Chapter 2
Probabilities: Bayesian Classifiers

The earliest attempts to predict an example's class based on the known attribute values go back to well before World War II—prehistory, by the standards of computer science. Of course, nobody used the term "machine learning," in those days, but the goal was essentially the same as the one addressed in this book.

Here is the essence of the solution strategy they used: using the Bayesian probabilistic theory, calculate for each class the probability of the given object belonging to it, and then choose the class with the highest value.

2.1 The Single-Attribute Case

Let us start with something so simple as to be unrealistic: a domain where each example is described with a single attribute. Once we have developed the basic principles, we will generalize them so that they can be used in more practical domains.

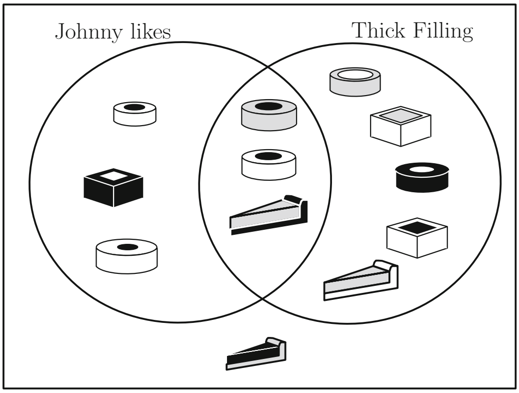
Probabilities The basics are easily explained using the toy domain from the previous chapter. The training set consists of twelve pies ($N_{all} = 12$), of which six are positive examples of the given concept ($N_{pos} = 6$) and six are negative ($N_{neg} = 6$). Assuming that the examples represent faithfully the real situation, the probability of Johnny liking a randomly picked pie is therefore 50%:

$$P(\mathtt{pos}) = \frac{N_{pos}}{N_{all}} = \frac{6}{12} = 0.5 \tag{2.1}$$

Let us now take into consideration one of the attributes, say, `filling-size`. The training set contains eight examples with thick filling ($N_{thick} = 8$). Out of these, three are labeled as positive ($N_{pos|thick} = 3$). This means that the "*conditional probability*

M. Kubat, *An Introduction to Machine Learning*,
DOI 10.1007/978-3-319-63913-0_2

Fig. 2.1 The prior probabilities, $P(\texttt{pos}) = \frac{6}{12}$ and $P(\texttt{thick}) = \frac{8}{12}$; the conditional probabilities, $P(\texttt{pos}|\texttt{thick}) = \frac{3}{8}$ and $P(\texttt{thick}|\texttt{pos}) = \frac{3}{6}$; and the joint probability, $P(\texttt{likes}, \texttt{thick}) = \frac{3}{12}$



of an example being positive given that `filling-size=thick`" is 37.5%—this is what the relative frequency of positive examples among those with thick filling implies:

$$P(\texttt{pos}|\texttt{thick}) = \frac{N_{pos|thick}}{N_{thick}} = \frac{3}{8} = 0.375 \tag{2.2}$$

Applying Conditional Probability to Classification Importantly, the relative frequency is calculated only for pies with the given attribute value. Among these same eight pies, five represented the negative class, which means that $P(\texttt{neg}|\texttt{thick}) = 5/8 = 0.625$. Observing that $P(\texttt{neg}|\texttt{thick}) > P(\texttt{pos}|\texttt{thick})$, we conclude that the probability of Johnny disliking a pie with thick filling is greater than the probability of the opposite case. It thus makes sense for the classifier to label all examples with `filling-size=thick` as negative instances of the "pie that Johnny likes."

Note that conditional probability, $P(\texttt{pos}|\texttt{thick})$, is more trustworthy than the prior probability, $P(\texttt{pos})$, because of the additional information that goes into its calculation. This is only natural. In a DayCare center where the number of boys is about the same as that of girls, we expect a randomly selected child to be a boy with $P(\texttt{boy}) = 0.5$. But the moment we hear someone call the child Johnny, we increase this expectation, knowing that it is rare for a girl to have this name. This is why $P(\texttt{boy}|\texttt{Johnny}) > P(\texttt{boy})$.

Joint Probability Conditional probability should not be confused with *joint probability* of two events occurring simultaneously. Be sure to use the right notation: in joint probability, the terms are separated by commas, $P(\texttt{pos}, \texttt{thick})$; in conditional probability, by a vertical bar, $P(\texttt{pos}|\texttt{thick})$. For a randomly picked pie, $P(\texttt{pos}, \texttt{thick})$ denotes the probability that the example is positive *and* its filling is `thick`; whereas $P(\texttt{pos}|\texttt{thick})$ refers to the occurrence of a positive example among those that have `filling-size=thick`.

A Concrete Example Figure 2.1 illustrates the terms. The rectangle represents all pies. The positive examples are contained in one circle and those with `filling-size=thick` in the other; the intersection contains three instances that satisfy both conditions; one pie satisfies neither, and is therefore left outside both circles. The conditional probability, $P(\texttt{pos}|\texttt{thick}) = 3/8$, is obtained by dividing the size of the intersection (three) by the size of the circle `thick` (eight). The joint probability, $P(\texttt{pos}, \texttt{thick}) = 3/12$, is obtained by dividing the size of the intersection (three) by the size of the entire training set (twelve). The prior probability of $P(\texttt{pos}) = 6/12$ is obtained by dividing the size of the circle `pos` (six) with that of the entire training set (twelve).

Obtaining Conditional Probability from Joint Probability The picture convinces us that joint probability can be obtained from prior probability and conditional probability:

$$P(\texttt{pos}, \texttt{thick}) = P(\texttt{pos}|\texttt{thick}) \cdot P(\texttt{thick}) = \frac{3}{8} \cdot \frac{8}{12} = \frac{3}{12}$$

$$P(\texttt{thick}, \texttt{pos}) = P(\texttt{thick}|\texttt{pos}) \cdot P(\texttt{pos}) = \frac{3}{6} \cdot \frac{6}{12} = \frac{3}{12}$$

Note that joint probability can never exceed the value of the corresponding conditional probability: $P(\texttt{pos}, \texttt{thick}) \leq P(\texttt{pos}|\texttt{thick})$. This is because conditional probability is multiplied by prior probability, $P(\texttt{thick})$ or $P(\texttt{pos})$, which can never be greater than 1.

Another fact to notice is that $P(\texttt{thick}, \texttt{pos}) = P(\texttt{pos}, \texttt{thick})$ because both represent the same thing: the probability of `thick` and `pos` co-occurring. Consequently, the left-hand sides of the previous two formulas have to be equal, which implies the following:

$$P(\texttt{pos}|\texttt{thick}) \cdot P(\texttt{thick}) = P(\texttt{thick}|\texttt{pos}) \cdot P(\texttt{pos})$$

Dividing both sides of this last equation by $P(\texttt{thick})$, we obtain the famous Bayes formula, the foundation for the rest of this chapter:

$$P(\texttt{pos}|\texttt{thick}) = \frac{P(\texttt{thick}|\texttt{pos}) \cdot P(\texttt{pos})}{P(\texttt{thick})} \tag{2.3}$$

If we derive the analogous formula for the probability that pies with `filling-size = thick` will belong to the negative class, we obtain the following:

$$P(\texttt{neg}|\texttt{thick}) = \frac{P(\texttt{thick}|\texttt{neg}) \cdot P(\texttt{neg})}{P(\texttt{thick})} \tag{2.4}$$

Comparison of the values calculated by these two formulas will tell us which class, pos of neg, is more probable. Things are simpler than they look: since the denominator, $P($thick$)$, is the same for both classes, we can just as well ignore it and simply choose the class for which the numerator is higher.

