Problems to look at before Chapter 40

▷ Exercise 40.1.[2] What is $\sum_{K=0}^{N} \binom{N}{K}$?

[The symbol $\binom{N}{K}$ means the combination $\frac{N!}{K!(N-K)!}$.]

▷ Exercise 40.2.[2] If the top row of Pascal's triangle (which contains the single number '1') is denoted row zero, what is the sum of all the numbers in the triangle above row N?

▷ Exercise 40.3.[2] 3 points are selected at random on the surface of a sphere. What is the probability that all of them lie on a single hemisphere?

This chapter's material is originally due to Polya (1954) and Cover (1965) and the exposition that follows is Yaser Abu-Mostafa's.

40

Capacity of a Single Neuron

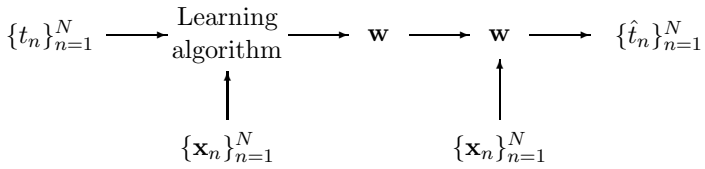

Figure 40.1. Neural network learning viewed as communication.

▶ 40.1 Neural network learning as communication

Many neural network models involve the adaptation of a set of weights $\mathbf{w}$ in response to a set of data points, for example a set of N target values $D_N = \{t_n\}_{n=1}^N$ at given locations $\{\mathbf{x}_n\}_{n=1}^N$. The adapted weights are then used to process subsequent input data. This process can be viewed as a communication process, in which the sender examines the data D_N and creates a message $\mathbf{w}$ that depends on those data. The receiver then uses $\mathbf{w}$; for example, the receiver might use the weights to try to reconstruct what the data D_N was. [In neural network parlance, this is using the neuron for 'memory' rather than for 'generalization'; 'generalizing' means extrapolating from the observed data to the value of t_{N+1} at some new location $\mathbf{x}_{N+1}$.] Just as a disk drive is a communication channel, the adapted network weights $\mathbf{w}$ therefore play the role of a communication channel, conveying information about the training data to a future user of that neural net. The question we now address is, 'what is the capacity of this channel?' – that is, 'how much information can be stored by training a neural network?'

If we had a learning algorithm that either produces a network whose response to all inputs is +1 or a network whose response to all inputs is 0, depending on the training data, then the weights allow us to distinguish between just two sorts of data set. The maximum information such a learning algorithm could convey about the data is therefore 1 bit, this information content being achieved if the two sorts of data set are equiprobable. How much more information can be conveyed if we make full use of a neural network's ability to represent other functions?

▶ 40.2 The capacity of a single neuron

We will look at the simplest case, that of a single binary threshold neuron. We will find that the capacity of such a neuron is *two bits per weight*. A neuron with K inputs can store $2K$ bits of information.

To obtain this interesting result we lay down some rules to exclude less interesting answers, such as: 'the capacity of a neuron is infinite, because each

of its weights is a real number and so can convey an infinite number of bits'. We exclude this answer by saying that the receiver is not able to examine the weights directly, nor is the receiver allowed to probe the weights by observing the output of the neuron for arbitrarily chosen inputs. We constrain the receiver to observe the output of the neuron at the same fixed set of N points $\{\mathbf{x}_n\}$ that were in the training set. What matters now is how many different distinguishable functions our neuron can produce, given that we can observe the function only at these N points. How many different binary labellings of N points can a linear threshold function produce? And how does this number compare with the maximum possible number of binary labellings, 2^N? If nearly all of the 2^N labellings can be realized by our neuron, then it is a communication channel that can convey all N bits (the target values $\{t_n\}$) with small probability of error. We will identify the capacity of the neuron as the maximum value that N can have such that the probability of error is very small. [We are departing a little from the definition of capacity in Chapter 9.]

