CHAPTER 1

INTRODUCTION

Neural networks were originally motivated by an interest in modelling the organic brain (McCulloch and Pitts, 1943; Hebb, 1949). They consist of independent processors that return a very simple function of their total input. In turn, their outputs form the inputs to other processing units. "Connectionism" was an early and revealing name for this work as the capabilities of the brain were felt to lie in the connections of neurons rather than in the capabilities of the individual neurons. Despite many debates over the years about the biological plausibility of various models, this is still the prevailing paradigm in neuro-science.

Important sources for the history of "connectionism" are McCulloch and Pitts (1943), Hebb (1949), Rosenblatt (1962), Minsky and Papert (1969), and Rumelhart et al. (1986). Anderson and Rosenfeld (1988) reproduce many of the historic papers in one volume and Widrow and Lehr (1990) give a history of the development.

However, the modern area of neural networks has fragmented somewhat and there is no attempt at biological plausibility in the artificial neural networks that are used for such tasks as grading olive oil (Goodacre et al., 1992), interpreting sonar signals (Gorman and Sejnowski, 1988) or inferring surface temperatures and water vapor content from remotely sensed data (Aires et al., 2004).

This book, and artificial neural networks in general, sit somewhere in a shared space between the disciplines of Statistics and Machine Learning (ML), which is in turn a cognate discipline of Artificial Intelligence. Table 1.1 summarizes some of the correspondences between the discipline concerns of Machine Learning and Statistics.

A Statistical Approach to Neural Networks for Pattern Recognition by Robert A. Dunne

Table 1.1 Some correspondences between the discipline concerns of Machine Learning and Statistics.

machine learning		statistics
supervised learning	"learning with a teacher"	classification
unsupervised learning	"learning without a teacher"	clustering

Friedman (1991b) carries the teacher analogy further and suggests that a useful distinction between ML and statistics is that statistics takes into account the fact that the "teacher makes mistakes." This is a very accurate and useful comment. A major strand of this work is understanding what happens when "the teacher makes a mistake" and allowing for it (this comes under the heading of "robustness" in statistics).

Clearly one of the differences between ML and statistics is the terminology, which of course is a function of the history of the disciplines. This is exacerbated by the fact that in some cases statistics has its own terminology that differs from the one standard in mathematics. Another easily spotted difference is that ML has better names for its activities[1].

More substantial differences are that:

- machine learning tends to have a emphasis on simple, fast heuristics. It has this aspect in common with data mining and artificial intelligence.
- following on from the first point, whereas statistics tends to start with a model for the data, often there is no real data model (or only a trivial one) in machine learning

Breiman in his article "Statistical modelling: The two cultures" (2001) talks about the divergence in practice between Statistics and Machine Learning and their quite different philosophies. Statistics is a "data modeling culture," where a function $f : x \rightarrow y$ is modeled in the presence of noise. Both linear and generalized linear models fall within this culture. However, Machine Learning is termed by Breiman an "algorithmic modeling culture." Within this culture, the function f is considered both unknown and unknowable. The aim is simply to predict a y value from a given x value.

Breiman argues that in recent years the most exciting developments have come from the ML community rather than the statistical community. Among these developments one could include:

- decision trees (Morgan and Sonquist, 1963);
- neural networks, in particular perceptron and multi-layer perceptron (MLP) models;
- support vector machines (and statistical learning theory) (Vapnik, 1995; Burges, 1998);
- boosting (Freund and Schapire, 1996).

[1] I believe that this comment was made in the lecture for which Friedman (1991b) are the notes.

Breiman in what, from the discussion[2], appears to have been received as quite a provocative paper, cautions against ignoring this work and the problem areas that gave rise to it.

While agreeing with Breiman about the significance of working with the algorithmic modeling culture we would make one point in favor of the statistical discipline. While these methodologies have been developed within the machine learning community, the contribution of the statistical community to a full understanding of these methodologies has been paramount. For example:

- decision trees were put on a firm foundation by Breiman et al. (1984). They clarified the desirability of growing a large tree and then pruning it as well as a number of other questions concerning the splitting criteria;

- neural networks were largely clarified by Cheng and Titterington (1994); Kraanowski and Marriott (1994); Bishop (1995a) and Ripley (1996) amongst others. The exaggerated claims made for MLP models prior to the intervention of statisticians no longer appear in the literature;

- boosting was demystified by Friedman et al. (1998) and Friedman (1999, 2000);

- support vector machines have yet to be widely investigated by statisticians, although Breiman (2001) and Hastie et al. (2001) have done a lot to explain the workings of the algorithm.