A Trivial Numeric Example That this formula leads to correct values is illustrated in Table 2.1 which, for the sake of simplicity, deals with the trivial case where the examples are described by a single boolean attribute. So simple is this single-attribute world, actually, that we might easily have obtained $P(\texttt{pos}|\texttt{thick})$ and $P(\texttt{neg}|\texttt{thick})$ directly from the training set, without having to resort to the mighty Bayes formula—this makes it easy to verify the correctness of the results.

When the examples are described by two or more attributes, the way of calculating the probabilities is essentially the same, but we need at least one more trick. This will be introduced in the next section.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- How is the Bayes formula derived from the relation between the conditional and joint probabilities?
- What makes the Bayes formula so useful? What does it enable us to calculate?
- Can the joint probability, $P(x, y)$, have a greater value than the conditional probability, $P(x|y)$? Under what circumstances is $P(x|y) = P(x, y)$?

2.2 Vectors of Discrete Attributes

Let us now proceed to the question how to apply the Bayes formula in domains where the examples are described by vectors of attributes such as $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

Multiple Classes Many realistic applications have more than two classes, not just the pos and neg from the "pies" domain. If c_i is the label of the i-th class, and if $\mathbf{x}$ is the vector describing the object we want to classify, the Bayes formula acquires the following form:

$$P(c_i|\mathbf{x}) = \frac{P(\mathbf{x}|c_i)P(c_i)}{P(\mathbf{x})}$$

Table 2.1 Illustrating the principle of Bayesian decision making

Let the training examples be described by a single attribute, `filling-size`, whose value is either `thick` or `thin`. We want the machine to recognize the positive class (`pos`). Here are the eight available training examples:

	ex1	ex2	ex3	ex4	ex5	ex6	ex7	ex8
Size	thick	thick	thin	thin	thin	thick	thick	thick
Class	pos	pos	pos	pos	neg	neg	neg	neg

The probabilities of the individual attribute values and class labels are obtained by their relative frequencies. For instance, three out of the eight examples are characterized by `filling-size=thin`; therefore, $P(\texttt{thin}) = 3/8$.

$P(\texttt{thin}) = 3/8$
$P(\texttt{thick}) = 5/8$
$P(\texttt{pos}) = 4/8$
$P(\texttt{neg}) = 4/8$

The conditional probability of a concrete attribute value within a given class is, again, determined by relative frequency. Our training set yields the following values:

$P(\texttt{thin}|\texttt{pos}) = 2/4$
$P(\texttt{thick}|\texttt{pos}) = 2/4$
$P(\texttt{thin}|\texttt{neg}) = 1/4$
$P(\texttt{thick}|\texttt{neg}) = 3/4$

Using these values, the Bayes formula gives the following conditional probabilities:

$P(\texttt{pos}|\texttt{thin}) = 2/3$
$P(\texttt{pos}|\texttt{thick}) = 2/5$
$P(\texttt{neg}|\texttt{thin}) = 1/3$
$P(\texttt{neg}|\texttt{thick}) = 3/5$

(note that $P(\texttt{pos}|\texttt{thin}) + P(\texttt{neg}|\texttt{thin}) = P(\texttt{pos}|\texttt{thick}) + P(\texttt{neg}|\texttt{thick}) = 1$)

Based on these results, we conclude that an example with `filling-size=thin` should be classified as positive because $P(\texttt{pos}|\texttt{thin}) > P(\texttt{neg}|\texttt{thin})$. Conversely, an example with `filling-size = thick` should be classified as negative because $P(\texttt{neg}|\texttt{thick}) > P(\texttt{pos}|\texttt{thick})$.

The denominator being the same for each class, we choose the class that maximizes the numerator, $P(\mathbf{x}|c_i)P(c_i)$. Here, $P(c_i)$ is easy to estimate by the relative frequency of c_i in the training set. As for $P(\mathbf{x}|c_i)$, however, things are not so simple.

A Vector's Probability $P(\mathbf{x}|c_i)$ is the probability that a randomly selected representative of class c_i is described by vector $\mathbf{x}$. Can its value be estimated by relative frequency? Not really. In the "pies" domain, the size of the instance space was 108 different examples, of which the training set contained twelve. These twelve vectors were each represented by one training example, while none of the other vectors (the vast majority!) was represented at all. The relative frequency of $\mathbf{x}$ among the six positive examples was thus either $P(\mathbf{x}|\texttt{pos}) = 1/6$, when $\mathbf{x}$ was among them, or $P(\mathbf{x}|\texttt{pos}) = 0$, when it was not. Any $\mathbf{x}$ identical to a training example "inherits" this example's class label; if the vector is *not* found in the training set, we have $P(\mathbf{x}|c_i) = 0$ for any c_i. The numerator in the Bayes formula thus being always $P(\mathbf{x}|c_i)P(c_i) = 0$, we are unable to choose the most probable class. Evidently, we will not get very far calculating the probability of an event that occurs only once or not at all.

This, fortunately, is not the case with the individual attributes. For instance, shape=circle occurs four times among the positive examples and twice among the negative, the corresponding probabilities thus being $P(\texttt{shape} = \texttt{circle}|\texttt{pos}) = 4/6$ and $P(\texttt{shape} = \texttt{circle}|\texttt{neg}) = 2/6$. If an attribute can acquire only two or three values, chances are high that each of these values is represented in the training set more than once, thus offering better grounds for probability estimates.

Mutually Independent Attributes What is needed is a formula that combines probabilities of individual attribute values into the probability of the given attribute vector in the given class: $P(\mathbf{x}|c_i)$. As long as the attributes are independent of each other, this is simple. If $P(x_i|c_j)$ is the probability that the value of the i-th attribute of an example from class c_j is x_i, then the probability, $P(\mathbf{x}|c_j)$, that a random representative of c_j is described by $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, is calculated as follows:

$$P(\mathbf{x}|c_j) = \prod_{i=1}^{n} P(x_i|c_j) \tag{2.5}$$

An object will be labeled with c_j if this class maximizes the following version of the Bayes formula's numerator:

$$P(c_j) \cdot \prod_{i=1}^{n} P(x_i|c_j) \tag{2.6}$$

The Naive Bayes Assumption The reader may complain that the assumption of mutually independent attributes is rarely justified. Indeed, can the interrelation of diverse variables ever be avoided? An object's weight grows with its size, the quality of health care may be traced to an individual's income, an object's color can be derived from its physical properties. In short, domains where no two attributes are in any way related to each other are rare. No wonder that the above-described approach is known under the unflattering name, *Naive Bayes*.