We thus examine the following scenario. The sender is given a neuron with K inputs and a data set D_N which is a labelling of N points. The sender uses an adaptive algorithm to try to find a $\mathbf{w}$ that can reproduce this labelling exactly. We will assume the algorithm finds such a $\mathbf{w}$ if it exists. The receiver then evaluates the threshold function on the N input values. What is the probability that *all* N bits are correctly reproduced? How large can N become, for a given K, without this probability becoming substantially less than one?

General position

One technical detail needs to be pinned down: what set of inputs $\{\mathbf{x}_n\}$ are we considering? Our answer might depend on this choice. We will assume that the points are in *general position*.

Definition 40.1 *A set of points $\{\mathbf{x}_n\}$ in K-dimensional space are in* general position *if any subset of size $\leq K$ is linearly independent, and no $K+1$ of them lie in a $(K-1)$-dimensional plane.*

In $K = 3$ dimensions, for example, a set of points are in general position if no three points are colinear and no four points are coplanar. The intuitive idea is that points in general position are like random points in the space, in terms of the linear dependences between points. You don't expect three random points in three dimensions to lie on a straight line.

The linear threshold function

The neuron we will consider performs the function

$$y = f\left(\sum_{k=1}^{K} w_k x_k\right) \tag{40.1}$$

where

$$f(a) = \begin{cases} 1 & a > 0 \\ 0 & a \leq 0. \end{cases} \tag{40.2}$$

We will not have a bias w_0; the capacity for a neuron with a bias can be obtained by replacing K by $K+1$ in the final result below, i.e., considering one of the inputs to be fixed to 1. (These input points would not then be in general position; the derivation still works.)

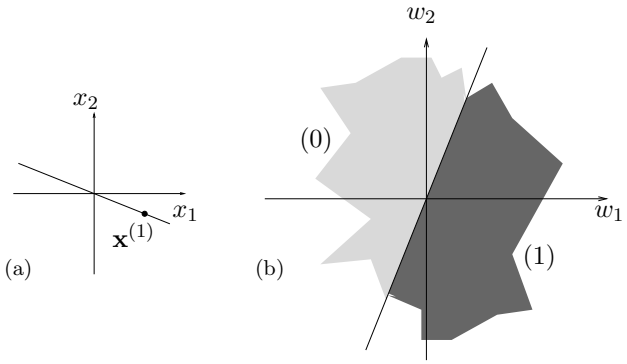

Figure 40.2. One data point in a two-dimensional input space, and the two regions of weight space that give the two alternative labellings of that point.

▶ 40.3 Counting threshold functions

Let us denote by $T(N, K)$ the number of distinct threshold functions on N points in general position in K dimensions. We will derive a formula for $T(N, K)$.

To start with, let us work out a few cases by hand.

In $K = 1$ dimension, for any N

The N points lie on a line. By changing the sign of the one weight w_1 we can label all points on the right side of the origin 1 and the others 0, or *vice versa*. Thus there are two distinct threshold functions. $T(N, 1) = 2$.

With $N = 1$ point, for any K

If there is just one point $\mathbf{x}^{(1)}$ then we can realize both possible labellings by setting $\mathbf{w} = \pm\mathbf{x}^{(1)}$. Thus $T(1, K) = 2$.

In $K = 2$ dimensions

In two dimensions with N points, we are free to spin the separating line around the origin. Each time the line passes over a point we obtain a new function. Once we have spun the line through 360 degrees we reproduce the function we started from. Because the points are in general position, the separating plane (line) crosses only one point at a time. In one revolution, every point is passed over twice. There are therefore $2N$ distinct threshold functions. $T(N, 2) = 2N$.

Comparing with the total number of binary functions, 2^N, we may note that for $N \geq 3$, not all binary functions can be realized by a linear threshold function. One famous example of an unrealizable function with $N = 4$ and $K = 2$ is the exclusive-or function on the points $\mathbf{x} = (\pm 1, \pm 1)$. [These points are not in general position, but you may confirm that the function remains unrealizable even if the points are perturbed into general position.]

