

Chapter 3

Probability and Information Theory

In this chapter, we describe probability theory and information theory.

Probability theory is a mathematical framework for representing uncertain statements. It provides a means of quantifying uncertainty and axioms for deriving new uncertain statements. In artificial intelligence applications, we use probability theory in two major ways. First, the laws of probability tell us how AI systems should reason, so we design our algorithms to compute or approximate various expressions derived using probability theory. Second, we can use probability and statistics to theoretically analyze the behavior of proposed AI systems.

Probability theory is a fundamental tool of many disciplines of science and engineering. We provide this chapter to ensure that readers whose background is primarily in software engineering with limited exposure to probability theory can understand the material in this book.

While probability theory allows us to make uncertain statements and reason in the presence of uncertainty, information theory allows us to quantify the amount of uncertainty in a probability distribution.

If you are already familiar with probability theory and information theory, you may wish to skip all of this chapter except for section 3.14, which describes the graphs we use to describe structured probabilistic models for machine learning. If you have absolutely no prior experience with these subjects, this chapter should be sufficient to successfully carry out deep learning research projects, but we do suggest that you consult an additional resource, such as [Jaynes \(2003\)](#).

3.1 Why Probability?

Many branches of computer science deal mostly with entities that are entirely deterministic and certain. A programmer can usually safely assume that a CPU will execute each machine instruction flawlessly. Errors in hardware do occur, but are rare enough that most software applications do not need to be designed to account for them. Given that many computer scientists and software engineers work in a relatively clean and certain environment, it can be surprising that machine learning makes heavy use of probability theory.

This is because machine learning must always deal with uncertain quantities, and sometimes may also need to deal with stochastic (non-deterministic) quantities. Uncertainty and stochasticity can arise from many sources. Researchers have made compelling arguments for quantifying uncertainty using probability since at least the 1980s. Many of the arguments presented here are summarized from or inspired by [Pearl \(1988\)](#).

Nearly all activities require some ability to reason in the presence of uncertainty. In fact, beyond mathematical statements that are true by definition, it is difficult to think of any proposition that is absolutely true or any event that is absolutely guaranteed to occur.

There are three possible sources of uncertainty:

1. Inherent stochasticity in the system being modeled. For example, most interpretations of quantum mechanics describe the dynamics of subatomic particles as being probabilistic. We can also create theoretical scenarios that we postulate to have random dynamics, such as a hypothetical card game where we assume that the cards are truly shuffled into a random order.
2. Incomplete observability. Even deterministic systems can appear stochastic when we cannot observe all of the variables that drive the behavior of the system. For example, in the Monty Hall problem, a game show contestant is asked to choose between three doors and wins a prize held behind the chosen door. Two doors lead to a goat while a third leads to a car. The outcome given the contestant's choice is deterministic, but from the contestant's point of view, the outcome is uncertain.
3. Incomplete modeling. When we use a model that must discard some of the information we have observed, the discarded information results in uncertainty in the model's predictions. For example, suppose we build a robot that can exactly observe the location of every object around it. If the

robot discretizes space when predicting the future location of these objects, then the discretization makes the robot immediately become uncertain about the precise position of objects: each object could be anywhere within the discrete cell that it was observed to occupy.

In many cases, it is more practical to use a simple but uncertain rule rather than a complex but certain one, even if the true rule is deterministic and our modeling system has the fidelity to accommodate a complex rule. For example, the simple rule “Most birds fly” is cheap to develop and is broadly useful, while a rule of the form, “Birds fly, except for very young birds that have not yet learned to fly, sick or injured birds that have lost the ability to fly, flightless species of birds including the cassowary, ostrich and kiwi. . .” is expensive to develop, maintain and communicate, and after all of this effort is still very brittle and prone to failure.

While it should be clear that we need a means of representing and reasoning about uncertainty, it is not immediately obvious that probability theory can provide all of the tools we want for artificial intelligence applications. Probability theory was originally developed to analyze the frequencies of events. It is easy to see how probability theory can be used to study events like drawing a certain hand of cards in a game of poker. These kinds of events are often repeatable. When we say that an outcome has a probability p of occurring, it means that if we repeated the experiment (e.g., draw a hand of cards) infinitely many times, then proportion p of the repetitions would result in that outcome. This kind of reasoning does not seem immediately applicable to propositions that are not repeatable. If a doctor analyzes a patient and says that the patient has a 40% chance of having the flu, this means something very different—we can not make infinitely many replicas of the patient, nor is there any reason to believe that different replicas of the patient would present with the same symptoms yet have varying underlying conditions. In the case of the doctor diagnosing the patient, we use probability to represent a **degree of belief**, with 1 indicating absolute certainty that the patient has the flu and 0 indicating absolute certainty that the patient does not have the flu. The former kind of probability, related directly to the rates at which events occur, is known as **frequentist probability**, while the latter, related to qualitative levels of certainty, is known as **Bayesian probability**.

If we list several properties that we expect common sense reasoning about uncertainty to have, then the only way to satisfy those properties is to treat Bayesian probabilities as behaving exactly the same as frequentist probabilities. For example, if we want to compute the probability that a player will win a poker game given that she has a certain set of cards, we use exactly the same formulas as when we compute the probability that a patient has a disease given that she

has certain symptoms. For more details about why a small set of common sense assumptions implies that the same axioms must control both kinds of probability, see [Ramsey \(1926\)](#).

Probability can be seen as the extension of logic to deal with uncertainty. Logic provides a set of formal rules for determining what propositions are implied to be true or false given the assumption that some other set of propositions is true or false. Probability theory provides a set of formal rules for determining the likelihood of a proposition being true given the likelihood of other propositions.

3.2 Random Variables

A **random variable** is a variable that can take on different values randomly. We typically denote the random variable itself with a lower case letter in plain typeface, and the values it can take on with lower case script letters. For example, x_1 and x_2 are both possible values that the random variable x can take on. For vector-valued variables, we would write the random variable as \mathbf{x} and one of its values as \mathbf{x} . On its own, a random variable is just a description of the states that are possible; it must be coupled with a probability distribution that specifies how likely each of these states are.

Random variables may be discrete or continuous. A discrete random variable is one that has a finite or countably infinite number of states. Note that these states are not necessarily the integers; they can also just be named states that are not considered to have any numerical value. A continuous random variable is associated with a real value.

3.3 Probability Distributions

A **probability distribution** is a description of how likely a random variable or set of random variables is to take on each of its possible states. The way we describe probability distributions depends on whether the variables are discrete or continuous.

3.3.1 Discrete Variables and Probability Mass Functions

A probability distribution over discrete variables may be described using a **probability mass function** (PMF). We typically denote probability mass functions with a capital P . Often we associate each random variable with a different probability

mass function and the reader must infer which probability mass function to use based on the identity of the random variable, rather than the name of the function; $P(x)$ is usually not the same as $P(y)$.