Yet practical experience is not bad at all. True, the violation of the "independence requirement" renders the probability estimates inaccurate. However, this does not necessarily make them point to the wrong classes. Remember? **x** is labeled with the class that maximizes $P(\mathbf{x}|c_i) \cdot P(c_i)$. If the product's value is 0.8 for one class and 0.2 for the other, then the classifier's behavior will not change even if the probability estimates miss the accuracy mark by ten or 20%. And so, while requesting that the attributes in principle be independent, we will do reasonably well even if they are not.

When Mutual Independence Cannot Be Assumed This said, we have to ask how to handle the case where attribute interdependence *cannot* be ignored. A scientist's first instinct may be to suggest more sophisticated ways of estimating $P(\mathbf{x}|c_i)$. These do indeed exist, but their complexity grows with the number of attributes, and they contain terms whose values are hard to determine. The practically minded engineer doubts that the trouble is justified by the benefits it brings.

A more pragmatic approach will therefore seek to reduce the attribute dependence by appropriate data pre-processing. A good way to start is to get rid of redundant attributes, those whose values are known to depend on others. For instance, if the set of attributes contains `age, date-of-birth`, and `current-date`, chances are that Naive Bayes will do better if we use only `age`.

We can also try to replace two or three attributes by an artificially created one that combines them. Thus in the "pies" domain, a baker might have told us that `filling-size` is not quite independent of `crust-size`: if one is `thick`, the other is `thin` and vice versa. In this event, we may benefit from replacing the two attributes with a new one, say, `CF-size`, that acquires only two values: `thick-crust-and-thin-filling` or `thin-crust-and-thick-filling`.

In the last resort, if we are prejudiced against advanced methods of multivariate probability estimates, and if we want to avoid data pre-processing, there is always the possibility of giving up on Bayesian classifiers altogether, preferring some of the machine-learning paradigms from the later chapters of this book.

A Numeric Example To get used to the mechanism in which Naive Bayes is used for classification purposes, the reader may want to go through the example in Table 2.2. Here the class of a previously unseen pie is established based on the training set from Table 1.1. The Bayes formula is used, and the attributes are assumed to be mutually independent.

The procedure is summarized by the pseudocode in Table 2.3.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- Under what circumstances shall we assume that the individual attributes are mutually independent? What benefit does this assumption bring for the estimates of $P(\mathbf{x}|c_i)$?
- Discuss the conflicting aspects of this assumption.

2.3 Probabilities of Rare Events: Exploiting the Expert's Intuition

In the first approximation, probability is almost identified with relative frequency: having observed x thirty times in one hundred trials, we assume that $P(x) = 0.3$. This is how we did it in the previous sections.

Table 2.2 Bayesian classification: examples described by vectors of independent attributes

Suppose we want to apply the Bayesian formula to the training set from Table 1.1 in order to determine the class of the following object:

```
x = [shape=square, crust-size=thick, crust-shade=gray
       filling-size=thin, filling-shade=white]
```

There are two classes, pos and neg. The procedure is to calculate the numerator of the Bayes formula separately for each of them, and then choose the class with the higher value. In the training set, each class has the same number of representatives: $P(\text{pos}) = P(\text{neg}) = 0.5$. The remaining terms, $\prod_{i=1}^{n} P(x_i|\text{pos})$ and $\prod_{i=1}^{n} P(x_i|\text{neg})$, are calculated from the following conditional probabilities:

P(shape=square|pos)	= 1/6	P(shape=square|neg)	= 2/6
P(crust-size=thick|pos)	= 5/6	P(crust-size=thick|neg)	= 5/6
P(crust-shade=gray|pos)	= 1/6	P(crust-shade=gray|neg)	= 2/6
P(filling-size=thin|pos)	= 3/6	P(filling-size=thin|neg)	= 1/6
P(filling-shade=white|pos)	= 1/6	P(filling-shade=white|neg)	= 2/6

Based on these values, we obtain the following probabilities:

$$P(\text{x}|\text{pos}) = \prod_{i=1}^{n} P(x_i|\text{pos}) = \frac{1}{6} \cdot \frac{5}{6} \cdot \frac{1}{6} \cdot \frac{3}{6} \cdot \frac{1}{6} = \frac{15}{6^5}$$

$$P(\text{x}|\text{neg}) = \prod_{i=1}^{n} P(x_i|\text{neg}) = \frac{2}{6} \cdot \frac{5}{6} \cdot \frac{2}{6} \cdot \frac{1}{6} \cdot \frac{2}{6} = \frac{40}{6^5}$$

Since $P(\mathbf{x}|\text{pos}) < P(\mathbf{x}|\text{neg})$, we label $\mathbf{x}$ with the negative class.

Table 2.3 Classification with the Naive-Bayes principle

The example to be classified is described by $\mathbf{x} = (x_1, \ldots, x_n)$.

1. For each x_i, and for each class c_j, calculate the conditional probability, $P(x_i|c_j)$, as the relative frequency of x_i among those training examples that belong to c_j.
2. For each class, c_j, carry out the following two steps:

   i) estimate $P(c_j)$ as the relative frequency of this class in the training set;
   ii) calculate the conditional probability, $P(\mathbf{x}|c_j)$, using the "naive" assumption of mutually independent attributes:

$$P(\mathbf{x}|c_j) = \prod_{i=1}^{n} P(x_i|c_j)$$

3. Choose the class with the highest value of $P(c_j) \cdot \prod_{i=1}^{n} P(x_i|c_j)$.

To be fair, though, such estimates can be trusted only when supported by a great many observations. It is conceivable that a coin flipped four times comes up heads three times, and yet it will be overhasty to interpret this observation as meaning that $P(\texttt{heads}) = 0.75$; the physics of the experiment suggests that a fair coin should come up heads 50% of the time. Can this *prior expectation* help us improve probability estimates in domains with insufficient numbers of observations?

The answer is, "Yes, we can use the *m*-estimate."

The Essence of an *m*-Estimate Let us illustrate the principle using the case of an unfair coin where one side comes up somewhat more frequently than the other. In the absence of any better guidance, the prior expectation of heads is $\pi_{head} = 0.5$. An auxiliary parameter, m, helps the engineer tell the class-predicting program *how confident* he is in this value, how much the prior expectation can be trusted (higher m indicating higher confidence).