In $K = 2$ dimensions, from the point of view of weight space

There is another way of visualizing this problem. Instead of visualizing a plane separating points in the two-dimensional input space, we can consider the two-dimensional *weight space*, colouring regions in weight space different colours if they label the given datapoints differently. We can then count the number of threshold functions by counting how many distinguishable regions there are in weight space. Consider first the set of weight vectors in weight

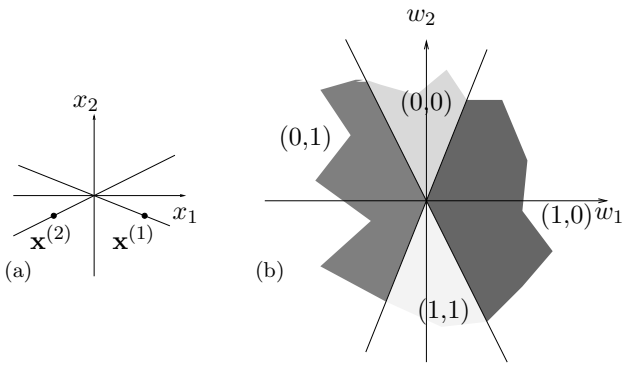

Figure 40.3. Two data points in a two-dimensional input space, and the four regions of weight space that give the four alternative labellings.

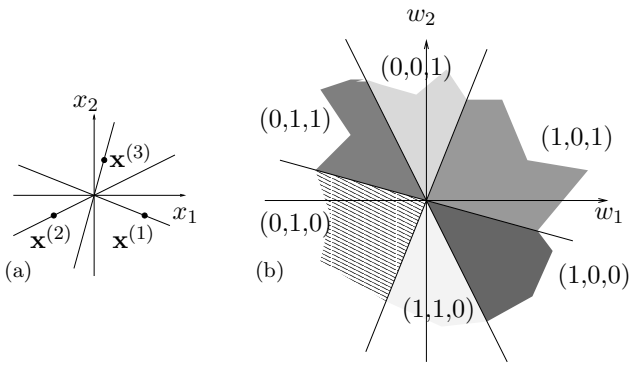

Figure 40.4. Three data points in a two-dimensional input space, and the six regions of weight space that give alternative labellings of those points. In this case, the labellings $(0,0,0)$ and $(1,1,1)$ cannot be realized. For any three points in general position there are always two labellings that cannot be realized.

space that classify a particular example $\mathbf{x}^{(n)}$ as a 1. For example, figure 40.2a shows a single point in our two-dimensional $\mathbf{x}$-space, and figure 40.2b shows the two corresponding sets of points in $\mathbf{w}$-space. One set of weight vectors occupy the half space

$$\mathbf{x}^{(n)} \cdot \mathbf{w} > 0, \tag{40.3}$$

and the others occupy $\mathbf{x}^{(n)} \cdot \mathbf{w} < 0$. In figure 40.3a we have added a second point in the input space. There are now 4 possible labellings: $(1,1)$, $(1,0)$, $(0,1)$, and $(0,0)$. Figure 40.3b shows the two hyperplanes $\mathbf{x}^{(1)} \cdot \mathbf{w} = 0$ and $\mathbf{x}^{(2)} \cdot \mathbf{w} = 0$ which separate the sets of weight vectors that produce each of these labellings. When $N = 3$ (figure 40.4), weight space is divided by three hyperplanes into six regions. Not all of the eight conceivable labellings can be realized. Thus $T(3,2) = 6$.

In $K = 3$ dimensions

We now use this weight space visualization to study the three dimensional case.

Let us imagine adding one point at a time and count the number of threshold functions as we do so. When $N = 2$, weight space is divided by two hyperplanes $\mathbf{x}^{(1)} \cdot \mathbf{w} = 0$ and $\mathbf{x}^{(2)} \cdot \mathbf{w} = 0$ into four regions; in any one region all vectors $\mathbf{w}$ produce the same function on the 2 input vectors. Thus $T(2,3) = 4$.

