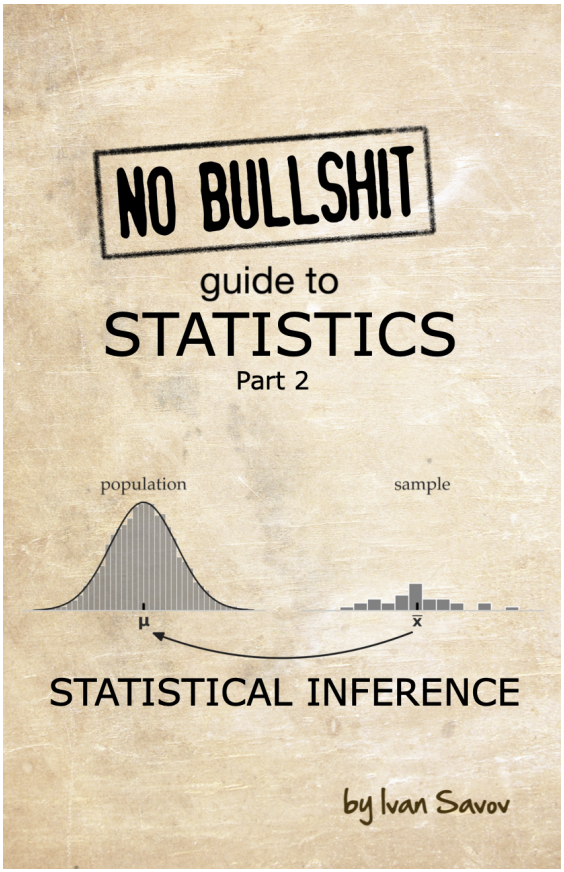
No Bullshit Guide to Statistics
Part 2: Statistical Inference

Extended book preview



The full book has 656 pages. This preview has been chosen to showcase the main ideas from Part 1 of the book and incudes chapter intros, key formulas, figures, and code examples. Get the eBook here: https://gum.co/noBSstats.

Contents

Statistical inference concept map

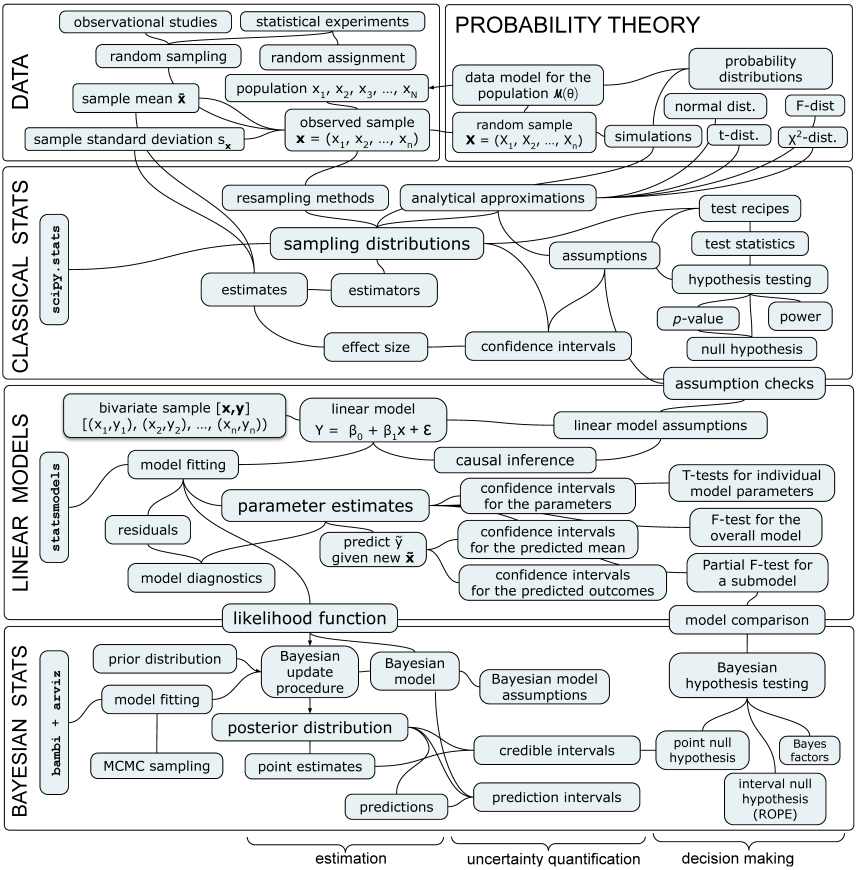


Figure 1: This concept map shows the statistical inference topics and concepts we'll learn in Part 2 of the book. We'll start with classical (frequentist) statistics in Chapter 3, then follow up with linear models in Chapter 4, and finally discuss Bayesian statistics in Chapter 5.

We'll see the same three statistical inference tasks come up in each chapter: *estimation* of model parameters, *quantification* of uncertainty in our estimates, and *decision-making* about unknown model parameters.

Introduction

Statistics is the systematic use of data and probability models to obtain knowledge. Historically, statistics was a specialized topic reserved for researchers and academics, but in the 21st century, statistical literacy is increasingly important for business people, technologists, and the general audience.

In Part 1 of the book, we learned about *descriptive statistics*, which allow us to summarize the properties of a particular data sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. Descriptive statistics techniques are relatively simple because we know all the relevant data and we're only summarizing and visualizing it.

We're often interested in making broader conclusions beyond a particular sample. We assume the sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is representative of some wider population $\{x_1, x_2, x_3, \ldots, x_N\}$, and we want to estimate the properties of this wider population based on the properties of the sample $\mathbf{x}$. The process of learning about a population based on a sample of observations from that population is called *inferential statistics*. Inferential statistics is inherently hard because the true population is unknown, and we only get a "glimpse" of it through the sample of observations.

The focus of Part 2 of the book is on learning statistical inference techniques that allow us to make precise statements about unknown populations based on samples taken from these populations.

Prerequisites for statistical inference

The data management and probability theory topics we learned in Part 1 of the book are key prerequisites for understanding statistics.

Data skills Doing statistics requires familiarity with data concepts like populations, samples, random selection, random assignment, and the *data generating process* that describes how samples from the population are obtained. Your practical experience with data

manipulation and data visualization from Chapter 1 will be useful for all the statistical analysis scenarios we'll see in this book.

Probability theory skills We assume that the unknown population is described by a *probability model*, which is a mathematical model for the variability of the values in the population. Your familiarity with the probability concepts from Chapter 2 and the practical experience with probability calculations will be essential for understanding the statistical inference topics we'll learn in Part 2 of the book.

Statistical inference

The starting point of the statistical inference process is a probability model for the population, which we'll denote $\mathcal{M}(\theta)$. We assume $\mathcal{M}$ is a known distribution family, and θ (the Greek letter *theta*) describes the unknown parameters that characterize the population we're studying. Probability theory tells us what kind of samples we might expect to observe from the population model $\mathcal{M}(\theta)$, as described by the forward arrow in Figure 2.

Inferential statistics studies the *inverse problem*: we're trying to reverse-engineer the data generating process, starting from the observed sample $\mathbf{x}$, and working backward to *infer* the population parameters θ that generated the sample $\mathbf{x}$, as illustrated by the backward arrow.

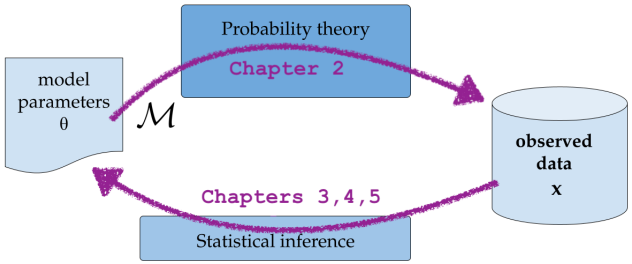


Figure 2: Probability theory describes how data samples are generated from the population data model $\mathcal{M}(\theta)$. Statistical inference studies the inverse problem: we start with the observed sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and we want to find the population model parameters θ that produced this sample.

There are many approaches to statistical inference, which we'll learn in chapters 3, 4, and 5. In each chapter, we'll build statistical models that specify the data generating process, then develop various techniques that work backward, to infer the model parameters θ that led to the observed data.

Statistical inference example

Consider a public health researcher who wants to study the weight distribution of men in a small city. The population consists of the weight measurements for the $N = 10\,000$ adult men in the city $\{w_1, w_2, w_3, \ldots, w_{10000}\}$. The researcher doesn't have the budget to collect weight measurements from all 10 000 individuals. Instead, she has selected $n = 30$ individuals at random from the population and obtained the sample of weights $\mathbf{w} = (w_1, w_2, \ldots, w_{30})$.

The researcher knows that biological measurements like height and weight follow a normal distribution. Based on this knowledge, she decides to model the weights as a normally distributed random variable $W \sim \mathcal{N}(\mu_W, \sigma_W)$, for some choice of the parameters μ_W and σ_W, as shown in Figure 3. The population mean μ_W (the Greek letter *mu*) describes the average weight in the population. The population standard deviation σ_W (the Greek letter *sigma*) measures the dispersion of the distribution of weights.

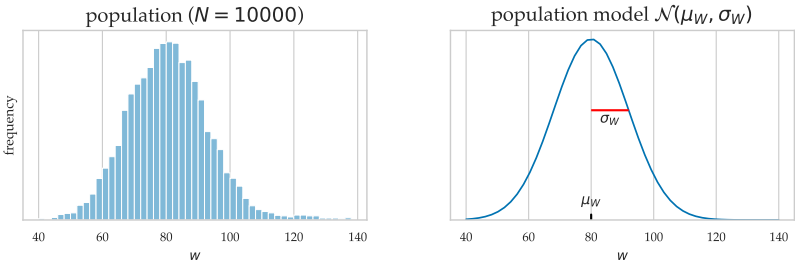


Figure 3: The distribution of weights in the population (left) can be approximated using the normal probability model $W \sim \mathcal{N}(\mu_W, \sigma_W)$ (right).

The histogram on the left shows the data of the whole population. This is the histogram she would obtain if she could measure the weights of all men in the city. The plot on the right shows the probability density function of the population model $W \sim \mathcal{N}(\mu_W, \sigma_W)$, which is an approximation to the real-world distribution of weights.

Figure 4 illustrates a typical statistical inference scenario. The forward arrow in the figure illustrates the random selection process that the researcher used to obtain the sample $\mathbf{w}$. The backward arrow shows the statistical inference that generalizes from the properties of the particular sample $\mathbf{w} = (w_1, w_2, \ldots, w_{30})$ to the properties of the population data model $W \sim \mathcal{N}(\mu_W, \sigma_W)$. Specifically, the researcher can use the sample mean $\overline{\mathbf{w}}$ as an estimate for the population mean μ_W, and the sample standard deviation $s_\mathbf{w}$ as an estimate for the population standard deviation σ_W.

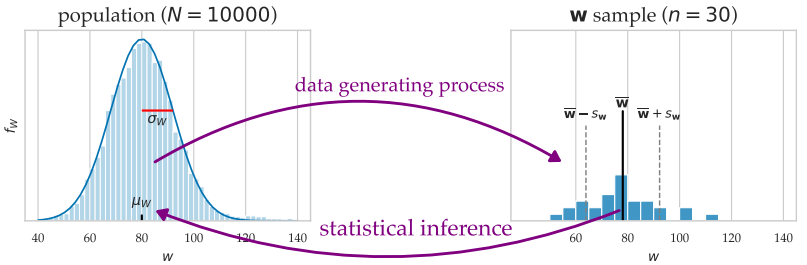


Figure 4: The goal of statistical inference is to estimate the unknown parameters of the population model $W \sim \mathcal{N}(\mu_W, \sigma_W)$, based on a sample of weights $\mathbf{w} = (w_1, w_2, \ldots, w_{30})$ selected at random from the population.

Statistical inference tasks

Statistical inference includes estimating unknown quantities, quantifying the uncertainty in our estimates, and making decisions using hypothesis testing. We'll now briefly describe each of these tasks.

Estimation

Estimation is the process of "guessing" the model parameters of the unknown population based on a sample from that population. For example, the sample mean $\overline{\mathbf{w}} = \mathbf{mean}(\mathbf{w})$ is an estimate for the population mean μ_W. Estimation is the fundamental statistical inference task on which all other tasks depend.

Uncertainty quantification

Using the sample mean as an estimate for the population mean is a useful approximation, $\overline{\mathbf{w}} \approx \mu_W$, but how accurate is this approximation? What is the uncertainty of the estimate $\overline{\mathbf{w}}$? All self-respecting statisticians quantify the uncertainty in the estimates they produce.

Instead of reporting a single number (a *point estimate*), we can report a range of numbers that we believe contains the unknown population parameter μ_W, which is called an *interval estimate*. For example, we can report a 90% confidence interval for the population mean, denoted $\mathbf{ci}_{\mu_W, 0.9} = [\mathbf{l}_{\mu_W}, \mathbf{u}_{\mu_W}]$, which specifies a lower bound $\mathbf{l}_{\mu_W}$ and an upper bound $\mathbf{u}_{\mu_W}$ on the population mean μ_W.

Hypothesis testing

Hypothesis testing is a standardized procedure for formulating and answering research questions. The main idea is to set up a

competition between two hypotheses: a research hypothesis that says that some pattern or trend in the data exists, and a *null hypothesis* that says that the pattern doesn't exist. The goal of the hypothesis testing procedure is to reach a yes-or-no decision whether the data shows enough evidence to *reject* the null hypothesis.

For example, suppose the public health researcher wants to test the claim that the bike lanes that the city added one year ago have improved the overall fitness of the population. Her theory is that the addition of the bike lanes allowed more citizens to bike to work instead of driving, which improved their fitness. If this theory is true, then we expect to see a reduction in the average weight of men in this city μ_W, relative to the average weight from five years ago $\mu_{W_0} = 92\,\text{kg}$, which is before the bike lanes were added. The researcher's hypothesis about the reduction in population weight can be written as $H_A : \mu_W < \mu_{W_0}$. In words, the hypothesis H_A says that the unknown population parameter μ_W is less than the constant $\mu_{W_0} = 92\,\text{kg}$. The null hypothesis is a skeptical counter-claim that the addition of the bike lanes made no difference to the average weight in the population $H_0 : \mu_W = \mu_{W_0}$. The researcher can use a hypothesis testing procedure based on the observed sample $\mathbf{w}$ to decide between the two hypotheses.

Hypothesis testing is used as the first step of many research projects to justify the basic plausibility of the research hypothesis, by showing that the "data pattern" is unlikely to be the result of a coincidence. Most research currently done in academia and industry uses the language of hypothesis testing, so you need to know how hypothesis testing works to interpret the results presented in research papers, or to write your own papers and reports.

The journey so far

The data management and probability theory topics we learned in Part 1 of the book were specifically designed to prepare you for the statistical inference topics we'll study in Part 2 of the book. You can think of Part 1 as a "prerequisites adapter" that ensures that you will have the necessary background knowledge to understand statistics.

This two-part book is my elaborate scheme to explain statistics in full detail: I don't want you to have any excuse to cop-out when it comes to more complicated statistical procedures and say "this is too complicated for me, I'll just use the formula without understanding it." Given your solid background in data and probability, you have the skills to handle *all* the details. Going back to the bike trip analogy, we could say you have the muscle and lung capacity required to make it to the top of the statistics mountains in Part 2 of the book.

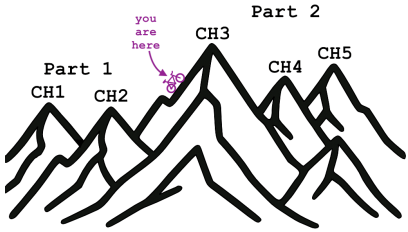


Figure 5: The bike trip in the statistics mountains continues. In Part 1, you learned the data and probability theory prerequisites, which prepared you for the big statistical inference uphills that you'll face in Part 2 of the book.

The journey ahead

We'll now briefly outline the statistical inference topics that we'll be discussing in the next three chapters. At the heart of each chapter, there is one key idea that we learned in the probability chapter. This means you're already ahead in the game: even if you're starting to learn about statistical inference, the mental effort you invested in Part 1 of the book has optimally prepared you to understand the topics discussed in Part 2 of the book.

Chapter 3: Classical statistics

We'll start with the core ideas of classical inferential statistics that form the basis of most STATS 101 courses. We'll define *estimators* (functions that take samples as inputs and produce estimates as outputs) and study their properties in Section 3.1. We'll then learn about confidence intervals (Section 3.2), and hypothesis testing (sections 3.3 through 3.7). In Section 3.8, we'll discuss important aspects of statistical practice, which you need to know to correctly apply statistical methods in the real world.