Let us denote by N_{all} the number of times the coin was flipped, and by N_{heads} the number of times the coin came up heads. The way to combine these values with the prior expectation and confidence is summarized by the following formula:

$$P_{heads} = \frac{N_{heads} + m\pi_{heads}}{N_{all} + m} \tag{2.7}$$

Note that the formula degenerates to the prior expectation, π_{heads}, if $N_{all} = N_{heads} = 0$. Conversely, it converges to that of relative frequency if N_{all} and N_{heads} are so large as to render the terms $m\pi_{heads}$ and m negligible. Using the values $\pi_{heads} = 0.5$ and $m = 2$, we obtain the following:

$$P_{heads} = \frac{N_{heads} + 2 \times 0.5}{N_{all} + 2} = \frac{N_{heads} + 1}{N_{all} + 2}$$

Illustrating Probability Estimates Table 2.4 shows how the values thus calculated gradually evolve in the course of five trials. The reader can see that the *m*-estimate is for small numbers of experiments more in line with common sense than relative

Table 2.4 For each successive trial, the second row gives the observed outcome; the third, the relative frequency of *heads*; the last, the m-estimate of the probability, assuming $\pi_{heads} = 0.5$ and $m = 2$

Toss number	1	2	3	4	5
Outcome	Heads	Heads	Tails	Heads	Tails
Relative frequency	1.00	1.00	0.67	0.75	0.60
m-estimate	0.67	0.75	0.60	0.67	0.57

frequency. Thus after two trials, m-estimate suggests a 0.75 chance of heads, whereas anybody espousing relative frequency will have to concede that, based on the two experiments, there is a zero chance that the coin will come up tails. As the number of trials increases, though, the values returned by m-estimate and relative frequency tend to converge.

The Impact of the User's Confidence Let us take a closer look at the effect of m, the user's confidence. A lot is revealed if we compare the two different settings below: $m = 100$ on the left and $m = 1$ on the right (in both cases, $\pi_{heads} = 0.5$).

$$\frac{N_{heads} + 50}{N_{all} + 100} \qquad\qquad \frac{N_{heads} + 0.5}{N_{all} + 1}$$

The version with $m = 100$ allows the prior estimate to be modified only if really substantial evidence is available ($N_{heads} \gg 50, N_{all} \gg 100$). By contrast, the version with $m = 1$ allows the user's opinion to be controverted with just a few experimental trials.

Domains with More Than Two Outcomes Although we have used a two-outcome domain, the formula is applicable also in multi-outcome domains. Rolling a fair die can result in six different outcomes, and we expect that the probability of seeing, say, three points is $\pi_{three} = 1/6$. Using $m = 6$, we obtain the following:

$$P_{three} = \frac{N_{three} + m\pi_{three}}{N_{all} + m} = \frac{N_{three} + 6 \cdot \frac{1}{6}}{N_{all} + 6} = \frac{N_{three} + 1}{N_{all} + 6}$$

Again, if N_{all} is so high that $m = 6$ and $m\pi_{three} = 1$ can be neglected, the formula converges to relative frequency: $P_{three} = \frac{N_{three}}{N_{all}}$. If we do not want this to happen prematurely (perhaps because we have high confidence in the prior estimate, π_{three}), we prevent it by choosing a higher m.

The Limits of *m*-Estimates We should not forget that the m-estimate is only as good as the parameters it relies on. If we start from an unrealistic prior estimate, the result can be disappointing. Suppose that $\pi_{heads} = 0.9$ and $m = 10$. Equation (2.7) then turns into the following:

$$P_{heads} = \frac{N_{heads} + 9}{N_{all} + 10}$$

When we use this formula to recalculate the values from Table 2.4, we will realize that, after five trials, the probability is estimated as $P_{heads} = \frac{3+9}{5+10} = \frac{12}{15} = 0.8$, surely a less plausible value than the one obtained in the case of $\pi_{heads} = 0.5$ where we got $P_{heads} = 0.57$. The reader is encouraged to verify that the situation will somewhat improve if we reduce m.

Mathematical Soundness Let us make one last comment. A common understanding in mathematics is that the probabilities of all possible events should sum up to 1: if an experiment can have N different outcomes, and if P_i is the probability of the i-th outcome, then $\sum_{i=1}^{N} P_i = 1$. It is easy to verify that Eq. (2.7) satisfies this condition for any value of m. Suppose we are dealing with the coin-tossing domain where there are only two possible outcomes. If the prior estimates sum up to 1 ($\pi_{heads} + \pi_{tails} = 1$), then, given that $N_{heads} + N_{tails} = N_{all}$, we derive the following:

$$P_{heads} + P_{tails} = \frac{N_{heads}+m\pi_{heads}}{N_{all}+m} + \frac{N_{tails}+m\pi_{tails}}{N_{all}+m}$$

$$= \frac{N_{heads}+N_{tails}+m(\pi_{heads}+\pi_{tails})}{N_{all}+m} = 1$$

The interested reader will easily generalize this to any finite number of classes.

Why This May Be Useful In the problem presented in Table 2.5, we want to classify example x using the Bayesian classifier. To be able to do that, we first need to calculate the requisite conditional probabilities. Trying to do so for the positive class, however, we realize that, since the training set is so small, none of the training examples has `crust-shade=gray`, the value observed in **x**. If the probabilities are estimated by relative frequency, this concrete conditional probability would be 0. As a result, $P(\texttt{x}|\texttt{pos}) = 0$, regardless of all the other probabilities. This simply does not seem right.

The problem disappears if we use m-estimate instead of relative frequency because the m-estimate is non-zero even if the concrete value has never being observed in the training set.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- Under what circumstances is relative frequency ill-suited for estimates of discrete probabilities?
- What is the impact of parameter m in Eq. (2.7)? Under what circumstances will you prefer large m, and when will you rather go for small m?
- What is the impact of the prior estimate, π_{heads}, in Eq. (2.7)? How is the credibility of m-estimates affected by unrealistic values of π_{heads}?

Table 2.5 An example of one reason for using *m*-estimates in Bayesian classification

Let us return to the "pies" domain from Table 1.1. Remove from the table the first example, then use the rest for the calculation of the probabilities.

```
x = [shape=circle, crust-size=thick, crust-shade=gray
      filling-size=thick, filling-shade=dark]
```

Let us first calculate the probabilities of the individual attribute values:

$$\begin{aligned}
&\mathrm{P}(\texttt{shape=circle}|\texttt{pos}) &&= 3/5\\
&\mathrm{P}(\texttt{crust-size=thick}|\texttt{pos}) &&= 4/5\\
&\mathrm{P}(\texttt{crust-shade=gray}|\texttt{pos}) &&= 0/5\\
&\mathrm{P}(\texttt{filling-size=thick}|\texttt{pos}) &&= 2/5\\
&\mathrm{P}(\texttt{filling-shade=dark}|\texttt{pos}) &&= 3/5
\end{aligned}$$

Based on these values, we obtain the following probabilities:

$$P(\mathrm{x}|\texttt{pos}) = \frac{3\times4\times0\times2\times3}{5^5} = 0.$$

We see that the circumstance that none of the five positive examples has `crust-shade=gray` causes the corresponding conditional probability to equal 0.

The problem is solved if we calculate the probabilities using the *m*-estimate. In this case, none of the conditional probabilities will be 0.

2.4 How to Handle Continuous Attributes

Up till now, we limited our considerations to attributes that assume *discrete* values, estimating their probabilities either by relative frequency or by the *m*-estimate. This, however, is not enough. In many applications, we encounter attributes (such as `age, price` or `weight`) that acquire values from continuous domains.