The probability mass function maps from a state of a random variable to the probability of that random variable taking on that state. The probability that $x = x$ is denoted as $P(x)$, with a probability of 1 indicating that $x = x$ is certain and a probability of 0 indicating that $x = x$ is impossible. Sometimes to disambiguate which PMF to use, we write the name of the random variable explicitly: $P(x = x)$. Sometimes we define a variable first, then use \sim notation to specify which distribution it follows later: $x \sim P(x)$.

Probability mass functions can act on many variables at the same time. Such a probability distribution over many variables is known as a **joint probability distribution**. $P(x = x, y = y)$ denotes the probability that $x = x$ and $y = y$ simultaneously. We may also write $P(x, y)$ for brevity.

To be a probability mass function on a random variable x , a function P must satisfy the following properties:

- The domain of P must be the set of all possible states of x .
- $\forall x \in x, 0 \leq P(x) \leq 1$. An impossible event has probability 0 and no state can be less probable than that. Likewise, an event that is guaranteed to happen has probability 1, and no state can have a greater chance of occurring.
- $\sum_{x \in x} P(x) = 1$. We refer to this property as being **normalized**. Without this property, we could obtain probabilities greater than one by computing the probability of one of many events occurring.

For example, consider a single discrete random variable x with k different states. We can place a **uniform distribution** on x —that is, make each of its states equally likely—by setting its probability mass function to

$$P(x = x_i) = \frac{1}{k} \quad (3.1)$$

for all i . We can see that this fits the requirements for a probability mass function. The value $\frac{1}{k}$ is positive because k is a positive integer. We also see that

$$\sum_i P(x = x_i) = \sum_i \frac{1}{k} = \frac{k}{k} = 1, \quad (3.2)$$

so the distribution is properly normalized.

3.3.2 Continuous Variables and Probability Density Functions

When working with continuous random variables, we describe probability distributions using a **probability density function (PDF)** rather than a probability mass function. To be a probability density function, a function p must satisfy the following properties:

- The domain of p must be the set of all possible states of x .
- $\forall x \in \mathbf{x}, p(x) \geq 0$. Note that we do not require $p(x) \leq 1$.
- $\int p(x)dx = 1$.

A probability density function $p(x)$ does not give the probability of a specific state directly, instead the probability of landing inside an infinitesimal region with volume δx is given by $p(x)\delta x$.

We can integrate the density function to find the actual probability mass of a set of points. Specifically, the probability that x lies in some set \mathbb{S} is given by the integral of $p(x)$ over that set. In the univariate example, the probability that x lies in the interval $[a, b]$ is given by $\int_{[a,b]} p(x)dx$.

For an example of a probability density function corresponding to a specific probability density over a continuous random variable, consider a uniform distribution on an interval of the real numbers. We can do this with a function $u(x; a, b)$, where a and b are the endpoints of the interval, with $b > a$. The “;” notation means “parametrized by”; we consider x to be the argument of the function, while a and b are parameters that define the function. To ensure that there is no probability mass outside the interval, we say $u(x; a, b) = 0$ for all $x \notin [a, b]$. Within $[a, b]$, $u(x; a, b) = \frac{1}{b-a}$. We can see that this is nonnegative everywhere. Additionally, it integrates to 1. We often denote that x follows the uniform distribution on $[a, b]$ by writing $x \sim U(a, b)$.

3.4 Marginal Probability

Sometimes we know the probability distribution over a set of variables and we want to know the probability distribution over just a subset of them. The probability distribution over the subset is known as the **marginal probability** distribution.

For example, suppose we have discrete random variables x and y , and we know $P(x, y)$. We can find $P(x)$ with the **sum rule**:

$$\forall x \in \mathbf{x}, P(x = x) = \sum_y P(x = x, y = y). \quad (3.3)$$

The name “marginal probability” comes from the process of computing marginal probabilities on paper. When the values of $P(x, y)$ are written in a grid with different values of x in rows and different values of y in columns, it is natural to sum across a row of the grid, then write $P(x)$ in the margin of the paper just to the right of the row.

For continuous variables, we need to use integration instead of summation:

$$p(x) = \int p(x, y) dy. \quad (3.4)$$

3.5 Conditional Probability

In many cases, we are interested in the probability of some event, given that some other event has happened. This is called a **conditional probability**. We denote the conditional probability that $y = y$ given $x = x$ as $P(y = y \mid x = x)$. This conditional probability can be computed with the formula

$$P(y = y \mid x = x) = \frac{P(y = y, x = x)}{P(x = x)}. \quad (3.5)$$

The conditional probability is only defined when $P(x = x) > 0$. We cannot compute the conditional probability conditioned on an event that never happens.

It is important not to confuse conditional probability with computing what would happen if some action were undertaken. The conditional probability that a person is from Germany given that they speak German is quite high, but if a randomly selected person is taught to speak German, their country of origin does not change. Computing the consequences of an action is called making an **intervention query**. Intervention queries are the domain of **causal modeling**, which we do not explore in this book.

3.6 The Chain Rule of Conditional Probabilities

Any joint probability distribution over many random variables may be decomposed into conditional distributions over only one variable:

$$P(x^{(1)}, \dots, x^{(n)}) = P(x^{(1)}) \prod_{i=2}^n P(x^{(i)} \mid x^{(1)}, \dots, x^{(i-1)}). \quad (3.6)$$

This observation is known as the **chain rule** or **product rule** of probability. It follows immediately from the definition of conditional probability in equation 3.5.

For example, applying the definition twice, we get

$$\begin{aligned} P(a, b, c) &= P(a \mid b, c)P(b, c) \\ P(b, c) &= P(b \mid c)P(c) \\ P(a, b, c) &= P(a \mid b, c)P(b \mid c)P(c). \end{aligned}$$

3.7 Independence and Conditional Independence

Two random variables x and y are **independent** if their probability distribution can be expressed as a product of two factors, one involving only x and one involving only y :

$$\forall x \in \mathbf{x}, y \in \mathbf{y}, p(x = x, y = y) = p(x = x)p(y = y). \quad (3.7)$$

Two random variables x and y are **conditionally independent** given a random variable z if the conditional probability distribution over x and y factorizes in this way for every value of z :

$$\forall x \in \mathbf{x}, y \in \mathbf{y}, z \in \mathbf{z}, p(x = x, y = y \mid z = z) = p(x = x \mid z = z)p(y = y \mid z = z). \quad (3.8)$$

We can denote independence and conditional independence with compact notation: $x \perp y$ means that x and y are independent, while $x \perp y \mid z$ means that x and y are conditionally independent given z .

3.8 Expectation, Variance and Covariance

The **expectation** or **expected value** of some function $f(x)$ with respect to a probability distribution $P(x)$ is the average or mean value that f takes on when x is drawn from P . For discrete variables this can be computed with a summation:

$$\mathbb{E}_{x \sim P}[f(x)] = \sum_x P(x)f(x), \quad (3.9)$$

while for continuous variables, it is computed with an integral:

$$\mathbb{E}_{x \sim p}[f(x)] = \int p(x)f(x)dx. \quad (3.10)$$

When the identity of the distribution is clear from the context, we may simply write the name of the random variable that the expectation is over, as in $\mathbb{E}_x[f(x)]$. If it is clear which random variable the expectation is over, we may omit the subscript entirely, as in $\mathbb{E}[f(x)]$. By default, we can assume that $\mathbb{E}[\cdot]$ averages over the values of all the random variables inside the brackets. Likewise, when there is no ambiguity, we may omit the square brackets.