Brad Efron, in discussing Breiman's paper, suggests that it appears to be an argument for "black boxes with lots of knobs to twiddle." This is a common statistical criticism of ML. It arises, not so much from the number of knobs on the black box, which is often comparable to the number of knobs in a statistical model[3], but from the lack of a data model.

When we have a data model, it gives confidence in the statistical work. The data model arises from an understanding of the process generating the data and in turn assures us that we have done the job when the model is fitted to the data. There are then generally some diagnostic procedures that can be applied, such as examining the residuals. The expectation is that, as the data model matches the process generating the data, the residuals left over after fitting the model will be random with an appropriate distribution. Should these prove to have a pattern, then the modelling exercise may be said to have failed (or to be as good as we can do). In the absence of a data model, there seems little to prevent the process being reduced to an ad-hock empiricism with no termination criteria.

However the ML community is frequently working in areas where no plausible data model suggests itself, due to our lack of knowledge of the generating mechanisms.

[2]As Breiman (2001) was the leading paper in that issue of *Statistical Science*, it was published with comments from several eminent statisticians and a rejoinder from Leo Breiman.

[3]In some instances the number of "knobs" may be fewer for ML algorithms than for comparable statistical models. See Breiman's rejoinder to the discussion.

1.1 THE PERCEPTRON

We will start with Rosenblatt's perceptron learning algorithm as the foundation of the area of neural networks. Consider a set of data as shown in Figure 1.1. This is a 2-dimensional data set in that 2 variables, x_1 and x_2, have been measured for each observation. Each observation is a member of one of two mutually exclusive classes labelled "×" and "+" in the figure.

To apply the perceptron algorithm we need to have a numeric code for each of the classes. We use

$$y = \begin{cases} 1 & \text{for class } \times \text{ and} \\ -1 & \text{for class } +. \end{cases}$$

The model then consists of a function f, called an "activation function," such that:

$$f = \begin{cases} 1 & \omega_0 + \omega^T x \geq 0 \\ -1 & \text{otherwise.} \end{cases}$$

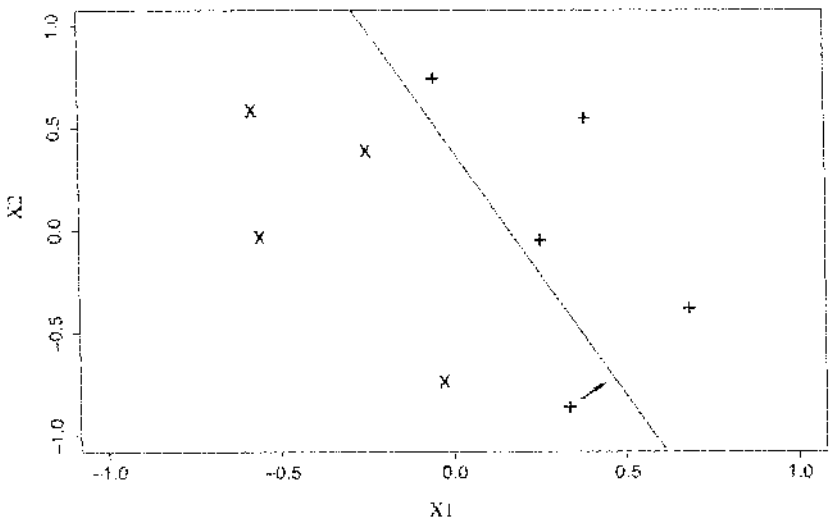

Figure 1.1 *A 2-dimensional data set consisting of points in two classes, labelled "×" and "+". A perceptron decision boundary $\omega_0 + \omega_1 x_1 + \omega_2 x_2$ is also shown. One point is misclassified, that is, it is on the wrong side of the decision boundary. The margin of its misclassification is indicated by an arrow. On the next iteration of the perceptron fitting algorithm (1.1) the decision boundary will move to correct the classification of that point. Whether it changes the classification in one iteration depends on the value of η, the step size parameter.*

