and with weights and thresholds drawn from uniform distributions in $[-3, 3]$. Both distributions appear to be symmetric ($p_\pi(\pi_0) = p_\pi(1 - \pi_0)$), as we would expect, since the weight distribution is symmetric about 0. The degeneracy of each model is also made immediately evident by the appearance of sharp peaks in

the observed π-distribution. In both cases the degeneracies appear near $\pi = \frac{1}{2}$, which we would expect for class balanced (that is, $\Pr[f(x) = 1] \simeq \Pr[f(x) = 0]$) problems when the degeneracy is due to the zero and one hypotheses.

3.2.1 Model Symmetry

Many parameterized classifier models used in practice (and specifically the neural network models described above) can immediately be shown to be set-symmetric for an appropriate parameter range. In the bin model framework, symmetry of the learning model leads to some straightforward results. We mention several of these here, which may seem trivial but will prove to be useful later.

Under a symmetric learning model

$$p_\pi(\pi_0) = p_\pi(1 - \pi_0) \tag{3.10}$$

$$Pr[\nu(g) = \nu_0] = Pr[\nu(g) = 1 - \nu_0] \tag{3.11}$$

$$\pi(\nu_0) = 1 - \pi(1 - \nu_0) \tag{3.12}$$

$$\pi(1/2) = 1/2 \tag{3.13}$$

$$\mathrm{E}_g[\pi(\nu(g))] = \mathrm{E}_g[\nu(g)] = \mathrm{E}_g[\pi(g)] = 1/2, \tag{3.14}$$

and from the monotonicity of $\pi(\nu)$ and (3.13),

$$\pi(\nu) \leq \frac{1}{2} \Leftrightarrow \nu \leq \frac{1}{2}. \tag{3.15}$$

Also, for a single input x, since each hypothesis that classifies x correctly has a counterpart that classifies x incorrectly (with the same prior probability), $\mathrm{E}_g[e(g(x), f(x))] = \frac{1}{2}$, and since $e(\cdot, \cdot) \in \{0, 1\}$, $\mathrm{Var}_g[e(g(x), f(x))] = \frac{1}{4}$.

Note that these results rely on the symmetry of the π-distribution described by (3.10), rather than the symmetry of the learning model. Therefore, these results still hold in the presence of uniform BSC noise, since the transformation (2.17) preserves this symmetry.

3.2.2 Model Decomposition

Given learning models $\mathcal{G}_i$ and associated prior distributions p_{G_i}, we define the composition of these models, written $\mathcal{G} = \sum_i \alpha_i \mathcal{G}_i$ for constants α_i with $\sum_i \alpha_i = 1$, to be the model with $\mathcal{G} = \bigcup_i \mathcal{G}_i$ and prior distribution $p_G(g) = \sum_i \alpha_i p_{G_i}(g)$. We can write the generalization curve for the composite model in terms of the generalization curves under the individual models. For the composite model,

$$\pi(\nu_0) = \mathrm{E}_g[\pi(g)|\nu(g) = \nu_0] \tag{3.16}$$

$$= \sum_i \mathrm{E}_g[\pi|\nu(g) = \nu_0, \mathcal{G}_i] \Pr[\mathcal{G}_i|\nu(g) = \nu_0] \tag{3.17}$$

$$= \frac{\sum_i \alpha_i \Pr[\nu(g) = \nu_0|\mathcal{G}_i]\mathrm{E}_g[\pi(g)|\nu(g) = \nu_0, \mathcal{G}_i]}{\Pr[\nu(g) = \nu_0]} \tag{3.18}$$

$$= \frac{\sum_i \alpha_i \Pr_i[\nu(g) = \nu_0]\pi_i(\nu_0)}{\Pr[\nu(g) = \nu_0]}, \tag{3.19}$$

where $\pi_i(\nu_0) = \mathrm{E}_g[\pi|\nu(g) = \nu_0, \mathcal{G}_i]$ and $\Pr_i[\nu(g) = \nu_0] = \Pr[\nu(g) = \nu_0|\mathcal{G}_i]$ are respectively the generalization curve and probability of observing in-sample error ν_0 under model $\mathcal{G}_i$.

In order to investigate the effects of a degeneracy on the overall generalization, we decompose a given learning model $\mathcal{G}$ into three components. We let $\mathcal{G} = \mathcal{G}_1 + \mathcal{G}_2 + \mathcal{G}_3$, where $\mathcal{G}_1$ is a simple symmetric degenerate model

$$\mathcal{G}_1 = \{g_1, \overline{g_1}\}, \quad p_{G_1}(g_1) = p_{G_1}(\overline{g_1}) = \frac{1}{2}, \tag{3.20}$$

$\mathcal{G}_2$ is a model with uniform π-distribution and $\mathcal{G}_3$ can be any model. Any learning model with $p_\pi(\pi_0) > \epsilon$ for all $\pi_0 \in [0, 1]$ and some $\epsilon > 0$ with a symmetric degeneracy can be decomposed this way with $\alpha_1, \alpha_2 > 0$. We analyze the behavior of $\mathcal{G}_1$ and $\mathcal{G}_2$ independently, then look at their influence on the generalization of the composite model.

For the degenerate model $\mathcal{G}_1$, we assume w.l.o.g. that $\pi(g_1) = \gamma < \frac{1}{2}$ and that

$\gamma > 0$ ($\mathcal{G}_1$ is not strictly degenerate). Then

$$Pr_1[\nu(g) = \nu_0] = \frac{1}{2}\binom{N}{N\nu_0}\left(\gamma^{N\nu_0}(1-\gamma)^{N(1-\nu_0)} + \gamma^{N(1-\nu_0)}(1-\gamma)^{N\nu_0}\right) \tag{3.21}$$

and

$$\pi_1(\nu_0) = \gamma(1-\gamma)\left(\frac{\gamma^{N(1-2\nu_0)-1} + (1-\gamma)^{N(1-2\nu_0)-1}}{\gamma^{N(1-2\nu_0)} + (1-\gamma)^{N(1-2\nu_0)}}\right). \tag{3.22}$$

For the model $\mathcal{G}_2$ with uniform p_π,

$$Pr_2[\nu(g) = \nu_0] = \frac{1}{N+1} \tag{3.23}$$

for every ν_0, and

$$\pi_2(\nu_0) = \frac{N\nu_0 + 1}{N+2}. \tag{3.24}$$

Consider the generalization behavior of $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}$ in the limit of large N. As $N \to \infty$, $\pi_2(\nu) \to \nu$ for all ν. In contrast, $\gamma < \pi_1(\nu) < 1-\gamma$ for all ν and N. Thus, the uniform π model will tend to improve the generalization behavior as the amount of data increases, whereas the degenerate model will degrade generalization behavior for $\nu < \gamma$ and $\nu > 1-\gamma$. The influence of each model is mediated by the corresponding probability of ν being observed.

The two terms in expression (3.21) for $Pr_1[\nu(g) = \nu_0]$ represent binomial distributions, which, by the central limit theorem, will converge to normal distributions $\mathcal{N}(\gamma, \gamma(1-\gamma)/N)$ and $\mathcal{N}(1-\gamma, \gamma(1-\gamma)/N)$. Hence, if $\nu_0 < \gamma - \epsilon$ for some $\epsilon > 0$, then $Pr_1[\nu(g) = \nu_0]$ goes to zero like $\sqrt{N}e^{-N}$ as $N \to \infty$. $Pr_2[\nu(g) = \nu_0]$ goes to zero like $1/N$, and will quickly dominate $Pr_1[\nu(g) = \nu_0]$. Thus, while the presence of a degeneracy has a negative effect on generalization, this effect quickly becomes insignificant as the amount of data increases.

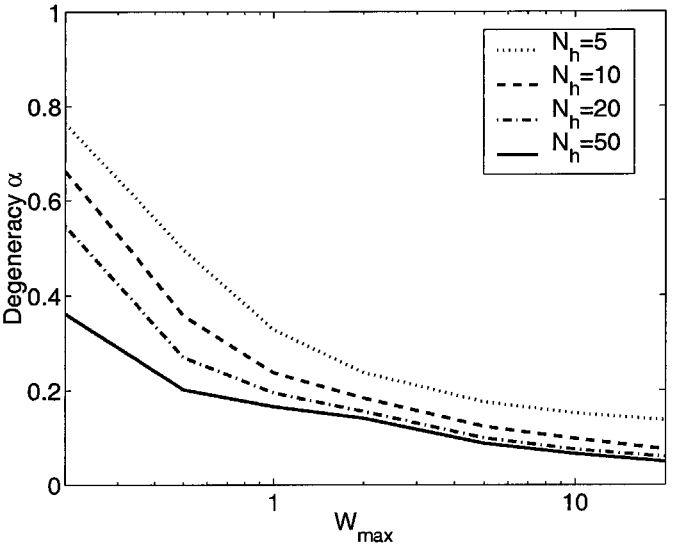


Figure 3.4: Degeneracy of neural network models. The level α of the degeneracy of the all-zero and all-one hypotheses is shown for neural networks with varying number of hidden units and varying weight distribution.

3.2.3 Levels of Degeneracy

We have shown that a neural network model may have a degeneracy. We now investigate with what level (in the notation of the previous section, that is, with what α_1) the degeneracy will occur. For a neural network with tanh hidden units, the output of each hidden unit is in $[-1, 1]$, so the output from the hidden layer $\mathbf{v}(x) \in [-1, 1]^{N_h}$ when there are N_h hidden units. $\mathbf{v}(x) \cdot \mathbf{w}$ is bounded by $\sum |w_i|$, so the hypotheses $g \equiv 0$ and $g \equiv 1$ will occur at least when $\sum |w_i| < |t|$. When w_i and t are chosen uniformly in $[-W_{max}, W_{max}]$, this occurs with probability $t^{N_h}/N_h!$, and the total weight of the degeneracy is at least $1/(N_h + 1)!$. This is a weak lower bound for the level of the degeneracy, and in general the true level will depend on the particular architecture of the network. A qualitative inspection, though, tells us that the chance of a degeneracy based solely on a bad choice of $\mathbf{w}$ and t goes quickly to zero as N_h increases. In fact, this holds for any weight distribution, since, if w_i and t are i.i.d., then $Pr[|t| > |w_i|] = Pr[|t| < |w_i|] \leq \frac{1}{2}$, and so $Pr[|t| > \max_i |w_i|] \leq 2^{-N_h}$.

Empirical estimates of the degeneracy of a neural network model are shown in Figure 3.4. The weights and thresholds for the networks were chosen uniformly from

$[-W_{max}, W_{max}]$, and W_{max} was varied from 0.2 to 20. Networks with different numbers of hidden units were compared, with N_h being 5, 10, 20 or 50. For the smallest network and weight range, the degeneracy has $\alpha > 0.75$, indicating that more than 3/4 of randomly chosen networks will result in trivial functions. We see that the dependence on N_h agrees qualitatively with what we expect from the preceding discussion, the degeneracy decreasing as N_h increases. An increase in W_{max} also results in a decreasing degeneracy level.

A larger number of hidden units N_h and maximum weight size W_{max} are usually associated with greater model complexity, which in turn is associated with poorer generalization. The results of this section illustrate that constraining these parameters to be small in an exhaustive learning setting can also degrade generalization due to the increased degeneracy.

3.3 Two-Stage Learning

In the preceding sections we have presented a very rudimentary analysis of some specific learning models. This analysis falls far short of describing the full generalization behavior for arbitrary target functions. Furthermore, it still does not allow us to benefit from a sophisticated learning algorithm associated with the model. In real applications, the exhaustive learning algorithm—selecting and evaluating random hypotheses—is impractical. For a more practical learning algorithm, though, the selected hypotheses will depend on the training set in nontrivial ways and the learning process cannot be cast in the bin model framework. Nevertheless, we can use an arbitrary learning algorithm and still take advantage of the bin model analysis through use of a validation set.

Given a data set $\mathcal{D}$ of size N, we partition it into a training set $\mathcal{D}_T$ of size N_T and a validation set $\mathcal{D}_V$ of size $N_V = N - N_T$. The training set will be used to select a hypothesis using a learning algorithm $\mathcal{A}$. This is the training stage, and we can consider the set of hypotheses produced by $\mathcal{A}$ to be the new learning model. The new distribution over hypothesis is implied by the results of $\mathcal{A}$ given random training sets

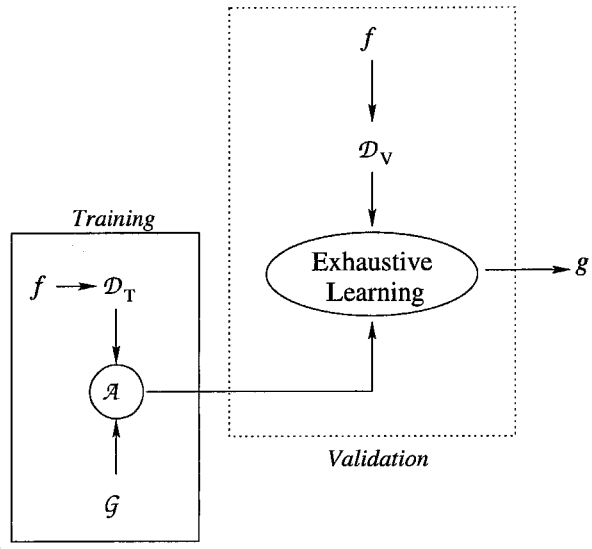
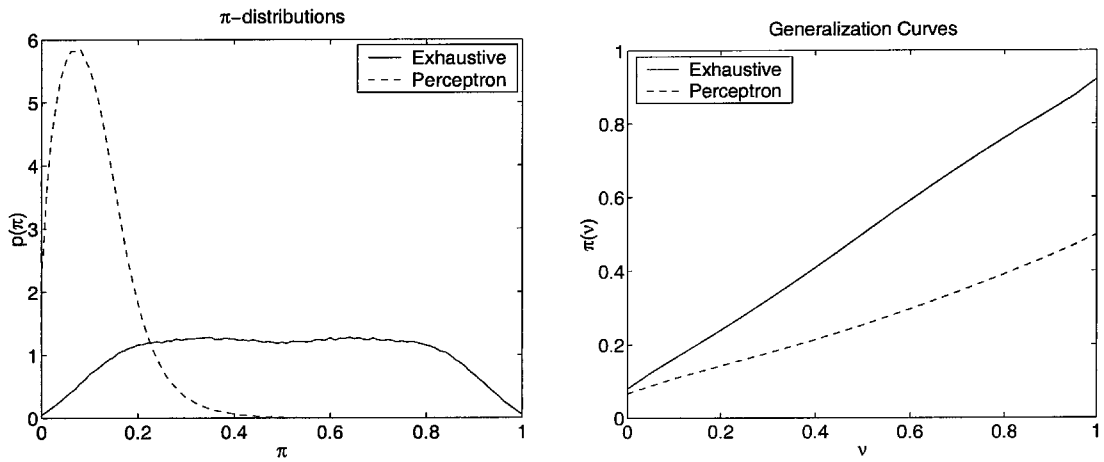


Figure 3.5: The two-stage learning scenario. The prior distribution over hypotheses is now induced by the results of a learning algorithm $\mathcal{A}$ on random data sets. Given this distribution, the *Validation* stage can be studied with the bin model analysis.

$\mathcal{D}_T$ of size N_T. This process is illustrated in Figure 3.5. The box labelled *Training* now plays the role that the model $\mathcal{G}$ alone played in Figure 1.1.

There is now a training set underlying the selection process, but for the purpose of generalization analysis, the second stage of learning is the same (the box labelled *Validation* in Figure 3.5). The hypotheses are produced according to some distribution p_G, and we can find the generalization curve with respect to the new p_π, now using $\mathcal{D}_V$ to compute the in-sample error. We now have a smaller set of examples, but presumably the hypotheses produced are better.