Expectations are linear, for example,

$$\mathbb{E}_x[\alpha f(x) + \beta g(x)] = \alpha \mathbb{E}_x[f(x)] + \beta \mathbb{E}_x[g(x)], \quad (3.11)$$

when α and β are not dependent on x .

The **variance** gives a measure of how much the values of a function of a random variable x vary as we sample different values of x from its probability distribution:

$$\text{Var}(f(x)) = \mathbb{E} \left[(f(x) - \mathbb{E}[f(x)])^2 \right]. \quad (3.12)$$

When the variance is low, the values of $f(x)$ cluster near their expected value. The square root of the variance is known as the **standard deviation**.

The **covariance** gives some sense of how much two values are linearly related to each other, as well as the scale of these variables:

$$\text{Cov}(f(x), g(y)) = \mathbb{E} [(f(x) - \mathbb{E}[f(x)]) (g(y) - \mathbb{E}[g(y)])]. \quad (3.13)$$

High absolute values of the covariance mean that the values change very much and are both far from their respective means at the same time. If the sign of the covariance is positive, then both variables tend to take on relatively high values simultaneously. If the sign of the covariance is negative, then one variable tends to take on a relatively high value at the times that the other takes on a relatively low value and vice versa. Other measures such as **correlation** normalize the contribution of each variable in order to measure only how much the variables are related, rather than also being affected by the scale of the separate variables.

The notions of covariance and dependence are related, but are in fact distinct concepts. They are related because two variables that are independent have zero covariance, and two variables that have non-zero covariance are dependent. However, independence is a distinct property from covariance. For two variables to have zero covariance, there must be no linear dependence between them. Independence is a stronger requirement than zero covariance, because independence also excludes nonlinear relationships. It is possible for two variables to be dependent but have zero covariance. For example, suppose we first sample a real number x from a uniform distribution over the interval $[-1, 1]$. We next sample a random variable

s . With probability $\frac{1}{2}$, we choose the value of s to be 1. Otherwise, we choose the value of s to be -1 . We can then generate a random variable y by assigning $y = sx$. Clearly, x and y are not independent, because x completely determines the magnitude of y . However, $\text{Cov}(x, y) = 0$.

The **covariance matrix** of a random vector $\mathbf{x} \in \mathbb{R}^n$ is an $n \times n$ matrix, such that

$$\text{Cov}(\mathbf{x})_{i,j} = \text{Cov}(x_i, x_j). \quad (3.14)$$

The diagonal elements of the covariance give the variance:

$$\text{Cov}(x_i, x_i) = \text{Var}(x_i). \quad (3.15)$$

3.9 Common Probability Distributions

Several simple probability distributions are useful in many contexts in machine learning.

3.9.1 Bernoulli Distribution

The **Bernoulli** distribution is a distribution over a single binary random variable. It is controlled by a single parameter $\phi \in [0, 1]$, which gives the probability of the random variable being equal to 1. It has the following properties:

$$P(x = 1) = \phi \quad (3.16)$$

$$P(x = 0) = 1 - \phi \quad (3.17)$$

$$P(x = x) = \phi^x (1 - \phi)^{1-x} \quad (3.18)$$

$$\mathbb{E}_x[x] = \phi \quad (3.19)$$

$$\text{Var}_x(x) = \phi(1 - \phi) \quad (3.20)$$

3.9.2 Multinoulli Distribution

The **multinoulli** or **categorical** distribution is a distribution over a single discrete variable with k different states, where k is finite.¹ The multinoulli distribution is

¹ “Multinoulli” is a term that was recently coined by Gustavo Lacerdo and popularized by [Murphy \(2012\)](#). The multinoulli distribution is a special case of the **multinomial** distribution. A multinomial distribution is the distribution over vectors in $\{0, \dots, n\}^k$ representing how many times each of the k categories is visited when n samples are drawn from a multinoulli distribution. Many texts use the term “multinomial” to refer to multinoulli distributions without clarifying that they refer only to the $n = 1$ case.

parametrized by a vector $\mathbf{p} \in [0, 1]^{k-1}$, where p_i gives the probability of the i -th state. The final, k -th state's probability is given by $1 - \mathbf{1}^\top \mathbf{p}$. Note that we must constrain $\mathbf{1}^\top \mathbf{p} \leq 1$. Multinoulli distributions are often used to refer to distributions over categories of objects, so we do not usually assume that state 1 has numerical value 1, etc. For this reason, we do not usually need to compute the expectation or variance of multinoulli-distributed random variables.

The Bernoulli and multinoulli distributions are sufficient to describe any distribution over their domain. They are able to describe any distribution over their domain not so much because they are particularly powerful but rather because their domain is simple; they model discrete variables for which it is feasible to enumerate all of the states. When dealing with continuous variables, there are uncountably many states, so any distribution described by a small number of parameters must impose strict limits on the distribution.

3.9.3 Gaussian Distribution

The most commonly used distribution over real numbers is the **normal distribution**, also known as the **Gaussian distribution**:

$$\mathcal{N}(x; \mu, \sigma^2) = \sqrt{\frac{1}{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x - \mu)^2\right). \quad (3.21)$$

See figure 3.1 for a plot of the density function.

The two parameters $\mu \in \mathbb{R}$ and $\sigma \in (0, \infty)$ control the normal distribution. The parameter μ gives the coordinate of the central peak. This is also the mean of the distribution: $\mathbb{E}[x] = \mu$. The standard deviation of the distribution is given by σ , and the variance by σ^2 .

When we evaluate the PDF, we need to square and invert σ . When we need to frequently evaluate the PDF with different parameter values, a more efficient way of parametrizing the distribution is to use a parameter $\beta \in (0, \infty)$ to control the **precision** or inverse variance of the distribution:

$$\mathcal{N}(x; \mu, \beta^{-1}) = \sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{1}{2}\beta(x - \mu)^2\right). \quad (3.22)$$

Normal distributions are a sensible choice for many applications. In the absence of prior knowledge about what form a distribution over the real numbers should take, the normal distribution is a good default choice for two major reasons.