Adding a third point in general position produces a third plane in $\mathbf{w}$ space, so that there are 8 distinguishable regions. $T(3,3) = 8$. The three bisecting planes are shown in figure 40.5a.

At this point matters become slightly more tricky. As figure 40.5b illustrates, the fourth plane in the three-dimensional $\mathbf{w}$ space cannot transect all eight of the sets created by the first three planes. Six of the existing regions are cut in two and the remaining two are unaffected. So $T(4,3) = 14$. Two

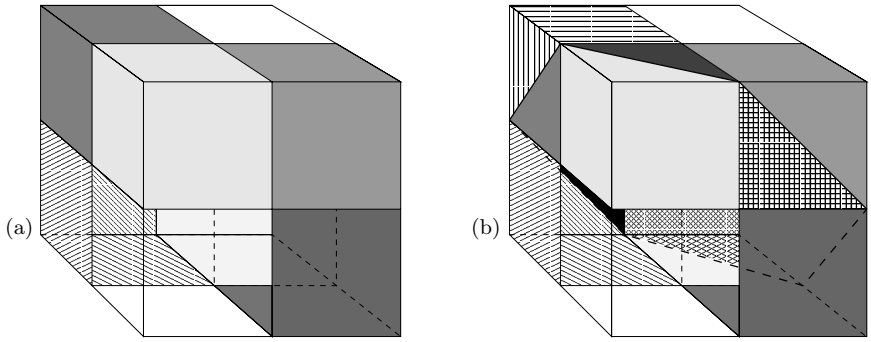

Figure 40.5. Weight space illustrations for $T(3,3)$ and $T(4,3)$. (a) $T(3,3) = 8$. Three hyperplanes (corresponding to three points in general position) divide 3-space into 8 regions, shown here by colouring the relevant part of the surface of a hollow, semi-transparent cube centred on the origin. (b) $T(4,3) = 14$. Four hyperplanes divide 3-space into 14 regions, of which this figure shows 13 (the 14th region is out of view on the right-hand face. Compare with figure 40.5a: all of the regions that are not coloured white have been cut into two.

	K							
N	1	2	3	4	5	6	7	8
1	2	2	2	2	2	2	2	2
2	2	4	4					
3	2	6	8					
4	2	8	14					
5	2	10						
6	2	12						

Table 40.6. Values of $T(N,K)$ deduced by hand.