To illustrate the potential benefits of two-stage learning, we consider a two-dimensional classification problem. The target function is a linear classifier, and $\mathcal{G}$ is a linear perceptron learning model. Given 20 examples, we consider two different learning approaches. First, we consider exhaustive learning with all of the available data. Alternatively, we can use the two-stage learning process, using 10 examples to train a perceptron model with the perceptron learning algorithm [Rosenblatt 1962], and reserving 10 examples as a validation set. In the first case, the π-distribution is the distribution of $\pi(g)$ for randomly selected hypotheses (here with a uniform weight



(a) (b)

Figure 3.6: Example of two-stage learning. (a) shows the empirically measured π-distributions for a linear perceptron model under exhaustive learning and using the perceptron learning rule with 10 examples. (b) shows the corresponding generalization curves. The exhaustive learning result assumes 20 examples. Using the perceptron rule with 10 examples, the expected generalization error is lower for all values of ν, even though only 10 examples are available for validation.

distribution), and the generalization curve is given by (2.7) with $N = 20$. In the second case, the π-distribution now depends on the hypotheses that result from applying the perceptron learning rule to random sets of 10 points, and for the calculation of $\pi(\nu)$, now $N = 10$. Figure 3.6 shows the empirically estimated π-distributions and their respective generalization curves.

The random selection results in a π-distribution that is symmetric and relatively uniform. Using a training stage yields a π-distribution that is heavily skewed towards lower values of π. The result is that the generalization curve shows a lower $\pi(\nu)$ for all ν when we use the two-stage learning process. This example illustrates the advantage of the two-stage approach—the hypotheses produced in the training stage have much better performance, on average, than randomly selected ones, yet the bin model analysis can still be applied to the validation stage. We are thus no longer restricted to the inefficient exhaustive learning algorithm, but have not sacrificed all hope for a good out-of-sample error estimate.

3.3.1 Practical Implementation

The use of a validation set to estimate the out-of-sample error is not a novel idea, but the usual procedure is to use the sample error on this set directly as the estimate. We expect to obtain better estimates using the bin model results, but to do so we will need to estimate the π-distribution. We can accomplish this by sampling hypotheses produced by the training stage using data sets generated by partitioning $\mathcal{D}_T$ or by bootstrapping [Efron and Tibshirani 1993].

Instead of applying the learning algorithm to the set $\mathcal{D}_T$ of N_T examples available to the training stage, we construct a first stage training set $\mathcal{D}_0$ of N_0 examples and a first stage "test set" $\mathcal{D}_1$ of N_1 examples. We then apply the algorithm $\mathcal{A}$ to $\mathcal{D}_0$ to produce a hypothesis g, and take the sample error $\langle e(g(x), y)\rangle_{\mathcal{D}_1}$ as an estimate of $\pi(g)$. We are free to construct different $\mathcal{D}_0$ and $\mathcal{D}_1$ and repeat the training as often as we like in order to estimate the distribution of $\pi(g)$.

If we take $\mathcal{D}_0$ and $\mathcal{D}_1$ as a partition of $\mathcal{D}_T$, then $\mathcal{D}_1$ can genuinely be considered "out-of-sample" with respect to $\mathcal{D}_0$, and the estimate of π will be unbiased. Alternatively, we can construct $\mathcal{D}_0$ and $\mathcal{D}_1$ by sampling with replacement (bootstrapping) from $\mathcal{D}_T$. In this case the error $\langle e(g(x), y)\rangle_{\mathcal{D}_1}$ will be a biased estimate of π for a hypothesis g selected by $\mathcal{A}$, but it is not necessary that $N_0 + N_1 \leq N_T$.

Having estimated the π-distribution, we can apply $\mathcal{A}$ to a final training set $\mathcal{D}_0$. The resulting hypothesis can be considered to be randomly selected from a distribution p_G that produces the estimated p_π.

Unfortunately, the data sets $\mathcal{D}_0$ cannot, in general, reflect the true input distribution. Hence we cannot be assured of the accuracy of our estimate of p_π, especially when N_T is small. Nevertheless, in the next section we present experimental results that demonstrate the effectiveness of two-stage learning in practice.

3.4 Experiments

Two-stage learning eliminates the limitation to exhaustive learning and we have described how to estimate the π-distribution. It is now straightforward to apply our theoretical results to practical problems.

3.4.1 Artificial Data

In order to illustrate the two-stage learning process, a Monte Carlo simulation was run with target functions chosen randomly from a class of feed-forward neural networks. The input dimension was 34 and 200 data points were available to the learning algorithm (these were chosen to match the ionosphere data set discussed in the next section). $N_V = 50$ examples were reserved as the validation set $\mathcal{D}_V$. The data sets $\mathcal{D}_0$ and $\mathcal{D}_1$ of size 150 and 50 respectively were constructed by bootstrapping from the remaining 150 examples. A final test set of 1000 examples was used to estimate the out-of-sample error. In the training stage, a neural network learning model was trained on $\mathcal{D}_0$ for a fixed number (1000) of epochs of gradient descent, and the mean error on $\mathcal{D}_1$ was calculated for the resulting network. This was repeated for 1000 different bootstrapped $\mathcal{D}_0$ and $\mathcal{D}_1$, and the distribution of errors on $\mathcal{D}_1$ was taken as the estimate for the π-distribution. An example of this distribution is shown in Figure 3.7. From this distribution we compute an estimate $\pi_{\text{est}}(\nu)$ of the generalization curve.

After estimating the π-distribution and learning curve, a final hypothesis was selected by training a network on a new $\mathcal{D}_0$. The in-sample (validation) error ν on $\mathcal{D}_V$ was reported on this hypothesis. The error on the 1000 example test set was calculated as an estimate of the out-of-sample error π. Finally, using the estimate from the first learning stage, the value of $\pi_{\text{est}}(\nu)$ for the observed ν was reported. The average results of these three errors over 50000 runs (corresponding to 50000 different targets randomly chosen from a neural network model) are illustrated in Figure 3.8.

The dotted line shows ν and indicates the error one might expect by taking the validation set error to be a good indicator of the out-of-sample error ($\pi \simeq \nu$). The

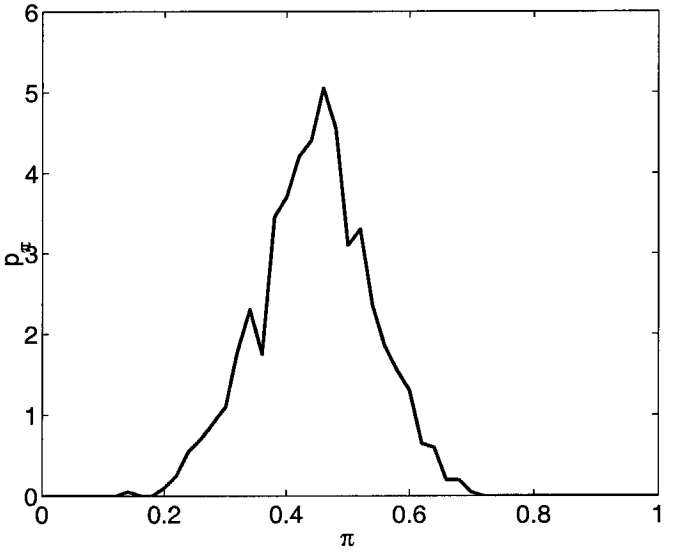


Figure 3.7: Estimated π-distribution for a toy problem. The distribution of errors on $\mathcal{D}_1$ made by a network trained on $\mathcal{D}_0$ is shown for 1000 pairs of bootstrapped data sets.

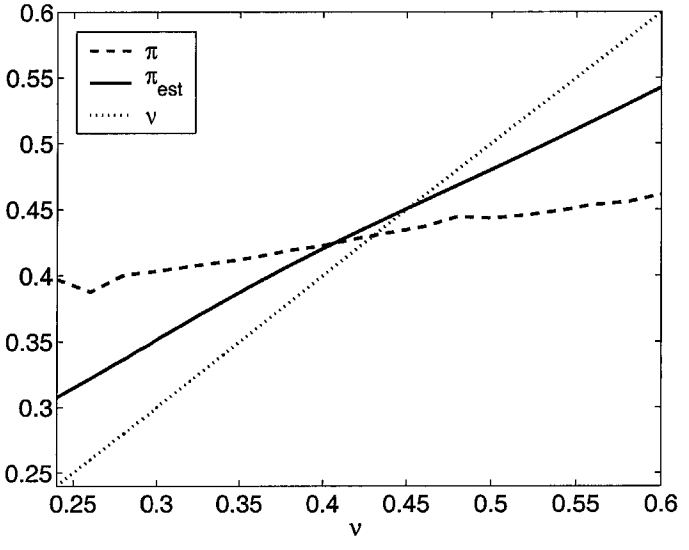


Figure 3.8: Average errors for a two-stage learning process. The dashed curve shows average out-of-sample errors π as a function of the validation error ν. The solid curve shows the average generalization curve $\pi_{\text{est}}(\nu)$ estimated with the two-stage learning process.

dashed line shows the mean out-of-sample error as estimated on the 1000 final test points. It is evident that ν is an underestimate of the true out-of-sample error when the error is small and an overestimate when the error is large. From the sharp peak of the π-distribution, we would expect that that errors far from the mean are unlikely. When such errors are observed, they are likely due to random variation in the data, and we must adjust our expectations of generalization ability. The solid line in Figure 3.7 shows the mean value of $\pi_{\text{est}}(\nu)$ calculated using estimates of the π-distributions from resampling in the first learning stage. It is evident that this estimate is (on average) a better approximation to the out-of-sample error than the in-sample error for almost any observed ν.

3.4.2 Ionosphere Radar Data

To illustrate the value of our approach for real problems, we ran experiments with the Ionosphere Radar data set from the UCI Machine Learning Repository [Blake and Merz 1998]. The data set comprises only 351 examples, each consisting of 17 pairs of real numbers (associated with radar returns) and a classification as either "good" or "bad" (suitable for further analysis or not; see [Sigillito *et al.* 1989] for more details).

For this problem, N_0, N_1 and N_V were the same as in the previous section, leaving 151 examples for a final test set. The first stage data sets were selected in the same manner, and the same neural network learning model was trained on $\mathcal{D}_0$ using gradient descent for 1000 epochs. The resulting π-distribution was estimated based on 5000 pairs of bootstrapped data sets, and an example of this estimate is shown in Figure 3.9. For an additional 10000 choices of $\mathcal{D}_0$, a network trained on $\mathcal{D}_0$ was tested on the validation set $\mathcal{D}_V$ and the 151 example test set, and $\pi(\nu)$ was calculated based on the estimated π-distribution. This entire experiment was repeated ten times, each with a different (random) splits of the data into training, validation and test sets.[2]

[2]The original order of examples in the data set as available from the UCI Machine Learning Repository and as used in [Sigillito *et al.* 1989] has the first 200 examples selected to be balanced by classification (101 good, 99 bad). This results in the final 151 examples having more than 82% good examples. The artificial balancing of the training set creates a different input distribution for the in-sample and out-of-sample data, which is inconsistent with our model of the learning problem. Therefore, we do not work with this division of the data and our results may not be quantitatively

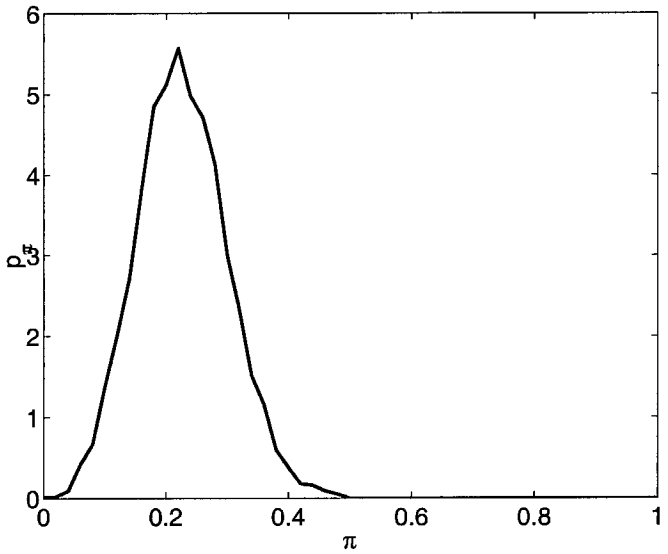


Figure 3.9: π-distribution estimate for the ionosphere radar problem. The distribution of training stage errors for 5000 pairs of bootstrapped data sets is shown.

The dashed curve in Figure 3.10 plots the resulting mean test error as a function of the validation error. As for the experiments on artificial data, we see that the validation error is a poor estimate when the observed error is far from the expectation. In fact, although the observed error ν may go to zero, the expected test error $\pi(\nu)$ never goes below 0.2. The solid curve in Figure 3.10 shows the estimated generalization curve $\pi_{\text{est}}(\nu)$ associated with the estimated π-distributions. For nearly all values of ν, the estimate π_{est} is better than the validation error as an indicator of the out-of-sample error. We can state this quantitatively by comparing the mean squared differences $\Delta(\pi, \nu) = 5.4 \times 10^{-3}$ and $\Delta(\pi, \pi_{est}) = \mathrm{E}_g[(\pi(g) - \pi_{\text{est}}(\nu(g)))^2] = 6.3 \times 10^{-4}$.

The two-stage learning process allows us to find improved estimates of the out-of-sample error, but we must sacrifice possibly valuable training data. The result of a smaller training set is shown in Table 3.1. For a learning system that uses all of the available data for learning, the in-sample error shown is the final training error reached, and is a poor indication of the expected out-of-sample error, but no other estimate is available. For the two-stage learning, the in-sample error corresponds to ν observed in the validation stage, and is, on average, a good estimate of the out-of-

comparable to those of [Sigillito *et al.* 1989].

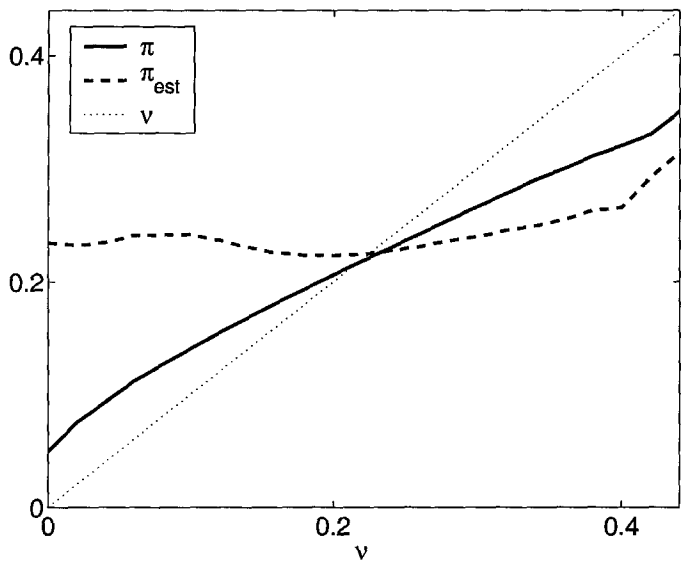


Figure 3.10: Average errors for the ionosphere radar problem. The average test error π and expected out-of-sample error estimate π_{est} are shown for different validation errors ν.

	In-Sample	Out-Of-Sample
No Validation	0.0120 ± 0.0002	0.2217 ± 0.0012
Two-Stage	0.2210 ± 0.0002	0.2311 ± 0.0001

Table 3.1: Average errors for the ionosphere radar classification problem. The in-sample error shown is the training error for the learning system without a validation stage, and is the validation stage ν for the two-stage learning system.

sample error. The consequence of removing of 50 points from the available training data, however, is an increase in the out-of-sample error of approximately 0.01.

3.5 Training Set Dependence

In Chapter 2, the major theoretical results (Theorems 2.4.1 and 2.5.1) implied that there can be no overfitting with exhaustive learning. The experimental $\pi(\nu)$ curves shown in figures 3.8 and 3.10 are not monotonically nondecreasing. How do we reconcile the disagreement between theory and experiment?