The perceptron learning algorithm tries to minimize the distance of a misclassified point to the straight line $\omega_0 + \omega_1 x_1 + \omega_2 x_2$. This line forms the "decision boundary" in that points are classified as belonging to class "×" or "+" depending

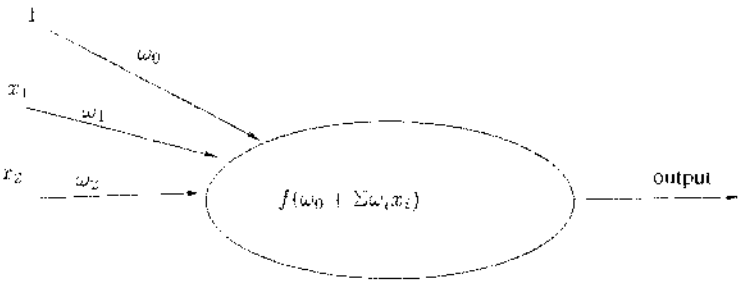

Figure 1.2 The perceptron learning model can be represented as a processing node that receives a number of inputs, forms their weighted sum, and gives an output that is a function of this sum.

on which side of the line they are on. The misclassification rate is the proportion of observations that are misclassified, so in Figure 1.1 it is 1/9. Two classes that can be separated with 0 misclassification error are termed "linearly separable."

The algorithm uses a cyclic procedure to adjust the estimates of the ω parameters. Each point x is visited in turn and the ωs are updated by

$$\omega_i \leftarrow \omega_i + \eta[y - f(\omega_0 + \omega^T x)]x. \tag{1.1}$$

This means that only incorrectly classified points move the decision boundary. The η term has to be set in advance and determines the step size. This is generally set to a small value in order to try to prevent overshooting the mark.

Where the classes are linearly separable it can be shown that the algorithm converges to a separating hyperplane in a finite number of steps. Where the data are not linearly separable, the algorithm will not converge and will eventually cycle through the same values. If the period of the cycle is large this may be hard to detect.

Where then is the connection with brains and neurons? It lies in the fact that the *algorithm* can be represented in the form shown in Figure 1.2 where a processing node (the "neuron") receives a number of weighted inputs, forms their sum, and gives an output that is a function of this sum.

Interest in the perceptron as a computational model flagged when Minsky and Papert (1969) showed that it was not capable of learning some simple functions. Consider two logical variables A and B that take values in the set {TRUE, FALSE}. Now consider the truth values of the logical functions AND, OR, and XOR (eXclusive OR, which is true if and only if one of its arguments is true) as shown in Table 1.2.

We can recast the problem of learning a logical function as a geometric problem by encoding {TRUE, FALSE} as $\{1, 0\}$. Now for the XOR function, in order to get a 0 classification error the perceptron would have to put the points $\{1, 1\}$ and $\{0, 0\}$ on one side of a line and $\{1, 0\}$ and $\{0, 1\}$ on the other. Clearly this is not possible (see Figure 1.3).

We note here the very different flavor of this work to traditional statistics. Linear discriminant analysis (see Chapter 3, p. 19, and references therein) is the classical statistical technique for classification. It can not achieve a zero error on the geo-

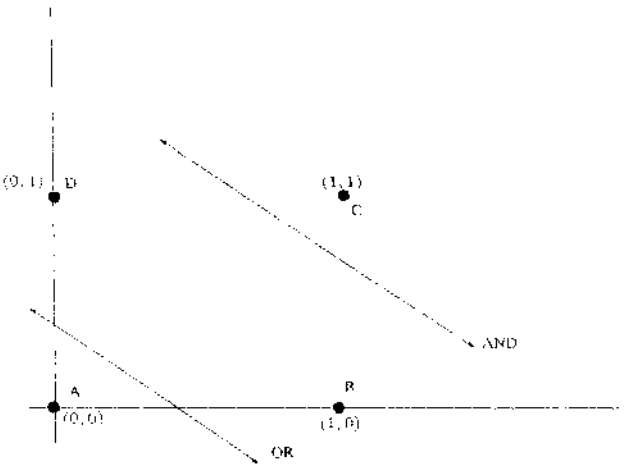

Figure 1.3 The problem of learning a logical function can be recast as a geometric problem by encoding {TRUE, FALSE} as {1, 0}. The figure shows decision boundaries that implement the function OR and AND. The XOR function would have to have points A and C on one side of the line and B and D on the other. It is clear that no single line can achieve that, although a set of lines defining a region or a non-linear boundary can achieve it.

metric XOR problem any more than the perceptron can. However, as far as I know, no statistician has ever shown a lot of concern about this fact.