The key probability idea that enables the techniques we'll learn in Chapter 3 is the notion of a *sampling distribution*, which describes the variability of the estimates we obtain from random samples. Recall, we discussed the properties of random samples in Section 2.8, where we defined the sampling distribution of the mean (see page 263 in Part 1 of the book). We computed the sampling distribution of the mean by simulating thousands of samples and computing the mean of each sample (see code block 2.8.10 in Part 1, for example). We also learned about the *central limit theorem* (page 272 in Part 1), which provides an analytical formula for the sampling distribution of the mean. This prior experience with sampling distributions will make it easier for you to understand the probability calculations in Chapter 3, and help you fully comprehend the logic of statistical inference.

Chapter 4: Linear models

Linear models are an important family of statistical models with numerous practical applications. The simplest example of a linear model is the "best fit" line through a set of data points. We'll give a general introduction to linear models in Section 4.1, then extend the linear model idea to different contexts. We'll learn to fit linear models with multiple predictors (Section 4.2) and explain how to interpret model parameters (Section 4.3). We'll then discuss linear models with categorical variables (Section 4.4), and also talk about generalized linear models (Section 4.6). Section 4.5 includes an important discussion about *causality* and the challenges of discovering causal relationships between variables.

The key idea from Chapter 2 that enables the fitting of linear models is the *likelihood function*, which tells us the likelihood of different choices of parameters for a fixed dataset (see page 282 in Part 1). We define the "best fit" model parameters as the ones that have the highest likelihood of producing the observed data.

Chapter 5: Bayesian statistics

Bayesian statistics is an approach to statistical inference that treats unknown parameters as random variables rather than fixed numbers. In Bayesian statistics, we use probability distributions to represent our *knowledge* and *uncertainty* about unknown quantities. Bayesian inference is the process of updating our knowledge about the parameters based on observed data.

The Bayes' rule formula we saw in Chapter 2 (see page 146 and page 210 in Part 1 of the book) is the basis for all the statistical inference techniques we'll learn in Chapter 5. In Bayesian statistics, we start with some initial knowledge about the model parameters (the *prior* distribution), then apply Bayes' rule to update our knowledge to the *posterior* distribution, which represents our knowledge of the parameters after observing the data. The Bayesian "knowledge update" procedure is a unifying principle with applications in all kinds of statistical inference scenarios.

Applications of statistical inference

Before we jump into hundreds of pages on inferential statistics, we should say a few words about why knowing statistics is so useful. What are the real-world applications of inferential statistics?

In many real-world data analysis scenarios, it is impractical or impossible to collect measurements for the entire population, which contains thousands or millions of individuals. We're often limited to

working with samples from the population, which means we depend on inferential statistics to make statements about the population.

Statistics is used to make informed decisions in many areas of science, government, and business. All science papers and business reports are made more convincing when statistical data analysis is provided in support. Indeed, we could go as far as to say that statistical reasoning is a synonym for the "scientific method," since most scientific research involves some form of statistical analysis.

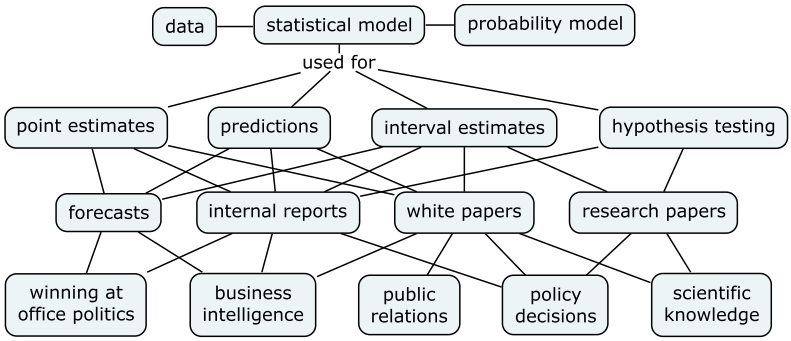


Figure 6: High-level concept map showing the applications of statistics.

Results in fields like medicine (clinical trials), genetics, pharmacology (drug safety, bioequivalence), nutrition, and other life sciences are all based on statistical analysis of sample data, with the goal of reaching general conclusions about a wider population. Similarly, research in psychology, biology, chemistry, physics, engineering, and computer science is all backed by statistical analysis.

Statistics is also used in government and policy circles. Governments collect data about demographics, the economy, weather, public health, and education, which helps them make informed decisions (supposedly). Statistical inference from samples to populations plays a key role here too, since it's not practical to always collect data on all citizens.

Statistics is becoming increasingly important in industry and business too. Domains like marketing, advertising, analytics, finance, insurance, and the tech sector all depend on the analysis of data for specific business objectives.

The common thread in all domains where statistics is used, is the need to estimate the properties of unknown populations and make informed decisions.

Computational statistics for the win

A unique advantage of this book is that we take a computational approach to statistics. In Part 1 of the book, we used Python for the data manipulation tasks and probability computations. In Part 2 of the book, we'll use Python for statistical inference procedures, which often involve multiple steps and probability calculations. The code description of statistical procedures provides a compact way to represent statistical concepts and procedures. Trust me, it's much easier to understand ten lines of Python code than ten pages of text and formulas!

We'll use simple Python based on the `numpy` and `scipy` libraries to illustrate how statistical calculations work from scratch, mirroring directly the explanations given in the text and formulas. We'll also show examples using high-level libraries like `statsmodels` and `bambi` that allow us to perform entire statistical analysis using one or two lines of code.

Real talk

Dear readers, I'm not going to lie to you and tell you learning statistics is going to be easy. Statistical inference is a hard subject that requires a substantial investment of "brain sweat" to learn and understand deeply. What I can promise you, though, is that the route I've chosen for our journey is a very smooth climb, introducing the complexity gradually along the way. Your existing knowledge of data and probability theory will enable you to learn statistical inference topics thoroughly, and even enjoy the process!

Ready? Grab a coffee and let's get started!

Chapter 3

Classical statistics

In this chapter, we'll describe the most common statistical inference techniques taught to university students and used by researchers to prepare reports and scientific papers. Classical statistics consists of various procedures for estimation, uncertainty quantification, and hypothesis testing derived from the properties of sampling distributions. The primary goal of the statistical techniques we'll learn in this chapter is to obtain reusable procedures for common data analysis scenarios.

Plan of attack

The topics we'll learn in this chapter have a reputation for being difficult to understand. Statistical inference requires **knowing about populations and samples**, formulating research questions, translating research questions into statistical questions, **doing probability calculations** to answer the statistical questions, and correctly reporting the results of the statistical analysis. That's a lot of moving parts, so we'd better have a plan for how we're going to approach this complexity.

The organization of this book is an elaborate scheme to make your first contact with classical statistics topics as smooth as possible. Part 1 of the book built up your knowledge of prerequisite concepts like populations and samples, and got you comfortable with doing probability calculations with random variables. You already know the parts shown in **bold** above, which is half the work.

This chapter will make learning inferential statistics easy by taking a computational approach to statistics procedures. Instead of looking at complicated stats formulas right away, like in most textbooks, we'll approach statistics calculations by running computer simulations. For example, to learn about sampling distributions

(Section 3.1), we can generate thousands of samples from the population, compute estimates from each sample, and visualize the results. Simulations and visualizations provide a direct, intuitive way to understand statistics concepts. We'll also present the math formulas, but they will no longer be difficult to understand since you're already familiar with the underlying concepts that the formulas describe. The book presents parallel (sometimes redundant) narratives based on text, math formulas, code examples, and visualizations to give you multiple ways to understand what's going on.

Don't get me wrong—statistics is still complicated, but I've done my best to make learning the topics as painless as possible. You'll still have to put in the effort to understand the concepts, but I'll try to minimize the moments of confusion, and I'll never make you memorize formulas without explaining them first. It's not going to be easy, but you can totally do it!

In this chapter, I'm going to show you how to …

- **generate** sampling distributions using simulation
- **approximate** sampling distributions using analytical formulas
- **construct** confidence intervals using the bootstrap and analytical formulas
- **design** statistical tests with desired error rates
- **choose** the appropriate statistical test for use in different situations
- **test** hypotheses using resampling and analytical methods
- **use** Python code for statistics calculations
- **identify** statistical modelling assumptions
- **interpret** p-values and confidence intervals correctly
- **report** statistical results honestly and objectively
- **assess** the limitations of statistical procedures
- **identify** common misconceptions about p-values and confidence intervals

3.1 Estimators

The term *estimator* is a fancy way of talking about **the function that we use to compute estimates** from samples. The *sampling distribution* of an estimator describes the variability of the estimates we might obtain from different samples. Estimates, estimators, and their sampling distributions are the main building blocks of classical inferential statistics, so this is where we'll start.

The goal of statistical inference is to use the properties of a sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ to estimate the parameters of the population from which the sample was collected. For example, we can use the sample mean $\overline{\mathbf{x}} = \mathbf{mean}(\mathbf{x})$ as an estimate of the population mean μ_X.

What are estimators used for?

Figure 3.1 shows some of the applications of estimators in statistics. In later sections of this chapter, we'll learn about *confidence intervals* and *hypothesis testing*, which are the two major topics in classical inferential statistics. We'll now describe these topics briefly, so you'll have an idea of the statistical calculations you'll be able to do using the math tools you'll learn in this section.

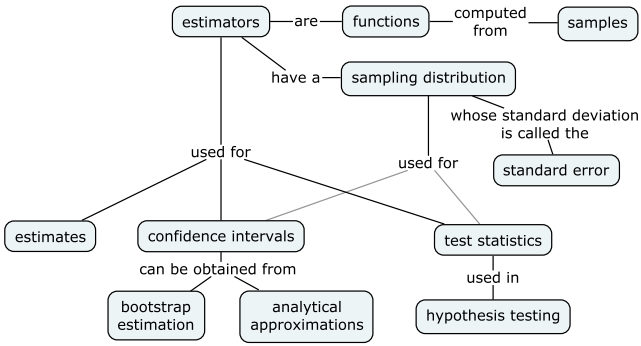


Figure 3.1: Overview of estimator concepts and their uses in statistics.

Confidence intervals A confidence interval describes the region of the parameter space that contains most of the probability mass. For example, a 90% confidence interval describes a region that contains 0.9 of the total weight of some distribution. We've already seen confidence intervals in Chapter 2 when we talked about the "bulk" of a distribution. Figure 2.52 (page 201) shows examples of intervals that contain 68.2%, 95.4%, and 99.7% proportions of the total probability. A confidence interval is a two-point summary of

a distribution that tells you the smallest and the largest values you might observe. In Section 3.2, we'll learn how to compute confidence intervals for unknown population parameters of interest.

Test statistics A *test statistic* is a particular type of estimate that is used as part of a *hypothesis testing* procedure. Examples of test statistics are the z-statistic (or z-score), the t-statistic, the χ^2-statistic (pronounced *khai square*), and the F-statistic. We compute a test statistic from a sample, and compare the observed value of the test statistic to the sampling distribution of a hypothetical population. The probability of the observed test statistic under the hypothetical sampling distribution is used as the basis of a decision rule, which tells us if we can "reject the null hypothesis." You'll learn more about hypothesis testing in sections 3.3, 3.4, 3.5, 3.6, and 3.7.

Estimators and sampling distributions are the essential math tools we use to construct confidence intervals and perform hypothesis tests. Getting to know the "math machinery" of estimators will make it easy for you to learn statistics, and give you a deep understanding of the power (and limitations) of statistical procedures.

Heads up, this section is the longest and the most cognitively demanding in this chapter. You'll be exposed to dozens of math formulas and code examples as part of the explanations. The good news is that **once you understand estimators and their sampling distributions, all the rest of the chapter will be comparatively easy**, since confidence intervals and hypothesis testing are straightforward calculations based on sampling distributions.

In order to make your journey through this section as smooth as possible, I've prepared some exercises for you. At the end of each section, you'll find exercises that ask you to do a math calculation, run a computer simulation, or generate a plot. These exercises will give you some hands-on practice with all the concepts explained in the text. I've already created a notebook containing the exercise questions for you, so you just need to continue the calculations. Visit this URL `bit.ly/est-exrc-nb` to get started.

3.1.1 Definitions

Statistical inference requires a combination of the data manipulation techniques we discussed in Chapter 1 and the probability modelling skills you developed in Chapter 2. This is why we invested so much energy in these prerequisite chapters—it was the necessary setup so that we can do statistics right!

It's been a while since we discussed DATA and PROB topics, so let's review the definitions of the relevant concepts that we need to "import" from the prerequisites chapters in Part 1 of the book.

Review of data concepts

- *Population*: the entire group of individuals we're interested in. We generally assume we can't collect measurements for the entire population, because it consists of thousands or millions of individuals.
- *Sample* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$: a sequence of n observations from the population. Each x_i corresponds to a measurement from one individual randomly selected from the population.
- *Sample statistic*: any quantity computed from the sample $\mathbf{x}$.
- *Descriptive statistics*: statistics that describe or summarize the characteristics of the sample $\mathbf{x}$. Examples of descriptive statistics include the sample mean $\overline{\mathbf{x}}$, the sample variance $s_{\mathbf{x}}^2$, and the sample standard deviation $s_{\mathbf{x}}$. Table 3.1 lists all the descriptive statistics that we learned about in Section 1.3.

Statistic	Name	Measurement of
$\overline{\mathbf{x}} = \mathbf{mean}(\mathbf{x})$	mean	Central tendency
$s_{\mathbf{x}}^2 = \mathbf{var}(\mathbf{x})$	variance	Dispersion
$s_{\mathbf{x}} = \mathbf{std}(\mathbf{x})$	standard deviation	Dispersion
$\mathbf{min}(\mathbf{x})$	minimum	Position
$\mathbf{Q}_1(\mathbf{x})$	first quartile	Position
$\mathbf{Q}_2(\mathbf{x}) = \mathbf{med}(\mathbf{x})$	median	Central tendency
$\mathbf{Q}_3(\mathbf{x})$	third quartile	Position
$\mathbf{P}_{90}(\mathbf{x})$	90^{th} percentile	Position
$\mathbf{max}(\mathbf{x})$	maximum	Position

Table 3.1: Examples of descriptive statistics computed from a sample $\mathbf{x}$.

The goal of the descriptive statistics we learned in Chapter 1 was limited to obtaining *numerical summaries* of the observed sample $\mathbf{x}$, without making any generalizations about a wider population.

The goal of *statistical inference* is to use the properties of the sample $\mathbf{x}$ to make generalizations about the properties of the population from which the sample was collected. Specifically, we'll make inferences (educated guesses) about the *parameters* of the probability model for the population.

Review of probability concepts

Recall the numerous probability distributions that we introduced in sections 2.3 and 2.6. This "inventory" of probability models includes distributions like the uniform, Poisson, exponential, normal, etc. We can use these probability distributions to describe the variability of real-world populations of interest. Let's review the main concepts and the notation we used to describe probability models:

- $X \sim \mathcal{M}(\theta)$: the *population probability model* describes the variability of the data in the population as a random variable X with probability distribution f_X.
- $\mathcal{M}$: the *model family* describes the general "shape" of the distribution. Examples of probability model families include the uniform $\mathcal{U}$, the normal $\mathcal{N}$, and the exponential Expon.
- θ: the *model parameters* describe the specific population that we're interested in. Examples of model parameters: α and β for uniform models, μ and σ for normal models, and λ for Poisson and exponential models.
- `rvX`: the *computer model* of the random variable $X \sim \mathcal{M}(\theta)$, constructed as an instance of one of the predefined models from `scipy.stats`. Table 2.3 (see page 182) lists all the methods available on `scipy.stats` objects.
- *Random sample* $\mathbf{X} = (X_1, X_2, \ldots, X_n)$: a sequence of n independent instances of the random variable $X \sim \mathcal{M}(\theta)$.

In this book, we'll focus on *parametric inference* procedures, which assume that the probability model family $\mathcal{M}$ for the population is known, and we want to estimate the unknown model parameters θ, which characterize the specific population that we're interested in. We'll also describe some *nonparametric inference* techniques in Section 3.7, which don't make any assumptions about the population model family $\mathcal{M}$, but we'll dedicate much less time to them.

For the next hundred pages, when we talk about *statistical inference*, you can assume we're talking about the process of "guessing" the model parameters θ of the population model $\mathcal{M}(\theta)$ that best describe the observed sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. See Figure 4 (page 4) for a high-level description of the statistical inference process.