The binomial distribution of ν given in (2.3) is correct for randomly chosen training sets. It assumes that the errors $e(g(x), y)$ are independent Bernoulli trials with mean $\pi(g)$. For a given, fixed training set, however, dependencies may be introduced between errors on different data points for a selected hypothesis, or for different hypotheses on a given data point. We illustrate the possible problems with two pathological cases.

First, consider a training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ with N replicas of the same point, that is $(x_1, y_1) = (x_2, y_2) = \ldots = (x_N, y_N)$. The errors on these points given a hypothesis g are not independent. In fact, all the errors are the same, and only two possible values of ν are possible, depending on the value of $g(x_1)$.

$$\nu_{\mathcal{D}}(g) = \begin{cases} 0 & g(x_1) = y_1 \\ 1 & g(x_1) \neq y_1 \end{cases} . \tag{3.25}$$

The distribution of ν may depend on $\pi(g)$, but is not binomial in general. The bin model analysis does not apply, and in fact, since $\Pr[\nu = \frac{i}{N}] = 0$ for $0 < i < N$, the generalization curve is not defined.

Now consider a point x_1 for which $g(x_1) = \zeta_1$ for all $g \in \mathcal{G}$. Then the errors on this point for different hypotheses will also all take on the same value, $e(\zeta_1, y_1)$. If the training set consists entirely of such points, so that $g(x_i) = \zeta_i$ for all g, then only one in-sample error will be observed, $\nu(g) = N^{-1} \sum_{i=1}^N e(\zeta_i, y_i)$ for all g. Again the bin model analysis does not apply, and the generalization curve is not defined.

These two examples illustrate the extremes of error interdependence, but the effects may be present to varying degrees. Although the results of the bin model analysis describe the average behavior of a learning system, for a practical learning problem with only a single training set to consider, anomalies in the generalization behavior (including overfitting) may be observed. In the exhaustive learning formalism of [Schwartz *et al.* 1990], this amounts to a violation of the "self-averaging" assumption, and was identified as a problem by Wolpert and Lapedes [Wolpert and Lapedes 1992]. The analysis of the effects of specific training sets and individual points on the generalization curve is the subject of the next chapter.

3.6 Discussion

The main shortcomings of the analysis of Chapter 2 are its inability to address practical learning models and the dependence on the exhaustive learning algorithm. In this chapter we addressed these problems by looking at various practical learning systems.

For a simple linear model, we were able to calculate the exact form of the π-distribution for targets in the learning model. We discussed qualitative aspects of the π-distribution that can be inferred for neural network models, and discussed the effects of symmetry and degeneracy on the generalization curve.

The two-stage learning process introduced in section 3.3 frees us from the inefficient exhaustive learning algorithm. By estimating the π-distribution in the first stage, we are thus able to apply the generalization analysis of Chapter 2 to a practical learning problem. The experimental results of section 3.4 demonstrated that this procedure gives estimates of the out-of-sample error that are better than validation error estimates.

We briefly discussed the effects of a finite data sample on the generalization of a learning system, and showed that the error dependencies that arise in practical learning scenarios can adversely affect generalization. The dependence of the generalization behavior on particular examples and data sets is studied in much greater detail in the next chapter.

Chapter 4

Data Valuation

The results of Chapter 2 describe the *expected* generalization behavior of a learning system. In section 3.5 we briefly introduced practical problems related to using a single realization of the training set. We now look more closely at the question of which data sets lead to better or worse generalization, and the role played by individual training examples.

4.1 Overfitting

We return to the issue of overfitting in the context of the bin model analysis. Consider two data sets $\mathcal{D}_1 = \{(x_i, y_i)\}_{i=1}^N$ and $\mathcal{D}_2 = \{(x_i, 1 - y_i)\}_{i=1}^N$ containing examples with the same input values x_i but with opposite classifications. As long as $\Pr[x \in \{x_i\}_{i=1}^N] = 0$, a learning system that uses $\mathcal{D}_1$ as the training set (with exhaustive learning) will exhibit generalization behavior that is quite different from what it will show using $\mathcal{D}_2$. In fact, the two data sets will result in what is, in a sense, "opposite" generalization, $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_1}(g) = \nu_0] = \mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_2}(g) = 1 - \nu_0]$. This follows from

the fact that

$$\nu_{\mathcal{D}_1}(g) = \frac{1}{N}\sum_{i=1}^{N} e(g(x_i), y_i) \tag{4.1}$$

$$= \frac{1}{N}\sum_{i=1}^{N} (1 - e(g(x_i), 1 - y_i)) \tag{4.2}$$

$$= 1 - \nu_{\mathcal{D}_2}(g) \tag{4.3}$$

for all g. If training with $\mathcal{D}_1$ leads to what we consider to be good generalization (low ν implies low π), then training with $\mathcal{D}_2$ will lead to poor generalization and vice versa. The existence of good data sets implies the existence of bad data sets. We state this formally with the following theorem.

Theorem 4.1.1 *For any N and for any learning system for which the learning curve $\pi(\nu)$ for data sets of size N is not constant, there exists a data set $\mathcal{D}_N$ of size N for which $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_N}(g) = \nu_0]$ is not monotonically nondecreasing in ν_0.*

Proof of Theorem 4.1.1:

If for a data set $\mathcal{D}$ the expectation $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}}(g)] = c$ is constant, then we must have $c = \mathrm{E}_{\nu,g}[\pi|\nu_{\mathcal{D}} = \nu] = \mathrm{E}_g[\pi]$. Therefore, if $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_N}(g)]$ is constant for every data set $\mathcal{D}_N$ of size N, then

$$\pi(\nu_0) = \mathrm{E}_g[\pi(g)|\nu(g) = \nu_0] \tag{4.4}$$

$$= \mathrm{E}_{\mathcal{D}_N}[\mathrm{E}_g[\pi(g)]] \tag{4.5}$$

$$= \mathrm{E}_g[\pi(g)], \tag{4.6}$$

which is constant. By assumption this is not the case, so there must be a data set $\mathcal{D}_N^*$ for which $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_N^*}(g)]$ takes on at least two values. Consider the generalization curves given $\mathcal{D}_N^*$ and its complement

$$\overline{\mathcal{D}_N^*} = \{(x, 1 - y)|(x, y) \in \mathcal{D}_N^*\}. \tag{4.7}$$

Since $\mathrm{E}[\pi(g)|\nu_{\overline{\mathcal{D}_N^*}}(g) = \nu_0] = \mathrm{E}[\pi(g)|\nu_{\mathcal{D}_N^*}(g) = 1 - \nu_0]$, we cannot have both $\mathrm{E}[\pi(g)|\nu_{\overline{\mathcal{D}_N^*}}(g) = \nu_0]$ and $\mathrm{E}[\pi(g)|\nu_{\mathcal{D}_N^*}(g) = \nu_0]$ monotonically nondecreasing in ν_0 unless they are equal and constant. Since $\mathrm{E}_g[\pi(g)|\nu_{\mathcal{D}_N^*}(g)]$ is not constant by assumption, at least one of these generalization curves must not be monotonically nondecreasing.

■

The requirement that the generalization curve is not constant is quite reasonable, since a constant generalization curve implies that the in-sample error gives us no information about the expected out-of-sample error. For any problem with nontrivial generalization behavior, Theorem 4.1.1 ensures us that there are bad data sets, that is, data sets for which overfitting will be observed to some degree. If our training set is bad in this sense, then the monotonicity conditions promised by Theorems 2.4.1 and 2.5.2 will be violated.

The existence of bad data sets is perhaps not surprising, since the construction of the complementary data set in (4.7) requires at least some noise. By no means does this require a pathological case, as the following corollary of Theorem 4.1.1 makes clear.

Corollary 4.1.1 *For any learning system for which the input distribution is continuous, and for $\varepsilon > 0$ arbitrarily small, there exists a data set $\mathcal{D}_N$ of size N for which $\mathrm{E}_g[\pi|\nu_{\mathcal{D}_N}]$ is not monotonically nondecreasing in $\nu_{\mathcal{D}_N}$ and for which $\mathcal{D}_N = \{(x_i, \tilde{f}(x_i))\}_{i=1}^N$, where $\tilde{f}$ is a noisy realization of the target function with uniform noise with level ε.*

Many real-world learning problems have inputs that take on continuous values (for example, the ionosphere radar classification problem of section 3.4.2) and can be considered to have a continuous input distribution. In such a problem, any positive noise level, no matter how small, may result in a training set that yields an undesirable learning curve.

Having established that some training sets will lead to poor generalization, we would like to have a way to distinguish good and bad data sets a priori. With the

ability to make such a judgment, given a set of examples we would like to be able to select a subset that will give good generalization results.

4.2 Error Correlations

Heuristically, overfitting arises with exhaustive learning when the errors on the points in the training set are bad indicators of the overall error rate. That is, the errors $e(g(x_i), y_i)$ are poorly correlated with the expected errors $\pi(g)$. For a data set $\mathcal{D} = \{(x_i, f(x_i))\}$, we denote by $\rho(\mathcal{D})$ the error correlation

$$\rho(\mathcal{D}) = \mathrm{corr}_g[\nu_{\mathcal{D}}(g), \pi(g)] \tag{4.8}$$

$$= \frac{\mathrm{E}_g[\nu_{\mathcal{D}}(g)\pi(g)] - \mathrm{E}_g[\nu_{\mathcal{D}}(g)]\mathrm{E}_g[\pi(g)]}{\sqrt{\mathrm{Var}_g[\nu_{\mathcal{D}}(g)]\mathrm{Var}_g[\pi(g)]}}, \tag{4.9}$$

and for each point $x \in \mathcal{X}$, we denote by $\rho(x)$ the correlation

$$\rho(x) = \mathrm{corr}_g[e(g(x), f(x)), \pi(g)] \tag{4.10}$$

$$= \frac{\mathrm{E}_g[e(g(x), f(x))\pi(g)] - \mathrm{E}_g[e(g(x), f(x))]\mathrm{E}_g[\pi(g)]}{\sqrt{\mathrm{Var}_g[e(g(x), f(x))]\mathrm{Var}_g[\pi(g)]}}. \tag{4.11}$$

A correlation $\rho(\mathcal{D})$ very close to 1 indicates a nearly linear relationship between $\pi(g)$ and $\nu_{\mathcal{D}}(g)$ with a positive slope. In this case, we do not expect to observe overfitting. On the other hand, a correlation $\rho(\mathcal{D})$ less than 0 indicates that some overfitting must occur, and for correlations near -1 we expect that increasing $\nu_{\mathcal{D}}$ will almost always improve π. For one point x, if $\rho(x) = 0$, then the errors on x tell us nothing about the magnitude of π on average. If $\rho(x) < 0$, then the hypotheses that classify x correctly tend to have larger out-of-sample errors than those that make a mistake on x. Thus, at least qualitatively, ρ reflects how useful examples at particular input points (or sets) are for learning. We take $\rho(x)$ to be a measure of the value of an example with x correctly classified.

When the learning model is symmetric, we can use the results of section 3.2.1 to

write

$$\rho(x) = C_1 \mathrm{E}_g[e(g(x), f(x))\pi(g)] + C_2, \tag{4.12}$$

where $C_1 = 2/\sqrt{\mathrm{Var}_g[\pi(g)]}$ and $C_2 = 1/(2\sqrt{\mathrm{Var}_g[\pi(g)]})$. Looking at the mean square difference for one example we can write

$$\Delta_1(\pi, \nu) = \mathrm{E}_g[\pi(g)^2] + \mathrm{E}_g[\pi - 2\mathrm{E}_g[e(g(x), f(x))\pi(g)] \tag{4.13}$$

$$= C_3 - 2\mathrm{E}_g[e(g(x), f(x))\pi(g)]. \tag{4.14}$$

Comparing (4.12) and (4.13), we see that a larger $\rho(x)$ corresponds directly to a better generalization curve, and the single point x for which $\rho(x)$ is maximized is the point for which $\Delta_1(\pi, e(g(x), f(x)))$ is minimized.

For a data set $\mathcal{D}_N = \{(x_i, f(x_i)\}_{i=1}^N$, we can write $\rho(\mathcal{D}_N)$ in terms of $\rho(x_i)$.

$$\rho(\mathcal{D}_N) = \frac{\mathrm{E}_g[\nu_{\mathcal{D}_N}(g)\pi(g)] - \mathrm{E}_g[\nu_{\mathcal{D}_N}(g)]\mathrm{E}_g[\pi(g)]}{\sqrt{\mathrm{Var}_g[\nu_{\mathcal{D}_N}(g)]\mathrm{Var}_g[\pi(g)]}} \tag{4.15}$$

$$= \frac{\frac{1}{N}\sum_{i=1}^N \mathrm{E}_g[e(g(x_i), f(x_i))\pi(g)] - \frac{1}{N}\sum_{i=1}^N \mathrm{E}_g[e(g(x_i), f(x_i))]\mathrm{E}_g[\pi(g)]}{\sqrt{\mathrm{Var}_g[\nu_{\mathcal{D}_N}(g)]\mathrm{Var}_g[\pi(g)]}} \tag{4.16}$$

$$= \frac{1}{N}\sum_{i=1}^N \rho(x_i)\sqrt{\frac{\mathrm{Var}_g[e(g(x_i), f(x_i))]}{\mathrm{Var}_g[\nu_{\mathcal{D}_N}(g)]}}. \tag{4.17}$$

For a single point, the best choice maximizes ρ, but (4.17) illustrates a tradeoff in the choice of best data sets between points with large ρ (increasing $\rho(x_i)$ in the numerator) and sets with low variability in ν (decreasing $\mathrm{Var}_g[\nu_{\mathcal{D}}(g)]$ in the denominator).

Consider a simple example, illustrated in Figure 4.1. A target function defined on $\mathcal{X} = [0, 1]$ is shown in the upper half of the figure. We take p_X to be uniform and use a learning model of step functions

$$\mathcal{G} = \{\mathrm{sgn}(x - \alpha) | \alpha \in [0, 1]\} \cup \{\mathrm{sgn}(\alpha - x) | \alpha \in [0, 1]\} \tag{4.18}$$

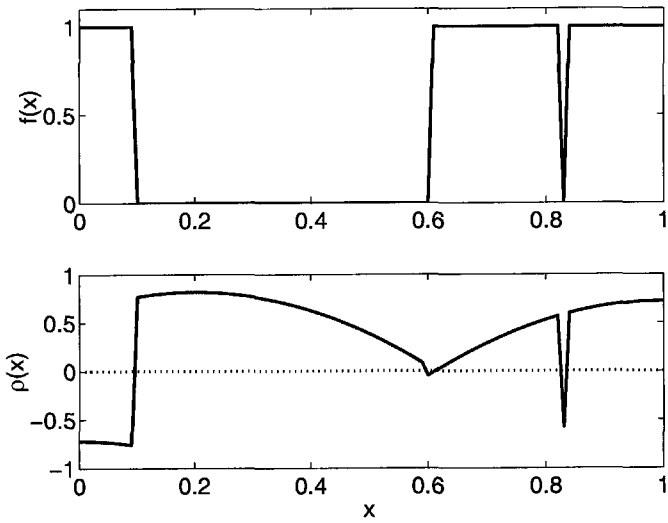


Figure 4.1: ρ values for a simple problem. The target function $f(x)$ is plotted in the upper graph. The value of $\rho(x)$ for a uniform symmetric step function model is plotted in the lower graph. The "outlier" near 0.83 has $\rho \ll 0$. The region $x < 0.1$ also has $\rho < 0$, indicating that these points may be detrimental to generalization using this learning model.

with uniform prior $p_G(\text{sgn}(x-\alpha)) = p_G(\text{sgn}(\alpha - x)) = \frac{1}{2}$ for all α. With this learning model, the value of $\rho(x)$ can be computed exactly, and is plotted in the lower half of the figure. The single point classified as 0 with $x \simeq 0.83$ appears to be an outlier and has $\rho \ll 0$, correctly indicating that it would form a bad example. All x in the region $0 \leq x \leq 0.1$ also have $\rho \ll 0$, even though we would not consider them "outliers." The complexity of the target function is such that the learning model cannot fit these points and simultaneously have low overall out-of-sample error, thus these points are detrimental to generalization under exhaustive learning. It is also apparent that, unlike more sophisticated learning algorithms, points far from the decision boundary are more useful for generalization in this context than are points near the boundary.