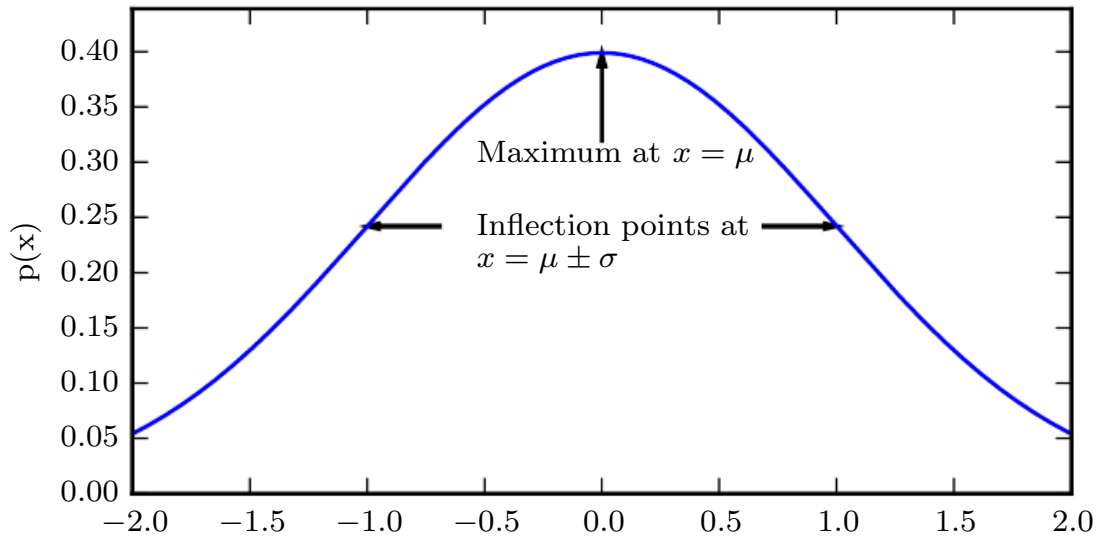


Figure 3.1: **The normal distribution:** The normal distribution $\mathcal{N}(x; \mu, \sigma^2)$ exhibits a classic “bell curve” shape, with the x coordinate of its central peak given by μ , and the width of its peak controlled by σ . In this example, we depict the **standard normal distribution**, with $\mu = 0$ and $\sigma = 1$.

First, many distributions we wish to model are truly close to being normal distributions. The **central limit theorem** shows that the sum of many independent random variables is approximately normally distributed. This means that in practice, many complicated systems can be modeled successfully as normally distributed noise, even if the system can be decomposed into parts with more structured behavior.

Second, out of all possible probability distributions with the same variance, the normal distribution encodes the maximum amount of uncertainty over the real numbers. We can thus think of the normal distribution as being the one that inserts the least amount of prior knowledge into a model. Fully developing and justifying this idea requires more mathematical tools, and is postponed to section 19.4.2.

The normal distribution generalizes to \mathbb{R}^n , in which case it is known as the **multivariate normal distribution**. It may be parametrized with a positive definite symmetric matrix Σ :

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sqrt{\frac{1}{(2\pi)^n \det(\boldsymbol{\Sigma})}} \exp \left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right). \quad (3.23)$$

The parameter $\boldsymbol{\mu}$ still gives the mean of the distribution, though now it is vector-valued. The parameter $\boldsymbol{\Sigma}$ gives the covariance matrix of the distribution. As in the univariate case, when we wish to evaluate the PDF several times for many different values of the parameters, the covariance is not a computationally efficient way to parametrize the distribution, since we need to invert $\boldsymbol{\Sigma}$ to evaluate the PDF. We can instead use a **precision matrix** $\boldsymbol{\beta}$:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\beta}^{-1}) = \sqrt{\frac{\det(\boldsymbol{\beta})}{(2\pi)^n}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\beta}(\mathbf{x} - \boldsymbol{\mu})\right). \quad (3.24)$$

We often fix the covariance matrix to be a diagonal matrix. An even simpler version is the **isotropic** Gaussian distribution, whose covariance matrix is a scalar times the identity matrix.

3.9.4 Exponential and Laplace Distributions

In the context of deep learning, we often want to have a probability distribution with a sharp point at $x = 0$. To accomplish this, we can use the **exponential distribution**:

$$p(x; \lambda) = \lambda \mathbf{1}_{x \geq 0} \exp(-\lambda x). \quad (3.25)$$

The exponential distribution uses the indicator function $\mathbf{1}_{x \geq 0}$ to assign probability zero to all negative values of x .

A closely related probability distribution that allows us to place a sharp peak of probability mass at an arbitrary point μ is the **Laplace distribution**

$$\text{Laplace}(x; \mu, \gamma) = \frac{1}{2\gamma} \exp\left(-\frac{|x - \mu|}{\gamma}\right). \quad (3.26)$$

3.9.5 The Dirac Distribution and Empirical Distribution

In some cases, we wish to specify that all of the mass in a probability distribution clusters around a single point. This can be accomplished by defining a PDF using the Dirac delta function, $\delta(x)$:

$$p(x) = \delta(x - \mu). \quad (3.27)$$

The Dirac delta function is defined such that it is zero-valued everywhere except 0, yet integrates to 1. The Dirac delta function is not an ordinary function that associates each value x with a real-valued output, instead it is a different kind of

mathematical object called a **generalized function** that is defined in terms of its properties when integrated. We can think of the Dirac delta function as being the limit point of a series of functions that put less and less mass on all points other than zero.

By defining $p(x)$ to be δ shifted by $-\mu$ we obtain an infinitely narrow and infinitely high peak of probability mass where $x = \mu$.

A common use of the Dirac delta distribution is as a component of an **empirical distribution**,

$$\hat{p}(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^m \delta(\mathbf{x} - \mathbf{x}^{(i)}) \quad (3.28)$$

which puts probability mass $\frac{1}{m}$ on each of the m points $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}$ forming a given dataset or collection of samples. The Dirac delta distribution is only necessary to define the empirical distribution over continuous variables. For discrete variables, the situation is simpler: an empirical distribution can be conceptualized as a multinoulli distribution, with a probability associated to each possible input value that is simply equal to the **empirical frequency** of that value in the training set.

We can view the empirical distribution formed from a dataset of training examples as specifying the distribution that we sample from when we train a model on this dataset. Another important perspective on the empirical distribution is that it is the probability density that maximizes the likelihood of the training data (see section 5.5).

3.9.6 Mixtures of Distributions

It is also common to define probability distributions by combining other simpler probability distributions. One common way of combining distributions is to construct a **mixture distribution**. A mixture distribution is made up of several component distributions. On each trial, the choice of which component distribution generates the sample is determined by sampling a component identity from a multinoulli distribution:

$$P(\mathbf{x}) = \sum_i P(c = i)P(\mathbf{x} \mid c = i) \quad (3.29)$$

where $P(c)$ is the multinoulli distribution over component identities.

We have already seen one example of a mixture distribution: the empirical distribution over real-valued variables is a mixture distribution with one Dirac component for each training example.

The mixture model is one simple strategy for combining probability distributions to create a richer distribution. In chapter 16, we explore the art of building complex probability distributions from simple ones in more detail.