New estimator concepts

Building on the DATA and PROB definitions, we can now introduce the notation and terminology around estimators, which are the main building blocks of the STATS chapter:

- *Estimate* $\hat{\theta}$: a statistic computed from the sample $\mathbf{x}$ for the purpose of making inferences about a population parameter θ. For example, the sample mean $\bar{\mathbf{x}}$ is an estimate of the population mean μ_X.
- *Estimator* $g : \mathcal{X}^n \to \mathbb{R}$: the function that we use to compute the estimate $\hat{\theta}$ from a given sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. We denote this as $\hat{\theta} \stackrel{\text{def}}{=} g(\mathbf{x})$. For example, the estimator **mean** is defined as the function $\textbf{mean}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} x_i$. We use the estimator **mean** to compute the sample mean estimate $\bar{\mathbf{x}}$. Table 3.2 shows examples of estimates, estimators, and the corresponding population parameters they estimate.
- The *sampling distribution* of the estimator g is denoted $\widehat{\Theta} \stackrel{\text{def}}{=} g(\mathbf{X})$ and describes the estimates we can expect to observe from random samples from the population $\mathbf{X} = (X_1, X_2, \ldots, X_n)$. For example, the sampling distribution of the mean is the random variable $\overline{\mathbf{X}} \stackrel{\text{def}}{=} \textbf{mean}(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} X_i$.
- The *standard error* of an estimator g is denoted $\mathbf{se}_{\hat{\theta}}$ and describes the standard deviation of the estimator's sampling distribution. For example, the standard error of the mean is written $\mathbf{se}_{\bar{\mathbf{x}}}$.

Table 3.2 lists all the estimators that we'll discuss in this section. When describing general results that apply to all estimators, we'll use the notation g to describe the estimator, which is a generic name often used to describe functions in math. We'll denote the population parameter as θ, and denote the estimate computed from a sample as $\hat{\theta} = g(\mathbf{x})$. We use the "hat" notation to denote estimates, since it gives us a hint that $\hat{\theta}$ is an approximation to θ.

Estimate $\hat{\theta}$	Estimator g	Population parameter θ
$\bar{\mathbf{x}}$	**mean**	μ_X (mean)
$s^2_{\mathbf{x}}$	**var**	σ^2_X (variance)
$s_{\mathbf{x}}$	**std**	σ_X (standard deviation)
$\hat{d} = \bar{\mathbf{x}} - \bar{\mathbf{y}}$	**dmeans**	$\Delta = \mu_X - \mu_Y$

Table 3.2: List of the estimates (numbers), estimators (functions used to compute estimates), and the population parameters that we want to infer.

The notation for the mean $\bar{\mathbf{x}}$, the variance $s^2_{\mathbf{x}}$, and the standard deviation $s_{\mathbf{x}}$, do not follow the "hat" convention for estimates, because we have already introduced the notation for these quantities in Chapter 1 and we prefer to reinforce the connections with your prior knowledge. If we wanted to follow the "hat" convention, we would write $\bar{\mathbf{x}}$ as $\hat{\mu}$, $s^2_{\mathbf{x}}$ as $\hat{\sigma}^2$, and $s_{\mathbf{x}}$ as $\hat{\sigma}$.

Speaking of prior knowledge, if you compare the estimators listed in Table 3.2 to the list of descriptive statistics from Table 3.1, you'll notice that you've seen most of them already! Indeed, the mean, the variance, and the standard deviation estimates are just a new way for talking about the sample statistics $\overline{\mathbf{x}}$, $s_{\mathbf{x}}^2$, and $s_{\mathbf{x}}$, which we introduced in Section 1.3. It's the same quantities, but used for different applications. Old ponies, new tricks.

Sampling distributions The sampling distribution of an estimator is the most important new concept you'll learn in this chapter. It's also a tricky concept, so you need to be ready for this, and persist in your efforts even if you find it difficult to understand.

Computationally speaking, the sampling distribution of the estimator g corresponds to the distribution of the estimates $\hat{\theta}_j = g(\mathbf{x}_j)$ we would obtain if we repeatedly generated thousands of samples $\mathbf{x}_j$ from the population model. Suppose we generate $N = 1000$ random samples $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, where each $\mathbf{x}_j$ is a sample of size n from the population. If we then compute the value of the estimator g from each of these samples, we'll obtain the list of estimates $[g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)]$. We can obtain an approximation of the sampling distribution of the estimator $g(\mathbf{X})$ by plotting a histogram of the N estimates we computed from the simulated samples. Look ahead to Figure 3.5 and Figure 3.6, which show histograms of sampling distributions of the **mean** and **std** produced in this way.

Mathematically speaking, the sampling distribution is the distribution of the random variable $\widehat{\Theta} = g(\mathbf{X})$, which is the estimate we obtain when the input to the estimator is a random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$. Specifically, the sampling distribution of the estimator g is the probability density function of the random variable $\widehat{\Theta}$, which we can denote as $f_{\widehat{\Theta}}$ or $f_{g(\mathbf{X})}$. Since the sampling distribution is described by a random variable $\widehat{\Theta}$, all the probability rules and formulas for working with random variables we learned in Chapter 2 also apply to calculations involving the sampling distribution $\widehat{\Theta}$. In other words, you'll have to learn about some weird new "hat quantities" in this section, but you already know all the math tools needed to work with them.

3.1.2 Estimates and estimators

Estimators are functions that take samples as inputs and produce estimates as outputs. We use the "hat" notation $\hat{\theta}$ to denote the estimate of the population parameter θ, and the name g to denote the

function that computes $\hat{\theta}$ from a given sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$:

$$\hat{\theta} \quad = \quad g(\mathbf{x}).$$

Figure 3.2 shows the context in which the estimate $\hat{\theta}$ is produced. We assume the population is described by the model $X \sim \mathcal{M}(\theta)$, where θ is an unknown population parameter. We have obtained a sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ that contains n observations selected at random from the population. We then obtain the estimate $\hat{\theta}$ for the population parameter θ by computing the output of the θ-estimating estimator function g on the input $\mathbf{x}$.

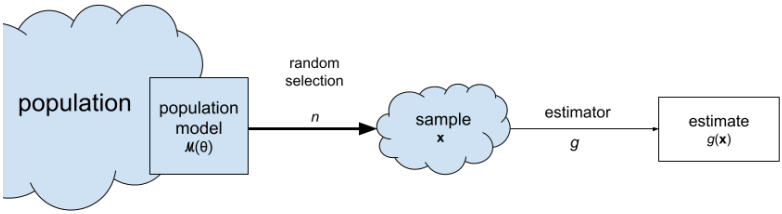


Figure 3.2: Computing the estimate $\hat{\theta} = g(\mathbf{x})$ based on a sample $\mathbf{x}$, which consists of n random observations from the population. If g is a good estimator, then the estimate $\hat{\theta}$ will be close to the population parameter θ.

For example, the sample mean estimator **mean** is the function that we use to compute the sample mean estimate $\overline{\mathbf{x}} = \mathbf{mean}(\mathbf{x})$ from a given sample $\mathbf{x}$, which is an approximation to the population mean μ_X. Table 3.2 lists the most important estimators we'll discuss in this chapter, and the population parameters they estimate.

We'll now go over each of these estimators to describe them as math formulas and also as Python functions.

Sample mean estimator

The sample mean estimator **mean** computes the average value of the sample $\mathbf{x}$:

$$\overline{\mathbf{x}} = \mathbf{mean}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} x_i.$$

The sample mean $\overline{\mathbf{x}}$ is an estimate for the population mean μ_X. Here is a Python function that computes the sample mean:

code 3.1.1
```
>>> def mean(sample):
        return sum(sample) / len(sample)
```

The code is very concise because we're using the Python built-in function `sum` to compute the sum $\sum_{i=1}^{n} x_i$, and the built-in function `len` to compute the sample size $n =$ `len(sample)`.

E3.3 Compute the difference between the means of the sleep `scores` for the doctors working in `rural` and `urban` locations in the doctors dataset `datasets/doctors.csv`.

Hint: Use the code `doctors[doctors["loc"]=="rur"]` to select the subset of the doctors working in a rural location.

* * *

The estimates we obtained from the above calculations are very useful, but they correspond to only half of the "deliverables" we expect from any statistical analysis. Every self-respecting statistician will report the *uncertainty* associated with all estimates they compute. Knowing the uncertainty associated with the estimates allows us to better interpret the results of statistical experiments. In order to describe the uncertainty in the estimates $\overline{\mathbf{a}}$, $s^2_{\mathbf{a}}$, $\hat{d}$ we computed above, we'll need to find the *sampling distribution* of the estimators **mean**, **var**, and **dmeans**, which is what we'll do next.

3.1.3 Sampling distributions

Sampling distributions are the essential building blocks for statistics, so it's important that you become proficient with the terminology, notation, formulas, and computational techniques associated with sampling distributions. Once you learn to describe the variability of estimates you can expect from random samples from the population, all the other statistics topics in this chapter will fall easily into place.

The *sampling distribution* of the estimator g describes the output of the estimator when the input is a *random sample* $\mathbf{X} = (X_1, X_2, \ldots, X_n)$:

$$\widehat{\Theta} = g(\mathbf{X}).$$

For example, the *sampling distribution of the mean* is $\overline{\mathbf{X}} = \mathbf{mean}(\mathbf{X}) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} X_i$. Note the formula for the sampling distribution of the mean is the same as the formula we use to compute the sample mean $\overline{\mathbf{x}} = \mathbf{mean}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} x_i$, but computed from a random sample $\mathbf{X}$ instead of a particular sample $\mathbf{x}$. When the input to an estimator is a random variable, its output is also a random variable, so we denote sampling distributions using capital letters, following the usual convention for denoting random variables.

The random variable $\widehat{\Theta} = g(\mathbf{X})$ describes the distribution of estimates we might observe from **all possible samples of size** n drawn from the population $X \sim \mathcal{M}(\theta)$. Figure 3.3 illustrates the generative process for the sampling distribution of the estimator g. Note the similarity between Figure 3.3 and Figure 3.2 (page 19).

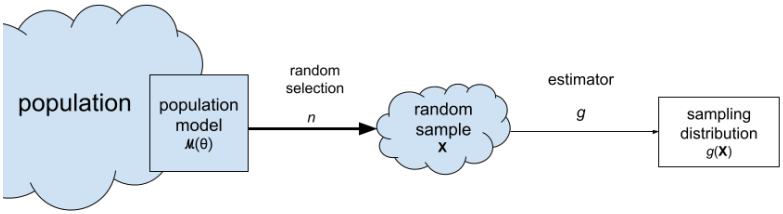


Figure 3.3: The *sampling distribution* of the estimator g is defined as the random variable $\widehat{\Theta} = g(\mathbf{X})$, which is the output of the estimator g when the input is a random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$, which consists of n independent instances of the random variable X.

Example 4: samples from a normal distribution

Let's obtain the sampling distribution of the mean, **mean** = `mean`, and the standard deviation, **std** = `std`, computed from random samples of size $n = 20$ taken from the standard normal distribution $Z \sim \mathcal{N}(0, 1)$.

We can start by generating $N = 10$ samples $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \ldots, \mathbf{z}_{10}$ of size $n = 20$ from $Z \sim \mathcal{N}(0, 1)$, and computing the mean and the standard deviation in each sample. Figure 3.4 shows the data from ten such samples. The diamond markers indicate the position of the sample means computed from each sample: $[\overline{\mathbf{z}}_1, \overline{\mathbf{z}}_2, \overline{\mathbf{z}}_3, \ldots, \overline{\mathbf{z}}_{10}]$. The sample standard deviations $[s_{\mathbf{z}_1}, s_{\mathbf{z}_2}, s_{\mathbf{z}_3}, \ldots, s_{\mathbf{z}_{10}}]$ are indicated by the vertical bars positioned relative to the mean in each sample.

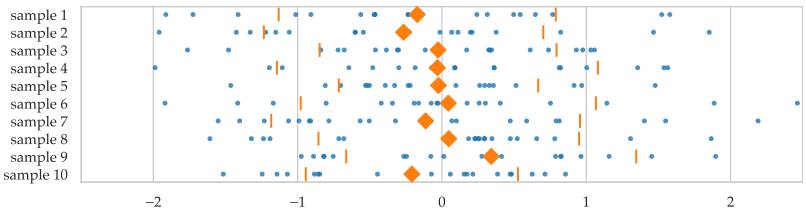


Figure 3.4: Strip plots from ten samples of size $n = 20$ from the random variable $Z \sim \mathcal{N}(0, 1)$. The sample means are indicated by the diamonds. The vertical bars denote the standard deviations computed from each sample.

Using visual inspection of the diamonds in Figure 3.4, we can start to get an idea about the variability of the means we can expect to observe for different random samples from the standard normal $Z \sim \mathcal{N}(0, 1)$. The diamonds seem to be all clustered around 0, which is the population mean $\mu_Z = 0$. Note also the vertical bars all seem to be roughly 1 unit away from the sample mean, which is consistent with the standard deviation of the population $\sigma_Z = 1$.

Sampling distribution of the mean Now imagine we generate 990 more samples to obtain a total of $N = 1000$ samples from the population model: $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \ldots, \mathbf{z}_{1000}$. We can visualize the sampling distribution of the mean $\overline{\mathbf{Z}} = \mathbf{mean}(\mathbf{Z})$ by plotting a histogram of the means computed from the 1000 random samples, $[\overline{\mathbf{z}}_1, \overline{\mathbf{z}}_2, \overline{\mathbf{z}}_3, \ldots, \overline{\mathbf{z}}_{1000}]$, where $\overline{\mathbf{z}}_j$ denotes the sample mean computed from the data in the j^{th} sample, $\overline{\mathbf{z}}_j = \mathbf{mean}(\mathbf{z}_j)$.

Figure 3.5 shows the sampling distribution of the mean computed from samples of size $n = 20$ from the standard normal. Each of the diamond shapes in the strip plot (bottom of the figure) corresponds to one of the sample means computed from the 1000 simulated samples. The histogram shows the "density of diamond shapes," and provides a representation of the sampling distribution of the mean $\overline{\mathbf{Z}} = \mathbf{mean}(\mathbf{Z})$.

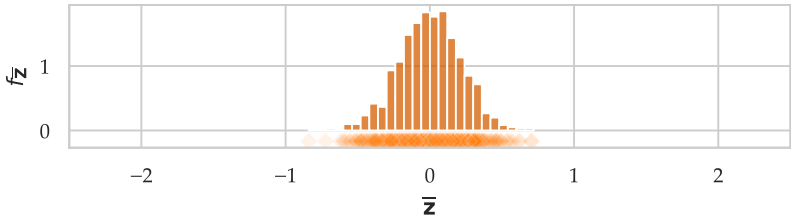


Figure 3.5: Sampling distribution of the mean $\overline{\mathbf{Z}} = \mathbf{mean}(\mathbf{Z})$. This histogram is an approximation to the probability density function $f_{\overline{\mathbf{Z}}}$, which describes the sampling distribution of the estimator **mean** = `mean` when computed from the random samples $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_{20})$ from the standard normal.

Sampling distribution of the standard deviation We can similarly obtain an approximation to the sampling distribution of the standard deviation $S_{\mathbf{Z}} = \mathbf{std}(\mathbf{Z})$ by plotting a histogram of the standard deviations computed from the 1000 simulated samples: $[s_{\mathbf{z}_1}, s_{\mathbf{z}_2}, s_{\mathbf{z}_3}, \ldots, s_{\mathbf{z}_{1000}}]$, where $s_{\mathbf{z}_j}$ denotes the standard deviation computed from the j^{th} sample: $s_{\mathbf{z}_j} = \mathbf{std}(\mathbf{z}_j) = \sqrt{\mathbf{var}(\mathbf{z}_j)}$.

I highly recommend that you take a *looooooong* look at the above figures and think carefully about what is going on. Sampling distributions are definitely not an obvious concept. Try visualizing how the distribution of diamonds we see in Figure 3.4 generates the histogram in Figure 3.5. Also, try explaining in your own words how the distribution of the vertical bars we see in Figure 3.4 generates the histogram shown in Figure 3.6.