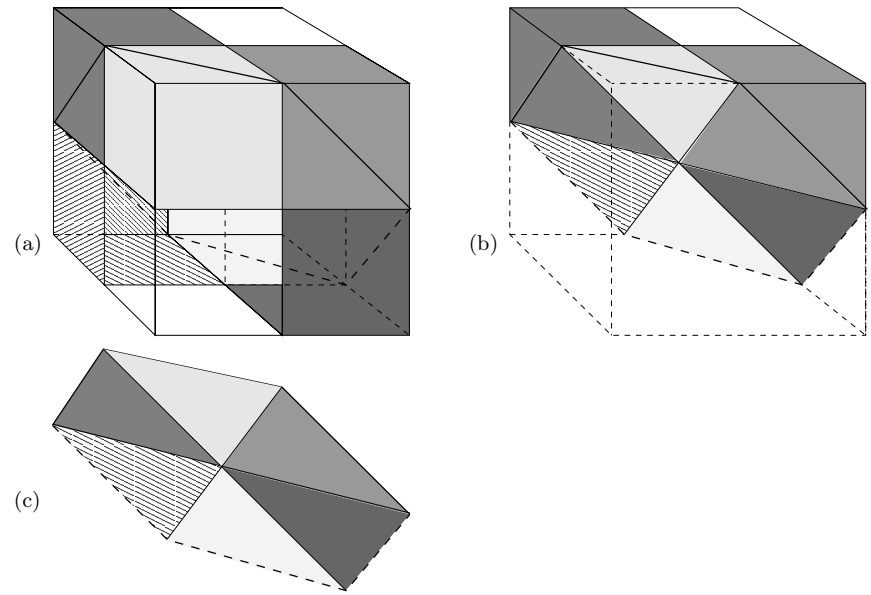

Figure 40.7. Illustration of the cutting process going from $T(3,3)$ to $T(4,3)$. (a) The eight regions of figure 40.5a with one added hyperplane. All of the regions that are not coloured white have been cut into two. (b) Here, the hollow cube has been made solid, so we can see which regions are cut by the fourth plane. The front half of the cube has been cut away. (c) This figure shows the new two dimensional hyperplane, which is divided into six regions by the three one-dimensional hyperplanes (lines) which cross it. Each of these regions corresponds to one of the three-dimensional regions in figure 40.7a which is cut into two by this new hyperplane. This shows that $T(4,3) - T(3,3) = 6$. Figure 40.7c should be compared with figure 40.4b.

of the binary functions on 4 points in 3 dimensions cannot be realized by a linear threshold function.

We have now filled in the values of $T(N, K)$ shown in table 40.6. Can we obtain any insights into our derivation of $T(4, 3)$ in order to fill in the rest of the table for $T(N, K)$? Why was $T(4, 3)$ greater than $T(3, 3)$ by six?

Six is the number of regions that the new hyperplane bisected in **w**-space (figure 40.7a b). Equivalently, if we look in the $K-1$ dimensional subspace that is the Nth hyperplane, that subspace is divided into six regions by the $N-1$ previous hyperplanes (figure 40.7c). Now this is a concept we have met before. Compare figure 40.7c with figure 40.4b. How many regions are created by $N-1$ hyperplanes in a $K-1$ dimensional space? Why, $T(N-1, K-1)$, of course! In the present case $N = 4$, $K = 3$, we can look up $T(3, 2) = 6$ in the previous section. So

$$T(4, 3) = T(3, 3) + T(3, 2). \tag{40.4}$$

Recurrence relation for any N, K

Generalizing this picture, we see that when we add an Nth hyperplane in K dimensions, it will bisect $T(N-1, K-1)$ of the $T(N-1, K)$ regions that were created by the previous $N-1$ hyperplanes. Therefore, the total number of regions obtained after adding the Nth hyperplane is $2T(N-1, K-1)$ (since $T(N-1, K-1)$ out of $T(N-1, K)$ regions are split in two) plus the remaining $T(N-1, K) - T(N-1, K-1)$ regions not split by the Nth hyperplane, which gives the following equation for $T(N, K)$:

$$T(N, K) = T(N-1, K) + T(N-1, K-1). \tag{40.5}$$

Now all that remains is to solve this recurrence relation given the boundary conditions $T(N, 1) = 2$ and $T(1, K) = 2$.

Does the recurrence relation (40.5) look familiar? Maybe you remember building Pascal's triangle by adding together two adjacent numbers in one row to get the number below. The N, K element of Pascal's triangle is equal to

$$C(N, K) \equiv \binom{N}{K} \equiv \frac{N!}{(N-K)!K!}. \tag{40.6}$$

Table 40.8. Pascal's triangle.

	K							
N	0	1	2	3	4	5	6	7
0	1							
1	1	1						
2	1	2	1					
3	1	3	3	1				
4	1	4	6	4	1			
5	1	5	10	10	5	1		

Combinations $\binom{N}{K}$ satisfy the equation

$$C(N, K) = C(N-1, K-1) + C(N-1, K), \quad \text{for all } N > 0. \tag{40.7}$$

[Here we are adopting the convention that $\binom{N}{K} \equiv 0$ if $K > N$ or $K < 0$.] So $\binom{N}{K}$ satisfies the required recurrence relation (40.5). This doesn't mean $T(N, K) = \binom{N}{K}$, since many functions can satisfy one recurrence relation.