4.3 Data Set Selection

In this section we demonstrate that the ideas of section 4.2 can be applied to a real learning problem, and that ρ based data selection tends to improve generalization.

Several things hinder the estimation of ρ in practice. We do not have access to the true value of $\pi(g)$, but can only approximate it by using a validation set. We cannot generally find the true correlation with respect to p_g, although we may sample arbitrarily often from the learning model, and hence may get as good an estimate as our patience allows.

4.3.1 ρ Estimation

Here we describe how to estimate ρ from the data, and how accurate we can expect it to be. We are restricted to a finite training set $\mathcal{D}$ of N points, but, in principle, we may sample from the learning model as often as we like. We will discuss the estimation of ρ using a sample S of N_g hypotheses selected randomly (according to p_G) from the learning model.

For each example $(x_i, f(x_i))$ in $\mathcal{D}$, we estimate the out-of-sample error on a hypothesis by the leave-one-out error

$$\nu^{(i)}(g) = \frac{1}{N-1} \sum_{j \neq i} e(g(x_j), f(x_j)). \tag{4.19}$$

We then compute an estimate $\widehat{\rho}(x_i)$ of the correlation $\rho(x)$ by taking the sample correlation over S, that is,

$$\widehat{\rho}(x_i) = \frac{\left\langle \nu^{(i)}(g) e(g(x_i), f(x_i)) \right\rangle_S - \left\langle \nu^{(i)}(g) \right\rangle_S \left\langle e(g(x_i), f(x_i)) \right\rangle_S}{\widehat{\sigma}(\nu^{(i)}(g)) \widehat{\sigma}(e(g(x_i), f(x_i))}, \tag{4.20}$$

where $\widehat{\sigma}(\nu^{(i)}(g))$ and $\widehat{\sigma}(e(g(x_i), f(x_i))$ denote the sample estimates of the standard deviations of $\nu^{(i)}(g)$ and $e(g(x_i), f(x_i)$ taken over S. If we know that our learning model is symmetric, we can replace the known statistics in (4.20) by their true values. Applying the results of section 3.2.1, we can write

$$\rho(x) = \frac{2\left(\mathrm{E}_{g,x}[\nu(g) e(g(x), f(x))] - 1/4\right)}{\sqrt{\mathrm{Var}_g[\pi(g)]}} \tag{4.21}$$

for the true correlation, and

$$\widehat{\rho}(x_i) = \frac{2\left(\left\langle \nu^{(i)}(g) e(g(x_i), f(x_i))\right\rangle_S - 1/4\right)}{\widehat{\sigma}(\nu^{(i)}(g))} \tag{4.22}$$

for the estimate. For symmetric models (4.22) should provide a better estimate of $\rho(x_i)$ than (4.20), but it does not represent a true correlation and may give values outside of $[-1, 1]$.

We can make some quantitative statements about the accuracy of our estimates. The Hoeffding bound [Hoeffding 1963] ensures us that

$$\Pr[|\nu^{(i)}(g) - \pi(g)| > \epsilon] \leq 2\exp(-2\epsilon^2(N-1)), \tag{4.23}$$

and therefore that

$$\Pr[\max|\nu^{(i)}(g) - \pi(g)| > \epsilon] \leq 2N_g \exp(-2\epsilon^2(N-1)) \tag{4.24}$$

when we consider the maximum deviation over the sampled hypotheses. Since we may select N_g to be arbitrarily large, we can instead use the VC bound

$$\Pr[\max|\nu^{(i)}(g) - \pi(g)| > \epsilon] \leq 6\mathrm{e}^{2\epsilon}\Delta(2(N-1))\exp(-\epsilon^2(N-1)), \tag{4.25}$$

where $\Delta(\cdot)$ is the 'growth function' and obeys

$$\Delta(k) \begin{cases} = 2^k & k \leq d_{\mathrm{VC}} \\ \leq (\mathrm{e}(N-1)/d_{\mathrm{VC}})^{d_{\mathrm{VC}}} & k > d_{\mathrm{VC}} \end{cases} \tag{4.26}$$

for a model with VC dimension d_{VC} [Parrondo and Van den Broeck 1993]. The Hoeffding bound further ensures us that, since $0 \leq e(g(x), f(x))\pi(g) \leq 1$,

$$Pr[|\langle e(g(x), f(x))\pi(g)\rangle - \mathrm{E}_g[e(g(x), f(x))\pi(g)]| > \epsilon] \leq 2\exp(-2\epsilon^2 N_g). \tag{4.27}$$

In our estimation of ρ, we compute the sample mean $\langle e(g(x_i), f(x_i))\nu^{(i)}(g)\rangle$ in place of

the expectation $E_g[e(g(x), f(x))\pi(g)]$. A large deviation between these values implies a large deviation of the type in either (4.24) or (4.27),

$$\begin{aligned}
&\left|\langle e(g(x_i), f(x_i))\nu^{(i)}(g)\rangle - E_g[e(g(x), f(x))\pi(g)]\right| > \epsilon \\
&\quad \Rightarrow \Big(\left|\langle e(g(x_i), f(x_i))\nu^{(i)}(g)\rangle - \langle e(g(x), f(x))\pi(g)\rangle\right| > \frac{\epsilon}{2} \\
&\qquad \text{or } \left|\langle e(g(x_i), f(x_i))\pi(g)\rangle - E_g[e(g(x), f(x))\pi(g)]\right| > \frac{\epsilon}{2}\Big) .
\end{aligned} \tag{4.28}$$

Therefore,

$$\begin{aligned}
&\Pr[\left|\langle e(g(x_i), f(x_i))\nu^{(i)}(g)\rangle - \mathrm{E}_g[e(g(x), f(x))\pi(g)]\right| > \epsilon] \\
&\qquad \leq 2N_g \exp(-\epsilon^2 N/2) + 2\exp(-\epsilon^2 N_g/2)
\end{aligned} \tag{4.29}$$

using (4.24), or alternatively

$$\begin{aligned}
&\Pr[\left|\langle e(g(x_i), f(x_i))\nu^{(i)}(g)\rangle - \mathrm{E}_g[e(g(x), f(x))\pi(g)]\right| > \epsilon] \\
&\qquad \leq 6\mathrm{e}^{\epsilon}\Delta(2(N-1))\exp(-\epsilon^2(N-1)/4) + 2\exp(-\epsilon^2 N_g/2)
\end{aligned} \tag{4.30}$$

using (4.25). The implication of these bounds can be seen by comparing (4.21) and (4.22). Since $\mathrm{Var}_g[\pi(g)]$ and $\widehat{\sigma}(\nu^{(i)}(g))$ may be arbitrarily small (in the presence of noise with $\epsilon \to \frac{1}{2}$, for example), our error in the estimate of ρ may be large. With high probability, however, the numerators of (4.21) and (4.22) will be close, allowing us to determine the correct sign of ρ given sufficient data (and provided $\widehat{\rho} \neq 0$). As we will see in section 4.4.1, under certain conditions it is appropriate to discard all data with $\rho < 0$, so knowing the sign of ρ is sufficient for data selection.

It should be noted that data valuation based on $\rho(x)$ differs from active learning [Cohn *et al.* 1995], which also attempts to select an example that will be useful for learning. In the active learning scenario, examples may be chosen arbitrarily, and the value is ordinarily determined by how well an example will improve a working hypothesis. In contrast, the ρ valuation is a priori given the learning model, and our goal is only to eliminate bad data from a given training set. This selection also differs

from pure outlier detection, as ρ values may indicate that examples are detrimental to learning even in the absence of noise (as in the example of Figure 4.1).

4.3.2 Experimental Results

We demonstrate the effectiveness of data set selection by ρ valuation using neural network models and artificial target functions. Given a data set $\mathcal{D}$ of N examples, we wish to construct a training set $\mathcal{D}_T \subset \mathcal{D}$ that will yield good generalization behavior.

We showed above that $\rho(x) < 0$ can be taken as an indication that $(x, f(x))$ is a "bad" example. Inspection of (4.17), however, indicated that inclusion of some examples with $\rho(x) < 0$ in a training set may result in a better $\rho(\nu)$. Furthermore, we saw above that an estimate of $\widehat{\rho} < 0$ may correspond to a $\rho > 0$ when $|\widehat{\rho}|$ is sufficiently small. In order to reject bad data without risking good data in the process, we consider data sets constructed by rejecting points with $\rho < \rho_t$, for some appropriate threshold $\rho_t \leq 0$.

To test the effectiveness of this procedure, we ran Monte Carlo simulations with neural network models with tanh hidden units. The input distribution was uniform in $[-5, 5]^3$, and the learning model consisted of neural networks with 3 hidden units. Random target functions were selected from a neural network model with 4 hidden units, and targets for which the classifications were heavily unbalanced ($\Pr[f(x) = 0] < 0.4$ or $\Pr[f(x) = 1] < 0.4$) were rejected. Data sets $\mathcal{D}$ of $N = 200$ examples were generated and $\widehat{\rho}(x)$ was estimated for each one based on $N_g = 500$ sampled hypotheses. Training sets $\mathcal{D}_T(\rho_t)$ were constructed by discarding examples with $\widehat{\rho}(x) < \rho_t$ from $\mathcal{D}$ for ρ_t ranging between -0.09 and 0. The neural network model was trained using gradient descent for 1000 epochs $\mathcal{D}$ and each of the $\mathcal{D}_T(\rho_t)$, and the resulting mean squared errors between the (continuous) network output and the target value was reported on a test set of 10000 examples.

The improvement in MSE resulting from the use of $\mathcal{D}_T(\rho_t)$ instead of $\mathcal{D}$ averaged over 500 targets is shown in Figure 4.2. When no noise is added to the data, discarding data always results in a larger error. When there is noise, however, we obtain a better

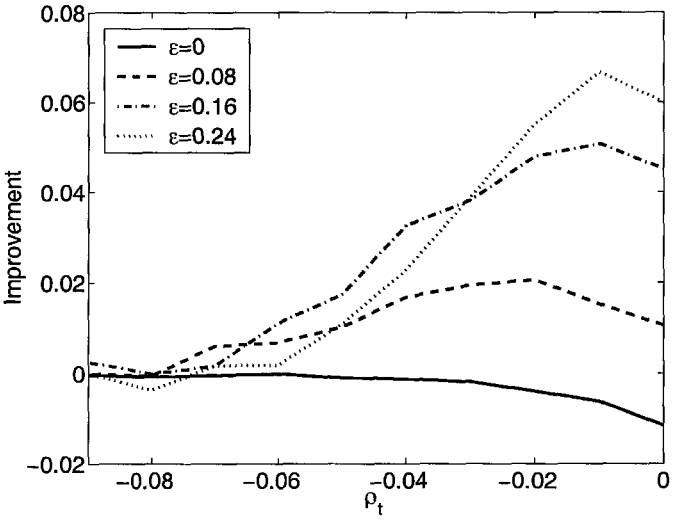


Figure 4.2: Generalization error improvement using ρ based data selection. The average improvement in out-of-sample error is shown for neural networks trained with data sets constructed by discarding examples with $\rho < \rho_t$. Improvement is relative to the same model trained with all available examples. The different curves correspond to different levels of uniform noise added to the data.

out-of-sample error by discarding data for almost all choices of ρ_t. In particular, using $\rho_t = 0$ results in a significant improvement for all noise levels.

If we have enough confidence in labelling examples as bad that we consider it appropriate to discard them, then we might instead consider reclassifying them $(x, y) \rightarrow (x, 1 - y)$. Given a data set $\mathcal{D}$, we construct the reclassified set $\mathcal{D}_R(\rho_t)$ by replacing each example for which $\widehat{\rho}(x) < \rho_t$ by its reclassified version. Experimental results for data reclassification are compared with data rejection in Figure 4.3 for data sets with 12% noise. Although reclassification is slightly worse than rejection for ρ_t near 0, the out-of-sample error is lower for networks trained on $\mathcal{D}_R(\rho_t)$ than for those trained on $\mathcal{D}$ for every ρ_t.

For these experiments we have used gradient descent and have reported mean squared errors on the real valued outputs of our neural network model. Valuation based on ρ is justified by an analysis based on exhaustive learning, but even in this complicated learning scenario we find that selection of a training set based on estimates of ρ results in better out-of-sample performance. This can be seen as a con-

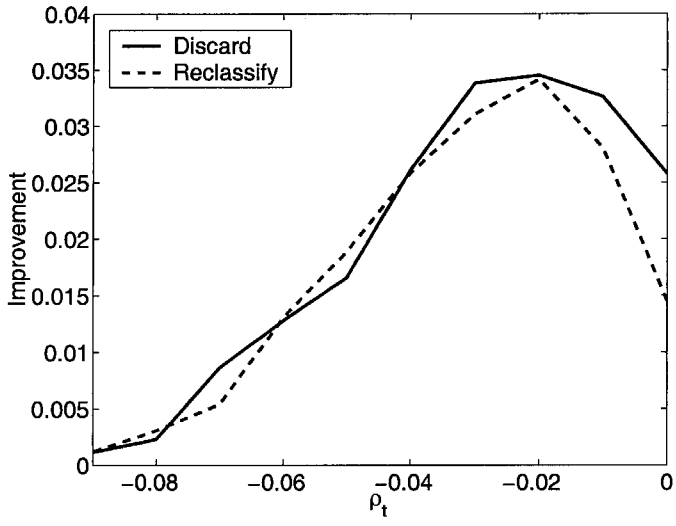


Figure 4.3: Error improvements with data rejection and reclassification. The solid curve shows error improvements resulting from data sets constructed by discarding points with $\rho < \rho_t$ as a function of ρ_t. The dashed curve shows the corresponding improvement if we keep all examples but reclassify those with $\rho < \rho_t$. Both curves show results averaged over random neural network targets with 12% uniform noise.

sequence of our valuation identifying examples that are actually outliers—examples that carry incorrect information about the target function. Rejection (or reclassification) of outliers should increase the usefulness of a training set for almost any learning system.

4.4 Noise

In the definition of $\rho(x)$, we consider the correlation of $\pi(g)$ with the error g makes on x with respect to the true target value $f(x)$. The assumption of correctly classified examples carries over into our definition of the estimate $\widehat{\rho}$ in (4.20). In section 4.3.2 we experiment with noisy data, so this assumption is implicitly violated. We can deal with this by assuming an observed realization of the noisy target to be the true f, but it is more convenient to generalize the definition of ρ to allow for examples (x, y)

with $y \neq f(x)$. We write

$$\rho(x|y) = \mathrm{corr}_g[e(g(x), y)\pi(g)] \tag{4.31}$$

for an example (x, y) and form the analogous estimate $\widehat{\rho}(x_i|y_i)$ for an example (x_i, y_i) from a finite data set by replacing $f(x_i)$ with y_i in (4.20).

For any $x \in \mathcal{X}$ and any $g \in \mathcal{G}$ it is clear that $e(g(x), 0) = 1 - e(g(x), 1)$, from which it follows that

$$\begin{aligned}
\rho(x|0) &= \left(\mathrm{E}_g[e(g(x), 0)\pi(g)] - \mathrm{E}_g[e(g(x), 0)]\mathrm{E}_g[\pi(g)]\right)/C && (4.32)\\
&= \left(\mathrm{E}_g[(1 - e(g(x), 1))\pi(g)] - \mathrm{E}_g[(1 - e(g(x), 1))]\mathrm{E}_g[\pi(g)]\right)/C && (4.33)\\
&= \left(-\mathrm{E}_g[e(g(x), 1)\pi(g)] + \mathrm{E}_g[e(g(x), 1)]\mathrm{E}_g[\pi(g)]\right)/C && (4.34)\\
&= -\rho(x|1), && (4.35)
\end{aligned}$$

where $C = \sqrt{\mathrm{Var}_g[\pi(g)]\mathrm{Var}_g[e(g(x), 0)]} = \sqrt{\mathrm{Var}_g[\pi(g)]\mathrm{Var}_g[e(g(x), 1)]}$. Thus, reclassification of any single example results in a sign change of ρ. This provides a justification for the data reclassification used in section 4.3.2. Examples that we confidently consider to be bad (based on $\widehat{\rho}$) become examples that we can confidently consider good by reclassifying. It is also a starting point for studying the effects of noise.