Relative frequency is then impractical. While it is easy to establish that the probability of an engineering student being male is $P_{male} = 0.7$, the probability that this student's body weight is 184.5 pounds cannot be specified so readily: the number of different `weight` values being infinite, the probability of any one of them is infinitesimally small. What to do in this case?

Discretizing Continuous Attributes One possibility is to *discretize*. The simplest "trick" will split the attribute's original domain in two; for instance, by replacing `age` with the boolean attribute `old` that is *true* for `age` > 60 and *false* otherwise. However, at least part of the available information then gets lost: a person may be `old`, but we no longer know *how old*; nor do we know whether one `old` person is older than another `old` person.

The loss will be mitigated if we divide the original domain into not two, but several intervals, say, $(0, 10], \dots (90, 100]$.[1] Suppose we get ourselves a separate bin for each of these, and place a little black ball into the i-th bin for each training example whose value of `age` falls into the i-th interval.

Having done so, we may reach the situation depicted in Fig. 2.2. The upper part shows the bins, side by side, and the bottom part shows apiecewise constant function created in the following manner: if N is the size of the training set, and N_i is the number of balls in the i-th bin, then the function's value in the i-th interval is N_i/N—the relative frequency of the balls in the i-th bin. Since the area under the function is $\frac{\Sigma N_i}{N} = 1$, we have a mechanism to estimate the probability *not* of a concrete value of `age`, but rather of this value falling into the given interval.

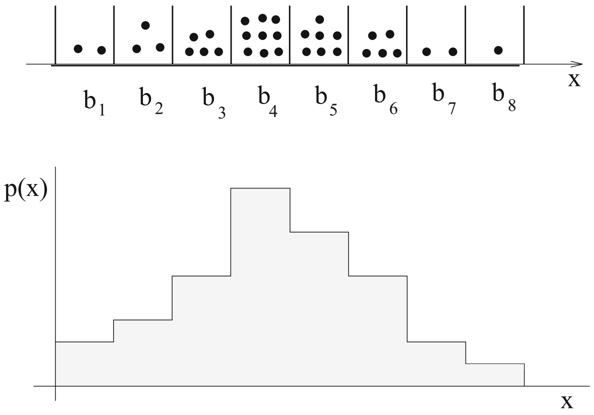
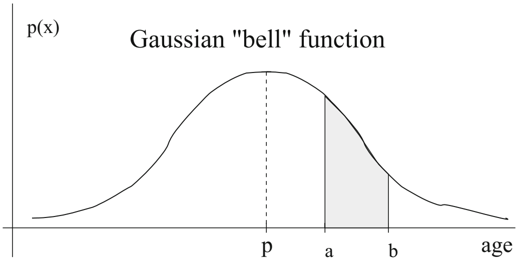


Fig. 2.2 A simple discretization method that represents each subinterval by a separate bin. The *bottom chart* plots the histogram over the individual subintervals

Probability Density Function If the step-function thus constructed seems too crude, we may fine-tune it by dividing the original domain into shorter—and thus more numerous—intervals, provided that the number of balls in each bin is sufficient for reliable probability estimates. If the training set is infinitely large, we can, theoretically speaking, keep reducing the lengths of the intervals until they become infinitesimally small. The result of the bin-filling exercise will then no longer be a step-function, but rather a continuous function, $p(x)$, such as the one in Fig. 2.3. Its interpretation follows from the way it has been created: a high value of $p(x)$ indicates that there are many examples with `age` close to x; conversely, a low value of $p(x)$ tells us that `age` in the vicinity of x is rare. This is why we call $p(x)$ a *probability density function*, often avoiding this mouthful by preferring the acronym *pdf*.

[1] We assume here that 100 is the maximum value observed in the training set. Alternatively, our background knowledge may inform us that the given attribute's value cannot exceed 100.

Fig. 2.3 When using the *pdf*, we identify the probability of $x \in [a, b]$ with the relative size of the area below the corresponding section of the *pdf*



Let us be careful about notation. The discrete probability of x is indicated by an uppercase letter, $P(x)$. The *pdf* at x is denoted by a lowercase letter, $p(x)$. And if we want to point out that the *pdf* has been created exclusively from examples belonging to class c_i, we do so by using a subscript, $p_{c_i}(x)$.

Bayes Formula for Continuous Attributes The good thing about the *pdf* is that it makes it possible to employ the Bayes formula even in the case of continuous attributes. We only replace the conditional probability $P(x|c_i)$ with $p_{c_i}(x)$, and $P(x)$ with $p(x)$. Let us begin with the trivial case where the object to be classified is described by a single continuous attribute, x. The Bayes formula then assumes the following form:

$$P(c_i \mid x) = \frac{p_{c_i}(x) \cdot P(c_i)}{p(x)} \tag{2.8}$$

Here, $P(c_i)$ is estimated by the relative frequency of c_i in the training set, $p(x)$ is the *pdf* created from all training examples, and $p_{c_i}(x)$ is the *pdf* created from those training examples that belong to c_i.

Again, the denominator can be ignored because it has the same value for any class. The classifier simply calculates, separately for each class, the value of the numerator, $p_{c_i}(x) \cdot P(c_i)$, and then labels the object with the class for which the product is maximized.

Naive Bayes Revisited When facing the more realistic case where the examples are described by vectors of attributes, we will avail ourselves of the same “trick” as before: the assumption that all attributes are mutually independent. Suppose we encounter an example described by $\mathbf{x} = (x_1, \ldots, x_n)$. The *pdf* at $\mathbf{x}$ is approximated by the product along the individual attributes:

$$p_{c_j}(\mathbf{x}) = \Pi_{i=1}^{n} p_{c_j}(x_i) \tag{2.9}$$

A statistician will be able to suggest formulas that are theoretically sounder; however, higher sophistication often fails to give satisfaction. For one thing, we may commit the sin of accurate calculations with imprecise numbers. Besides, the more complicated the technique, the higher the danger it will be applied incorrectly.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- What is the probability density function, *pdf*, and how does it help us in the context of Bayesian classification?
- Explain the discretization mechanism that helped us arrive at an informal definition of a *pdf*.
- How does the Bayes formula change in domains with continuous attributes? How do we estimate the values of the individual terms?

2.5 Gaussian “Bell” Function: A Standard *pdf*

One way to approximate a *pdf* is to employ the discretization technique from the previous section. Alternatively, we can capitalize on standardized models known to be applicable to many realistic situations. Perhaps the most popular among them is the *gaussian function*, named after the great German mathematician.

The Shape and the Formula Describing It The curve in Fig. 2.3 is an example; its shape betrays why many people call it a “bell function.” The maximum is reached at the mean, $x = \mu$, and the curve slopes down gracefully with the growing distance of x from μ. It is reasonable to expect that this is a good model of the *pdf* of such variables as the body temperature where the density peaks around $x = 99.7$ degrees Fahrenheit.