* * *

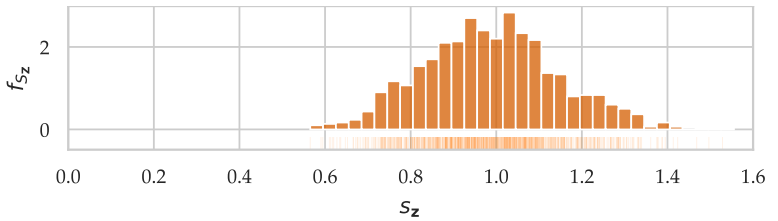


Figure 3.6: Sampling distribution of the standard deviation $S_{\mathbf{Z}} = \mathbf{std}(\mathbf{Z})$. The estimator $\mathbf{std} = \texttt{std}$ is computed from $N = 1000$ random samples of size $n = 20$ from the standard normal $Z \sim \mathcal{N}(0, 1)$.

If Chapter 3 were a movie, then sampling distributions would be the main actors. In this section, we're starting with some "character development" scenes, so you can get to know the main characters. In the remainder of the chapter, we'll watch a bunch of action scenes in which the main actors will appear in many kinds of statistics situations: all the supposedly complicated statistical concepts you've heard about will be revealed to be simple calculations based on sampling distributions.

Vocabulary for describing estimator properties

We'll now introduce some concepts for describing the properties of the estimator g, which are really the properties of its sampling distribution $\widehat{\Theta} = g(\mathbf{X})$.

- The *bias* of the estimator g measures the difference between the expected value of the estimator's sampling distribution and the true population parameter: $\mathbf{bias}(g) \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{X}}[\widehat{\Theta}] - \theta$.
- The *variance* of an estimator is $\mathbf{Var}(\widehat{\Theta}) = \mathbb{E}_{\mathbf{X}}\big[(\widehat{\Theta} - \mathbb{E}[\widehat{\Theta}])^2\big]$ and it measures the variability of the estimates around the average.
- The *standard error* of the estimator g is the square root of its variance and denoted $\mathbf{se}_{\hat{\theta}} = \sqrt{\mathbf{Var}(\widehat{\Theta})}$.

Figure 3.7 shows a visual representation of the concepts of bias and variance, through an analogy with accuracy and grouping properties of arrows hitting an archery target. The centre of the target corresponds to the true population parameter θ, and each arrow hit (denoted by the symbol ✸) corresponds to one estimate $\hat{\theta}_j$. In panel a), all the arrow hits are on target (low bias) and close together (low variance). This is the best-case scenario. In panel b), the arrow hits are still close to each other, which means the estimator has low variance, but the hits are consistently "off" the centre (below and to the right), which illustrates a biased estimator. Panel c) is an

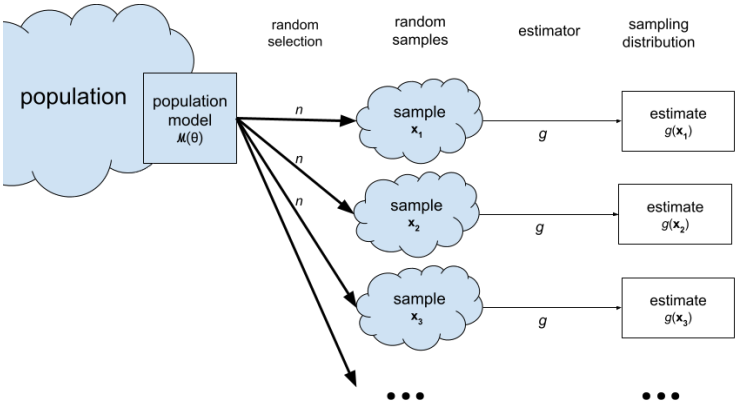


Figure 3.8: Sampling distribution of g can be obtained by repeatedly generating random samples $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$ and computing the values of the estimator from each sample $g(\mathbf{x}_1), g(\mathbf{x}_2), g(\mathbf{x}_3), \ldots$.

from the sampling distribution of the estimator `estfunc` computed from random samples of size `n` from the population model `rv`.

Recall we have already seen sampling distributions in Section 2.8, where we computed the sampling distribution of the mean **mean** = `mean` for samples of size $n = 30$ taken from the uniform, normal, and exponential probability models: $U \sim \mathcal{U}(0,1)$, $Z \sim \mathcal{N}(0,1)$, and $E \sim \text{Expon}(\lambda)$. Figure 2.81 on page 267 shows the sampling distribution $\overline{\mathbf{U}} = \mathbf{mean}(\mathbf{U})$. Figure 2.83 shows the sampling distributions of mean $\overline{\mathbf{Z}} = \mathbf{mean}(\mathbf{Z})$, where $\mathbf{Z}$ corresponds to random samples from the standard normal, and Figure 2.85 shows the sampling distribution of $\overline{\mathbf{E}} = \mathbf{mean}(\mathbf{E})$, for an exponentially distributed population described by the parameter $\lambda = 0.2$.

The function `gen_sampling_dist` allows us to generate observations from the sampling distribution of any estimator g, by passing in the estimator function g as the argument `estfunc`. In the next example, we'll see how to generate the sampling distributions of the estimators **mean** and **std** for samples from the kombucha volume population.

Example 5: normal population with known parameters

We want to generate the sampling distributions of the mean and the variance from samples from the normally distributed population $K \sim \mathcal{N}(\mu_K = 1000, \sigma_K = 10)$. This probability model describes the variability of the kombucha volume that goes into each bottle. Let's start by defining a computer model `rvK` that corresponds to the random variable K.

code
3.1.20

in the sample $\mathbf{k}_{02}$ = `ksample02`.

Bootstrap estimation

The *bootstrap* is a computational technique that allows us to obtain an approximation to the sampling distribution of any estimator using only a single sample of observations. The name "bootstrap" comes from the expression "pulling yourself up by your bootstraps," which describes a process that is performed without external help. In this case, we'll obtain the sampling distribution of the estimator without calling on external probability theory assumptions or analytical formulas, relying only on the data from a single sample of observations from the population.

The idea behind bootstrap estimation is to generate "new" samples from the population distribution by **reusing the observations from the sample we have**, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. Specifically, we generate bootstrap samples using **sampling with replacement** from the sample $\mathbf{x}$. Bootstrap estimation is an example of a *resampling procedure*, since it generates new observations based on an existing sample, treating the sample as if it was a population. The process of sampling with replacement can also be described as generating observations from the *empirical distribution* of the sample $\mathbf{x}$, which we denote $f_{\mathbf{x}}$. See Section 2.7.4 (page 242) if you want a reminder about empirical distributions, but don't worry about this terminology too much—it's just a fancy math way to talk about sampling with replacement.

We use an asterisk symbol to denote individual bootstrap observations. The value x^* corresponds to one of the observations from the list $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, selected at random. A *bootstrap sample* $\mathbf{x}^*$ is a sequence of n bootstrap observations:

$$\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*),$$

where each bootstrap observation x_i^* is obtained by sampling with replacement from the list of values $x_1, x_2, \ldots, x_n$ in the sample $\mathbf{x}$.

We can use the NumPy function `np.random.choice` to sample with replacement from a list of values. We'll illustrate how this works using a tiny sample that consists of four observations: $\mathbf{x} = (1, 2, 3, 4)$ = `sample`. To generate one bootstrap observation, we call the function `np.random.choice` passing in the sample as the first argument.

code 3.1.30

```
>>> xsample = [1,2,3,4]
>>> import numpy as np
>>> x_boot = np.random.choice(xsample)
>>> x_boot
3
```

To generate a bootstrap sample $\mathbf{x}^* =$ `bsample` of the same size as the original sample $\mathbf{x}$, we must specify the second argument to the function `np.random.choice`, which determines the `size` of the bootstrap sample we want to generate.

```
>>> bsample = np.random.choice(xsample, size=len(xsample))
>>> bsample
[3, 4, 1, 3]
```

code 3.1.31

Note the bootstrap sample `bsample` includes the observation 3 twice. This is to be expected since we're doing sampling with replacement. In contrast, the observation 2 did not get selected at all in this bootstrap sample.

The *bootstrap estimation* procedure is based on repeatedly generating thousands of bootstrap samples and calculating the sampling distribution of the estimator from these bootstrap samples. We generate a list of B bootstrap samples $\mathbf{x}_1^*, \mathbf{x}_2^*, \mathbf{x}_3^*, \ldots, \mathbf{x}_B^*$, where each $\mathbf{x}_j^*$ is a bootstrap sample of size n obtained from the original sample $\mathbf{x}$. To obtain the sampling distribution of the estimator g, we compute the *bootstrap estimates* from each of the bootstrap samples $\hat{\theta}_1^* = g(\mathbf{x}_1^*)$, $\hat{\theta}_2^* = g(\mathbf{x}_2^*)$, $\hat{\theta}_3^* = g(\mathbf{x}_3^*)$, $\ldots$, $\hat{\theta}_B^* = g(\mathbf{x}_B^*)$, and plot a histogram of these estimates. Generating $B = 5000$ bootstrap samples is usually sufficient to obtain a smooth histogram plot and accurate numerical results, so this is the default value we'll use for all the bootstrap estimation procedures in this book.

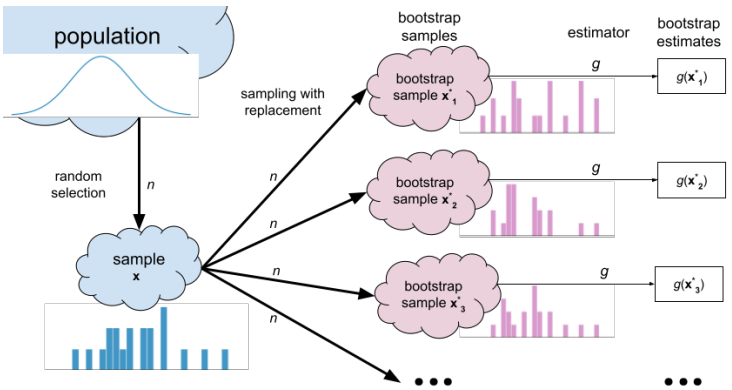


Figure 3.13: The bootstrap distribution of the estimator g is obtained by generating bootstrap samples $\mathbf{x}_1^*, \mathbf{x}_2^*, \mathbf{x}_3^*, \ldots$ from a single sample $\mathbf{x}$ and computing the bootstrap estimates $\hat{\theta}_1^* = g(\mathbf{x}_1^*), \hat{\theta}_2^* = g(\mathbf{x}_2^*), \hat{\theta}_3^* = g(\mathbf{x}_3^*), \ldots$.

Let's write a Python function for generating bootstrap estimates from the sampling distribution of the estimator function `estfunc` based on a `sample` of observations.

code 3.1.32

```
>>> def gen_boot_dist(sample, estfunc, B=5000):
        n = len(sample)
        bestimates = []
        for i in range(0, B):
            bsample = np.random.choice(sample, size=n)
            bestimate = estfunc(bsample)
            bestimates.append(bestimate)
        return bestimates
```

The function `gen_boot_dist` takes three arguments as inputs: the `sample` of observations, the estimator function `estfunc`, and the optional argument `B` that tells us how many bootstrap samples to generate. We start by creating the list `bestimates` where we'll store the bootstrap estimates. We then run a for-loop in which we repeatedly generate bootstrap samples $\mathbf{x}_j^*$ using `np.random.choice`, compute the bootstrap estimate for that sample $\hat{\theta}_j^* = \texttt{estfunc}(\mathbf{x}_j^*)$, and append the result to the end of the list `bestimates`. By the end the for-loop, the list `bestimates` will contain 5000 bootstrap observations of the estimator `estfunc`, which is an approximation to the sampling distribution of the estimator.

Note the logic behind the function `gen_boot_dist` is very similar to the function `gen_sampling_dist` (code block 3.1.19 on page 30) that we used to generate sampling distributions in the case when the population distribution is known. The bootstrap approach is essentially the same as the simulation approach, but instead of repeatedly generating new samples from the population, we're generating bootstrap samples from the sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. Essentially, we're treating the sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ as if it were a population.

The function `gen_boot_dist` is a general-purpose tool you can use to obtain the sampling distribution of any estimator—you just need to pass in the estimator function as the `estfunc` argument.

Example 6: bootstrap means To calculate the sampling distribution of the mean from the sample `ksample02`, we simply need to pass the estimator function `mean` as the `estfunc` argument.

code 3.1.33

```
>>> kbars_boot = gen_boot_dist(ksample02, estfunc=mean)
>>> sns.histplot(kbars_boot, stat="density")
See Figure 3.14.
```

Figure 3.14 shows the histogram of the bootstrap means `kbars_boot`. We'll use the suffix `_boot` to indicate quantities obtained using bootstrap estimation. The variability in `kbars_boot` comes from the random choices we made when generating the bootstrap samples. The conceptually simple idea of sampling with replacement within the existing sample turns out to produce a meaningful approximation of the variability of the sample mean estimator. We can

We can now compute the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}} = \frac{s_{\mathbf{k}}}{\sqrt{n}}$, which we'll denote as `sehat03` in the code (the `03`-suffix is a reminder that this estimate is computed from Batch 03).

code 3.1.41
```
>>> sehat03 = std(ksample03) / np.sqrt(7)
>>> sehat03
3.220066336410536
```

The standard error estimate $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}$ we obtained from `ksample03` is an underestimate of the true standard error, which is $\mathbf{se}_{\overline{\mathbf{k}}} = \frac{\sigma_K}{\sqrt{7}} = 3.78$.

Normal approximation to the sampling distribution

Let's see what happens when we replace the true standard error $\mathbf{se}_{\overline{\mathbf{k}}}$ with the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}$ in the central limit formula. We define a new random variable $N_{\overline{\mathbf{K}}}$ as follows:

$$\overline{\mathbf{K}} \approx N_{\overline{\mathbf{K}}} \quad \sim \quad \mathcal{N}\left(\mu_K, \widehat{\mathbf{se}}_{\overline{\mathbf{k}}}\right),$$

where $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}} = \frac{s_{\mathbf{k}}}{\sqrt{n}}$. The normal model $N_{\overline{\mathbf{K}}}$ corresponds to the plug-in approximation to the central limit theorem, and is an approximation to the sampling distribution of the mean, $\overline{\mathbf{K}} \approx N_{\overline{\mathbf{K}}}$. We can build a computer model `rvNKbar` that corresponds to the normal model, and compare its probability density function to the true sampling distribution `kbars7`, which we obtained earlier through simulation.

code 3.1.42
```
>>> sehat03 = std(ksample03) / np.sqrt(7)
>>> rvNKbar = norm(loc=muK, scale=sehat03)
>>> ax = sns.histplot(kbars7, stat="density", bins=100)
>>> plot_pdf(rvNKbar, ax=ax)
See Figure 3.17.
```

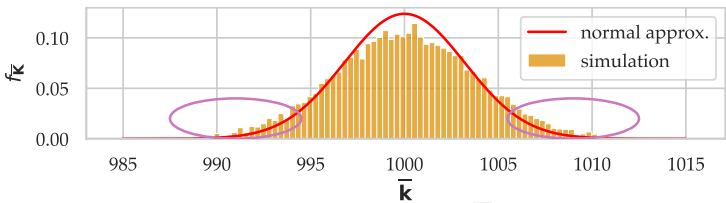


Figure 3.17: Sampling distribution of the mean $\overline{\mathbf{K}}$ obtained from samples of size $n = 7$ from the normal population $K \sim \mathcal{N}(\mu_K = 1000, \sigma_K = 10)$. The histogram shows the true distribution obtained from simulation. The line plot shows the probability density function of the normal approximation $N_{\overline{\mathbf{K}}} \sim \mathcal{N}\left(\mu_K, \widehat{\mathbf{se}}_{\overline{\mathbf{k}}}\right)$ based on the plug-in estimate for the standard error.

Figure 3.17 shows that the normal approximation $N_{\overline{\mathbf{K}}} \sim \mathcal{N}\left(\mu_K, \widehat{\mathbf{se}}_{\overline{\mathbf{k}}}\right)$ based on the plug-in estimate $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}$ is a very rough approximation to the true histogram of the true sampling distribution $\overline{\mathbf{K}}$. The normal

approximation is narrower than the true sampling distribution, and there are important differences in the tails of the distribution (highlighted by the two ellipses). The fact we're using the estimate $s_{\mathbf{k}}$ instead of the true value σ_K causes us to **underestimate the variance of the sampling distribution**.

In 1904, the statistician William Gosset noticed this tendency of the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}$ to underestimate the true standard error $\mathbf{se}_{\overline{\mathbf{k}}}$, and tried to find a way compensate for it.

A better approximation using Student's t-distribution

Gosset, working under the pseudonym Student, invented a new probability distribution that precisely compensates for the tendency of the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}$ to underestimate the true standard error $\mathbf{se}_{\overline{\mathbf{x}}}$, which we now call Student's t-distribution[Stu08].