Consider a learning problem with uniform BSC noise with level ε. The noisy out of sample errors $\tilde{\pi}(g)$ are given by the linear transformation (2.17) of $\pi(g)$ (with positive scaling). Since $\mathrm{corr}[A, B] = \mathrm{corr}[\alpha A + \beta, B]$ for constants α, β with $\alpha > 0$, a noisy correlation

$$\tilde{\rho}(x|y) = \mathrm{corr}_g[e(g(x), y)\tilde{\pi}(g)] \tag{4.36}$$

will have the same value as the noiseless $\rho(x|y)$ under BSC noise. The effect of noise

on a single example is therefore

$$\tilde{\rho}(x|y) = \rho(x|y) \tag{4.37}$$

$$= \begin{cases} \rho(x) & y = f(x) \\ -\rho(x) & y = 1 - f(x) \end{cases} . \tag{4.38}$$

Thus, the addition of uniform noise to the targets does not change the value of $\rho(x)$ when x is correctly classified. Points that are misclassified, however, will have $\tilde{\rho}(x|y) = -\rho(x)$. (4.38) is quite simple, but its implications for the study of the effects of noise are significant.

4.4.1 Noise Estimation

Without some prior belief about the possible form of the target function, it is not generally possible to detect noise in the data or estimate the noise level [Magdon-Ismail 2000]. Given a prior distribution over target functions, however, analysis of the ρ valuations can give us insight into the noise distribution, and can allow us to detect and reject noisy data with confidence. Assuming a prior distribution over noiseless target functions, there is an associated distribution p_ρ for the values of ρ for randomly selected inputs and targets. The addition of uniform noise with probability ε results in a fraction ε of the points having $\tilde{\rho}(x) = -\rho(x)$ and the remaining $1 - \varepsilon$ having $\tilde{\rho}(x) = \rho(x)$. Hence the distribution of $\tilde{\rho}$ will take the form

$$p_{\tilde{\rho}}(r) = (1 - \varepsilon)p_\rho(r) + \varepsilon p_\rho(-r). \tag{4.39}$$

This transformation is illustrated in Figure 4.4. The noiseless p_ρ has a single peak near 0.3. The addition of noise results in a distribution of $\tilde{\rho}$ with two peaks, one near 0.3 (corresponding to unaffected data) and one near -0.3 (corresponding to noisy data).

Given a suitable prior over noiseless target functions, we can therefore estimate the noise level by estimating p_ρ from random targets and $p_{\tilde{\rho}}$ from the available examples.

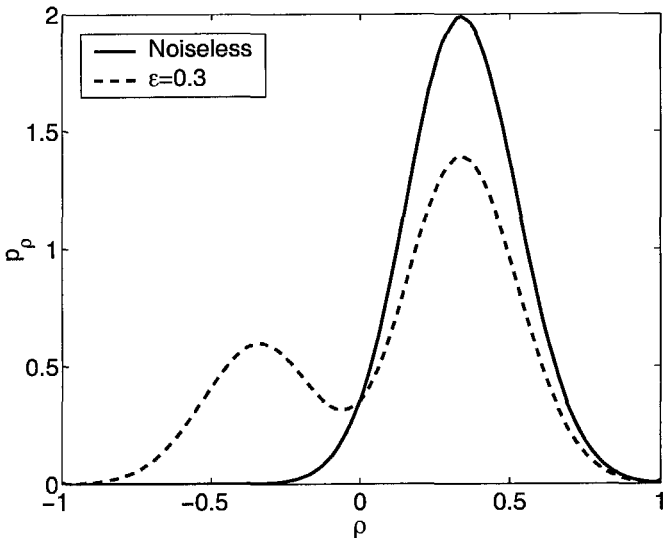


Figure 4.4: Noiseless and noisy ρ-distributions. The solid curve shows a hypothetical ρ-distribution and the dashed curve shows the corresponding expected ρ-distribution when 30% uniform noise is added.

Since we can only estimate π and ρ from a finite data set, our estimate of ρ can only take on finitely many values. For a target prior distribution p_F, we approximate the ρ-distribution by the discrete distribution described by

$$\Pr[\rho = r] = \int_{x \in \mathcal{X}} \int_{\{f | \widehat{\rho}(x|f(x)) = r\}} p_F(s) p_X(t) ds \; dt. \tag{4.40}$$

Given a noisy training set, we approximate the noisy distribution $p_{\tilde{\rho}}(r)$ by the relative frequency of observing the estimate $\widehat{\rho}(x_i|y_i) = r$. The value of ε for which these distributions most closely satisfy (4.39) can be taken as an estimate of the level of noise in the data.

The first and second terms on the right side of (4.39) correspond to contributions to the distribution of ρ by correctly and incorrectly classified examples respectively. A comparison of the magnitude of each term for a given $\tilde{\rho}$ can tell us whether an example is more likely correctly or incorrectly classified. Given the true a priori distribution of ρ, the conditional probability of an example being free of noise given

its noisy ρ is

$$\Pr[y = f(x)|\tilde{\rho} = r] = \frac{(1-\varepsilon)p_\rho(r)}{p_{\tilde{\rho}}(r)}. \tag{4.41}$$

(4.41) gives the a posteriori probability that an example is correctly classified. This suggests a solution to the problem encountered in section 4.3.2 of how to select and appropriate threshold ρ_t for labelling data as noisy. An example (x, y) should be labelled as noisy when

$$Pr[y \neq f(x)|\tilde{\rho} = \rho(x|y)] > Pr[y = f(x)|\tilde{\rho} = \rho(x|y)]. \tag{4.42}$$

For an arbitrary p_ρ, the set of $\rho(x|y)$ may not necessarily be easy to describe. Intuitively, we would expect that, given a reasonable learning model and with no noise in the data, most x should have $\rho(x) > 0$, and very few should have $\rho(x) \ll 0$. We define a certain class of "well behaved" ρ-distributions that have these properties.

Definition 4.4.1 *Define $\mathcal{P}$ to be the set of unimodal probability distributions p on $[-1, 1]$ that have $p(r) = 0$ for all $r < -m(p)$, where $m(p)$ denotes the mode of p.*

Qualitatively, a distribution $p \in \mathcal{P}$ must have a peak for some $m(p) > 0$ and a tail on the left that dies off before it gets to $-m(p)$. The distribution in Figure 4.4 is in $\mathcal{P}$, since it is unimodal with $m(p) \simeq 0.34$ and has $p(r) = 0$ for all $r < -0.25$.

The unimodal requirement ensures that $p(r)$ is monotonically nondecreasing in r for $-m(p) \leq r \leq m(p)$. This, in turn, guarantees that $p(-r)$ is monotonically nonincreasing in r for $-m(p) \leq r \leq m(p)$. It follows that, for any $p \in \mathcal{P}$ and $\varepsilon \in [0, 1]$, there is a unique $\rho_t(\varepsilon) \in [-m(p), m(p)]$ such that

$$\varepsilon p(-r) > (1-\varepsilon)p(r) \quad \forall r < \rho_t \tag{4.43}$$

and

$$\varepsilon p(-r) \leq (1-\varepsilon)p(r) \quad \forall r \geq \rho_t. \tag{4.44}$$

Thus, while the condition (4.42) describes the general rule for declaring examples as noisy, if p_ρ satisfies the conditions for membership in $\mathcal{P}$, then we see that a simple thresholding rule applies, and an appropriate threshold ρ_t solves

$$\varepsilon p(-\rho_t) = (1-\varepsilon)p(\rho_t). \tag{4.45}$$

We have described a procedure for estimating the noise level, and given the noise level, a procedure for indicating which points are most likely misclassified. These require a prior distribution over targets in order to find an estimate of the noiseless distribution p_ρ.

4.5 Application to Image Denoising

We illustrate the use of ρ valuation for noise estimation and outlier detection with an example of black-and-white image denoising. The pixel data can be interpreted as examples of target functions in a learning problem, and noise in the data corresponds to degradation of the image. Restoration of digital images has been widely studied and has applications in many scientific disciplines [Banham and Katsaggelos 1997].

We can consider the rows of an image to be functions of the horizontal pixel position. For a black-and-white (one bit) images, the output (pixel color) can be represented by $\{0,1\}$, and the row can be considered a binary classifier. To make this precise, we represent a row of an image by a function $f : \{1,2,\ldots,N_X\} \to \{0,1\}$, where N_X is the horizontal dimension of the image (the number of pixels in a row) and $f(x) = 0$ if the pixel at location x is black, and $f(x) = 1$ if the pixel is white. Figure 4.5(a) shows one such function with $N_X = 240$.

For our learning model $\mathcal{G}$, we use the set of functions represented by the rows of a set of natural images. For the experiments described in this section, the learning model consists of 1470 hypotheses, corresponding to the 1470 rows of the images illustrated in Figure 4.6.[1] The set of functions corresponding to the 400 rows of

[1]The images used for the learning model are black-and-white thresholded versions of images provided with the MATLAB Image Processing Toolbox.

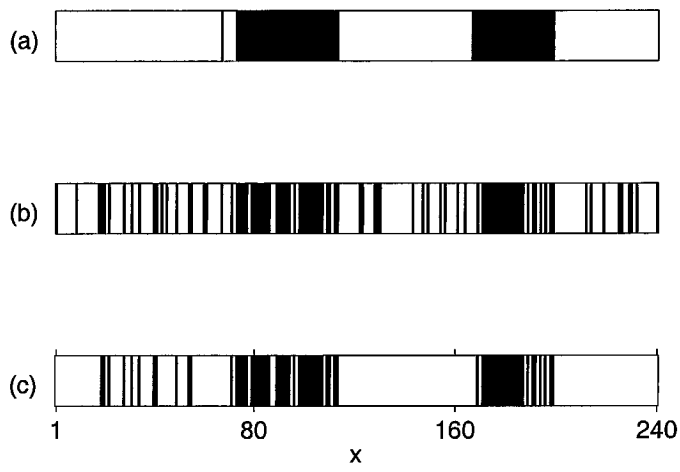


Figure 4.5: Black-and-white image row target function. One row of a black-and-white image defines a binary target function in (a). The input is discrete in $\{1, 2, \ldots, 240\}$ and corresponds to the position of a pixel. The addition of BSC noise with probability 0.2 results in the noisy function shown in (b). Application of a denoising procedure based on ρ to (b) yields the row shown in (c).

the image shown in Figure 4.7(a) were used as target functions. Since an image row provides a classification for every point in the discrete input space, the study of learning and generalization is of little interest. We look instead at the problems of noise detection and image restoration. To do so, we add BSC noise to the image uniformly with varying intensity. Figure 4.5(b) shows the single function of 4.5(a) with 20% noise added, and Figure 4.7(b) shows the result of adding 30% noise to each target row. The pixel data for each row in the noisy image corresponds to the value of $\tilde{f}(x)$ for each point in the input space, and the in-sample error ν is exactly the noisy out-of-sample error $\tilde{\pi}$.

For each target row, we can estimate the value of $\rho(x)$ for each pixel location as

$$\widehat{\rho}(x) = \mathrm{corr}_{g \in \mathcal{G}}[e(g(x), \tilde{f}(x)), \tilde{\pi}(g)]. \tag{4.46}$$

Following the analysis of section 4.4, we consider points with $\rho < \rho_t(\varepsilon)$ to be outliers and reclassify them.

We consider the learning model to be an appropriate prior distribution over possible target functions. Using uniform distributions over the input space and learning

Figure 4.6: Natural image row learning model. The set of rows of five black-and-white images was used as a learning model for noise detection experiments.

model in (4.40), our approximation to the ρ-distribution becomes

$$\Pr[\rho = r] = \frac{1}{N_X|\mathcal{G}|} \sum_{x=1}^{N_X} \sum_{g_i \in \mathcal{G}} \widehat{\rho}(x|g_i(x)). \tag{4.47}$$

The approximate noiseless ρ-distribution is shown in Figure 4.8. We can use this ρ-distribution and (4.45) to find appropriate $\rho_t(\varepsilon)$ for varying noise levels. (This distribution does not completely satisfy the conditions of definition 4.4.1, but there is one dominant mode, and so, to a reasonable approximation, a single $\rho_t(\varepsilon)$ suffices for outlier detection.) For $0 < \epsilon < \frac{1}{2}$, Figure 4.9 shows the resulting thresholds. When the data is noiseless $\rho_t = -1$, indicating that no data should be discarded. As $\varepsilon \to \frac{1}{2}$, $\rho_t \to 0$, indicating that any point with $\rho < 0$ is most likely misclassified.

Given an estimate of the noiseless ρ-distribution and thresholds ρ_t for any given noise level, we are left with the task of estimating the noise in the data. We find the distribution of ρ for the examples of the given targets, and find the value of ε for which the noisy distribution $p_{\tilde{\rho}}$ given by (4.39) most closely matches the observed distribution (in the sense of mean squared difference). An example is shown in Figure 4.10. The distribution of ρ occurring in the image with 20% noise is plotted as a solid

(a) (b)

Figure 4.7: Clean and noisy targets for image denoising experiment. The 400 rows of the image on the left are used as target functions. On the right is the image with bits flipped with probability 0.3.

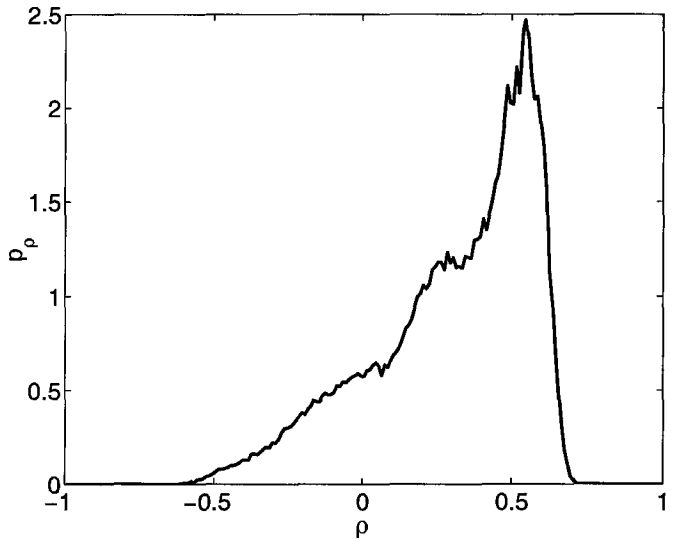


Figure 4.8: Noiseless ρ distribution. Using the learning model (Figure 4.6) as the prior over targets, the resulting ρ distribution is shown as a histogram.

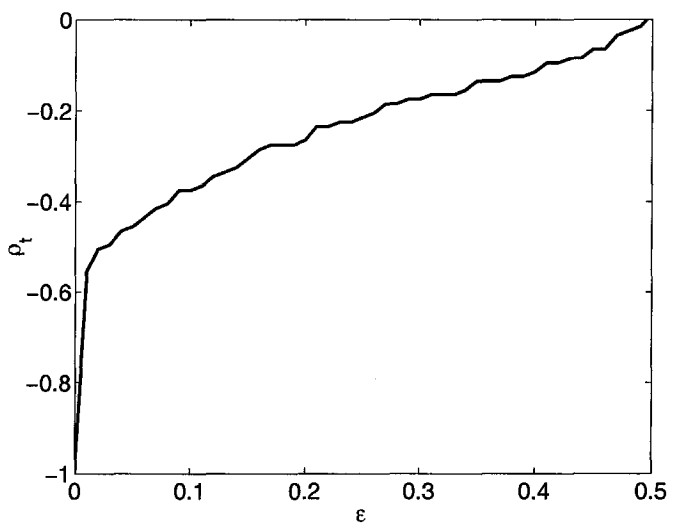


Figure 4.9: ρ_t as a function of noise level. Thresholds for outlier detection were determined by finding the minimum ρ_t satisfying equation (4.45) for the ρ-distribution of Figure 4.8.