Expressed mathematically, the gaussian function is defined by the following formula where e is the base of the natural logarithm:

$$p(x) = k \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.10}$$

Parameters Note that the greater the difference between x and μ, the greater the exponent’s numerator, and thus the smaller the value of $p(x)$ because the exponent is negative. The reason the numerator is squared, $(x - \mu)^2$, is to make sure that the value slopes down with the same angle on both sides of the mean, μ; the curve is symmetric. How steep the slope is depends on σ^2, a parameter called *variance*. Greater variance means smaller sensitivity to the difference between x and μ, and thus a “flatter” bell curve; conversely, smaller variance defines a narrower bell curve.

The task for the coefficient k is to make the area under the bell function equal to 1 as required by the theory of probability. It would be relatively easy to prove that this happens when k is determined by the following formula:

$$k = \frac{1}{\sqrt{2\pi\sigma^2}} \tag{2.11}$$

Setting the Parameter Values To be able to use this model when approximating $p_{c_i}(x)$ in a concrete application, we only need to estimate the values of its parameters, μ and σ^2. This is easy. Suppose that class c_i has m representatives among the training examples. If x_i is the value of the given attribute in the i-th example, then the mean and variance, respectively, are calculated using the following formulas:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.12}$$

$$\sigma^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \mu)^2 \tag{2.13}$$

In plain English, the gaussian center, μ, is obtained as the arithmetic average of the values observed in the training examples, and the variance is obtained as the average of the squared differences between x_i and μ. Note that, when calculating variance, we divide the sum by $m - 1$, and not by m, as we might expect. The intention is to compensate for the fact that μ itself is only an estimate. The variance should therefore be somewhat higher than what it would be if we divided by m. Of course, this matters only if the training set is small: for large m, the difference between m and $m - 1$ is negligible.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- Give an example of a continuous variable whose *pdf* can be expected to follow the gaussian distribution.
- What parameters define the bell function? How can we establish their values using the training set?
- How—and why—do we normalize the bell function?

2.6 Approximating PDFs with Sets of Gaussians

While the bell function represents a good mechanism to approximate the *pdf* in many realistic domains, it is not a panacea. Some variables simply do not behave that way. Just consider the distribution of `body-weight` in a group that mixes grade-school children with their parents. If we create the *pdf* using the discretization method, we will observe two peaks: one for the kids, and the other for the grownups. There may be three peaks if it turns out that body-weight of fathers is distributed around a higher mean than that of the mothers. And the number of peaks can be higher still if the families come from diverse ethnic groups.

Combining Gaussian Functions No doubt, a single bell function would misrepresent the situation. But what if we combine two or more of them? If we knew the diverse sources of the examples, we might create a separate gaussian for each source, and then superimpose the bell functions on each other. Would this solve our problem?

The honest answer is, "yes, in this ideal case." In reality, though, prior knowledge about diverse sources is rarely available. A better solution will divide the body-weight values into great many random groups. In the extreme, we may even go as far as to make each example a "group" of its own, and then identify a gaussian center with this example's body-weight, thus obtaining *m* bell functions (for *m* examples).

The Formula to Combine Them Suppose we want to approximate the *pdf* of a continuous attribute, x. If we denote by μ_i the value of x in the i-th example, then the *pdf* is approximated by the following sum of *m* functions:

$$p(x) = k \cdot \Sigma_{i=1}^{m} e^{-\frac{(x-\mu_i)^2}{2\sigma^2}} \tag{2.14}$$

As before, the normalization constant, k, is here to make sure that the area under the curve is 1. This is achieved when k is calculated as follows:

$$k = \frac{1}{m\sigma\sqrt{2\pi}} \tag{2.15}$$

From mathematics, we know that if m is sufficiently high, Eq. (2.14) approximates the *pdf* with almost arbitrary accuracy.

Illustrating the Point Figure 2.4 illustrates the approach using a training set consisting of $m = 3$ examples, the values of attribute x being $x_1 = 0.4, x_2 = 0.5$ and $x_3 = 0.7$. The upper three charts show three bell functions, each centered at one of these points, the variance always being $\sigma^2 = 1$. The bottom chart shows the composed *pdf* created by putting together Eqs. (2.14) and (2.15), using the means, $\mu_1 = 0.4$, $\mu_2 = 0.5$, and $\mu_3 = 0.7$, and $\sigma^2 = 1$:

$$p(x) = \frac{1}{3\sqrt{2\pi}} \cdot [e^{-\frac{(x-0.4)^2}{2}} + e^{-\frac{(x-0.5)^2}{2}} + e^{-\frac{(x-0.7)^2}{2}}]$$

The Impact of Concrete Parameter Values The practical utility of the *pdf* thus obtained (its success when used in the Bayes formula) depends on the choice of σ^2. In Fig. 2.4, we used $\sigma^2 = 1$, but there is no guarantee that this will work in any future application. To be able to adjust it properly, we need to understand how it affects the shape of the composite *pdf*.

Inspecting the gaussian formula, we realize that the choice of a very small value of σ^2 causes great sensitivity to the difference between x and μ_i; the individual bell functions will be "narrow," and the resulting *pdf* will be marked by steep peaks separated by extended "valleys." Conversely, the consequence of a high σ^2 will be an almost flat *pdf*. Seeking a compromise between the two extremes, we will do well if we make σ^2 dependent on the distances between examples.

The simplest solution will use $\sigma^2 = \mu_{max} - \mu_{min}$, where μ_{max} and μ_{min} are the maximum and minimum values of μ_i, respectively. If you think this too crude, you may consider normalizing the difference by the number of examples: $\sigma^2 = (\mu_{max} - \mu_{min})/m$. Large training sets (with high m) will then lead to smaller variations that will narrow the contributing gaussians. Finally, in some domains we might argue that each of the contributing bell functions should have a variance of its own, proportional to the distance from the center of the nearest other bell function. In this case, however, we are no longer allowed to set the value of k by Eq. (2.15).

A Numeric Example The example in Table 2.6 illustrates the whole procedure on a concrete case of a small training set and a vector to be classified. The reader is encouraged to go through all its details to get used to the way the formulas are put together. See also the illustration in Fig. 2.5.

When There Are Too Many Examples For a training set of realistic size, it is impractical to identify each training example with one gaussian centers; nor is it necessary. More often than not, the examples are grouped in *clusters* that can be detected by *cluster analysis* techniques—see Chap. 14. Once the clusters have been found, we identify the gaussian centers with the centroids of the clusters.

What Have You Learned?

To make sure you understand the topic, try to answer the following questions. If needed, return to the appropriate place in the text.

- Under what circumstances is the gaussian function a poor model of the *pdf*?
- Why does the composite *pdf* have to be normalized by k?
- How do we establish the centers and variances of the individual bell functions?

2.7 Summary and Historical Remarks

- Bayesian classifiers calculate the product $P(\mathbf{x}|c_i)P(c_i)$ separately for each class, c_i, and then label the example, $\mathbf{x}$, with the class where this product has the highest value.
- The main problem is how to calculate the probability, $P(\mathbf{x}|c_i)$. Most of the time, the job is simplified by making the assumption that the individual attributes are mutually independent, in which case $P(\mathbf{x}|c_i) = \prod_{j=1}^{n} P(x_j|c_i)$, where n is the number of attributes.