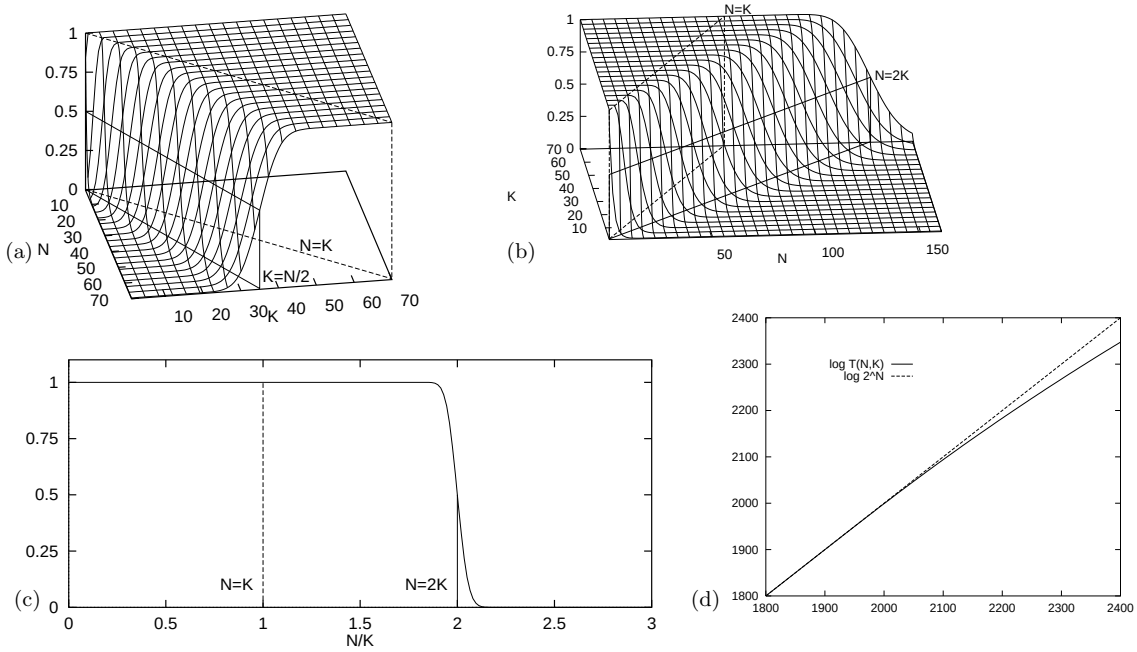

Figure 40.9. The fraction of functions on N points in K dimensions that are linear threshold functions, $T(N,K)/2^N$, shown from various viewpoints. In (a) we see the dependence on K, which is approximately an error function passing through 0.5 at $K = N/2$; the fraction reaches 1 at $K = N$. In (b) we see the dependence on N, which is 1 up to $N = K$ and drops sharply at $N = 2K$. Panel (c) shows the dependence on N/K for $K = 1000$. There is a sudden drop in the fraction of realizable labellings when $N = 2K$. Panel (d) shows the values of $\log_2 T(N,K)$ and $\log_2 2^N$ as a function of N for $K = 1000$. These figures were plotted using the approximation of $T/2^N$ by the error function.

But perhaps we can express $T(N,K)$ as a linear superposition of combination functions of the form $C_{\alpha,\beta}(N,K) \equiv \binom{N+\alpha}{K+\beta}$. By comparing tables 40.8 and 40.6 we can see how to satisfy the boundary conditions: we simply need to translate Pascal's triangle to the right by 1, 2, 3, ...; superpose; add; multiply by two, and drop the whole table by one line. Thus:

$$T(N,K) = 2\sum_{k=0}^{K-1} \binom{N-1}{k}. \tag{40.8}$$

Using the fact that the Nth row of Pascal's triangle sums to 2^N, that is, $\sum_{k=0}^{N-1} \binom{N-1}{k} = 2^{N-1}$, we can simplify the cases where $K-1 \geq N-1$.

$$T(N,K) = \begin{cases} 2^N & K \geq N \\ 2\sum_{k=0}^{K-1} \binom{N-1}{k} & K < N. \end{cases} \tag{40.9}$$

Interpretation

It is natural to compare $T(N,K)$ with the total number of binary functions on N points, 2^N. The ratio $T(N,K)/2^N$ tells us the probability that an arbitrary labelling $\{t_n\}_{n=1}^N$ can be memorized by our neuron. The two functions are equal for all $N \leq K$. The line $N = K$ is thus a special line, defining the maximum number of points on which *any* arbitrary labelling can be realized. This number of points is referred to as the *Vapnik–Chervonenkis dimension* (VC dimension) of the class of functions. The VC dimension of a binary threshold function on K dimensions is thus K.