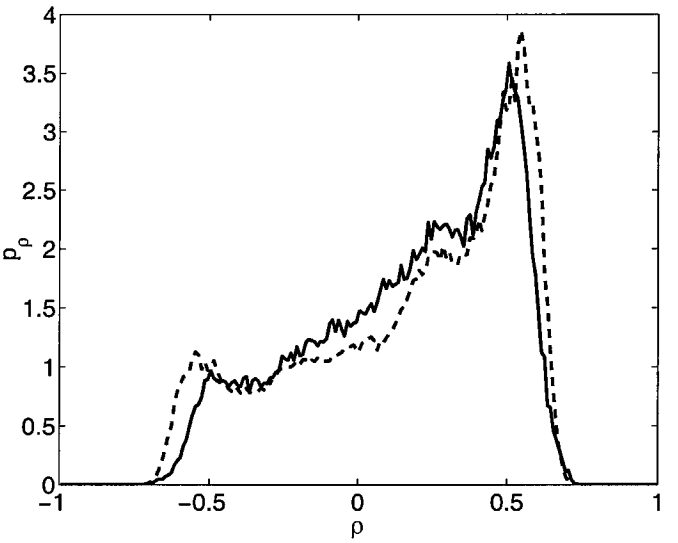


Figure 4.10: Estimation of the level of noise in the image. The solid curve shows the distribution of ρ observed with the targets with 20% noise added. The dashed curve shows the closest approximation of the form (4.39), giving a noise estimate of $\varepsilon = 0.22$.

curve. The closest approximation (shown as a dashed curve) is for the estimate $\varepsilon = 0.22$. The noise estimation was repeated for varying levels of added noise, and the estimates produced by this approach are plotted in Figure 4.11 as a function of the true added noise level. The original targets are estimated to have 4% noise with respect to the learning model, and in general this results in a slight overestimate when ε is small. For large noise values, the estimates become poor. This is not surprising, given that the estimates of ρ from the data become poorer as the noise level increases.

To assess the results of this denoising approach, we compared the results to smoothing with a median filter. Since we have considered image rows independently, only one-dimensional information is used in the ρ-based approach. Accordingly, we have compared our results to those using a 5×1 filter. The results of applying these two techniques to the noisy image in Figure 4.7(b) are shown in Figure 4.12. The one-dimensional constraint manifests itself as horizontal streaks in the images. Although perhaps less visually appealing, the ρ-based approach does a better job of cleaning

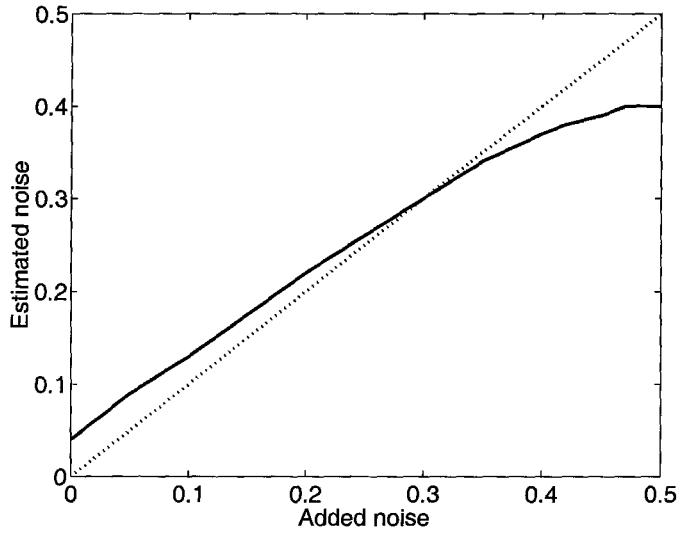


Figure 4.11: Noise estimates as a function of actual noise level. The solid curve shows estimated noise levels for images with noise ranging from 0 to 50%.

up the noise in large contiguous regions of color.[2]

To be of practical interest, it is perhaps foolish to ignore the two-dimensional information in the image, and therefore we also compare to results using a 3×3 filter. The comparative results for all three denoising strategies are shown in Figure 4.13. Following [Banham and Katsaggelos 1997] we measure the noise reduction in terms of improvement in signal-to-noise ratio (ISNR). For target f, noisy target $\tilde{f}$ and restored version g, the ISNR is given by

$$ISNR = 10 \cdot \log_{10} \left(\frac{\sum_{x=1}^{N_X} (f(x) - \tilde{f}(x))^2}{\sum_{x=1}^{N_X} (f(x) - g(x))^2} \right). \tag{4.48}$$

For all noise levels, the two-dimensional filter outperforms the one-dimensional filter. Both median filters outperform the ρ-based approach when the noise level is low. The use of a median filter results in a smoothing of the image, which works well when the noise is sparse. The ρ-based approach uses a prior based on the learning model and continues to perform relatively well for higher noise levels. It outperforms the one-dimensional filter for $\varepsilon \geq 0.3$, and also outperforms the two-dimensional filter

[2]Some large regions that are uniform in two dimensions appear to be well restored, but the algorithm considers each row independently.

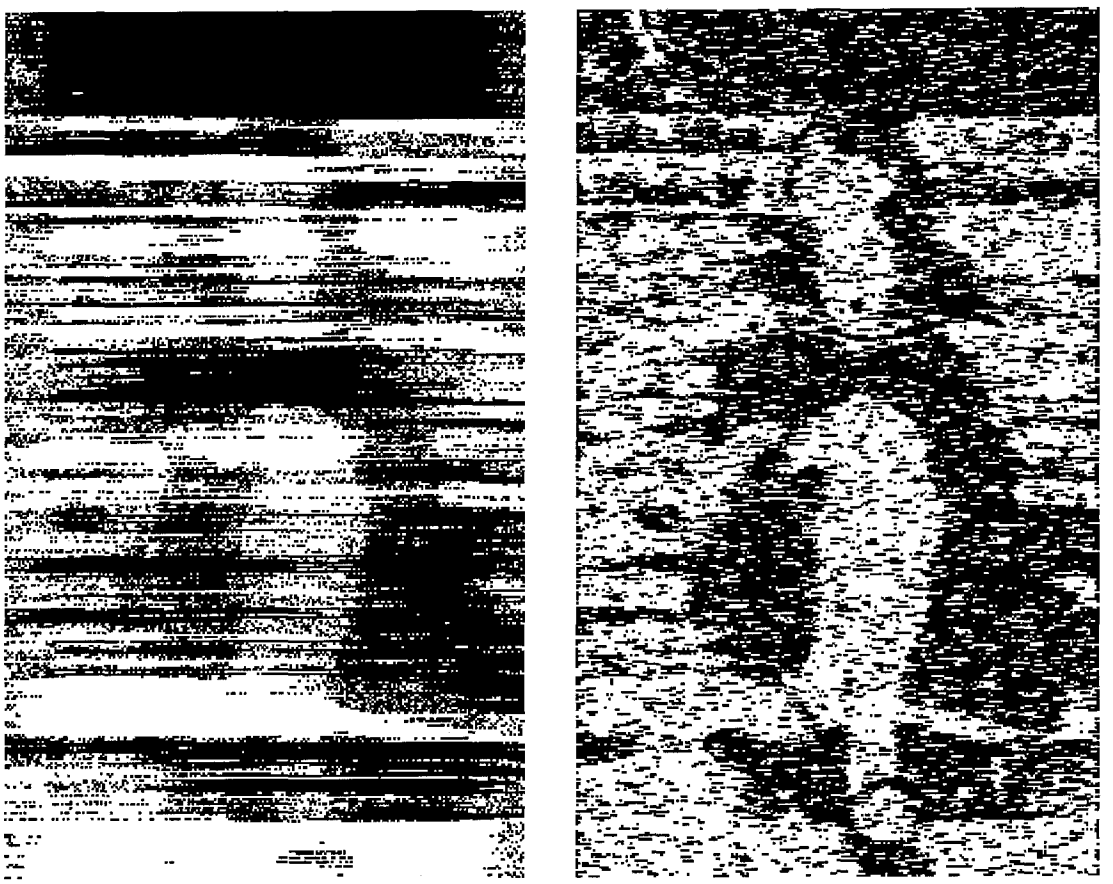


Figure 4.12: Images restored by ρ cleaning and median filtering. The image on the left shows the image with 30% noise after ρ-based denoising. On the right is the same imaged smoothed with a one-dimensional median filter.

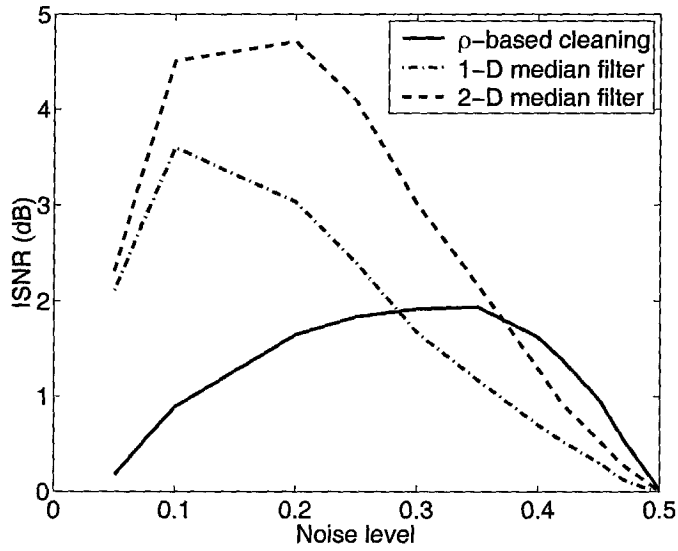


Figure 4.13: SNR improvement for image denoising techniques. The curves show the improvement in the signal-to-noise ratio for image denoising with varying noise levels.

when $\varepsilon \geq 0.4$. Although the peak performance is lower, the ρ-based approach shows much more uniform results, indicating a robustness to noise.

4.6 Linear Models and Financial Data

The noiseless ρ-distribution plays an important role in the noise estimation and correction procedures described in section 4.4 and illustrated in section 4.5. Building on the analysis of section 3.1, we are able to derive an expression for $\rho(x)$ for all x under this model, from which p_ρ can be computed.

4.6.1 ρ-Distributions for Linear Models

We computed the π-distributions for linear classifiers with decision boundaries that pass through the origin in section 3.1. When the input has dimension d,

$$p_\pi(s) = \frac{\mathcal{S}(d-1)}{\mathcal{S}(d)} \pi \sin(\pi s). \tag{4.49}$$

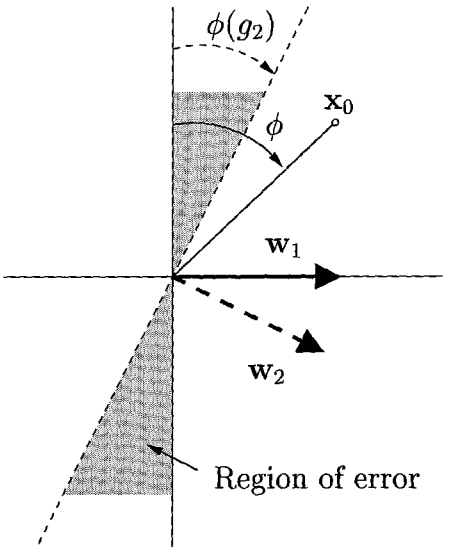


Figure 4.14: Evaluation of $\rho(x)$ in two dimensions. For a hypothesis g_2 with normal at angle $\phi(g_2)$ from $\mathbf{w}_1$, g_1 and g_2 disagree on points in the shaded region.

If we assume that p_G is a good prior over target functions (in this case, since p_G is symmetric this is equivalent to the assumption that $f \in \mathcal{G}$), then we can also compute the expected noiseless ρ-distributions.

For $g_1(x) = \text{sgn}(\mathbf{w}_1 \cdot x)$ chosen from the learning model, we find the distribution of $\rho(x, g_1(x))$. In the computation of ρ we take the expectation over choices of $g_2(x) = \text{sgn}(\mathbf{w}_2 \cdot x)$. We can say w.l.o.g. that $\mathbf{w}_1 = (1, 0, \ldots, 0)$ and by symmetry, it is sufficient to consider $|\mathbf{w}_2| = 1$. In order to compute ρ, we need to find

$$\mathrm{E}_{g_2}[\pi(g_2) e(g_2(x), g_1(x))]. \tag{4.50}$$

We begin by analyzing the two-dimensional case, then generalize to higher dimensions. The scenario is illustrated in Figure 4.14. We fix g_1, then consider its errors on a particular x_0 with all hypotheses g_2. For x_0 at an angle ϕ_x from the decision boundary of g_1,

$$e(g_1(x_0), g_2(x_0)) = \begin{cases} 0 & 0 \leq \Theta(g_2) \leq \phi_x, \ \pi + \phi_x < \phi(g_2) \leq 2\pi \\ 1 & \phi_x < \phi(g_2) \leq \pi + \phi_x \end{cases} \tag{4.51}$$

when $\phi_x \leq \pi$, and $e(g_1(x_0), g_2(x_0)) = e(g_1(-x_0), g_2(-x_0))$ when $\phi_x > \pi$.

Recall from (3.1) that $\pi(g_2) = \phi(\mathbf{w}_2)/\pi$. If we have a spherically symmetric weight distribution, then the distribution of $\phi(\mathbf{w}_2)$ is uniform, and we can compute

$$\mathrm{E}_{g_2}[\pi(g_2)e(g_2(x), g_1(x))] = \frac{1}{2\pi}\int_{\phi_x}^{\pi} \frac{t}{\pi}dt + \frac{1}{2\pi}\int_{\pi}^{\pi+\phi_x} \frac{2\pi - t}{\pi}dt \tag{4.52}$$

$$= \frac{1}{4}\left(2 - \frac{\phi_x^2}{\pi^2} - \left(1 - \frac{\phi_x}{\pi}\right)^2\right). \tag{4.53}$$

Furthermore, in two dimensions the π-distribution is uniform, and hence

$$\mathrm{E}_{g_2}[e(g_1(x), g_2(x))] = \mathrm{E}_{g_2}[\pi(g_2)] = \frac{1}{2} \tag{4.54}$$

and

$$\mathrm{Var}_{g_2}[e(g_1(x), g_2(x))] = \frac{1}{4}, \quad \mathrm{Var}_{g_2}[\pi(g_2)] = \frac{1}{12}. \tag{4.55}$$

Substituting (4.53) into (4.21), we get

$$\rho(\phi_x) = \sqrt{3}\left(1 - \frac{\phi_x^2}{\pi^2} - \left(1 - \frac{\phi_x}{\pi}\right)^2\right). \tag{4.56}$$

If the input distribution is spherically symmetrical, then ϕ_x is also uniformly distributed, giving the cumulative distribution

$$\Pr[\rho < r] = 2\Pr[\phi_x < \pi(1 - \sqrt{1 - 2r/\sqrt{3}})] \tag{4.57}$$

for $0 \leq r \leq \sqrt{3}/2$. The ρ-distribution is therefore given by

$$p_\rho(r) = \begin{cases} \frac{1}{\sqrt{3-2\sqrt{3}r}} & 0 \leq \rho < \frac{\sqrt{3}}{2} \\ 0 & \text{otherwise} \end{cases}. \tag{4.58}$$

