Bayes's theorem

A definition of conditional probability is

$$p(x|y) = p(x,y)/p(y) \tag{1}$$

which says, the probability of $X = x$ given $Y = y$ is the probability of $X = x$ and $Y = y$ divided by the probability of $Y = y$. It is easy to see this is a good definition by multiplying across by $p(y)$:

$$p(x,y) = p(x|y)p(y) \tag{2}$$

which says that the probability of $X = x$ and $Y = y$ is the probability of $X = x$ if $Y = y$ multiplied by the probability of $Y = y$, so to get x and y we need to get y and, having gotten y we need to get x; this makes sense. Now, this makes just as much sense the other way:

$$p(x,y) = p(y|x)p(x) \tag{3}$$

Putting the two equations together by eliminating $p(x,y)$ gives

$$p(y|x)p(x) = p(x|y)p(y) \tag{4}$$

or

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{5}$$

This is known as Bayes's theorem or Bayes's rule. It follows from the definition of the conditional probability in a straight forward way, but is immensely useful and also useful in thinking about the structure of science; we will return to that in due course.

For now we'll just think about its usefulness, first and foremost it is useful for turning around probabilities, often we know $p(x|y)$ but $p(y|x)$ is what we need; often Bayes's rule allows us to work out one from the other.

Often the example given is related to testing. Lets say 5% of steaks sold as beef steak are actually made of horse and imagine we have a horsiness test which is positive 90% of the time when tested on horse and 10% of the time when tested on beef. If a piece of steak tests positive for horse, what is the chance it is horse? Let H be the random variable for horsiness, $H = h$ means the steak is horse, $H = b$ means it is beef. Let T correspond to the

test, with $T = y$ when the test says 'yes the steak is horse' and $T = n$ when the test indicates the steak isn't beef.

Now we know $p(h) = 0.05$ and $p(y|h) = 0.9$. What we actually want is $p(h|y)$ and this is what Bayes's rule is useful for:

$$p(h|y) = \frac{p(y|h)p(h)}{p(y)} \tag{6}$$

We don't have $P(y)$ but we can work it out:

$$p(y) = p(y, h) + p(y, b) \tag{7}$$

just by summing the two possibilities, the test is yes and the meat is horse plus the test is yes and the meat is beef. Using the usual expression for the joint probability, $p(x, y) = p(x|y)p(y)$, we have

$$p(y) = p(y|h)p(h) + P(y|b)p(b) \tag{8}$$

Hence

$$p(y) = 0.9 \times 0.05 + 0.1 \times 0.95 = 0.14 \tag{9}$$

Thus

$$p(h|y) = \frac{0.9 \times 0.05}{0.14} = 0.32 \tag{10}$$

Hence, surprisingly, if a steak tests positive for horsiness it is still more likely to be beef. Basically, because there are so many more beef steaks than horse steaks, the relatively small false positive rate for beef still leads to a reasonably high chance a piece of steak that tests positive for horse is nonetheless beef.

There is a particular terminology associated with Bayes' rule; it is sometimes written:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}} \tag{11}$$

The *posterior* is the probability estimated after the evidence is gathered, for example, the chance of horsiness after we have found the test is positive. The *likelihood* is how likely the evidence is given the event, in the example above, it is $p(Y|H)$; the *prior* is the probability estimated before the evidence is gathered, that is $p(H)$, finally *evidence* measure the probability of the evidence, $p(Y)$. This fomulation of Bayes's law is related to picture of how we do science called Bayesian inference, we won't worry about that now.

Naïve Bayes estimator

Many learning algorithms can be thought of as machines for estimating probabilities, often in the face of insufficient data to estimate the probabilities required. A common example used to illustrate this is a spam filter. Let W represent an ordered list of words that may be in an email, say:

$$W = (\text{enlargement}, \text{xxx}, \text{cheapest}, \text{pharmaceuticals}, \text{satisfied}, \text{leeds}) \quad (12)$$

It isn't enough to look at these words on their own; an email with the word 'enlargement' might be talking about photographs, someone might actually be from Leeds. For this reason it is more useful to look at combinations. Say $\mathbf{w}$ is a vector of zeros and ones indicating the presence or absence of different potential spam words in an email. Thus, an email that includes the words 'enlargement', 'xxx' and 'leeds' but not 'cheapest', 'pharmaceuticals' and 'satisfied' would be represented by

$$\mathbf{w} = (1, 1, 0, 0, 0, 1) \quad (13)$$

Now let $S = s$ represent the event of an email being spam. The objective with a spam filter is to estimate $P(s|\mathbf{w})$ for every possible vector $\mathbf{w}$ and then use a cut-off to label any email with a high probability of being spam as 'spam'.

Obviously if you have a truly huge amount of data you could estimate this probability by counting:

$$P(S|(1,1,0,0,0,1)) = \frac{\#\{\text{spam emails with the words enlargement, xxx and leeds}\}}{\#\{\text{all emails with the words enlargement, xxx and leeds}\}} \quad (14)$$

where by 'spam emails with the words enlargement, xxx and leeds' we mean spam emails with those words, but not the three others, cheapest, pharmaceuticals and satisfied; these correspond to zeros in the $\mathbf{w}$ vector. Now, the problem is there are $2^6 = 64$ possible $\mathbf{w}$ vectors, and of course in a real example you'd need many more than six words, thus, for anything but an infeasibly large data set, the amount of emails with the precise combination of words represented by a given $\mathbf{w}$ will be tiny, leading to a poor estimate of the probabilities. For example, if there are 30 words being considered, still nothing like enough to think about when building a spam filter, then there are just over a billion possible $\mathbf{w}$ vectors; that means for most $\mathbf{w}$ vectors the

number of spam emails corresponding to $\mathbf{w}$ will be small, even if you have collected a billion spam emails.

An alternative approach is to use Bayes' rule to get

$$p(S|\mathbf{w}) = \frac{p(\mathbf{w}|s)p(s)}{p(\mathbf{w})} \tag{15}$$

This doesn't look any better, $p(\mathbf{w}|s)$ is no easier to estimate than $P(s|\mathbf{w})$. However, in the naïve Bayes estimator it is additionally assumed that the different words are independent so that

$$\begin{aligned} p((1,1,0,0,0,1)|S) &= p(\text{enlargement}|s)p(\text{xxx}|s)[1-p(\text{cheapest}|s)]\times \\ &\quad [1-p(\text{pharmaceuticals}|s)][1-p(\text{satisfied}|s)]\times \\ &\quad p(\text{leeds}|s) \end{aligned} \tag{16}$$

This is clearly inaccurate, a spam email containing 'enlargement' is more likely to contain 'satisfied' than one that doesn't, that is why it is a 'naïve' classifier. The advantage though is that the individual probabilities are much easier to estimate, there will be more emails with 'leeds' than there will be emails with the exact combination of words represented by $(1,1,0,0,0,1)$ and so counting occurrences will be much more accurate. The same approach can be used to calculate $P(\mathbf{w})$. Although the assumption that the words are independent is not correct, these estimators are quite effective.

The important thing is that we are assuming **conditional independence**, clearly an email that contains the word 'xxx' is more-likely to be spam and is therefore more likely to contain the word 'enlargement', $p(\text{xxx}, \text{enlargement})$ is likely to be very different from $p(\text{xxx})p(\text{enlargement})$. However, the assumption in the naïve Bayes classifier is different, the assumption is that for a spam email probability of the word enlargement appearing is the same irrespective of whether or not the word xxx appears. This is unlikely to be exactly true, but it is a very useful assumption since it vastly increases our ability to estimate the probabilities we require.