What is interesting is (for large K) the number of points N such that *almost* any labelling can be realized. The ratio $T(N,K)/2^N$ is, for $N < 2K$, still greater than $1/2$, and for large K the ratio is very close to 1.

For our purposes the sum in equation (40.9) is well approximated by the error function,

$$\sum_{0}^{K} \binom{N}{k} \simeq 2^N \, \Phi\left(\frac{K-(N/2)}{\sqrt{N}/2}\right), \tag{40.10}$$

where $\Phi(z) \equiv \int_{-\infty}^{z} \exp(-z^2/2)/\sqrt{2\pi}$. Figure 40.9 shows the realizable fraction $T(N,K)/2^N$ as a function of N and K. The take-home message is shown in figure 40.9c: although the fraction $T(N,K)/2^N$ is less than 1 for $N > K$, it is only negligibly less than 1 up to $N = 2K$; there, there is a catastrophic drop to zero, so that for $N > 2K$, only a tiny fraction of the binary labellings can be realized by the threshold function.

Conclusion

The capacity of a linear threshold neuron, for large K, is 2 bits per weight.

A single neuron can almost certainly memorize up to $N = 2K$ random binary labels perfectly, but will almost certainly fail to memorize more.

40.4 Further exercises

▷ Exercise 40.4.[2] Can a finite set of $2N$ distinct points in a two-dimensional space be split in half by a straight line

- if the points are in general position?
- if the points are not in general position?

Can $2N$ points in a K dimensional space be split in half by a $K-1$ dimensional hyperplane?

Exercise 40.5.[2, p.491] Four points are selected at random on the surface of a sphere. What is the probability that all of them lie on a single hemisphere? How does this question relate to $T(N,K)$?

Exercise 40.6.[2] Consider the binary threshold neuron in $K = 3$ dimensions, and the set of points $\{\mathbf{x}\} = \{(1,0,0),(0,1,0),(0,0,1),(1,1,1)\}$. Find a parameter vector $\mathbf{w}$ such that the neuron memorizes the labels: (a) $\{t\} = \{1,1,1,1\}$; (b) $\{t\} = \{1,1,0,0\}$.

Find an unrealizable labelling $\{t\}$.

▷ Exercise 40.7.[3] In this chapter we constrained all our hyperplanes to go through the origin. In this exercise, we remove this constraint.

How many regions in a plane are created by N lines in general position?

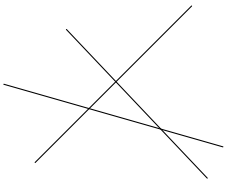

Figure 40.10. Three lines in a plane create seven regions.

Exercise 40.8.[2] Estimate in bits the total sensory experience that you have had in your life – visual information, auditory information, etc. Estimate how much information you have memorized. Estimate the information content of the works of Shakespeare. Compare these with the capacity of your brain assuming you have 10^{11} neurons each making 1000 synaptic connections, and that the capacity result for one neuron (two bits per connection) applies. Is your brain full yet?

▷ Exercise 40.9.[3] What is the capacity of the axon of a spiking neuron, viewed as a communication channel, in bits per second? [See MacKay and McCulloch (1952) for an early publication on this topic.] Multiply by the number of axons in the optic nerve (about 10^6) or cochlear nerve (about 50 000 per ear) to estimate again the rate of acquisition sensory experience.

▶ 40.5 Solutions

Solution to exercise 40.5 (p.490). The probability that all four points lie on a single hemisphere is

$$T(4,3)/2^4 = 14/16 = 7/8. \tag{40.11}$$