We generalize the previous analysis to the multidimensional case. In higher di-

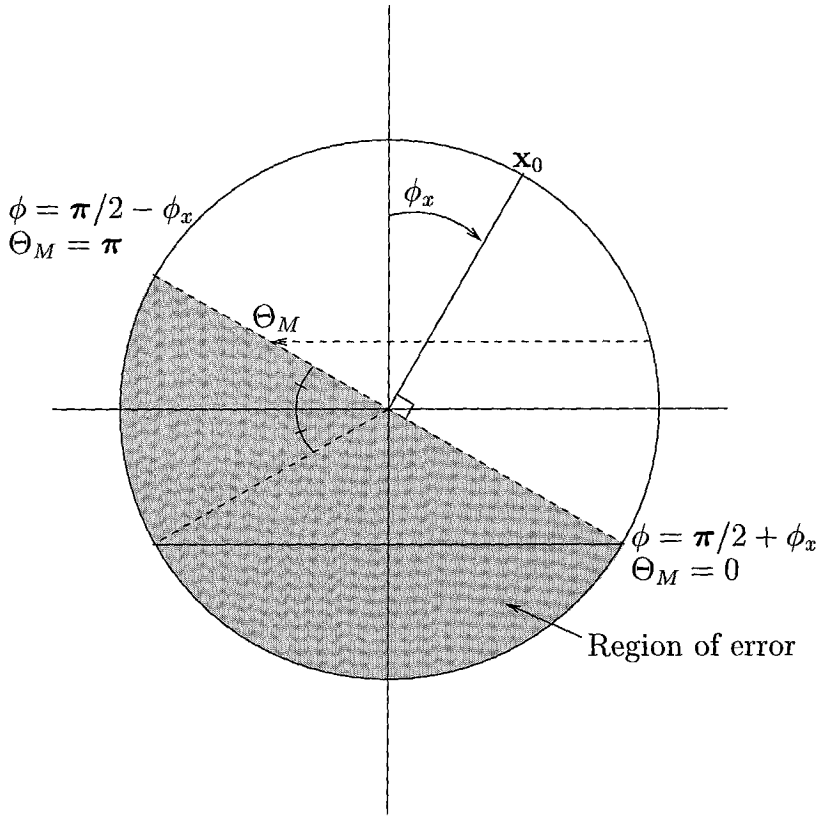


Figure 4.15: Evaluation of $\rho(x)$ in d dimensions. g_1 and g_2 disagree on x_0 in the shaded region, which includes the area with $\Theta_1 > \Theta_M(\phi)$ for $\pi/2 - \phi_x \leq \phi \leq \pi/2 + \phi_x$ and $0 \leq \Theta_1 \leq \pi$ for $\phi > \pi/2 + \phi_x$.

mensions we can still use symmetry arguments to say w.l.o.g. that

$$\mathbf{w}_1 = (1, 0, \ldots) \tag{4.59}$$

$$x_0 = (\cos(\phi_x), \sin(\phi_x), 0, \ldots) \tag{4.60}$$

$$|\mathbf{w}_2| = 1. \tag{4.61}$$

In order to compute the ρ distribution, we need to compute $\mathrm{E}_{g_2}[e(g_1(x), g_2(x))\pi(g_2)]$ for the d-dimensional model. Still restricting ourselves to the surface of unit d-ball, the region in which $e(g_1(x), g_2(x)) \neq 0$ is the hemisphere opposite x. As illustrated in

Figure 4.15, if $x = (\cos\phi_x, \sin\phi_x, 0, \dots)$, then the region of interest is that for which

$$\mathbf{x} \cdot \mathbf{w}_2 < 0 \tag{4.62}$$

$$\cos\phi_x \cos\phi(g_2) + \sin\phi_x \sin\phi(g_2) \cos\Theta_1(g_2) < 0 \tag{4.63}$$

$$\Theta_1(\mathbf{w}_2) < \arccos(-\cot\phi_x \cot\phi(g_2)). \tag{4.64}$$

We denote the minimum Θ_1 for which $e(g_1(\mathbf{x}), g_2(\mathbf{x})) = 1$ by

$$\Theta_M(\phi) = \begin{cases} \arccos(-\cot\phi_x \cot\phi) & \phi \in \left[\frac{\pi}{2} - \phi_x, \frac{\pi}{2} + \phi_x\right] \\ 0 & \phi \in \left[\frac{\pi}{2} + \phi_x, \pi\right] \end{cases}. \tag{4.65}$$

Since $\pi(g_2) = \phi(g_2)/\pi$,

$$\mathrm{E}_{g_2}[e(g_1(\mathbf{x}), g_2(\mathbf{x}))\pi(g_2)] \tag{4.66}$$

$$= \mathcal{S}(d)^{-1} \int_{\phi=\pi/2-\phi_x}^{\pi} \int_{\Theta_1=\Theta_M(\phi)}^{\pi} \cdots \int_{\Theta_{d-2}=0}^{\pi} \frac{\phi}{\pi} \mathbf{J}\, d\Theta_{d-2} \dots d\Theta_1 d\phi \tag{4.67}$$

$$= \frac{\mathcal{S}(d-2)}{\pi\mathcal{S}(d)} \int_{\pi/2-\phi_x}^{\pi/2+\phi_x} \phi \sin^{(d-2)}\phi \int_{\Theta_M(\phi)}^{\pi} \sin^{(d-3)}\Theta_1\, d\Theta_1 d\phi$$

$$+ \frac{\mathcal{S}(d-1)}{\pi\mathcal{S}(d)} \int_{\pi/2+\phi_x}^{\pi} \phi \sin^{(d-2)}\phi\, d\phi, \tag{4.68}$$

which is greatest for $\phi_x = 0$, monotonically decreasing in ϕ_x, and has minimum

$$\frac{\mathcal{S}(d-1)}{\pi\mathcal{S}(d)} \int_0^{\pi} \phi \sin^{(d-2)}\phi\, d\phi = \frac{1}{4} \tag{4.69}$$

when $\phi_x = \pi/2$ for every d.

Since these models are symmetric,

$$\mathrm{E}_g[e(g_1(x), g(x))] = \frac{1}{2}, \quad \mathrm{Var}_g[e(g_1(x), g(x))] = \frac{1}{4}, \tag{4.70}$$

and hence

$$\rho(x) = \frac{2(\mathrm{E}_{g_2}[e(g_1(x), g_2(x))\pi(g_2)] - \frac{1}{4})}{\sqrt{\mathrm{Var}_g[\pi(g)]}}. \tag{4.71}$$

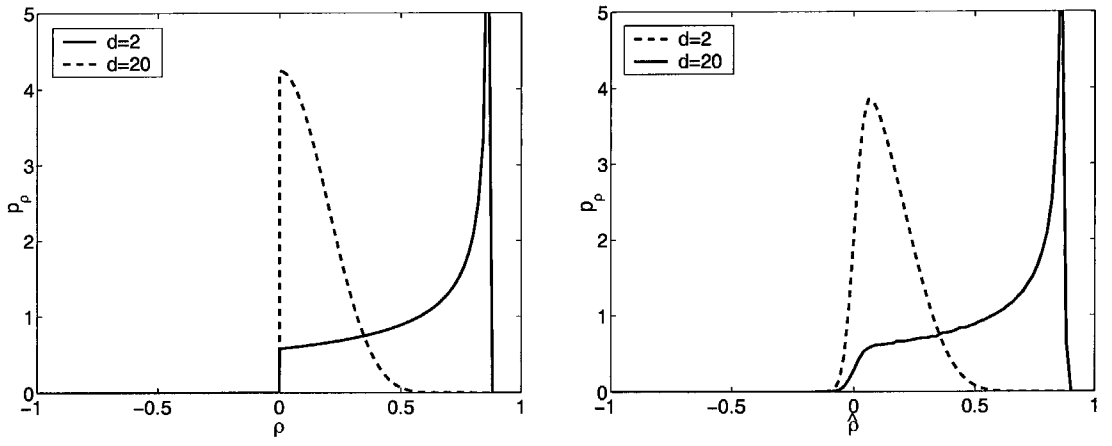


Figure 4.16: ρ-distributions for a linear model. (a) shows the expected ρ-distribution for linear targets and $d = 2, 20$. No points are expected to have $\rho(x) < 0$. Empirical measurements of the distribution of $\widehat{\rho}$ are illustrated in (b) for $d = 2, 20$ with random linear targets and for 10000 training examples

We can evaluate $\mathrm{Var}_g[\pi(g)]$ given (4.49) and can compute $\rho(x)$ by substituting (4.68) into (4.71). Since $\mathrm{E}_{g_2}[e(g_1(x), g_2(x))\pi(g_2)] \geq 1/4$ for all x and all d, every point has $\rho(x) > 0$ when there is no noise. Thus, every point can be considered a "good" data point, and the appropriate cutoff ρ_t for rejecting noisy data will be 0 for every positive noise level.

The cumulative distribution of ϕ_x is given by the surface of the spherical cap (compare with (3.3))

$$Pr[\phi_x < t] = \frac{\mathcal{S}(d-1)}{\mathcal{S}(d)} \int_{\phi=0}^{t} \sin^{d-2} \phi \; d\phi. \tag{4.72}$$

Using this distribution, we can numerically invert (4.71) to compute the distribution for ρ. We have carried this procedure through to yield the distribution for ρ with $d = 20$ shown in Figure 4.16(a). Also shown is p_ρ for $d = 2$ from (4.58). Figure 4.16(b) shows histograms of empirical estimates $\widehat{\rho}$ for linear targets and models based on 10000 training points.

We have demonstrated how it is possible to compute the expected ρ-distribution for simple linear models in arbitrary dimensions. Linear regression has been widely

studied, and we do not concern ourselves with extracting a linear target from noiseless data. Rather, we will motivate the preceding derivation with practical problem for which the input distribution is (approximately) spherically symmetric and for which the extraction of any information from very noisy data can be considered a success.

4.6.2 Financial Time Series

In analysis of financial time series, it is common to assume a lognormal random walk for asset price S

$$dS = \mu S dt + \sigma S dW \tag{4.73}$$

where μ and σ are constants and dW is a Wiener process (see, for example, [Wilmott *et al.* 1995]). As a result, the logarithms of proportional successive price changes are normally distributed.

We look at foreign exchange rates between the U.S. dollar and the German Mark. For this particular time series, we can make use of a symmetry hint [Abu-Mostafa 1995] to augment any available data set, ensuring that x has zero mean and that the classifications are balanced, that is, $\Pr[f(x) = 0] = \Pr[f(x) = 1] = \frac{1}{2}$. For this series, then, we can assume that sequential log price changes are i.i.d. normal random variables

$$\delta_t \sim \mathcal{N}(0, \sigma^2). \tag{4.74}$$

Then if we use a d-step price history $x_i = (\delta_{t-1}, \delta_{t-2}, \ldots, \delta_{t-d})$ as the input, the inputs can be considered multivariate normal random variables

$$x_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 I_d) \tag{4.75}$$

where $\mathbf{0}$ is the zero vector and I_d is the $d \times d$ identity matrix.

We assume that the target function is a thresholded linear function of the data

$$f(x) = \text{sgn}(\mathbf{w}_f \cdot x + \theta). \tag{4.76}$$

The symmetry hint cited above also implies that $\theta = 0$. We therefore choose a learning model containing hypotheses of the form $g_{\mathbf{w}} = \text{sgn}(\mathbf{w} \cdot x)$ with $|\mathbf{w}| = 1$. Of course, such a simple relation does not exist for real foreign exchange rates. For the sake of argument, however, we can attribute any deviation of the data from what is expected from (4.76) to noise in the data. The noise will be overwhelming, but using the techniques of section 4.4 we might hope to extract some sort of salient relationship from high dimensional data.

When the hypothesis prior distribution p_G is such that $\mathbf{w} \sim \mathcal{N}(0, \sigma_w^2)$, this learning problem fits into the model analyzed in sections 3.1 and 4.6.1. We are therefore able to compute the noiseless π- and ρ-distributions for this learning problem exactly.

4.6.3 Experiments

From historical quotes of the U.S. dollar/German Mark exchange rate we computed sequential log price changes, constructed inputs of 20-tick histories, and applied the symmetry hint to produce data that can be expected to be distributed as (4.75). The output classification was determined by the direction of the price movement over the following 5 minutes, being assigned 0 if the price 5 minutes in the future was lower and 1 otherwise.

From this data, we selected 10000 examples for the training set. We computed $\widehat{\rho}(x)$ for each input based on the training data. The distribution of the observed values of $\widehat{\rho}$ is shown in Figure 4.17. Estimates of the noise level were obtained using (4.39). Using the prior ρ-distribution illustrated in Figure 4.16(a) and using the experimental $\widehat{\rho}$-distribution of Figure 4.16(b) resulted in noise estimates of $\varepsilon = 0.499$ and $\varepsilon = 0.496$ respectively.

Thus, considering the exchange rate prediction to be a noisy linear function implies that the "noise" almost completely obscures the signal. As we might expect, the data

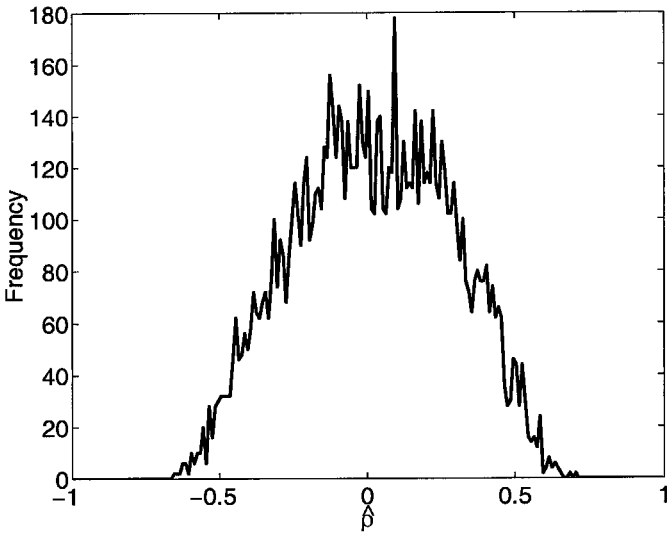


Figure 4.17: Empirical distribution of $\widehat{\rho}$ for foreign exchange data. The histogram of $\widehat{\rho}$ values for 10000 examples from the USD/DEM foreign exchange data is shown.

seems to be extremely noisy (with respect to our simple model), so, although we can compute the π-distribution and generalization curve, exhaustive learning is essentially hopeless. We would expect that extraction of any information from the data will be very difficult. For predicting financial time series, though, any level of performance that is statistically significantly better than random can be very lucrative.

We trained a two-layer neural network with 20 hidden units with tanh activation functions on the training set. The network was trained for 5000 epochs of gradient descent on squared error using sequential back-propagation [Rumelhart *et al.* 1986], with the goal of fitting the $\{0, 1\}$ classifications with a real value in $[0, 1]$. The resulting network was used to classify a test set of 10000 examples, giving a classification of $\text{sgn}(y - 1/2)$ for the real network output y. The resulting training and out-of-sample errors are shown in Table 4.1. Average and extreme cases are shown for 10 runs, corresponding to different random weight initializations for the neural network. The results fall in a very narrow range, and the training error never drops far below 0.25, which would be observed with random guessing.

Selecting data based on $\rho_t = 0$ results in the rejection of 4788 of the 10000 training examples. A neural network of the same architecture trained on the remaining 5212

	All Data	ρ Selected	Randomly Selected
Examples	10000	5212	5212
Training Error:			
Minimum	0.2443	0.0060	0.2400
Average	0.2456	0.0068	0.2412
Maximum	0.2467	0.0084	0.2424
Out-Of-Sample Error:			
Minimum	0.4803	0.4802	0.4836
Average	0.4880	0.4812	0.4878
+/−	0.0018	0.0003	0.0011
Maximum	0.4981	0.4832	0.4938

Table 4.1: Comparison of errors for ρ-based data selection.

examples was consistently able to reach a training error less than 0.01. Although the best observed performance was almost identical, the average performance on the truncated data set dropped to approximately 48.12%.

To ensure that these results do not just reflect smaller training set size, we also trained the neural network on a data set of 5212 examples randomly selected from the available data. While the training error was noticeably lower with this set than with the full data set, it did not go below 0.24. The mean out-of-sample error was slightly lower than that for the full training set, but the difference was not statistically significant. The error bars on these estimates of the mean are shown in the table row labelled '+/−.'