Student's t-distribution, denoted $T \sim \mathcal{T}(\nu)$, is a "heavy-tailed" version of the standard normal distribution $Z \sim \mathcal{N}(0,1)$. The *degrees of freedom parameter* ν (the Greek letter *nu*) controls the "heaviness" of the tails. If the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}$ is computed from a sample of size n, then the relevant distribution is Student's t-distribution with $\nu = n - 1$ degrees of freedom.

Before we continue with the analysis of the sample `ksample03`, let's do a quick review of the `location` and `scale` parameters that allow us to shift and scale random variables. Consider the random variable T distributed according to the standard Student t-distribution $T \sim \mathcal{T}(\nu)$, and a second random variable T_2 obtained by scaling and shifting the location of the random variable T as follows: $T_2 = mT + b$. The random variables T and T_2 both have the "shape" of a t-distribution, but the standard deviation of T_2 will be m times larger than the standard deviation of T, and the mean of T_2 will be b units larger than the mean of T. Since this type of location-and-scale transformation is so common, we can include it in the model definition as follows $T_2 \sim \mathcal{T}(\texttt{df} = \nu, \texttt{loc} = b, \texttt{scale} = m)$, which is equivalent to writing $T_2 \sim m\mathcal{T}(\nu) + b$.

Using Student's t-distribution, we can describe the sampling distribution of the mean as the following approximation:

$$\overline{\mathbf{X}} \;\approx\; T_{\overline{\mathbf{X}}} \;\sim\; \mathcal{T}(\texttt{df} = \nu,\ \texttt{loc} = \mu_X,\ \texttt{scale} = \widehat{\mathbf{se}}_{\overline{\mathbf{x}}}),$$

or equivalently as

$$\overline{\mathbf{X}} \;\approx\; T_{\overline{\mathbf{X}}} \;\sim\; \underbrace{\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}}_{\texttt{scale}} \cdot \mathcal{T}(\nu) \;+\; \underbrace{\mu_X}_{\texttt{loc}},$$

where $\mathcal{T}(\nu)$ is Student's t-distribution with $\nu = n - 1$ degrees of freedom. See page 223 in Section 2.6 if you're interested in the

exact mathematical definition of Student's t-distribution. The math details are not important here, since we'll use the computer model `scipy.stats.t` for all the calculations below.

Note the actual quantity we're interested in is the sampling distribution of the mean $\overline{\mathbf{X}}$. The random variable $T_{\overline{\mathbf{X}}}$ is an approximation to the sampling distribution, which is why we write $T_{\overline{\mathbf{X}}} \approx \overline{\mathbf{X}}$.

Let's see Gosset's improved approximation formula in action! We define the random variable $T_{\overline{\mathbf{K}}}$ as a model for the sampling distribution of the mean $\overline{\mathbf{K}}$ constructed from the standard error estimate $\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}$ = `sehat03` = 3.22 computed from `ksample03`. The sample size is $n = 7$, so we need to use Student's t-distribution with $\nu = 6$ degrees of freedom:

$$\overline{\mathbf{K}} \;\approx\; T_{\overline{\mathbf{K}}} \;\sim\; \mathcal{T}(\texttt{df}=6, \texttt{loc}=\mu_K, \texttt{scale}=\widehat{\mathbf{se}}_{\overline{\mathbf{k}}}) \;=\; \widehat{\mathbf{se}}_{\overline{\mathbf{k}}} \cdot \mathcal{T}(6) + \mu_K.$$

We define a computer model `rvTKbar` that corresponds to the random variable $T_{\overline{\mathbf{K}}}$ and plot its probability density function superimposed on the histogram of the true sampling distribution.

code 3.1.43

```
>>> from scipy.stats import t as tdist
>>> sehat03 = std(ksample03) / np.sqrt(7)
>>> rvTKbar = tdist(df=7-1, loc=muK, scale=sehat03)
>>> ax = sns.histplot(kbars7, stat="density", bins=100)
>>> plot_pdf(rvTKbar, ax=ax, xlims=[985,1015], color="m")
See Figure 3.18.
```

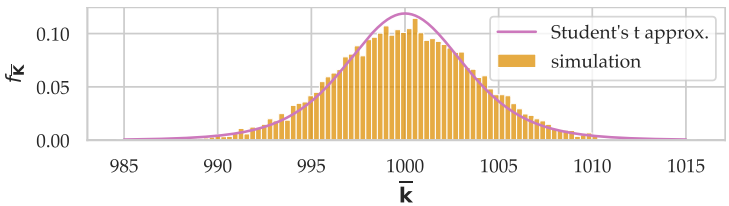


Figure 3.18: Approximation to the sampling distribution of the mean $\overline{\mathbf{K}}$ based on Student's t-distribution with $\nu = 6$ degrees of freedom.

Figure 3.18 shows the probability density plot of Student's t-distribution $T_{\overline{\mathbf{K}}}$ superimposed with a histogram of the true sampling distribution of the sample mean obtained through simulation. You can think of this figure a visual proof for the approximation $\overline{\mathbf{K}} \approx T_{\overline{\mathbf{K}}}$.

Note there is much better agreement in the tails as compared to the "naive" normal approximation we saw earlier in Figure 3.17. The "heaviness" in the tails of the t-distribution compensates for the fact that $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}$ is an underestimate of $\mathbf{se}_{\overline{\mathbf{x}}}$ and thus we obtain a better approximation for the sampling distribution.

3.1.9 Explanations

I want to now discuss some topics that I omitted from the main narrative, but which are important for your understanding of the material.

Computational approach to sampling distributions

Throughout this section, we used a mix of math descriptions (analytical formulas) and computational descriptions (simulations and bootstrap estimates). The math description of the sampling distribution of the estimator g for samples of size n from the population X is based on the concept of a random sample $\mathbf{X} = (X_1, \ldots, X_n)$. We obtain the computational description by generating $N = 10000$ samples $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, where each $\mathbf{x}_j$ is a sample of size n. Although the two descriptions seem very different on the surface, they both describe the same underlying concept: the variability we can expect to observe for random samples of size n from the model $X \sim \mathcal{M}(\theta)$. Here is a list of the key properties of the estimator g that highlights the correspondences between the two description types.

probability description		computational description
$\mathbf{X}$	$\leftrightarrow$	$[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$
$\widehat{\Theta} = g(\mathbf{X})$	$\leftrightarrow$	$[g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)]$
$\mathbb{E}[g(\mathbf{X})]$	$\leftrightarrow$	$\texttt{mean}([g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)])$
$\mathbf{var}(g(\mathbf{X}))$	$\leftrightarrow$	$\texttt{var}([g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)])$
$\mathbf{se}_g = \sqrt{\mathbf{var}(g(\mathbf{X}))}$	$\leftrightarrow$	$\texttt{std}([g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)])$

The two descriptions give us two different ways to visualize probability distributions and do probability calculations. For example, we can visualize the sampling distribution of the estimator g by plotting the probability density function $f_{\widehat{\Theta}}$ of the random variable $\widehat{\Theta} = g(\mathbf{X})$ or by plotting a histogram of the values $[g(\mathbf{x}_1), g(\mathbf{x}_2), \ldots, g(\mathbf{x}_N)]$. Indeed, if were to generate $N = \infty$ simulated samples, the shape of the histogram will become identical to the shape of the probability density function. In practice, we'll use $N = 5000$ or $N = 10000$ simulations, which provides enough precision to estimate probabilities, means, variances, standard errors, percentiles, and other properties we might be interested in.

Suppose we need to find the probability $\Pr(\{a \leqslant \widehat{\Theta} \leqslant b\})$, which describes the probability that the estimate computed from a random sample of size n will be between a and b. We can

3.2 Confidence intervals

Every self-respecting statistician reports the *uncertainty* of the estimates they compute. A *confidence interval* describes a lower bound and an upper bound on the possible values for the unknown parameter θ. For example, a 90% confidence interval for the unknown parameter θ is denoted $\mathbf{ci}_{\theta,0.9} = [\mathbf{l}_\theta, \mathbf{u}_\theta]$. We believe the unknown parameter θ falls inside this confidence interval, which we can write as the math statement $\theta \in \mathbf{ci}_{\theta,0.9}$, or equivalently as $\mathbf{l}_\theta \leqslant \theta \leqslant \mathbf{u}_\theta$. The "90% guarantee" is actually a statement about the reliability of the *procedure* that we used to construct the confidence interval, and not this particular interval $\mathbf{ci}_{\theta,0.9}$. It's a little tricky to explain in words, but I promise it will all make sense when you see the precise probability statements later on.

We can obtain such a 90% confidence interval by evaluating the position of the 5^{th} percentile and the 95^{th} percentile of the sampling distribution $\widehat{\Theta}$, followed by some math transformations (inverting a pivot), to obtain the values of the lower bound $\mathbf{l}_\theta$ and the upper bound $\mathbf{u}_\theta$.

In this section, we'll learn how to construct confidence intervals for the estimators that we studied in the previous section: **mean**, **var**, and **dmeans**. We'll continue the "parallel narratives" approach and describe both computational methods and analytical approximations for constructing confidence intervals. This will be a relatively easy section, since we already learned all the necessary math and computational techniques in the previous section.

3.2.1 Definitions

Review of estimators concepts

Let's start with a quick review of the terminology for estimators.

- $X \sim \mathcal{M}(\theta)$: the *population probability model* describes the variability of the data in the population X. We assume $\mathcal{M}$ is a known *model family* and θ are the unknown *model parameters* that characterize the specific population we're studying.
- *Sample* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$: a sequence of n observations from the population. Each x_i corresponds to a measurement from one individual randomly selected from the population.
- *Estimate* $\hat{\theta}$: a quantity computed from the sample $\mathbf{x}$ for the purpose of making inferences about the population parameter θ.
- *Estimator* $g : \mathcal{X}^n \to \mathbb{R}$: the function we use to compute the estimate $\hat{\theta}$ from a given sample $\mathbf{x}$. We write this as $\hat{\theta} \stackrel{\text{def}}{=} g(\mathbf{x})$.

- *Random sample* $\mathbf{X} = (X_1, X_2, \ldots, X_n)$: a sequence of n independent copies of the random variable $X \sim \mathcal{M}(\theta)$.
- The *sampling distribution* of the estimator g, denoted $\hat{\Theta} \stackrel{\text{def}}{=} g(\mathbf{X})$, describes the estimates we can expect to observe from random samples from the population.

Estimate $\hat{\theta}$	Estimator g	Population parameter θ	Pivotal quantity	Reference distribution
$\bar{\mathbf{x}}$	**mean**	μ_X	$T = \frac{\overline{\mathbf{X}}-\mu_X}{\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}}$	$\sim \mathcal{T}(n-1)$
$s^2_{\mathbf{x}}$	**var**	σ^2_X	$Q = \frac{(n-1)S^2_{\mathbf{X}}}{\sigma^2_X}$	$\sim \chi^2(n-1)$
$s_{\mathbf{x}}$	**std**	σ_X		
$\hat{d}$	**dmeans**	$\Delta = \mu_X - \mu_Y$	$T = \frac{\hat{D}-\Delta}{\widehat{\mathbf{se}}_{\hat{d}}}$	$\sim \mathcal{T}(\nu_d)$

Table 3.3: Summary of the estimates (numbers) and estimators (functions) that we studied in Section 3.1. The third column shows the population parameters we're trying to estimate. The last two columns list the pivotal quantities and the standard reference distributions that we used to obtain analytical approximations.

Table 3.3 shows the estimators we introduced in the previous section. The following code shows Python functions for computing these estimators from a given sample.

code 3.2.1
```
>>> def mean(sample):
        return sum(sample) / len(sample)

>>> def var(sample):
        xbar = mean(sample)
        sumsqdevs = sum([(xi-xbar)**2 for xi in sample])
        return sumsqdevs / (len(sample)-1)

>>> def std(sample):
        return np.sqrt(var(sample))

>>> def dmeans(xsample, ysample):
        return mean(xsample) - mean(ysample)
```

Nothing new to see here—we just copy-pasted the definitions of the estimator functions that we saw in Section 3.1.

Review of sampling distributions

The *sampling distribution* of the estimator g is denoted $\hat{\Theta} \stackrel{\text{def}}{=} g(\mathbf{X})$. The random variable $\hat{\Theta}$ gives us the complete information about the variability of the estimates $\hat{\theta}$ we can expect to observe. We can

describe the random variable $\hat{\Theta}$ either through its probability density function (PDF) $f_{\hat{\Theta}}$ or through its cumulative distribution function (CDF) $F_{\hat{\Theta}}$. The confidence interval calculations we'll show in this section will use the sampling distribution $\hat{\Theta}$ as the starting point.

In the previous section, we learned about the sampling distributions of the estimators **mean**, **var**, **std**, and **dmeans**. Let's review the two methods we can use to obtain the sampling distribution of an estimator: analytical approximations and bootstrap estimation.

Analytical approximations Starting from some assumptions about the population model $X \sim \mathcal{M}(\theta)$, we can build math models for the sampling distributions of the estimators **mean**, **var**, and **dmeans**, based on reference distributions like Student's t-distribution and the χ^2-distribution. The fifth column in Table 3.3 is a reminder of these reference distributions.

Recall the concept of a *pivotal quantity* that describes how we transform a problem-specific sampling distribution to one of the standard reference distributions. Here is a reminder of the three pivotal transformations we learned about in Section 3.1.

$$\frac{\overline{\mathbf{X}} - \mu_X}{\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}} \sim \mathcal{T}(n-1), \quad \frac{S_{\mathbf{x}}^2}{\sigma_X^2/(n-1)} \sim \chi^2(n-1), \quad \frac{\hat{D} - \Delta}{\widehat{\mathbf{se}}_{\hat{d}}} \sim \mathcal{T}(\nu_d).$$

Each of these transformations converts a problem-specific sampling distribution to a standard reference distribution with location zero and scale one.

Bootstrap estimation Recall the "sampling with replacement" trick that allows us to obtain an approximation to the sampling distribution of any estimator, as implemented by the helper function `gen_boot_dist` (see code block 3.1.32 on page 38). We just need to specify the sample and the estimator function as inputs to the function `gen_boot_dist`, and it will generate an approximation to the sampling distribution of the estimator based on $B = 5000$ bootstrap samples.

In this section, we'll continue the "parallel narratives" structure by presenting both analytical and computational methods for constructing confidence intervals. We'll have to double the work, but it's totally worth it for the deeper understanding and intuition about the statistical concepts that you'll achieve.

New confidence intervals concepts

Consider a statistical analysis of the population model $X \sim \mathcal{M}(\theta)$, where θ is an unknown parameter. We have obtained a sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ from this population, and we want to make inferences about the parameter θ based on this sample.

A *confidence interval* is a range of numbers that very likely contains the true parameter θ. Specifically, we'll develop a procedure for constructing confidence intervals with *coverage probability* γ (the Greek letter *gamma*), which means that $100\gamma\%$ of confidence intervals produced by this procedure will contain the true parameter. There are actually two different concepts that we call confidence intervals:

- A *random confidence interval* for the parameter θ with coverage probability γ. Confidence intervals are defined in terms of a lower limit and an upper limit computed from a random sample $\mathbf{X}$, and can be written using either interval notation or set notation:

  $$\mathbf{CI}_{\theta,\gamma} = [\mathbf{L}_\theta, \mathbf{U}_\theta] = \{\mathbf{L}_\theta \leqslant \theta \leqslant \mathbf{U}_\theta\},$$

  where the lower limit $\mathbf{L}_\theta$ and upper limit $\mathbf{U}_\theta$ are random variables computed from the random sample $\mathbf{X}$.
- A *particular confidence interval* computed from the sample $\mathbf{x}$ is denoted:

  $$\mathbf{ci}_{\theta,\gamma} = [\mathbf{l}_\theta, \mathbf{u}_\theta] = \{\mathbf{l}_\theta \leqslant \theta \leqslant \mathbf{u}_\theta\}.$$

  The lower limit $\mathbf{l}_\theta$ and upper limit $\mathbf{u}_\theta$ are estimates (numbers) computed from a particular sample $\mathbf{x}$.

