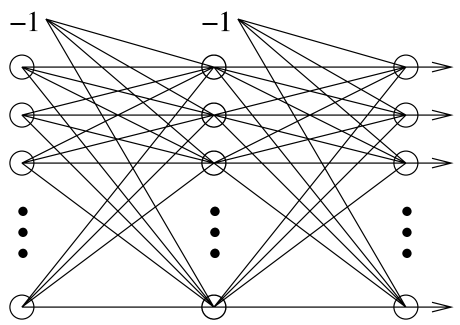
Tik-61.231 Principles of Pattern Recognition

Answers to exercise 6: 30.10.2000

1. **a)** The total amount of weights is the same as the total amount of connections, including the biases. So with the network containing N_x input neurons, N_h hidden layer neurons and N_y output neurons and being fully connected, as is shown in the figure, the total amount of weights is



$$N_w = (N_x + 1)N_h + (N_h + 1)N_y = N_h(N_x + N_y + 1) + N_y$$

b) $N_y = 10$ means that the number of character classes is ten and one output neuron has been assigned to each class, and the output of the network can be seen as a posteriori distribution of the classes. Thus the result is the class whose neuron shows the most activation.

A logical number of input neurons for a 16×16 is $N_x = 16 * 16 = 256$.

Widrow rule is written as

$$N_t \geq 10 \frac{N_w}{N_y}$$

In this case, since $N_y = 10$, this simplifies to $N_t = N_w$. The ratio at the limit can be explored from two directions,

1. $N_t = N_w = N_h(N_x + N_y + 1) + N_y \Rightarrow \frac{N_h}{N_t} = \frac{1-10/N_t}{267}$

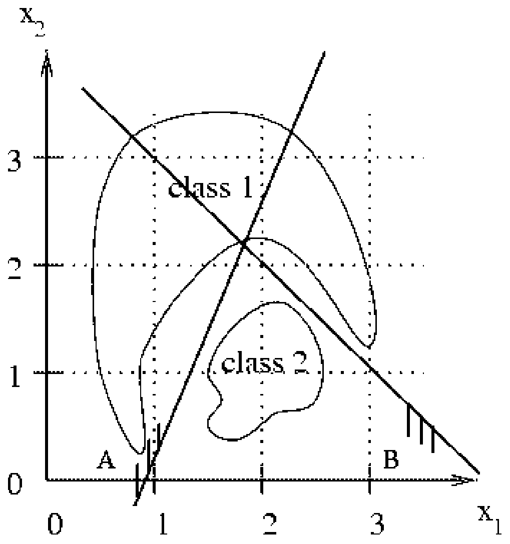
or

2. $\frac{N_h}{N_t} = \frac{N_w}{N_t} = \frac{1}{267+10/N_h}$

So if $N_h >> 1$ or $N_t >> 1$ the limit approaches $\frac{1}{267}$.

It should be noted that Widrows rule provides some kind of an estimate to the suitable network size; there is no proven analytic way of knowing the “right” network size for a given problem. But this estimate gives a reasonable value to start with.

2. A single perceptron can divide the feature space into two different parts based on which side of the decision line the point resides. Thus, two perceptrons can divide the feature space into four parts. So lets draw two decision lines into the figure as shown.



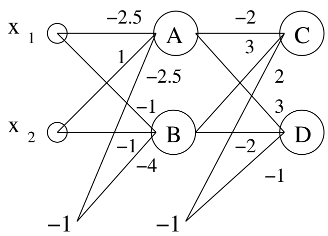
Now, all the samples of class 2 are in the area $C = \neg A \cap B$ and all the samples of class 1 are in $D = \neg(\neg A \cap B)$, with A lying above the first line and B below the second one as indicated with the short marks in the image.

A	B	C	D
0	0	1	0
0	1	0	1
1	0	1	0
1	1	1	0

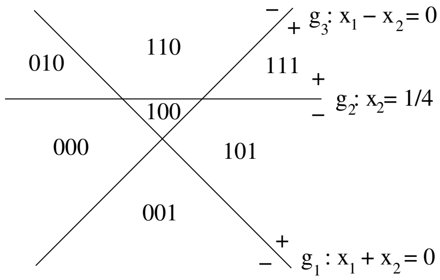
From the figure it can be estimated that for example the functions

$$y_1(x_1) = 2.5x_1 - 2.5$$
$$y_2(x_1) = -x_1 + 4$$

could separate the areas. Thus the network weights would be

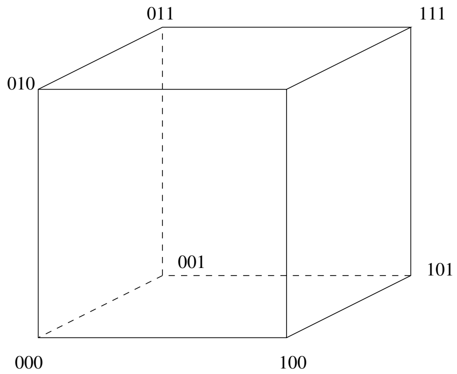


3. First, lets draw the lines together and mark which side of the line the equation gives positive values on. Then we go through each sector of the space separated by these lines and give them a binary code indicating which side of each of the lines it is on, for example 001 is on the negative side of g_1 and g_2 but on the positive side of g_3, the ith bit being one if the area resided on the positive side of g_i and zero if it resides on the negative side.