The mixture model allows us to briefly glimpse a concept that will be of paramount importance later—the **latent variable**. A latent variable is a random variable that we cannot observe directly. The component identity variable c of the mixture model provides an example. Latent variables may be related to \mathbf{x} through the joint distribution, in this case, $P(\mathbf{x}, c) = P(\mathbf{x} | c)P(c)$. The distribution $P(c)$ over the latent variable and the distribution $P(\mathbf{x} | c)$ relating the latent variables to the visible variables determines the shape of the distribution $P(\mathbf{x})$ even though it is possible to describe $P(\mathbf{x})$ without reference to the latent variable. Latent variables are discussed further in section 16.5.

A very powerful and common type of mixture model is the **Gaussian mixture** model, in which the components $p(\mathbf{x} | c = i)$ are Gaussians. Each component has a separately parametrized mean $\boldsymbol{\mu}^{(i)}$ and covariance $\boldsymbol{\Sigma}^{(i)}$. Some mixtures can have more constraints. For example, the covariances could be shared across components via the constraint $\boldsymbol{\Sigma}^{(i)} = \boldsymbol{\Sigma}, \forall i$. As with a single Gaussian distribution, the mixture of Gaussians might constrain the covariance matrix for each component to be diagonal or isotropic.

In addition to the means and covariances, the parameters of a Gaussian mixture specify the **prior probability** $\alpha_i = P(c = i)$ given to each component i . The word “prior” indicates that it expresses the model’s beliefs about c *before* it has observed \mathbf{x} . By comparison, $P(c | \mathbf{x})$ is a **posterior probability**, because it is computed *after* observation of \mathbf{x} . A Gaussian mixture model is a **universal approximator** of densities, in the sense that any smooth density can be approximated with any specific, non-zero amount of error by a Gaussian mixture model with enough components.

Figure 3.2 shows samples from a Gaussian mixture model.

3.10 Useful Properties of Common Functions

Certain functions arise often while working with probability distributions, especially the probability distributions used in deep learning models.

One of these functions is the **logistic sigmoid**:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \quad (3.30)$$

The logistic sigmoid is commonly used to produce the ϕ parameter of a Bernoulli

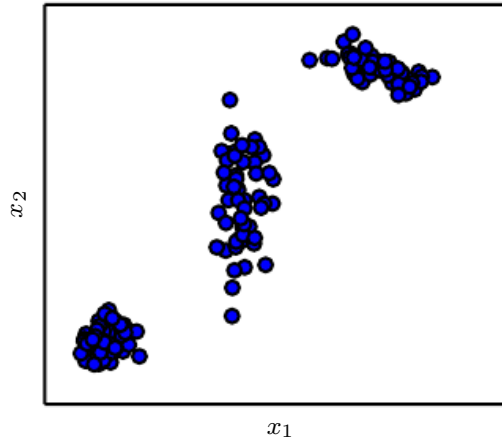


Figure 3.2: Samples from a Gaussian mixture model. In this example, there are three components. From left to right, the first component has an isotropic covariance matrix, meaning it has the same amount of variance in each direction. The second has a diagonal covariance matrix, meaning it can control the variance separately along each axis-aligned direction. This example has more variance along the x_2 axis than along the x_1 axis. The third component has a full-rank covariance matrix, allowing it to control the variance separately along an arbitrary basis of directions.

distribution because its range is $(0,1)$, which lies within the valid range of values for the ϕ parameter. See figure 3.3 for a graph of the sigmoid function. The sigmoid function **saturates** when its argument is very positive or very negative, meaning that the function becomes very flat and insensitive to small changes in its input.

Another commonly encountered function is the **softplus** function (Dugas *et al.*, 2001):

$$\zeta(x) = \log(1 + \exp(x)). \quad (3.31)$$

The softplus function can be useful for producing the β or σ parameter of a normal distribution because its range is $(0, \infty)$. It also arises commonly when manipulating expressions involving sigmoids. The name of the softplus function comes from the fact that it is a smoothed or “softened” version of

$$x^+ = \max(0, x). \quad (3.32)$$

See figure 3.4 for a graph of the softplus function.

The following properties are all useful enough that you may wish to memorize them:

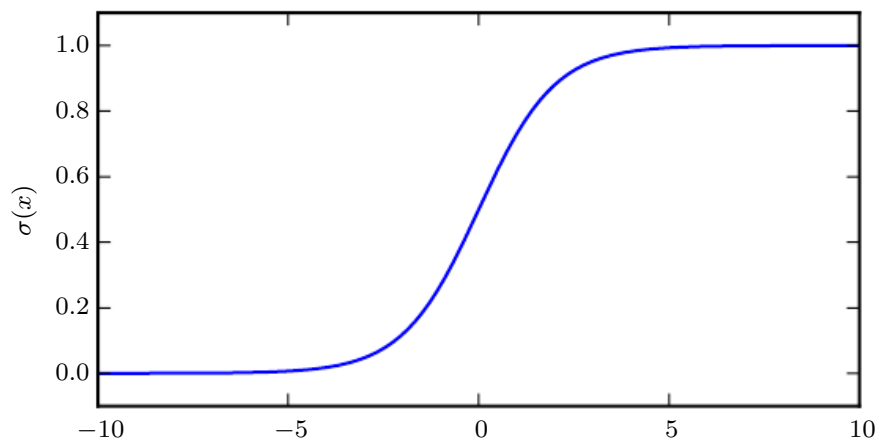


Figure 3.3: The logistic sigmoid function.

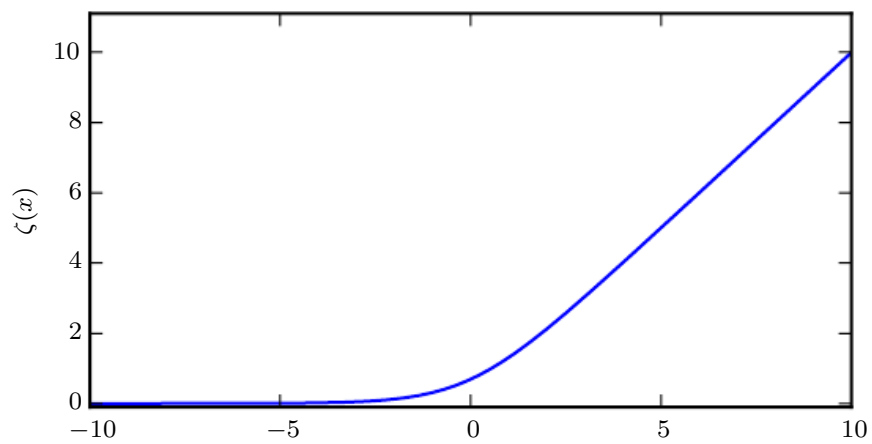


Figure 3.4: The softplus function.

$$\sigma(x) = \frac{\exp(x)}{\exp(x) + \exp(0)} \quad (3.33)$$

$$\frac{d}{dx}\sigma(x) = \sigma(x)(1 - \sigma(x)) \quad (3.34)$$

$$1 - \sigma(x) = \sigma(-x) \quad (3.35)$$

$$\log \sigma(x) = -\zeta(-x) \quad (3.36)$$

$$\frac{d}{dx}\zeta(x) = \sigma(x) \quad (3.37)$$

$$\forall x \in (0, 1), \sigma^{-1}(x) = \log\left(\frac{x}{1-x}\right) \quad (3.38)$$

$$\forall x > 0, \zeta^{-1}(x) = \log(\exp(x) - 1) \quad (3.39)$$

$$\zeta(x) = \int_{-\infty}^x \sigma(y)dy \quad (3.40)$$

$$\zeta(x) - \zeta(-x) = x \quad (3.41)$$

The function $\sigma^{-1}(x)$ is called the **logit** in statistics, but this term is more rarely used in machine learning.