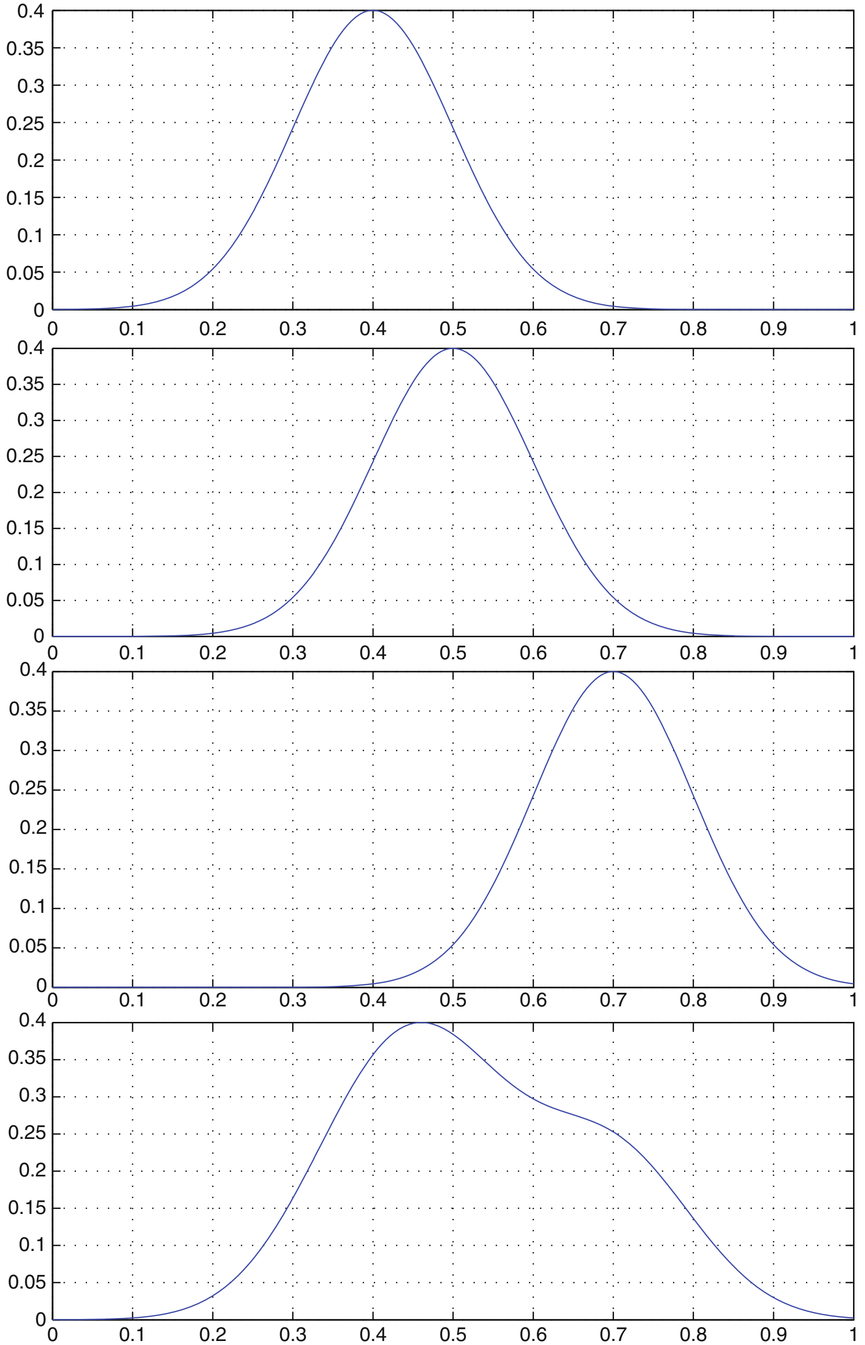


Fig. 2.4 Composing the *pdf* from three examples with the following values of attribute *x*: $\mu_1 = 0.4$, $\mu_2 = 0.5$, and $\mu_3 = 0.7$. The *upper three charts* show the contributing gaussians; the *bottom chart*, the composition. The variance is $\sigma^2 = 1$

- The so-called *m*-estimate makes it possible to take advantage of a user's estimate of an event's probability. This comes handy in domains with insufficient experimental evidence, where relative frequency cannot be relied on.
- In domains with continuous attributes, the role of the discrete probability, $P(\mathbf{x}|c_i)$, is taken over by $p_{c_i}(\mathbf{x})$, the probability density function, *pdf*, but otherwise the

Table 2.6 Using Naive Bayes in domains with three continuous attributes

Suppose we are given a training set that consists of the following six examples, ex$_1$, ..., ex$_6$, each described by three continuous attributes, at$_1$, at$_2$, ... at$_3$:

Example	at$_1$	at$_2$	at$_3$	Class
ex$_1$	3.2	2.1	2.1	pos
ex$_2$	5.2	6.1	7.5	pos
ex$_3$	8.5	1.3	0.5	pos
ex$_4$	2.3	5.4	2.45	neg
ex$_5$	6.2	3.1	4.4	neg
ex$_6$	1.3	6.0	3.35	neg

Using the Bayes formula, we are to find the most probable class of $\mathbf{x} = (9, 2.6, 3.3)$.

Our strategy is to evaluate $p_{\texttt{pos}}(\mathbf{x}) \cdot P(\texttt{pos})$ and $p_{\texttt{neg}}(\mathbf{x}) \cdot P(\texttt{neg})$. Observing that $P(\texttt{pos}) = P(\texttt{neg})$, we simply label $\mathbf{x}$ with pos if $p_{\texttt{pos}}(\mathbf{x}) > p_{\texttt{neg}}(\mathbf{x})$ and with neg otherwise. When constructing the *pdf*, we rely on the independent-attributes assumption. In accordance with Sect. 2.2, we have:

$$p_{\texttt{pos}}(\mathbf{x}) = p_{\texttt{pos}}(at_1) \cdot p_{\texttt{pos}}(at_2) \cdot p_{\texttt{pos}}(at_3) \text{ and}$$
$$p_{\texttt{neg}}(\mathbf{x}) = p_{\texttt{neg}}(at_1) \cdot p_{\texttt{neg}}(at_2) \cdot p_{\texttt{neg}}(at_3)$$

The terms on the right-hand sides are obtained by Eq. (2.14), in which we use $\sigma^2 = 1, m = 3$, and, therefore, $k = 1/\sqrt{(2\pi)^3}$. Thus for the first of these terms, we get the following:

$$p_{\texttt{pos}}(at_1) = \frac{1}{3\sqrt{2\pi}}[e^{-0.5(at_1-3.2)^2} + e^{-0.5(at_1-5.2)^2} + e^{-0.5(at_1-8.5)^2}]$$

Note that the values of at$_1$ in the positive examples are $\mu_1 = 3.2, \mu_2 = 5.2$, and $\mu_3 = 8.5$, respectively—see the exponents in the expression. The functions for the remaining five terms, obtained similarly, are plotted in the rightmost column of Fig. 2.5.