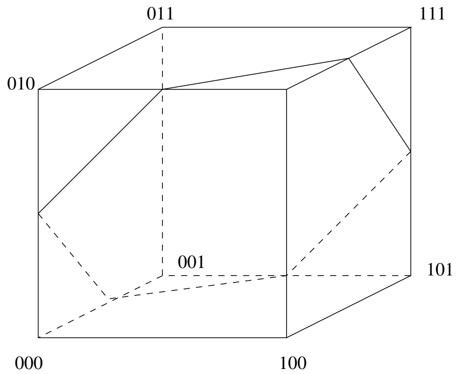


Now we can also see that no section received the value 011, this is perfectly normal. 011 is called a virtual polyhedra, which are quite common when dividing the two-dimensional space with multiple lines that do not all intersect at the same position.

Now, each of the areas is mapped onto the vertice of a three-dimensional cube.

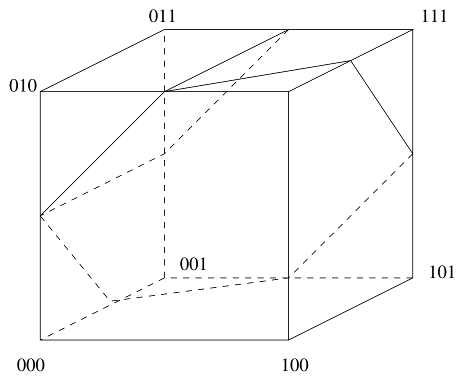


a) For the regions to be possible to classify with a two-layer network, it is necessary that the vertices of the cube that we want to join are linearly separable (in the cube projection). Thus we can choose for example the vertices 000, 100, 101 and 110. We can insert a plane of, for example, the form $y_1 - y_2 - y_3 + 1/2 = 0$, with the positive side being in the direction of 100, to separate these, as is illustrated in the image.

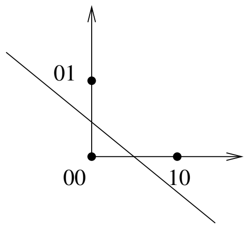


Thus we obtain the second-layer weight vector to be $\omega_{21} = [1 \; -1 \; -1; -1/2]^T$.

b) A three-layer network can separate any union of N_h vertices, where N_h is the amount of hidden-layer neurons. Thus it is possible to separate for example the vertices 000, 100, 101, 010 and 011. (Or looking at it the other way around, the remaining vertices of 110 and 111, which just happens to be the same number as the intended amount of hidden layer neurons). This can be accomplished for example by using the plane $-y_1 + y_2 - 1/2 = 0$, the positive side is in the direction of 010 and 001.



Now we obtain the second second-layer weight vector of $\omega_{22} = [-1\ 1\ 0; 1/2]^T$.
Further, we must now combine these hidden layer outputs. This can again be visualized by plotting the possible outputs 10, 00 and 01 onto a two-dimensional space and extracting a line to separate them. (Note again, that the situation 11 is now impossible; this is clear by looking at the planes and the cube vertices.)



Now these can be separated with for example the line $z_1 + z_2 + 1/2 = 0$, resulting in the final weight vector of $\omega_3 = [1\ 1\ -1/2]^T$.

4. The objective is to find the minimum regarding ω for the joint distribution expectation value,

$$\min_{\underline{\omega}} E_{x,d}[(y(x) - d)^2]$$

It helps to recall that $E_{x,d}[\cdots] = E_x[E_{d|x}[\cdots]]$ and $E_{d|x}[E_{d|x}[\cdots]] = E_{d|x}[\cdots]$
Now to the task at hand;

$$\begin{aligned}
E_{x,d}[(y(x) - d)^2] &= E_{x,d}[((y(x) - E_{d|x}[d]) + (E_{d|x}[d] - d))^2] \\
&= E_{x,d}[(y(x) - E_{d|x}[d])^2 + 2(y(x) - E_{d|x}[d])(E_{d|x}[d] - d) + (E_{d|x}[d] - d)^2] \\
&= E_x[E_{d|x}[(y(x) - E_{d|x}[d])^2] + 2(y(x) - E_{d|x}[d])(E_{d|x}[d] - E_{d|x}[d]) + (E_{d|x}[d] - E_{d|x}[d])^2] \\
&= E_x[E_{d|x}[(y(x) - E_{d|x}[d])^2]]
\end{aligned}$$

So $\min_{\underline{\omega}} E_{x,d}[(y(x) - d)^2] = \min_{\underline{\omega}} E_x[E_{d|x}[(y(x) - E_{d|x}[d])^2]]$. Since the term $(y(x) - E_{d|x}[d])^2 \geq 0$, the expectation is minimized when

$$(y(x) - E_{d|x}[d])^2 = 0 \Rightarrow y(x) = E_{d|x}[d]$$

And

$$E_{d|x}[d] = P(d = 1|x) * 1 + P(d = 0|x) * 1 = P(d = 1|x)$$

Which is the optimal discriminant function for a Bayesian classifier.