For example, a 90% ($\gamma = 0.9$) confidence interval for the parameter θ is denoted $\mathbf{CI}_{\theta,0.9} = [\mathbf{L}_\theta, \mathbf{U}_\theta]$, and describes a subset of the parameter space that should contain the unknown population parameter θ at least 90% of the time:

$$\Pr_{\mathbf{X}}\big(\{\theta \in \mathbf{CI}_{\theta,0.9}\}\big) \;=\; \Pr_{\mathbf{X}}\big(\{\mathbf{L}_\theta \leqslant \theta \leqslant \mathbf{U}_\theta\}\big) \;\geqslant\; 0.9.$$

See Figure 3.27 for an illustration of this interval.

Coverage and error probabilities The parameter γ is called the *coverage probability* or the *confidence level* of the confidence interval. We often describe confidence intervals in terms of the *error probability* α (the Greek letter *alpha*), which is the complement of the coverage probability, $\gamma = 1 - \alpha$. Common choices for coverage probabilities include $\gamma = 0.9$ (90%), $\gamma = 0.95$ (95%), and $\gamma = 0.99$ (99%), which correspond to error probabilities $\alpha = 0.1$, $\alpha = 0.05$, and $\alpha = 0.01$. The higher the coverage probability, the wider the interval needs to be.

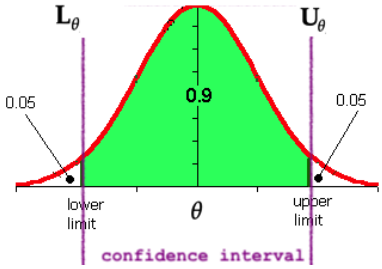


Figure 3.27: Illustration of a 90% confidence interval for the population parameter θ. The confidence interval describes the range of θ values in between the 5% and the 95% percentiles of the sampling distribution $f_{\widehat{\Theta}}$.

Interpreting confidence intervals

The confidence interval $\mathbf{CI}_{\theta,0.9}$ specifies a range of numbers that includes the "plausible" values for the population parameter θ. This confidence interval is a useful way to express our uncertainty about the unknown population parameter θ.

It's important to be clear exactly what the 90% coverage means so that we can correctly interpret confidence interval calculations. The coverage probability is a "quality guarantee" about **the procedure we used to compute the confidence interval**. The probability statement $\Pr_{\mathbf{X}}\left(\{\theta \in \mathbf{CI}_{\theta,0.9}\}\right) \geqslant 0.9$, tells us that the way we computed the confidence interval $\mathbf{CI}_{\theta,0.9}$ "works" (contains the true population parameter θ) 90% of the time. Note that **we can't give any guarantees about any particular confidence interval** $\mathbf{ci}_{\theta,0.9}$ computed from a sample $\mathbf{x}$. We can only make general statement about the confidence interval $\mathbf{CI}_{\theta,0.9}$ computed from a random sample $\mathbf{X}$.

The correct interpretation of the coverage probability associated with confidence intervals is the source of many misconceptions and misinterpretations. Let's make sure we know what's going on, by going back to the definition of the sampling distribution $\widehat{\Theta} = g(\mathbf{X})$ of the estimator. Confidence intervals are obtained from the sampling distribution, so if we're clear about the interpretation of sampling distributions, then we'll be clear about confidence intervals too.

The sampling distribution of the estimator g, denoted $\hat{\Theta} \stackrel{\text{def}}{=} g(\mathbf{X})$, describes the possible estimates we can expect to observe from *random samples* from the population. Recall we can obtain the sampling distribution by repeatedly generating samples $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$ from the population, computing the estimates $\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3, \ldots$, from the samples $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$, and plotting a histogram of the $\hat{\theta}_j$s.

The situation is analogous for confidence intervals. Imagine a sequence of confidence intervals $\mathbf{ci}_{\theta,0.9,1}$, $\mathbf{ci}_{\theta,0.9,2}$, $\mathbf{ci}_{\theta,0.9,3}$, ..., calcu-

lated from the samples $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$. The 90% coverage guarantee tells us that, on average, 90 out of 100 such confidence intervals will contain the true population parameter θ. Figure 3.28 illustrates the coverage probability as a proportion of successes. Since we're using a procedure with 90% guarantee, we expect 45 out of 50 confidence intervals to contain the true population mean, and around 5 out of 50 to fail. Indeed, we see there are only four confidence intervals that fail to include the population mean.

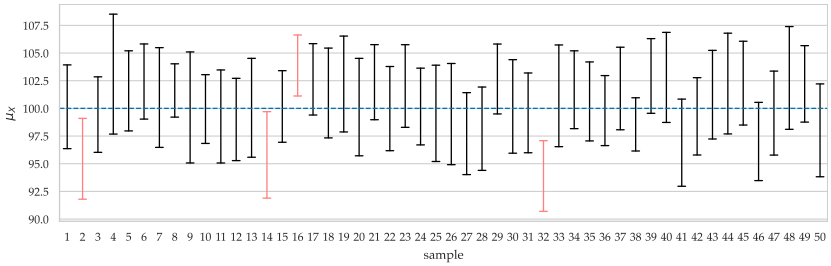


Figure 3.28: Illustration of 90% confidence intervals for the population mean. The plot shows confidence intervals $\mathbf{ci}_{\theta,0.9,1}$, $\mathbf{ci}_{\theta,0.9,2}$, $\mathbf{ci}_{\theta,0.9,3}$, $\ldots$, $\mathbf{ci}_{\theta,0.9,50}$, which we computed from 50 different samples of size $n = 20$ taken from the population $X \sim \mathcal{N}(100, 10)$. The true population mean is $\mu_X = 100$. Intervals that don't include μ_X are drawn in light red.

Note the confidence interval $\mathbf{ci}_{\theta,0.9}$ computed from any particular sample $\mathbf{x}$ either contains or doesn't contain the true population parameter θ, so there is no probability involved. The 90% probability guarantee $\Pr_{\mathbf{X}}\big(\{\theta \in \mathbf{CI}_{\theta,0.9}\}\big) \geqslant 0.9$ describes the **long term average reliability of the procedure for computing confidence intervals** when used on random samples $\mathbf{X}$ from the population. This type of guarantee is sometimes called *frequentist*, since it refers to the frequency (proportion) of successes we would observe if we repeat the confidence interval calculation from many samples.

3.2.2 Confidence interval constructions

To construct a $(1-\alpha)$-confidence interval $\mathbf{CI}_{\theta,(1-\alpha)}$ for the parameter θ is to find a procedure for computing the lower bound $\mathbf{L}_\theta$ and the upper bound $\mathbf{U}_\theta$, such that the interval $[\mathbf{L}_\theta, \mathbf{U}_\theta]$ contains the unknown population parameter θ at least $100(1-\alpha)\%$ of the time:

$$\Pr_{\mathbf{X}}\big(\{\mathbf{L}_\theta \leqslant \theta \leqslant \mathbf{U}_\theta\}\big) \geqslant 1 - \alpha.$$

Note the boundaries of the confidence interval $\mathbf{L}_\theta$ and $\mathbf{U}_\theta$ are random variables, since they are computed from random samples $\mathbf{X}$.

Comparing analytical formulas and bootstrap estimation

We learned how to construct confidence intervals using both analytical approximations and the bootstrap method. The confidence intervals we obtained from these methods were very similar. The bootstrap estimation approach is a general-purpose tool that you can use for any estimator, because it doesn't make any assumptions about the population model family. Bootstrap estimation is also robust to outliers. In contrast, the analytical approximation formulas we learned are special-purpose tools designed for a specific population family. All the formulas and proofs assume the population model is normally distributed. If we use the analytical approximation formulas for populations that are not normally distributed, we have no guarantee the results will be accurate.

Table 3.4 contains the evaluation of the width and coverage probabilities for several methods for computing confidence intervals. The first two columns show coverage probabilities of the analytical approximation and percentile bootstrap methods.. Third third column shows the results for the bias-corrected and accelerated (BCa) bootstrap method for computing confidence intervals, which we'll discuss next.

We have computed confidence intervals for different populations: uniform $U \sim \mathcal{U}(0,1)$, normal $Z \sim \mathcal{N}(0,1)$, a skewed distribution S (log-normal distribution with parameter 0.3), a long tailed distribution $E \sim \text{Expon}(\lambda = 5)$, and a bimodal distribution that consists of mixture 40-60 mixture of two gaussians, $B \sim 0.4\mathcal{N}(3,1) + 0.6\mathcal{N}(6.2,1)$.

		average interval width			coverage probability		
		approx.	percentile	BCa	approx.	percentile	BCa
U	$n = 20$	0.221	0.205	0.205	0.893	0.867	0.885
	$n = 40$	0.154	0.148	0.148	0.900	0.888	0.896
Z	$n = 20$	0.770	0.715	0.717	0.884	0.851	0.855
	$n = 40$	0.526	0.507	0.507	0.905	0.899	0.887
S	$n = 20$	0.244	0.226	0.230	0.886	0.868	0.859
	$n = 40$	0.169	0.163	0.164	0.902	0.890	0.892
E	$n = 20$	0.148	0.137	0.144	0.874	0.849	0.858
	$n = 40$	0.104	0.101	0.104	0.877	0.866	0.862
B	$n = 20$	1.420	1.316	1.321	0.902	0.884	0.903
	$n = 40$	0.983	0.947	0.949	0.908	0.902	0.905

Table 3.4: Comparison of the coverage probability of different methods for computing confidence intervals for the population mean for samples of size $n = 20$ and $n = 40$ from different populations.

		average interval width			coverage probability		
		approx.	percentile	BCa	approx.	percentile	BCa
U	$n = 20$	0.104	0.057	0.059	0.983	0.855	0.857
	$n = 40$	0.067	0.039	0.040	0.993	0.884	0.889
Z	$n = 20$	1.271	0.938	1.053	0.898	0.825	0.840
	$n = 40$	0.805	0.696	0.758	0.893	0.845	0.865
S	$n = 20$	0.128	0.107	0.122	0.795	0.730	0.757
	$n = 40$	0.085	0.088	0.100	0.804	0.828	0.826
E	$n = 20$	0.050	0.055	0.065	0.625	0.661	0.711
	$n = 40$	0.033	0.046	0.054	0.640	0.756	0.804
B	$n = 20$	4.332	2.602	2.775	0.961	0.846	0.861
	$n = 40$	2.824	1.863	1.948	0.976	0.893	0.872

Table 3.5: Comparison of the coverage probability of different methods for computing confidence intervals for the population variance for samples of size $n = 20$ and $n = 40$ from different populations.

3.2.9 Exercises

Alright, enough theory. It's time you try to build some confidence intervals on your own.

E3.17 Compute a confidence 90% confidence interval for the population mean based on the sample from Batch 04 of the `kombucha` dataset. Use **a)** analytical approximation and **b)** the bootstrap.

E3.18 Calculate a 90% confidence interval for the population variance based on the sample from Batch 05 of the `kombucha` dataset. Obtain answers using **a)** the analytical approximation based the χ^2-distribution, and **b)** the bootstrap.

E3.19 Compute a 95% confidence interval for the difference between rural and city sleep scores in the doctors dataset. Use **a)** an analytical approximation based on Student's t-distribution, and **b)** bootstrap.

E3.20 Calculate an 80% confidence interval for the difference between lecture and debate groups in the students dataset. Use **a)** analytical approximation based on Student's t-distribution, and **b)** the bootstrap.

E3.21 As part of a lab experiment, sixty-four two-week old rats were given a vitamin D supplement for a period of one month, and their weights were recored at the end of the month (30 days). The sample mean was 89.60 g with standard deviation 12.96 g. Calculate a 95% confidence interval for the mean weight for rats undergoing this treatment. Obtain analytical approximations based on **a)** the normal model, and **b)** Student's t-distribution. Compare your answers in a)

3.3 Introduction to hypothesis testing

Hypothesis testing is a statistical analysis technique we can use to detect "unexpected" patterns in a sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ collected from an unknown population $X \sim \mathcal{M}(\theta)$. We define what is "expected" in terms of a theoretical model $X_0 \sim \mathcal{M}(\theta_0)$, where both the model family $\mathcal{M}$ and the model parameter θ_0 are known.

The fundamental question we want to answer is whether the unknown population X from which the sample $\mathbf{x}$ was taken is the same as the expected theoretical model X_0, or different from it. More specifically, we'll learn about *parametric* hypothesis testing procedures, whose aim is to determine if the parameter θ of the unknown population is the same as the theoretical parameter θ_0. The population parameter θ is unknown, but we can use the estimate $\hat{\theta}_{\mathbf{x}}$ computed from the sample $\mathbf{x}$ as an approximation for θ.

Applications

Hypothesis testing is used in physics, chemistry, biology, medicine, finance, marketing, industrial process monitoring, and, more generally, in all domains where data analysis is needed to make informed decisions. Let's look at some examples that show how hypothesis testing is used in real-world situations.

- **Quality control** The operator of a kombucha bottling plant wants to detect *irregular* production batches. When the kombucha brewing and bottling process is working as expected, the volume in each bottle is described by the theoretical model $K_0 \sim \mathcal{N}(\mu_{K_0} = 1000, \sigma_{K_0} = 10)$. Performing a hypothesis test allows the operator to check if a sample $\mathbf{k}$ comes from a *regular* batch or an *irregular* batch.
- **Drug equivalence** A pharmaceutical startup has developed a new low-cost formulation for an important life-saving drug. They want to check if the new drug formulation works similarly to the old drug formulation by measuring some health indicator on a sample of patients treated with the new drug formulation. The old drug formulation has been around for many years, so there is a well-known model for its expected effects on the health indicator, $I_0 \sim \mathcal{M}(\theta_{I_0})$. Do the health indicator measurements of a sample of patients treated with the new drug formulation differ from the model I_0?
- **Software engineering** A software company wishes to detect when a software release (new version of the software) is unusually buggy. Based on observations from past releases, engineers have compiled a baseline model $B_0 \sim \mathcal{M}(\theta_{B_0})$ for the

number of errors that occur, on average, when the software is operating as expected. By comparing the number of errors that occur for a new software release to the baseline model, they can detect when the new software release is "extra buggy."

- **Scientific discovery** Suppose the baseline model $S_0 \sim \mathcal{M}(\theta_{S_0})$ represents the currently accepted scientific theory about some phenomenon of interest, say the speed of a chemical reaction. A chemistry researcher has discovered a new procedure that increases the speed of the chemical reaction, and they want to publish a paper about it. They perform an experiment in which they run the chemical reaction using the new procedure and collect a sample of reaction speed observations $\mathbf{s} = (s_1, \ldots, s_n)$. To show the new procedure is interesting and worthy of publication in an academic journal, the researcher must show that the observations obtained using the new procedure differ from the currently accepted theoretical model S_0.

In each of these examples, the baseline model contains some inherent variability, which is described by the theoretical model $X_0 \sim \mathcal{M}(\theta_0)$. **Our goal is to determine if the variability of the observed sample $\mathbf{x}$ exceeds the plausible variability in the baseline model.** If the observed sample is very unlikely to have occurred by chance under the theoretical model, then we have reason to suspect that there is something "special" about the sample $\mathbf{x}$, meaning it can't be explained by the baseline model X_0.

Overview of hypothesis testing

This section is your introduction to hypothesis testing, which will be the main topic for the next five sections. We'll start things off in "easy mode" by looking at the simplest possible statistical testing scenario (comparing one sample from an unknown population to a theoretical model) and using the simplest possible computational method (direct simulation of the sampling distribution using the function `gen_sampling_dist` that we saw in Section 3.1). In the next section (Section 3.4), we'll revisit the same statistical testing scenario (comparing one sample to a theoretical model) using analytical approximation methods based on standard reference distributions like Student's t-distribution and the χ^2-distribution. In later sections, we'll learn how to apply hypothesis testing in other statistical analysis scenarios.

Heads up, the topic of hypothesis testing is notoriously difficult to understand. Going back to the bike trip analogy, learning about hypothesis testing is the biggest "uphill" in the whole book. No worries though! The good news is that you have all the tools and

equipment (chapters 1 and 2), and adequate training (sections 3.1 and 3.2) to handle the hypothesis testing uphill. In fact, the whole structure of this book was intentionally designed to make your first contact with hypothesis testing a positive experience. Yes, hypothesis testing is a big climb, but you totally got this!

3.3.1 Definitions

You're already familiar with the math building blocks of hypothesis testing like *estimators* and their *sampling distributions*, but there are lots of new concepts and terminology you need to know about. Figure 3.32 shows the complete hypothesis testing workflow. As you can see, there are a lot of "moving pieces" that we need to define and explain, so we'd better get started by defining terms.