Equation 3.41 provides extra justification for the name “softplus.” The softplus function is intended as a smoothed version of the **positive part** function, $x^+ = \max\{0, x\}$. The positive part function is the counterpart of the **negative part** function, $x^- = \max\{0, -x\}$. To obtain a smooth function that is analogous to the negative part, one can use $\zeta(-x)$. Just as x can be recovered from its positive part and negative part via the identity $x^+ - x^- = x$, it is also possible to recover x using the same relationship between $\zeta(x)$ and $\zeta(-x)$, as shown in equation 3.41.

3.11 Bayes’ Rule

We often find ourselves in a situation where we know $P(y | x)$ and need to know $P(x | y)$. Fortunately, if we also know $P(x)$, we can compute the desired quantity using **Bayes’ rule**:

$$P(x | y) = \frac{P(x)P(y | x)}{P(y)}. \quad (3.42)$$

Note that while $P(y)$ appears in the formula, it is usually feasible to compute $P(y) = \sum_x P(y | x)P(x)$, so we do not need to begin with knowledge of $P(y)$.

Bayes' rule is straightforward to derive from the definition of conditional probability, but it is useful to know the name of this formula since many texts refer to it by name. It is named after the Reverend Thomas Bayes, who first discovered a special case of the formula. The general version presented here was independently discovered by Pierre-Simon Laplace.

3.12 Technical Details of Continuous Variables

A proper formal understanding of continuous random variables and probability density functions requires developing probability theory in terms of a branch of mathematics known as **measure theory**. Measure theory is beyond the scope of this textbook, but we can briefly sketch some of the issues that measure theory is employed to resolve.

In section 3.3.2, we saw that the probability of a continuous vector-valued \mathbf{x} lying in some set \mathbb{S} is given by the integral of $p(\mathbf{x})$ over the set \mathbb{S} . Some choices of set \mathbb{S} can produce paradoxes. For example, it is possible to construct two sets \mathbb{S}_1 and \mathbb{S}_2 such that $p(\mathbf{x} \in \mathbb{S}_1) + p(\mathbf{x} \in \mathbb{S}_2) > 1$ but $\mathbb{S}_1 \cap \mathbb{S}_2 = \emptyset$. These sets are generally constructed making very heavy use of the infinite precision of real numbers, for example by making fractal-shaped sets or sets that are defined by transforming the set of rational numbers.² One of the key contributions of measure theory is to provide a characterization of the set of sets that we can compute the probability of without encountering paradoxes. In this book, we only integrate over sets with relatively simple descriptions, so this aspect of measure theory never becomes a relevant concern.

For our purposes, measure theory is more useful for describing theorems that apply to most points in \mathbb{R}^n but do not apply to some corner cases. Measure theory provides a rigorous way of describing that a set of points is negligibly small. Such a set is said to have **measure zero**. We do not formally define this concept in this textbook. For our purposes, it is sufficient to understand the intuition that a set of measure zero occupies no volume in the space we are measuring. For example, within \mathbb{R}^2 , a line has measure zero, while a filled polygon has positive measure. Likewise, an individual point has measure zero. Any union of countably many sets that each have measure zero also has measure zero (so the set of all the rational numbers has measure zero, for instance).

Another useful term from measure theory is **almost everywhere**. A property that holds almost everywhere holds throughout all of space except for on a set of

²The Banach-Tarski theorem provides a fun example of such sets.

measure zero. Because the exceptions occupy a negligible amount of space, they can be safely ignored for many applications. Some important results in probability theory hold for all discrete values but only hold “almost everywhere” for continuous values.

Another technical detail of continuous variables relates to handling continuous random variables that are deterministic functions of one another. Suppose we have two random variables, \mathbf{x} and \mathbf{y} , such that $\mathbf{y} = g(\mathbf{x})$, where g is an invertible, continuous, differentiable transformation. One might expect that $p_y(\mathbf{y}) = p_x(g^{-1}(\mathbf{y}))$. This is actually not the case.

As a simple example, suppose we have scalar random variables x and y . Suppose $y = \frac{x}{2}$ and $x \sim U(0, 1)$. If we use the rule $p_y(y) = p_x(2y)$ then p_y will be 0 everywhere except the interval $[0, \frac{1}{2}]$, and it will be 1 on this interval. This means

$$\int p_y(y)dy = \frac{1}{2}, \quad (3.43)$$

which violates the definition of a probability distribution. This is a common mistake. The problem with this approach is that it fails to account for the distortion of space introduced by the function g . Recall that the probability of \mathbf{x} lying in an infinitesimally small region with volume $\delta\mathbf{x}$ is given by $p(\mathbf{x})\delta\mathbf{x}$. Since g can expand or contract space, the infinitesimal volume surrounding \mathbf{x} in \mathbf{x} space may have different volume in \mathbf{y} space.

To see how to correct the problem, we return to the scalar case. We need to preserve the property

$$|p_y(g(x))dy| = |p_x(x)dx|. \quad (3.44)$$

Solving from this, we obtain

$$p_y(y) = p_x(g^{-1}(y)) \left| \frac{\partial x}{\partial y} \right| \quad (3.45)$$

or equivalently

$$p_x(x) = p_y(g(x)) \left| \frac{\partial g(x)}{\partial x} \right|. \quad (3.46)$$

In higher dimensions, the derivative generalizes to the determinant of the **Jacobian matrix**—the matrix with $J_{i,j} = \frac{\partial x_i}{\partial y_j}$. Thus, for real-valued vectors \mathbf{x} and \mathbf{y} ,

$$p_x(\mathbf{x}) = p_y(g(\mathbf{x})) \left| \det \left(\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \right) \right|. \quad (3.47)$$

3.13 Information Theory

Information theory is a branch of applied mathematics that revolves around quantifying how much information is present in a signal. It was originally invented to study sending messages from discrete alphabets over a noisy channel, such as communication via radio transmission. In this context, information theory tells how to design optimal codes and calculate the expected length of messages sampled from specific probability distributions using various encoding schemes. In the context of machine learning, we can also apply information theory to continuous variables where some of these message length interpretations do not apply. This field is fundamental to many areas of electrical engineering and computer science. In this textbook, we mostly use a few key ideas from information theory to characterize probability distributions or quantify similarity between probability distributions. For more detail on information theory, see [Cover and Thomas \(2006\)](#) or [MacKay \(2003\)](#).