The use of linear classifiers to model foreign exchange rate movements seems like a futile task. In fact, our estimate of the noise under this model is above 49%. Nevertheless, data set selection based on ρ valuations resulted in training sets that could be fit more closely by a neural network model, and using this model resulted in a statistically significant improvement in the out-of-sample classification error.

4.7 Discussion

In this chapter, we introduced a procedure for evaluating training data. The main new idea is that training points have *a priori* different values in terms of selecting a

hypothesis that generalizes well. Training data selection based on empirical estimates of $\rho(x)$ was shown to yield training sets that improved generalization.

An analysis of the effects of noise led to procedures for noise estimation and outlier detection. These techniques were applied to image denoising and proved to be fairly robust to noise, with a one-dimensional ρ-based method outperforming one- and two-dimensional filters at high noise levels.

We demonstrated how the expected distribution of ρ could be computed for a simple linear model of arbitrary input dimension. Under this model we found that no inputs had $\rho < 0$ in the absence of noise. We demonstrated that a financial time series prediction task could be studied with this model, but that the effective level of noise was very close to 50%. Nevertheless, data selection based on ρ valuations estimated with the linear model resulted in generalization improvements for this task.

In the various experiments in this chapter, the learning model played several different roles. From the introduction of ρ, we might expect ρ valuation to be useful only for exhaustive learning systems. In section 4.3.2, however, we used ρ-selected data to obtain improved generalization after training. In section 4.6.3 we evaluated examples based on a linear learning model and showed a generalization improvement training neural networks with the selected data. In section 4.5 we never selected hypotheses from the learning model to approximate the targets. Instead, the learning model was only used to provide a ρ valuation for each example, and the classification was done based on ρ. For the image denoising problem, examples were provided for every input, so the reclassification was considered to be noise correction. In the absence of some data, however, the same approach could have provided classifications for unlabelled inputs. In the next chapter, we will extend this last idea to obtain a novel learning process.

Chapter 5

ρ Learning

The data valuation technique described in Chapter 4 has immediate application as a practical learning paradigm. New inputs can be assigned a classification without explicit selection of a hypothesis function.

5.1 The ρ Learning Idea

The idea behind ρ learning is quite simple. For any input value x we can consider the two possible valuations $\rho(x|0)$ and $\rho(x|1)$ independent of a particular concept of noise. By (4.35), if $\rho(x|0) = r_0$, then $\rho(x|1) = -r_0$. For p_ρ in $\mathcal{P}$ (definition 4.4.1), $p_\rho(r) > p_\rho(-r)$ for all $r > 0$, and hence a correctly classified example is more likely to have $\rho > 0$. In fact, we showed that, for the linear model of section 4.6.1, no point has $\rho(x|f(x)) < 0$. ρ learning relies on the assumption that this concept can be extended to an arbitrary learning problem, that is, that given an appropriate learning model (depending on the target function), $\rho(x|f(x)) > 0$ for every x.

Given an unlabelled input to be classified, we can consider $\widehat{\rho}(x|1)$ and $\widehat{\rho}(x|0)$ and choose the classification which gives a positive value. Based on the results of section 4.3.1 and (4.38), we can expect that given enough data, our estimate $\widehat{\rho}(x|1)$ will have the same sign as the true $\rho(x|1)$ with high probability, and hence, by the ρ learning assumption, that classifying x as $\text{sgn}(\widehat{\rho}(x|1))$ will result in the correct classification. If we denote by $R(x)$ the classification of x produced by ρ learning, then the possible scenarios are shown in Table 5.1.

	$f(x) = 0$	$f(x) = 1$
$\rho(x\|0)$	$+$	$-$
$\rho(x\|1)$	$-$	$+$
$\widehat{\rho}(x\|1)$	$-$	$+$
$R(x)$	0	1

Table 5.1: Possible scenarios for ρ learning. $+$ indicates a positive value, $-$ indicates a negative value. With high probability $\widehat{\rho}$ will match ρ. Assuming $\rho(x|f(x)) > 0$, $R(x) = \text{sgn}(\widehat{\rho}(x|1)) = f(x)$.

The ρ valuation is defined in terms of a learning model, but we do not select a hypothesis from the learning model as a guess for the target. Based on a statistic dependent on the whole learning model (or in practice a large sample of hypotheses), a classification $R(x)$ is produced for each input. In this way, ρ learning is not a learning system as described in section 1.1, and is akin to ensemble or aggregate methods [Dietterich 2000]. The resulting function $R(x)$ is implicitly dependent on the learning model, but is not, in general, equivalent to some $g \in \mathcal{G}$.

5.2 Edge Detection Example

We illustrate ρ learning and compare it to an alternative learning algorithm for a simple edge detection task. In one dimension, we can consider an 'edge' to be a step function $g_\alpha(x) = \text{sgn}(x - \alpha)$ or $g_\alpha(x) = \text{sgn}(\alpha - x)$.

As in section 4.2 we we take p_X to be uniform and we use the learning model of step functions

$$\mathcal{G} = \{\text{sgn}(x - \alpha)|\alpha \in [0, 1]\} \cup \{\text{sgn}(\alpha - x)|\alpha \in [0, 1]\} \tag{5.1}$$

with uniform prior $p_G(\text{sgn}(x - \alpha)) = p_G(\text{sgn}(\alpha - x)) = \frac{1}{2}$ for all α.

We choose a target function randomly from the learning model, and generate a data set of N examples with uniform noise level ε. We consider ρ learning in comparison to the learning algorithm that selects the step function that most closely matches the training data. Since there is ambiguity in this choice, we resolve it as

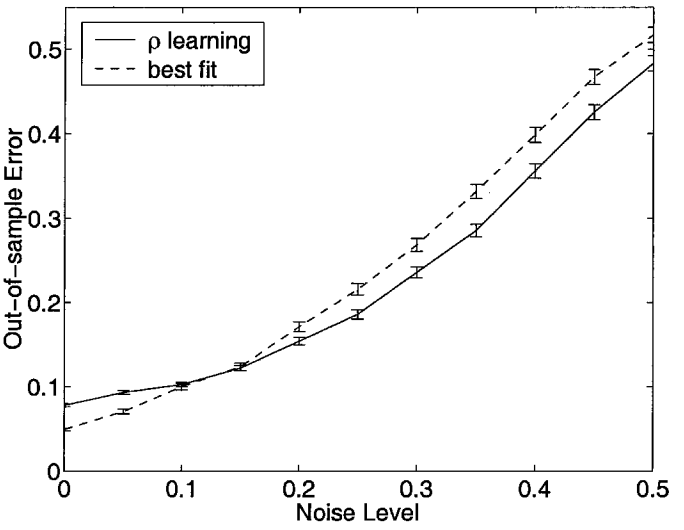


Figure 5.1: Performance comparison for an edge detection task. Noiseless out of sample error rates are shown, averaged over 1000 noisy data sets of $N = 10$ examples.

follows. If the best step function has a threshold that must lie between x_t and x_{t+1}, then we use $(x_t + x_{t+1})/2$ as the threshold. If the best step function classifies all of the examples as 0 (or 1), then we use $g(x) \equiv 0$ (or $g(x) \equiv 1$) as the hypothesis.

Figure 5.1 compares error rates for the two different learning approaches. For these experiments we used only 10 training examples. ρ learning for this problem is slightly worse for low noise levels, but performs noticeably better for $\varepsilon \geq 0.2$. In this example, the target function is in the learning model, and our benchmark is the algorithm that does an exhaustive search for a hypothesis that minimizes the in-sample error. Remarkably, this does not perform as well as ρ learning for noisy data sets.

5.3 Financial Time Series Prediction

The main advantage of the applications of ρ valuation discussed so far is an apparent robustness to noise. The example of the previous section indicates that this appears to be true for the ρ learning approach as well. We investigate the use of ρ learning for financial time series prediction, since, as we saw in section 4.6.3, these data sets can

be extremely noisy. We again look at the U.S. Dollar/German Mark foreign exchange rate prediction task.

On the same data set as in section 4.6.3, we used the ρ learning idea to classify points in the test set. For each new input x, the value of $\widehat{\rho}(x|1)$ was computed using the 10000 examples in the training set and x was classified as $R(x) = \text{sgn}(\widehat{\rho}(x|1))$. The resulting classification made 4810 errors on the 10000 test points. We can be confident that this performance is not due to pure luck, since the probability that random guessing results in an error that deviates this far from the mean is approximately 1.5×10^{-4}. This error rate is not as low as the best result for the neural networks trained in section 4.6.3, but is lower than the average error rate for any of the training sets considered there.

5.4 Discussion

The data valuation developed in Chapter 4 was intended to help select a good hypothesis from a learning model. The noise detection results, however, showed that this valuation could be used to determine the correct classification for individual points. In this chapter we use the learning model for data valuation, and use the valuation for classification. The hypothesis selection step is eliminated, and so no learning algorithm is required.

We illustrated the ρ learning approach with a simple example and a practical task, and found it to be competitive with (and in some cases better than) more conventional learning systems.

Chapter 6

Conclusion

In this thesis, we addressed the general learning problem from a probabilistic perspective, with the goal of describing and improving generalization performance. The the theoretical bin model framework was introduced in Chapter 2 and serves as the foundation for the subsequent results.

We showed in section 1.2 that generalization cannot be expected without any assumptions about the learning task at hand. An immediate result of the bin model analysis is the characterization of a learning problem in terms of its π-distribution. Given a π-distribution, we can describe the generalization behavior fully, without explicitly relying on a particular form of target or learning model.

In Chapter 3, we showed how the practical obstacles facing the bin model analysis could be (at least partially) overcome. A two-stage learning process that isolates the learning algorithm from error estimation allows us to apply the bin model results to more sophisticated learning systems. By estimating the π-distribution from the data, we can adjust error estimates on a validation set to be more in line with the true out-of-sample error.

The implications of fixed training sets for the preceding results led to a concept of data valuation described by the error correlation $\rho(x)$. The ideas that individual examples have a priori values with respect to generalization performance and that it may be appropriate to discard even noiseless data are novel in this study, as far as we know. An estimate of the ρ distribution, which can be obtained given a prior distribution over targets, allows us to estimate the level of noise in a data set. We

are then further able to construct good training sets by exclusion or reclassification of selected examples.

Reclassification based on ρ can be viewed as error correction, and was demonstrated to be useful for data denoising. An extension of this idea yields a new approach to learning. The new ρ learning concept introduced in Chapter 5 directly produces a classification rule given a learning model and data set, thus assuming the role of both learning model and hypothesis function.

These results, like any in machine learning, require certain assumptions about the problem at hand. As a result, there will be scenarios in which these techniques do not apply, and, in fact, the motivation for Chapters 3 and 4 was an investigation of exactly such situations for the preceding results. We have included experiments and practical examples throughout the thesis demonstrating the applicability of the major results to real learning problems, and hopefully convincing the reader of the potential for better generalization through their use.

Bibliography

[Abu-Mostafa 1989] Y. S. Abu-Mostafa. The Vapnik-Chervonenkis dimension: Information versus complexity in learning. *Neural Computation*, 1:312–317, 1989.

[Abu-Mostafa 1995] Y. S. Abu-Mostafa. Financial market applications of learning from hints. In A. Refenes, ed., *Neural Networks in the Capital Markets*, pp. 278–288. Wiley, 1995.

[Abu-Mostafa and Song 1996] Y. S. Abu-Mostafa and X. Song. Bin model for neural networks. In *Proceedings of ICONIP'96*, pp. 169–173. Hong Kong, 1996.

[Akaike 1970] H. Akaike. Statistical predictor identification. *Annals of the Institute of Statistical Mathematics*, 22:203, 1970.

[Angluin 1987] D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

[Banham and Katsaggelos 1997] M. R. Banham and A. K. Katsaggelos. Digital image restoration. *IEEE Signal Processing Magazine*, 14(2):24–41, 1997.

[Bishop 1995] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[Blake and Merz 1998] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[Blumer *et al.* 1987] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24(6):377–380, 1987.

[Cohn *et al.* 1995] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. In G. Tesauro, D. Touretzky, and T. Leen, eds., *Ad-*

vances in Neural Information Processing Systems, vol. 7, pp. 705–712. MIT Press, Cambridge, MA, 1995.

[Cover and Thomas 1991] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. Wiley, 1991.

[Devroye *et al.* 1996] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Aplications of Mathematics. Springer, New York, 1996.

[Dietterich 2000] T. G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, eds., *First International Workshop on Multiple Classifier Systems*, Lecture Notes in Computer Science, pp. 1–15. Springer Verlag, New York, 2000.

[Efron and Tibshirani 1993] B. Efron and R. J. Tibshirani. *An introduction to the bootstrap*. Chapman & Hall, New York, 1993.

[Haykin 1994] S. Haykin. *Neural Networks*. Macmillan College Publishing Company, New York, 1994.

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

[Kearns and Vazirani 1994] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, Cambridge, Massachusetts, 1994.

[Kirkpatrick *et al.* 1983] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[Lee *et al.* 1995] W. S. Lee, P. L. Bartlett, and R. C. Williamson. Lower bounds on the VC-dimension of smoothly parametrized function classes. *Neural Computation*, 7(5):1040–1053, 1995.

[Lee *et al.* 1997] W. S. Lee, P. L. Bartlett, and R. C. Williamson. Lower bounds on VC-dimension of smoothly parameterized function classes. *Neural Computation*, 9(4):765–769, 1997.

[Magdon-Ismail 2000] M. Magdon-Ismail. No free lunch for noise prediction. *Neural Computation*, 12(3):547–564, 2000.

[Moody 1992] J. E. Moody. The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, eds., *Advances in Neural Information Processing Systems 4*, vol. 4, pp. 847–854. Morgan Kaufmann, 1992.

[Murata and Amari 1999] N. Murata and S. Amari. Statistical analysis of learning dynamics. *Signal Processing*, 74(1):3–28, 1999.

[Nicholson 2000] A. Nicholson. A generalization model and learning in hardware. CSTR 2000.007, Caltech, 2000.

[Parrondo and Van den Broeck 1993] J. M. R. Parrondo and C. Van den Broeck. Vapnik-Chervonenkis bounds for generalization. *Journal of Physics A: Mathematical and General*, 26:2211–2233, 1993.

[Rissanen 1978] J. Rissanen. Modelling by shortest data description. *Automatica*, 14:465–471, 1978.

[Rosenblatt 1962] F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, Washington, DC, 1962.

[Rumelhart *et al.* 1986] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McCleland, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, pp. 318–362. MIT Press, Cambridge, MA, 1986.

[Schwartz *et al.* 1990] D. B. Schwartz, V. K. Samalam, S. A. Solla, and J. S. Denker. Exhaustive learning. *Neural Computation*, 2(2):374–385, 1990.

[Sigillito *et al.* 1989] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10(3):262–266, 1989.

[Song 1999] X. Song. *Contextual Pattern Recognition with Applications to Biomedical Image Identification*. Ph.D. thesis, California Institute of Technology, 1999.

[Sutton and Barto 1998] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, Massachusetts, 1998.

[Valiant 1984] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[Vapnik 1995] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer–Verlag, New York, 1995.

[Vapnik 1998] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc., New York, 1998.

[Vapnik and Chervonenkis 1971] V. N. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

[White 1989] H. White. Learning in artificial neural networks: A statistical perspective. *Neural Computation*, 1:425–464, 1989.

[White 1992] H. White. *Artificial Neural Networks*. Blackwell, Cambridge, Massachusetts, 1992.

[Wilmott *et al.* 1995] P. Wilmott, S. Howison, and J. Dewynne. *The Mathematics of Financial Derivatives*. Cambridge University Press, Cambridge, 1995.

[Wolpert 1996a] D. H. Wolpert. The existence of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1391–1420, 1996.

[Wolpert 1996b] D. H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, 1996.

[Wolpert and Lapedes 1992] D. H. Wolpert and A. Lapedes. An investigation of exhaustive learning. Tech. Rep. 92-04-20, Santa Fe Institute, 1992.