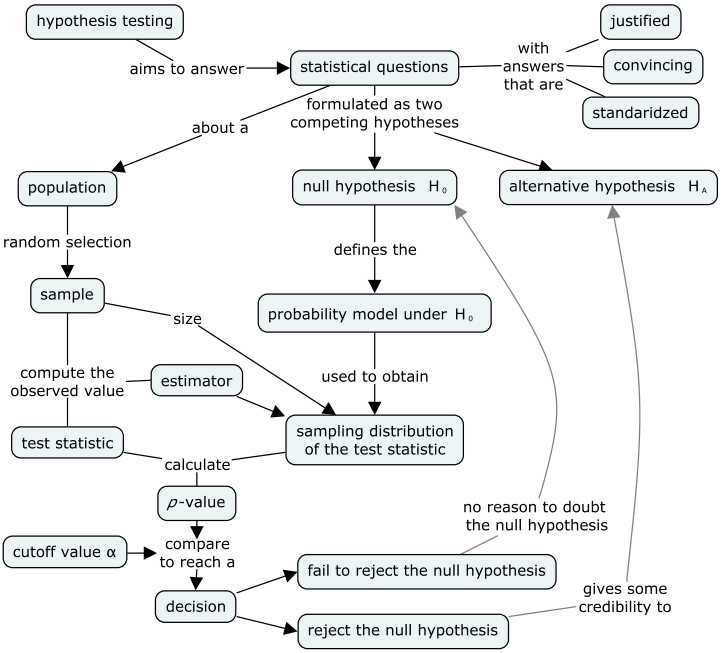


Figure 3.32: Overview of the hypothesis testing procedure. We want to differentiate between two competing hypotheses H_0 and H_A, based on the p-value of a test statistic $\hat{\theta}_{\mathbf{x}}$ computed under the sampling distribution $\widehat{\Theta}_0$, which assumes H_0 is true. If $\hat{\theta}_{\mathbf{x}}$ is unlikely to occur by chance under $\widehat{\Theta}_0$, then our decision will be to reject H_0. Otherwise, we have failed to reject H_0.

I know it's a lot to take in at once, but this is necessary complexity. We'll explain all of these concepts in the next five pages, and things

will become clear when we look at the real-world examples later in the section. For now, I'm just throwing all the new jargon and terminology at you, so you can learn the necessary vocabulary for talking about hypothesis testing.

Hypotheses

A *statistical hypothesis* is a precise way to describe the existence or non-existence of some "unexpected pattern" in the data. We usually state hypotheses as equations or inequalities between a parameter of the unknown population θ and the parameter of the baseline theoretical model θ_0.

To use the hypothesis testing procedure, we formulate two competing statistical hypotheses:

- The *alternative hypothesis*, denoted H_A, is a statement about some population parameter that corresponds to the condition of interest. For example, $H_A : \theta \neq \theta_0$.
- The *null hypothesis*, denoted H_0, is a skeptical claim about the population parameter that contradicts the alternative hypothesis. The null hypothesis is usually a "no effect" or "no difference" claim. For example, $H_0 : \theta = \theta_0$.

The alternative hypothesis describes the new or unexpected pattern, and it is the one we're usually interested in. For example, an abnormal kombucha bottle, a drug formulation that differs from the baseline model, an extra-buggy software release, or a discovery of a new chemical procedure.

In contrast, the null hypothesis corresponds to the baseline model that says no unexpected pattern exists, like the baseline models K_0, I_0, B_0, and S_0 in the above examples. You can think of the null hypothesis as saying there is "nothing to see here," since the unknown population is not different from the expected baseline model.

Kombucha bottling scenario revisited Let's look at the hypotheses we can use in the kombucha scenario. When the bottling process is working as expected, the volume in each bottle is supposed to be roughly 1000 ml, with standard deviation 10 ml, which we can model as the random variable $K_0 \sim \mathcal{N}(\mu_{K_0} = 1000, \sigma_{K_0} = 10)$.

Our goal is to detect when a production batch is *irregular*, meaning the average volume in this batch μ differs from the expected average μ_{K_0}. The two hypotheses we will consider are

$$H_A : \mu \neq \mu_{K_0} \qquad \text{and} \qquad H_0 : \mu = \mu_{K_0}.$$

The alternative hypothesis describes some deviation from the expected average volume, and it includes bottles that are too full $\mu > \mu_{K_0}$ or too empty $\mu < \mu_{K_0}$, on average. The null hypothesis H_0 describes the scenario when the bottling process is working as expected and the average volume in the current batch μ is equal to the expected average μ_{K_0}.

Test statistics

In order to decide between the null hypothesis H_0 (no difference exists) and the alternative hypothesis H_A (a difference exists), we choose an estimator g that helps us distinguish between the two hypotheses. We use the value of the estimator $\hat{\theta}_{\mathbf{x}} = g(\mathbf{x})$ computed from the sample $\mathbf{x}$ as an estimate of the parameter θ of the unknown population $X \sim \mathcal{M}(\theta)$. We refer to the value of the estimator computed from the sample $\mathbf{x}$ as a *test statistic*.

For example, we can use the sample mean $\overline{\mathbf{k}}$ computed from a kombucha sample $\mathbf{k}$ as an estimate for the unknown population mean μ. Knowing the sample mean $\overline{\mathbf{k}} \approx \mu$ can help us to distinguish between the two hypotheses $H_0 : \mu = \mu_{K_0}$ and $H_A : \mu \neq \mu_{K_0}$.

Sampling distribution of the test statistic under H_0

Assuming H_0 is true, we can compute the sampling distribution of the test statistic, denoted $\widehat{\Theta}_0 = g(\mathbf{X}_0)$, which describes the possible values of the test statistic we can observe for samples of size n from the null model $X_0 \sim \mathcal{M}(\theta_0)$. Recall $\mathbf{X}_0$ denotes a random sample from X_0. In other words, the sampling distribution of the test statistic $\widehat{\Theta}_0$ describes the expected variability of the test statistics we can expect to observe by chance under the null hypothesis.

For example, the null model for the kombucha volumes is the normal random variable $K_0 \sim \mathcal{N}(1000, 10)$. The sampling distribution of the mean $\overline{\mathbf{K}}_0 = \mathbf{mean}(\mathbf{K}_0)$ computed from a random sample $\mathbf{K}_0$ from the null model tells us the sample means we can expect to observe, when the kombucha bottling plant is working as expected.

In this section, we'll show how to approximate the sampling distribution under the null hypothesis using simulation. In later sections, we'll also show how to use analytical approximations to model sampling distributions.

Probability calculations for hypothesis testing

Hypothesis testing is based on the comparison between the observed value of the test statistic $\hat{\theta}_{\mathbf{x}}$ computed from the sample $\mathbf{x}$ and the

sampling distribution of the test statistic under the null hypothesis. Specifically, we compute the *p-value*, which is the probability of observing a test statistic *at least as extreme* as the observed estimate $\hat{\theta}_{\mathbf{x}}$ according to the sampling distribution of the test statistic $\widehat{\Theta}_0 = g(\mathbf{X}_0)$ under the null hypothesis.

The p-value is a measure of the "unexpectedness" or "surprisingness" of the observed sample $\mathbf{x}$ under the null hypothesis. For example, a p-value of 0.01 (1%) tells us the probability of observing the test statistic $\hat{\theta}_{\mathbf{x}}$ (or a more extreme value) is only 1 in 100 under the null hypothesis, which—we can agree—is very unlikely to occur by chance. Another way to describe a small p-value is to say the observed test statistic is "incompatible" with the null hypothesis.

Outcomes of statistical tests

We compare the p-value computed for the observed test statistic $\hat{\theta}_{\mathbf{x}}$ under the null hypothesis to a predetermined *cutoff value* α (usually a small number like $\alpha = 0.05$) to make a decision about the outcome of the statistical test. We can reach one of two possible decisions:

- We **reject the null hypothesis** when the p-value is less than or equal to the cutoff value α.
- We **fail to reject the null hypothesis** when the p-value is greater than the cutoff value α.

Let's talk about the two possible outcomes in more detail.

Rejecting the null hypothesis Observing a very small p-value for the test statistic $\hat{\theta}_{\mathbf{x}}$ under the sampling distribution $\widehat{\Theta}_0$ tells us the observed sample $\mathbf{x}$ is unlikely to occur by chance under H_0. A small p-value counts as evidence against the null hypothesis, which is why our decision is to *reject the null hypothesis*.

This conclusion means that the observed sample is unlikely to be explained by the baseline model H_0, which suggests we need to look for an alternative model. Note, however, that reaching the decision "reject H_0" does not mean that H_A is true. You can think of hypothesis testing as a minimum "sanity check" we perform to rule out the possibility that the observed sample occurred by chance under H_0.

Failing to reject the null hypothesis In contrast, a large p-value tells us $\hat{\theta}_{\mathbf{x}}$ is likely to have occurred by chance under $\widehat{\Theta}_0$, so there is no reason to reject the null hypothesis. We use the double negative "fail to reject H_0" to describe precisely what we did: we looked for

some "abnormality" in the sample $\mathbf{x}$ relative to the null model, and we didn't find any. Another way to say this is the observed sample $\mathbf{x}$ is *consistent* with the null hypothesis.

Make sure you understand the descriptions of the two possible outcomes and the is-this-sample-unexpected-under-H_0 reasoning behind them. Note the specific wording used. This is very tricky stuff, and it is easy to misinterpret the result, like saying we have "proved that H_0 is true" or "proved H_A is false," etc. We'll talk more about the logic of hypothesis testing in the next subsection, and have *a lot* more to say about the possible misinterpretations of hypothesis tests later in the book.

Look back at Figure 3.32 on page 105. You should now be familiar with all the terminology that appears in that concept map.

Effect size estimates

The *effect size*, denoted Δ (the Greek letter delta), describes the magnitude of the "difference" or "discrepancy" from the null model. For example, in the kombucha bottling scenario, we can use the difference between the population means $\Delta \stackrel{\text{def}}{=} \mu - \mu_{K_0}$ as a measure of the "irregularity" of the current batch, relative to the mean of the theoretical distribution μ_{K_0}. In practice, the population mean μ is unknown, so we can only compute estimates of the effect size. The *point estimate* for the effect size $\widehat{\Delta} = \overline{\mathbf{k}} - \mu_{K_0}$ is a single number that represents our "best guess" about the true effect size $\Delta = \mu - \mu_{K_0}$. We can also use the techniques we learned in Section 3.2 to construct a *confidence interval* for the effect size $\mathbf{ci}_{\Delta,\gamma} = [\mathbf{l}_\Delta, \mathbf{u}_\Delta]$.

Effect size estimates are often the most interesting part of any statistical analysis since they actually tell us what we want to know. Rejecting the null hypothesis is just a formality—a basic check we perform to satisfy a skeptical colleague who claims that the observed value could have occurred by chance when in reality no effect exists. Computing the effect size estimate $\widehat{\Delta}$ and the confidence interval $\mathbf{ci}_{\Delta,\gamma}$ tells us **how much** the observed sample differs from the theoretical model, which is what we're really interested in.

3.3.2 The logic of hypothesis testing

The basic argument we're making when we reach the decision to reject the null hypothesis is the following:

$$\hat{\theta}_{\mathbf{x}} \text{ unlikely to occur under } H_0 \quad \Rightarrow \quad H_0 \text{ is unlikely to be true.}$$

Example 2S: test for the mean of Batch 01

We'll now repeat the same statistical analysis based on the sample $\mathbf{k}_{01}$ from Batch 01. The null and alternative hypothesis are the same as in the previous example. We start by calculating the sample size and the observed sample mean $\overline{\mathbf{k}}_{01}$.

code 3.3.17
```
>>> batch01 = kombucha[kombucha["batch"]==1]
>>> ksample01 = batch01["volume"]
>>> len(ksample01)
40
>>> obsmean01 = mean(ksample01)
>>> obsmean01
999.10375
```

The next step is to simulate lots of observations from the sampling distribution of the mean under the null hypothesis by calling the function `gen_sampling_dist`, as we did earlier in code block 3.3.6.

code 3.3.18
```
>>> kbars40 = gen_sampling_dist(rvK0, estfunc=mean, n=40)
```

Note we generated the sampling distribution $\overline{\mathbf{K}}_0 \approx$ `kbars40` from random samples of the same size as $\mathbf{k}_{01}$.

We can now compute the p-value of the observed sample mean $\overline{\mathbf{k}}_{01} = 999.1$ by calculating the proportion of values in the list `kbars40` whose absolute deviation from the population mean μ_{K_0} is greater than the observed deviation $|\overline{\mathbf{k}}_{01} - \mu_{K_0}| =$ `obsdev01`.

code 3.3.19
```
>>> obsdev01 = abs(obsmean01 - muK0)
>>> tails = [v for v in kbars40 if abs(v-muK0) >= obsdev01]
>>> pvalue01 = len(tails) / len(kbars40)
>>> pvalue01
0.5711
```

Of the 10000 simulated sample means, 5711 satisfy the criterion "$\overline{\mathbf{k}}_{01}$ or more extreme." Figure 3.36 illustrates the p-value calculation. The p-value tells us that observing the sample mean $\overline{\mathbf{k}}_{01} = 999.1$ or more extreme has probability 57% under H_0, which is definitely not unlikely. The observed sample mean $\overline{\mathbf{k}}_{01}$ could very likely have occurred by chance under H_0.

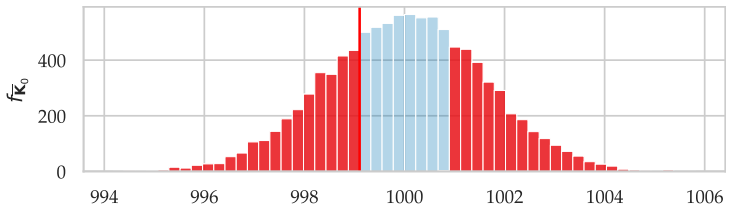


Figure 3.36: The p-value calculation of the observed mean $\overline{\mathbf{k}}_{01} = 999.1$ is obtained from the sampling distribution $\overline{\mathbf{K}}_0$ under the null model.

We now apply the logic of hypothesis testing to make a decision. Since the p-value is greater than the cutoff value $\alpha = 0.05$, we say the observed difference from the expected mean $\mu_{K_0} = 1000$ is *not statistically significant*, and our decision is "fail to reject H_0." The real-world interpretation of this result is that Batch 01 is probably a regular batch. There's nothing wrong with it, so we can ship it!

* * *

I know this was a lot of hoops to jump through, but in the end we have a standardized procedure for detecting "regular" and "irregular" production batches. This procedure is very useful to have if you're running the bottling plant. In problem P3.13, I'll ask you to write a Python function for performing all the steps of the one-sample simulation test of the mean.

Exercises 1

E3.22 Compute the p-value for hypothesis test of the mean in Batch 05 of the kombucha dataset.

3.3.5 Test for the variance

We'll now develop a hypothesis test for detecting variance deviations from a theoretically expected variance $\sigma^2_{X_0}$. In particular, we're interested in detecting the case when the variance of the unknown population σ^2 is higher than expected. We formulate the following two competing hypotheses:

$$H_A : \sigma^2 > \sigma^2_{X_0} \qquad \text{and} \qquad H_0 : \sigma^2 \leqslant \sigma^2_{X_0},$$

where $\sigma^2_{X_0}$ is the variance of the theoretical model $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$. Note the alternative hypothesis is stated as a "greater than" inequality, which means we're only interested in deviations from the theoretical variance $\sigma^2_{X_0}$ in the *positive* direction.

We'll use the sample variance $s^2_{\mathbf{x}}$ to estimate σ^2. We compute the sample variance using the estimator **var**. Here is a reminder of the Python function for computing the sample variance.

code 3.3.20

```
>>> def var(sample):
        xbar = mean(sample)
        sumsqdevs = sum([(xi-xbar)**2 for xi in sample])
        return sumsqdevs / (len(sample)-1)
```

3.4 Analytical approximations

When the theoretical model under the null hypothesis is normally distributed $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$, it is possible to obtain analytical approximations to the sampling distributions of the mean and the variance. Recall the formulas we learned in Section 3.1 for describing the sampling distribution of the mean in terms of Student's t-distribution and the sampling distribution of the variance in terms of the χ^2-distribution. This is really cool, because analytical approximations give us a more direct approach for computing p-values. Instead of doing all the "manual labour" of simulating the sampling distribution, we can compute p-values by evaluating the cumulative distribution function $F_{\widehat{\Theta}_0} = \texttt{rvTheta0.cdf}$ of the analytical approximation for the sampling distribution $\widehat{\Theta}_0 = \texttt{rvTheta0}$ under the null.