The basic intuition behind information theory is that learning that an unlikely event has occurred is more informative than learning that a likely event has occurred. A message saying “the sun rose this morning” is so uninformative as to be unnecessary to send, but a message saying “there was a solar eclipse this morning” is very informative.

We would like to quantify information in a way that formalizes this intuition. Specifically,

- Likely events should have low information content, and in the extreme case, events that are guaranteed to happen should have no information content whatsoever.
- Less likely events should have higher information content.
- Independent events should have additive information. For example, finding out that a tossed coin has come up as heads twice should convey twice as much information as finding out that a tossed coin has come up as heads once.

In order to satisfy all three of these properties, we define the **self-information** of an event $x = x$ to be

$$I(x) = -\log P(x). \quad (3.48)$$

In this book, we always use \log to mean the natural logarithm, with base e . Our definition of $I(x)$ is therefore written in units of **nats**. One nat is the amount of

information gained by observing an event of probability $\frac{1}{e}$. Other texts use base-2 logarithms and units called **bits** or **shannons**; information measured in bits is just a rescaling of information measured in nats.

When x is continuous, we use the same definition of information by analogy, but some of the properties from the discrete case are lost. For example, an event with unit density still has zero information, despite not being an event that is guaranteed to occur.

Self-information deals only with a single outcome. We can quantify the amount of uncertainty in an entire probability distribution using the **Shannon entropy**:

$$H(x) = \mathbb{E}_{x \sim P}[I(x)] = -\mathbb{E}_{x \sim P}[\log P(x)]. \quad (3.49)$$

also denoted $H(P)$. In other words, the Shannon entropy of a distribution is the expected amount of information in an event drawn from that distribution. It gives a lower bound on the number of bits (if the logarithm is base 2, otherwise the units are different) needed on average to encode symbols drawn from a distribution P . Distributions that are nearly deterministic (where the outcome is nearly certain) have low entropy; distributions that are closer to uniform have high entropy. See figure 3.5 for a demonstration. When x is continuous, the Shannon entropy is known as the **differential entropy**.

If we have two separate probability distributions $P(x)$ and $Q(x)$ over the same random variable x , we can measure how different these two distributions are using the **Kullback-Leibler (KL) divergence**:

$$D_{\text{KL}}(P\|Q) = \mathbb{E}_{x \sim P} \left[\log \frac{P(x)}{Q(x)} \right] = \mathbb{E}_{x \sim P} [\log P(x) - \log Q(x)]. \quad (3.50)$$

In the case of discrete variables, it is the extra amount of information (measured in bits if we use the base 2 logarithm, but in machine learning we usually use nats and the natural logarithm) needed to send a message containing symbols drawn from probability distribution P , when we use a code that was designed to minimize the length of messages drawn from probability distribution Q .

The KL divergence has many useful properties, most notably that it is non-negative. The KL divergence is 0 if and only if P and Q are the same distribution in the case of discrete variables, or equal “almost everywhere” in the case of continuous variables. Because the KL divergence is non-negative and measures the difference between two distributions, it is often conceptualized as measuring some sort of distance between these distributions. However, it is not a true distance measure because it is not symmetric: $D_{\text{KL}}(P\|Q) \neq D_{\text{KL}}(Q\|P)$ for some P and Q . This

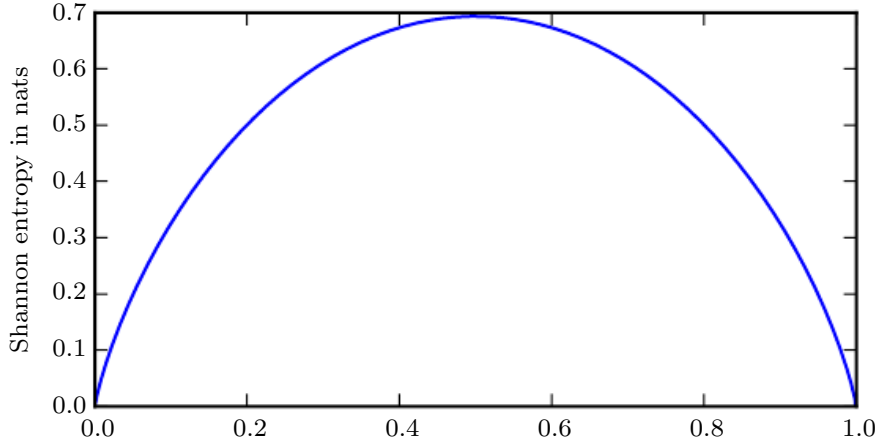


Figure 3.5: This plot shows how distributions that are closer to deterministic have low Shannon entropy while distributions that are close to uniform have high Shannon entropy. On the horizontal axis, we plot p , the probability of a binary random variable being equal to 1. The entropy is given by $(p-1)\log(1-p) - p\log p$. When p is near 0, the distribution is nearly deterministic, because the random variable is nearly always 0. When p is near 1, the distribution is nearly deterministic, because the random variable is nearly always 1. When $p = 0.5$, the entropy is maximal, because the distribution is uniform over the two outcomes.

asymmetry means that there are important consequences to the choice of whether to use $D_{\text{KL}}(P\|Q)$ or $D_{\text{KL}}(Q\|P)$. See figure 3.6 for more detail.

A quantity that is closely related to the KL divergence is the **cross-entropy** $H(P, Q) = H(P) + D_{\text{KL}}(P\|Q)$, which is similar to the KL divergence but lacking the term on the left:

$$H(P, Q) = -\mathbb{E}_{x \sim P} \log Q(x). \quad (3.51)$$

Minimizing the cross-entropy with respect to Q is equivalent to minimizing the KL divergence, because Q does not participate in the omitted term.

When computing many of these quantities, it is common to encounter expressions of the form $0 \log 0$. By convention, in the context of information theory, we treat these expressions as $\lim_{x \rightarrow 0} x \log x = 0$.

3.14 Structured Probabilistic Models

Machine learning algorithms often involve probability distributions over a very large number of random variables. Often, these probability distributions involve direct interactions between relatively few variables. Using a single function to