Substituting into these equations the coordinates of $\mathbf{x}$, namely $at_1 = 9, at_2 = 3.6$, and $at_3 = 3.3$, will give us the following:

$$p_{\texttt{pos}}(\mathbf{x}) = 0.0561 \times 0.0835 \times 0.0322 = 0.00015$$
$$p_{\texttt{neg}}(\mathbf{x}) = 0.0023 \times 0.0575 \times 0.1423 = 0.00001$$

Observing that $p_{pos}(\mathbf{x}) > p_{neg}(\mathbf{x})$, we label $\mathbf{x}$ with the class pos.

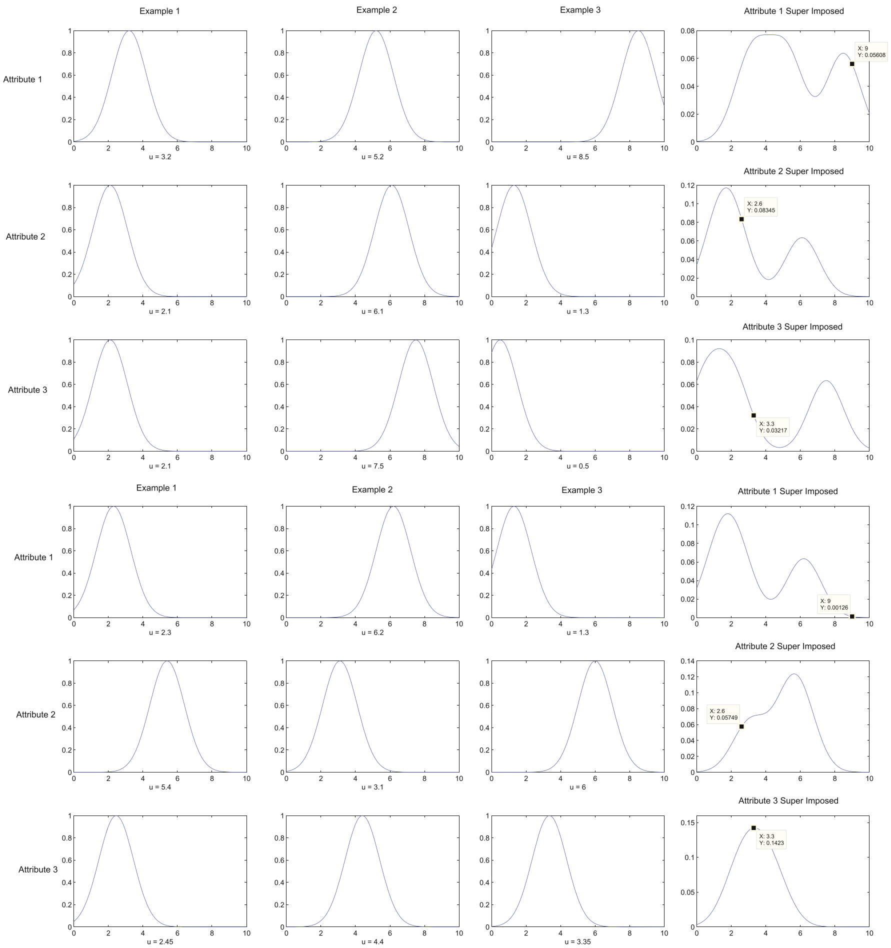


Fig. 2.5 Composing the *pdf*'s separately for the positive and negative class (with $\sigma^2 = 1$). Each row represents one attribute, and each of the *left three columns* represents one example. The *rightmost column* shows the composed *pdf*'s

procedure is the same: the example is labeled with the class that maximizes the product, $p_{c_i}(\mathbf{x})P(c_i)$.

- The concrete shape of the *pdf* is approximated by discretization, by the use of standardized *pdf*s, or by the sum of gaussians.
- The estimates of probabilities are far from perfect, but the results are often satisfactory even when rigorous theoretical assumptions are not satisfied.

Historical Remarks The first papers to use Bayesian decision theory for classification purposes were Neyman and Pearson [72] and [25], but the paradigm gained momentum only with the advent of the computer, when it was advocated by Chow

[14]. The first to use the assumption of independent attributes was Good [33]. The idea of approximating *pdf*'s by the sum of bell functions comes from Parzen [74].

When provided with perfect information about the probabilities, the Bayesian classifier is guaranteed to provide the best possible classification accuracy. This is why it is sometimes used as a reference to which the performance of other approaches is compared.

2.8 Solidify Your Knowledge

The exercises are to solidify the acquired knowledge. The suggested thought experiments will help the reader see this chapter's ideas in a different light and provoke independent thinking. Computer assignments will force the readers to pay attention to seemingly insignificant details they might otherwise overlook.

Exercises

1. A coin tossed three times came up *heads, tails*, and *tails*, respectively. Calculate the m-estimate for these outcomes, using $m = 3$ and $\pi_{heads} = \pi_{tails} = 0.5$.
2. Suppose you have the following training examples, described by three attributes, x_1, x_2, x_3, and labeled by classes c_1 and c_2.

x_1	x_2	x_3	Class
2.1	0.2	3.0	c_1
3.3	1.0	2.9	c_1
2.7	1.2	3.4	c_1
0.5	5.3	0.0	c_2
1.5	4.7	0.5	c_2

Using these data, do the following:

(a) Assuming that the attributes are mutually independent, approximate the following probability density functions: $p_{c_1}(\mathbf{x}), p_{c_2}(\mathbf{x}), p(\mathbf{x})$. Hint: use the idea of superimposed bell functions.

(b) Using the *pdf*'s from the previous step, decide whether $\mathbf{x} = [1.4, 3.3, 3.0]$ should belong to c_1 or c_2.

Give It Some Thought

1. How would you apply the *m*-estimate in a domain with three possible outcomes, $[A, B, C]$, each with the same prior probability estimate, $\pi_A = \pi_B = \pi_C = 1/3$? What if you trust your expectations of A but are not so sure about B and C? Is there a way to reflect this circumstance in the value of the parameter m?
2. Suggest the circumstances under which the accuracy of probability estimates will benefit from the assumption that attributes are mutually independent. Explain the advantages and disadvantages.
3. How would you calculate the probabilities of the output classes in a domain where some attributes are boolean, others discrete, and yet others continuous? Discuss the possibilities of combining different approaches.

Computer Assignments

1. Machine learning researchers often test their algorithms using publicly available benchmark domains. A large repository of such domains can be found at the following address: www.ics.uci.edu/~mlearn/MLRepository.html. Take a look at these data and see how they differ in the numbers of attributes, types of attributes, sizes and so on.
2. Write a computer program that will use the Bayes formula to calculate the class probabilities in a domain where all attributes are discrete. Apply this program to our "pies" domain.
3. For the case of continuous attributes, write a computer program that accepts the training examples in the form of a table such as the one in Exercise 3 above. Based on these, the program approximates the *pdf*s, and then uses them to determine the class labels of future examples.
4. Apply this program to a few benchmark domains from the UCI repository (choose from those where all attributes are continuous) and observe that the program succeeds in some domains better than in others.