Table 1.2 Consider two logical variables A and B that can take values in the set {TRUE, FALSE}. The truth values of the logical functions AND, OR, and XOR are shown.

A	B	AND	OR	XOR
T	T	T	T	F
F	F	F	F	F
T	F	F	T	T
F	T	F	T	T

Using a layered structure of perceptron as shown in Figure 2.1 (*p.* 10) overcame this problem and lead to a resurgence in interest in this research area. These are the "multi-layer perceptrons" (MLPs) that are the topic of this work. They required a different learning algorithm to the single perceptron and require that f be a differentiable function. It was the development of such algorithms that was the first step in their use. This has appeared several times in the literature, common early references being Werbos (1974) and Rumelhart et al. (1986).

Already a large number of questions are apparent: such as:

- what if there are more than two classes;
- what if the classes are not linearly separable – but there is a non-linear decision boundary that could separate them;

- do we want a classifier that performs well on this data set or on a new, as yet unseen, data set? Will they be the same *thing*?

These questions will be consider in the ensuing chapters

David Hand has recently written a interesting paper entitled "Classifier Technology and the Illusion of Progress" (Hand, 2006). The progress that he is questioning is the progress of sophisticated techniques like MLPs, support vector machines (Vapnik, 1995, Burges, 1998), and others. He shows that for many examples, the decrease in classification error using sophisticated techniques is only marginal. It may be so small, in fact, that it may be wiped out by the vagaries and difficulties in attendance with real word data sets.

I think that David Hand's caution should be taken to heart. Sophisticated techniques should be used with caution and with an appreciation of their limitations and idiosyncrasies. If there is a good data model available, for example, an understanding that the data are Gaussian, then there may be no justification for using an MLP model.

MLP models have not always been used with an appreciation of their characteristics. The fact that MLPs can be used in a "black box" fashion, and seem to produce reasonable results without a lot of effort being put into modeling the problem, has often led to them being used in this way. It appears that MLPs were being used on hard problems, such as speech recognition and vision, long before any real groundwork was done on understanding the behavior of the MLP as a classifier[4]

This has led to debate in the literature on such elementary points as the capabilities of MLPs with one hidden layer[5], and a lack of understanding of the possible roles of hidden layer units in forming separating boundaries between classes. However, such understanding can be readily arrived at by considering the behavior of the MLP in simple settings that are amenable both to analytic and graphical procedures. In this book the simplest case of two classes and two variables is often used as an example and some points that have been debated in the literature may be amongst the first things that an investigator will notice when confronted with a graphical representation of the output function of an MLP in this simple setting

The aim of this book is to reach a fuller understanding of the MLP model and extend it in a number of desirable ways. There are many introductions and surveys of multi-layer perceptrons in the literature (see below for references); however, none should be necessary in order to understand this book, which should contain the necessary introduction. Other works that could usefully be consulted to gain insight into the MLP model include Cheng and Titterington (1994), Krzanowski and Marriott (1994), Bishop (1995a), Ripley (1996) and Haykin (1999).

We use a number of examples from the area of remote sensing to illustrate various approaches. Richards and Jia (2006) is a good introduction to this problem area while Wilson (1992) and Kiiveri and Caccetta (1996) discuss some of the statistical issues involved. Once again, this work should be entirely self contained – with as much of the problem area introduced in each example as is needed for a full appreciation of the example. Multi-layer perceptrons have been used in the analysis of remotely sensed data in Bischof et al. (1992), Benediktsson et al. (1995)

[4]Early exceptions to this tendency to use MLPs without investigating their behavior are Gibson and Cowan (1990), Lee and Lippmann (1990) and Lui (1990). The situation has been changing markedly in recent years and many of the lacunae in the literature are now being filled.

[5]That is, are they capable of forming disjoint decision regions, see Lippmann (1987).

and Wilkinson et al. (1995). Paola and Schowengerdt (1995) give a review of the application of MLP models to remotely sensed data.