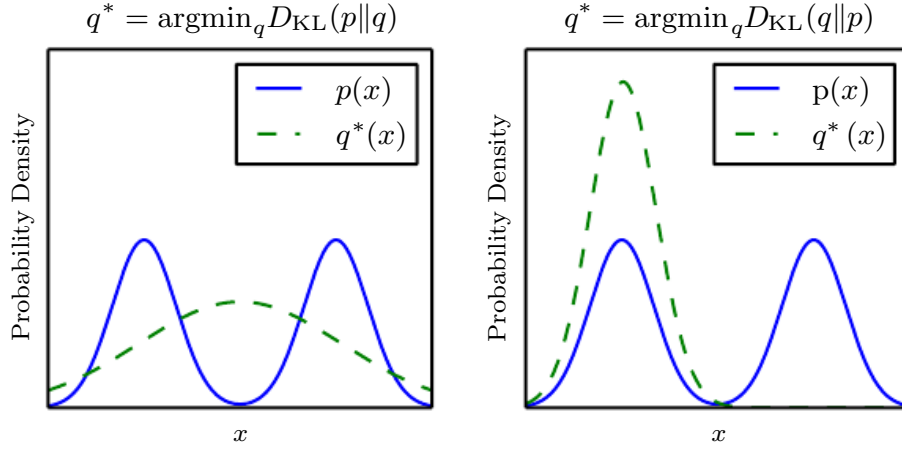


Figure 3.6: The KL divergence is asymmetric. Suppose we have a distribution $p(x)$ and wish to approximate it with another distribution $q(x)$. We have the choice of minimizing either $D_{\text{KL}}(p||q)$ or $D_{\text{KL}}(q||p)$. We illustrate the effect of this choice using a mixture of two Gaussians for p , and a single Gaussian for q . The choice of which direction of the KL divergence to use is problem-dependent. Some applications require an approximation that usually places high probability anywhere that the true distribution places high probability, while other applications require an approximation that rarely places high probability anywhere that the true distribution places low probability. The choice of the direction of the KL divergence reflects which of these considerations takes priority for each application. *(Left)* The effect of minimizing $D_{\text{KL}}(p||q)$. In this case, we select a q that has high probability where p has high probability. When p has multiple modes, q chooses to blur the modes together, in order to put high probability mass on all of them. *(Right)* The effect of minimizing $D_{\text{KL}}(q||p)$. In this case, we select a q that has low probability where p has low probability. When p has multiple modes that are sufficiently widely separated, as in this figure, the KL divergence is minimized by choosing a single mode, in order to avoid putting probability mass in the low-probability areas between modes of p . Here, we illustrate the outcome when q is chosen to emphasize the left mode. We could also have achieved an equal value of the KL divergence by choosing the right mode. If the modes are not separated by a sufficiently strong low probability region, then this direction of the KL divergence can still choose to blur the modes.

describe the entire joint probability distribution can be very inefficient (both computationally and statistically).

Instead of using a single function to represent a probability distribution, we can split a probability distribution into many factors that we multiply together. For example, suppose we have three random variables: a , b and c . Suppose that a influences the value of b and b influences the value of c , but that a and c are independent given b . We can represent the probability distribution over all three variables as a product of probability distributions over two variables:

$$p(a, b, c) = p(a)p(b | a)p(c | b). \quad (3.52)$$

These factorizations can greatly reduce the number of parameters needed to describe the distribution. Each factor uses a number of parameters that is exponential in the number of variables in the factor. This means that we can greatly reduce the cost of representing a distribution if we are able to find a factorization into distributions over fewer variables.

We can describe these kinds of factorizations using graphs. Here we use the word “graph” in the sense of graph theory: a set of vertices that may be connected to each other with edges. When we represent the factorization of a probability distribution with a graph, we call it a **structured probabilistic model** or **graphical model**.

There are two main kinds of structured probabilistic models: directed and undirected. Both kinds of graphical models use a graph \mathcal{G} in which each node in the graph corresponds to a random variable, and an edge connecting two random variables means that the probability distribution is able to represent direct interactions between those two random variables.

Directed models use graphs with directed edges, and they represent factorizations into conditional probability distributions, as in the example above. Specifically, a directed model contains one factor for every random variable x_i in the distribution, and that factor consists of the conditional distribution over x_i given the parents of x_i , denoted $Pa_{\mathcal{G}}(x_i)$:

$$p(\mathbf{x}) = \prod_i p(x_i | Pa_{\mathcal{G}}(x_i)). \quad (3.53)$$

See figure 3.7 for an example of a directed graph and the factorization of probability distributions it represents.

Undirected models use graphs with undirected edges, and they represent factorizations into a set of functions; unlike in the directed case, these functions

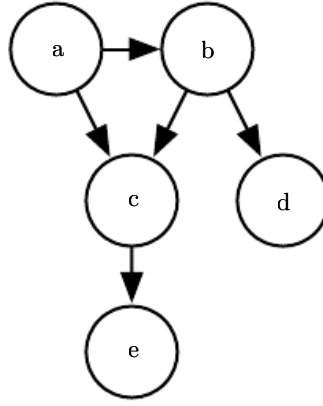


Figure 3.7: A directed graphical model over random variables a , b , c , d and e . This graph corresponds to probability distributions that can be factored as

$$p(a, b, c, d, e) = p(a)p(b | a)p(c | a, b)p(d | b)p(e | c). \quad (3.54)$$

This graph allows us to quickly see some properties of the distribution. For example, a and c interact directly, but a and e interact only indirectly via c .

are usually not probability distributions of any kind. Any set of nodes that are all connected to each other in \mathcal{G} is called a clique. Each clique $\mathcal{C}^{(i)}$ in an undirected model is associated with a factor $\phi^{(i)}(\mathcal{C}^{(i)})$. These factors are just functions, not probability distributions. The output of each factor must be non-negative, but there is no constraint that the factor must sum or integrate to 1 like a probability distribution.

The probability of a configuration of random variables is **proportional** to the product of all of these factors—assignments that result in larger factor values are more likely. Of course, there is no guarantee that this product will sum to 1. We therefore divide by a normalizing constant Z , defined to be the sum or integral over all states of the product of the ϕ functions, in order to obtain a normalized probability distribution:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_i \phi^{(i)}(\mathcal{C}^{(i)}). \quad (3.55)$$

See figure 3.8 for an example of an undirected graph and the factorization of probability distributions it represents.

Keep in mind that these graphical representations of factorizations are a language for describing probability distributions. They are not mutually exclusive families of probability distributions. Being directed or undirected is not a property of a probability distribution; it is a property of a particular **description** of a

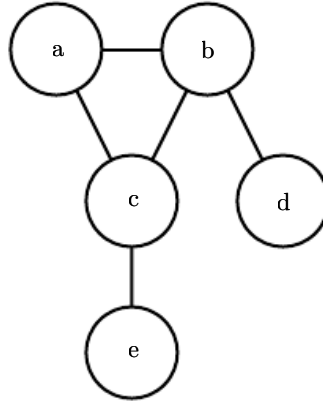


Figure 3.8: An undirected graphical model over random variables a , b , c , d and e . This graph corresponds to probability distributions that can be factored as

$$p(a, b, c, d, e) = \frac{1}{Z} \phi^{(1)}(a, b, c) \phi^{(2)}(b, d) \phi^{(3)}(c, e). \quad (3.56)$$

This graph allows us to quickly see some properties of the distribution. For example, a and c interact directly, but a and e interact only indirectly via c .

probability distribution, but any probability distribution may be described in both ways.

Throughout parts [I](#) and [II](#) of this book, we will use structured probabilistic models merely as a language to describe which direct probabilistic relationships different machine learning algorithms choose to represent. No further understanding of structured probabilistic models is needed until the discussion of research topics, in part [III](#), where we will explore structured probabilistic models in much greater detail.

This chapter has reviewed the basic concepts of probability theory that are most relevant to deep learning. One more set of fundamental mathematical tools remains: numerical methods.