The material we'll present in this section is a "rerun" of the hypothesis testing material that we saw in the previous section. We'll revisit the exact same statistical analysis scenarios related to detecting "irregular" batches at the kombucha bottling plant, but this time we'll use analytical formulas to compute p-values instead of using simulation. You'll have to learn how to do math calculations based on the cumulative distribution function $F_{\widehat{\Theta}_0}$, but they are not very complicated, and you already learned about the analytical approximations for sampling distributions in Section 3.1. The purpose of repeating the hypothesis testing material using different computational methods is to allow you to see more clearly the common structure of the hypothesis testing procedure: if the p-value is very small (less than some predetermined threshold like $\alpha = 0.05$), then we reject H_0. If the p-value is not small (larger than α), then we fail to reject H_0.

Figure 3.43 shows a high-level overview of the hypothesis testing procedure based on analytical approximations. If you compare the concept map in Figure 3.43 to the concept map from the previous section shown in Figure 3.33 (see page 114), you'll see the only difference is the "math machinery" used for p-value calculations. In this section, we'll compute the probability of "$\hat{\theta}_{\boldsymbol{x}}$ or more extreme" by evaluating the `rvTheta0.cdf` method of a computer model `rvTheta0` which corresponds to an analytical approximation for the sampling distribution $\widehat{\Theta}_0$ under the null hypothesis. Apart from this change, all the other steps of the hypothesis testing procedure are the same.

3.4.1 Definitions

There are actually no new concepts to define, since you've already seen the logic of hypothesis testing in the previous section, and

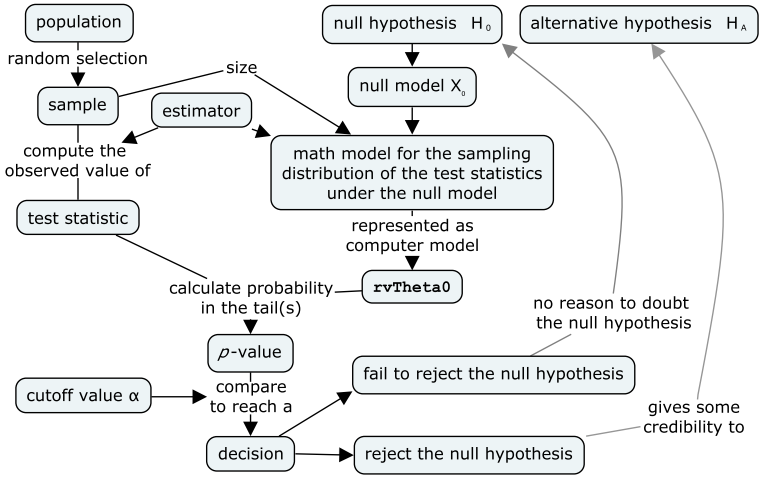


Figure 3.43: The hypothesis testing procedure based on analytical approximations. We compute the p-value for the hypothesis test based on the tails of the sampling distribution under the null, $\widehat{\Theta}_0 = \texttt{rvTheta0}$, which is usually one of the standard reference distributions like the normal distribution, the t-distribution, or the χ^2-distribution.

know about analytical approximations to sampling distributions from Section 3.1. Instead, we'll use the next few pages as reminders of the concepts we need to "import" from the previous sections to do hypothesis testing based on analytical approximations.

Context

Let's start with a quick review of the hypothesis testing concepts that we introduced in Section 3.1. Consider the sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ that we obtained using random sampling from an unknown population $X \sim \mathcal{M}(\theta)$. The fundamental question we're asking in hypothesis testing is whether the sample $\mathbf{x}$ is likely or unlikely to have come from the theoretical model $X_0 \sim \mathcal{M}(\theta_{X_0})$. To answer this question, we formulate two competing *statistical hypotheses*.

- H_A: the alternative hypothesis, which says the unknown parameter θ differs from θ_{X_0}.
- H_0: the null hypothesis, which states that θ is the same as θ_{X_0}.

We don't know the parameter of the unknown population θ, but we can use the *estimate* $\hat{\theta}_{\mathbf{x}} = g(\mathbf{x})$ computed from the observed sample $\mathbf{x}$, as an approximation for the population parameter θ. For example, we'll use the estimator $\textbf{mean} = \texttt{mean}$ in hypothesis tests for

the difference between the mean of the unknown population μ and the theoretically expected mean μ_{X_0}. We'll also use the estimator $\mathbf{var} = \texttt{var}$ to detect deviations of the variance σ^2 from a theoretical variance $\sigma^2_{X_0}$.

We refer to estimates that are used as part of a hypothesis testing procedure as *test statistics*. For example, the sample mean $\bar{\mathbf{x}} = \mathbf{mean}(\mathbf{x})$ and the sample variance $s^2_{\mathbf{x}} = \mathbf{var}(\mathbf{x})$ are two test statistics you're already familiar with. In this section, we'll also use standardized test statistics like $z_{\mathbf{x}}$, $t_{\mathbf{x}}$, and $q_{\mathbf{x}}$, which simplify p-value calculations.

The main thing we need to know to perform a hypothesis test is the sampling distribution of the test statistic under the null hypothesis, $\widehat{\Theta}_0 = g(\mathbf{X}_0)$, which describes the variability of the test statistics we can expect to observe for random samples $\mathbf{X}_0$ from the null model. The p-value of the observed statistic $\hat{\theta}_{\mathbf{x}}$ is defined as the probability of observing "$\hat{\theta}_{\mathbf{x}}$ or a more extreme value" according to the sampling distribution of the statistic under the null model $\widehat{\Theta}_0$. Depending on the size of the p-value, we reach one of two possible decisions: "reject H_0" or "fail to reject H_0."

Analytical approximations

Let's assume that the theoretical model under the null hypothesis is normally distributed $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$. This modelling assumption is valid for many naturally occurring phenomena in physics, chemistry, biology, medicine, quality control, environmental science, etc. Even in cases when the true distribution of the variable of interest is not normally distributed, we can still use normal models as approximations.

Starting from the assumption $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$, we can obtain the sampling distribution under the null $\widehat{\Theta}_0$ as an analytical approximation formula based on the normal distribution, Student's t-distribution, or the χ^2-distribution. Let's review the math formulas and the code required to build computer models for analytical approximations.

Normal approximation to the sample mean The central limit theorem tells us that the sampling distribution of the mean, denoted $\overline{\mathbf{X}}_0 = \mathbf{mean}(\mathbf{X}_0)$, for samples of size n taken from the normally distributed theoretical model $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$ is approximately normally distributed:

$$\overline{\mathbf{X}}_0 \;\approx\; N_{\overline{\mathbf{X}}_0} \;\sim\; \mathcal{N}\big(\texttt{loc} = \mu_{X_0},\, \texttt{scale} = \mathbf{se}_{\bar{\mathbf{x}},0}\big),$$

where $\mathbf{se}_{\overline{\mathbf{x}},0} = \frac{\sigma_{X_0}}{\sqrt{n}}$ is the standard error of the mean.

Here is the code for constructing a computer model `rvNXbar0` for the normal approximation $N_{\overline{\mathbf{X}}_0} \approx \overline{\mathbf{X}}_0 = \mathbf{mean}(\mathbf{X}_0)$.

code 3.4.1
```
>>> n = ...          # sample size
>>> muX0 = ...       # theoretical mean
>>> sigmaX0 = ...    # theoretical standard deviation
>>> se = sigmaX0 / np.sqrt(n)
>>> from scipy.stats import norm
>>> rvNXbar0 = norm(loc=muX0, scale=se)
```

The random variable $N_{\overline{\mathbf{X}}_0}$ = `rvNXbar0` tells us what sample means we can expect to observe for random samples of size n.

Student's t-approximation to the sample mean The normal approximation $N_{\overline{\mathbf{X}}_0} \approx \overline{\mathbf{X}}_0$ assumes that the standard deviation of the theoretical model σ_{X_0} = `sigmaX0` is known. In many statistical analysis situations, σ_{X_0} is not known, and we use the sample standard deviation $s_{\mathbf{x}}$ as an estimate for the theoretical standard deviation σ_{X_0}.

In these situations, the appropriate model to use is based on Student's t-distribution:

$$\overline{\mathbf{X}}_0 \;\approx\; T_{\overline{\mathbf{X}}_0} \;\sim\; \mathcal{T}(\texttt{df}=\nu,\ \texttt{loc}=\mu_{X_0},\ \texttt{scale}=\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}),$$

where $\mathcal{T}$ is Student's t-distribution, with degrees of freedom parameter $\nu = n-1$, centred at μ_{X_0}, and scale parameter equal to the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}} = \frac{s_{\mathbf{x}}}{\sqrt{n}}$, which is the plug-in estimate of true standard error $\mathbf{se}_{\overline{\mathbf{x}},0} = \frac{\sigma_{X_0}}{\sqrt{n}}$, obtained by replacing σ_{X_0} with $s_{\mathbf{x}} = \mathbf{std}(\mathbf{x})$. Recall Student's t-distribution is a heavy-tailed cousin of the normal distribution specially designed to compensate for the fact that the estimated standard error $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}$ tends to underestimate the true standard error $\mathbf{se}_{\overline{\mathbf{x}},0}$.

Let's review the steps for constructing a computer model `rvTXbar0` that corresponds to the analytical approximation for the sample mean $T_{\overline{\mathbf{X}}_0} \approx \overline{\mathbf{X}}_0 = \mathbf{mean}(\mathbf{X}_0)$ for samples of size n from the theoretical model $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$.

code 3.4.2
```
>>> sample = ...        # observed sample
>>> n = len(sample)     # sample size
>>> muX0 = ...          # theoretical mean
>>> sehat = std(sample) / np.sqrt(n)
>>> from scipy.stats import t as tdist
>>> rvTXbar0 = tdist(df=n-1, loc=muX0, scale=sehat)
```

The random variable $T_{\overline{\mathbf{X}}_0}$ = `rvTXbar0` tells us what sample means we can expect to observe for random samples of size n under the null hypothesis $H_0 : \mu = \mu_{X_0}$, based on the estimated standard deviation $s_{\mathbf{x}}$ = `std(sample)`, instead of assuming a known σ_{X_0}.

The χ^2-approximation to the sample variance The sampling distribution of the variance $S^2_{\mathbf{X}_0} = \mathbf{var}(\mathbf{X}_0)$ for samples of size n taken from the normally distributed theoretical model $X_0 \sim \mathcal{N}(\mu_{X_0}, \sigma_{X_0})$ is described by the χ^2-distribution with $\nu = n - 1$ degrees of freedom and `scale` parameter $\frac{\sigma^2_{X_0}}{n-1}$:

$$S^2_{\mathbf{X}_0} \approx Q_{S^2_{\mathbf{X}_0}} \sim \chi^2\Big(\texttt{df} = n-1,\ \texttt{scale} = \tfrac{\sigma^2_{X_0}}{n-1}\Big).$$

We can build a computer model `rvQSsq0` that corresponds to the analytical approximation $Q_{S^2_{\mathbf{X}}} \approx S^2_{\mathbf{X}_0} = \mathbf{var}(\mathbf{X}_0)$ based on the predefined model `scipy.stats.chi2` initialized with the appropriate choices of the `df` and `scale` parameters:

code 3.4.3
```
>>> n = ...          # sample size
>>> sigmaX0 = ...  # theoretical standard deviation
>>> scale = sigmaX0**2 / (n-1)
>>> from scipy.stats import chi2
>>> rvQSsq0 = chi2(df=n-1, scale=scale)
```

We can use the sampling distribution $Q_{S^2_{\mathbf{X}}} = \texttt{rvQSsq0}$ for hypothesis tests whose aim is to detect differences between the variance of the unknown population σ^2 and the theoretical variance $\sigma^2_{X_0}$. The sampling distribution $Q_{S^2_{\mathbf{X}_0}}$ tells us the sample variances we can expect to observe for random samples of size n under the null hypothesis $H_0 : \sigma^2 = \sigma^2_{X_0}$.

Calculating p-values

There are three types of p-value calculations we might need to perform, depending on the *direction* of the alternative hypothesis we are testing. Recall statistical hypotheses are defined as inequalities between the parameter θ, which represents the unknown distribution from which the sample $\mathbf{x}$ was obtained, and the theoretical parameter θ_0 under the null model.

The p-value is defined as the probability of observing the test statistic $\hat{\theta}_\mathbf{x} = g(\mathbf{x}) = \texttt{obs}$ (an estimate for θ), or a more extreme value, according to the sampling distribution of the estimator under the null model $\widehat{\Theta}_0 = g(\mathbf{X}_0)$. We calculate p-values using the cumulative distribution function $F_{\widehat{\Theta}_0}$ of the sampling distribution $\widehat{\Theta}_0$, which is available as the `.cdf` method of the computer model `rvTheta0`.

Figure 3.44 shows the three types of p-value calculations, which correspond to the possible interpretations of the phrase "$\hat{\theta}_\mathbf{x} = \texttt{obs}$ or more extreme." See also Table 3.6 on page 127.

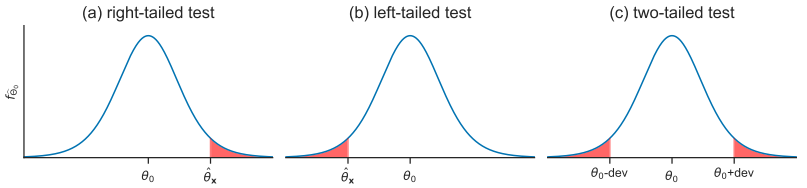


Figure 3.44: Illustration of the three types of p-value calculations for the test statistic $\hat{\theta}_{\mathbf{x}} = \texttt{obs}$ under the sampling distribution $\widehat{\Theta}_0 = g(\mathbf{X}_0) = \texttt{rvTheta0}$, which assumes the null hypothesis is true.

We'll now describe the probability calculations for the three cases using math notation based on $\widehat{\Theta}_0$, and also show code examples based on the computer model `rvTheta0`.

- **(a)** When we want to detect **positive deviations** of the population parameter θ from the theoretically expected parameter θ_0, we formulate the hypotheses $H_A : \theta > \theta_0$ and $H_0 : \theta \leqslant \theta_0$, and calculate the p-value based on the *right tail* of the probability distribution $\widehat{\Theta}_0 = \texttt{rvTheta0}$:

$$p = \Pr_{\widehat{\Theta}_0}\left(\{\widehat{\Theta}_0 \geqslant \hat{\theta}_{\mathbf{x}}\}\right) = 1 - F_{\widehat{\Theta}_0}(\hat{\theta}_{\mathbf{x}}) = \texttt{1 - rvTheta0.cdf(obs).}$$

- **(b)** When looking for **negative deviations** of the parameter θ from the theoretically expected parameter θ_0, we use the hypotheses $H_A : \theta < \theta_0$ and $H_0 : \theta \geqslant \theta_0$, and calculate the p-value based on the *left tail* of the probability distribution:

$$p = \Pr_{\widehat{\Theta}_0}\left(\{\widehat{\Theta}_0 \leqslant \hat{\theta}_{\mathbf{x}}\}\right) = F_{\widehat{\Theta}_0}(\hat{\theta}_{\mathbf{x}}) = \texttt{rvTheta0.cdf(obs).}$$

- **(c)** To detect both **positive and negative deviations** of the population parameter θ from the expected parameter θ_0, we formulate a two-sided alternative hypothesis $H_A : \theta \neq \theta_0$ and the null hypothesis $H_0 : \theta = \theta_0$. To calculate the p-value, we first compute the absolute value of the deviation of the observed statistic $\hat{\theta}_{\mathbf{x}}$ from the theoretical parameter $\theta_0 = \texttt{th0}$, which we denote $\texttt{dev} = |\hat{\theta}_{\mathbf{x}} - \theta_0|$. We then interpret the statement "$\hat{\theta}_{\mathbf{x}}$ or more extreme" to include all values of the sampling distribution $\widehat{\Theta}_0$ whose deviation from the theoretical

mean θ_0 is greater than the observed deviation `dev`:

$$\begin{aligned} p &= \Pr_{\widehat{\Theta}_0}\Big(\{|\widehat{\Theta}_0 - \theta_0| \geqslant \underbrace{|\hat{\theta}_{\mathbf{x}} - \theta_0|}_{\texttt{dev}}\}\Big) \\ &= \Pr_{\widehat{\Theta}_0}\Big(\{\widehat{\Theta}_0 \leqslant \theta_0 - \texttt{dev}\}\Big) \;+\; \Pr_{\widehat{\Theta}_0}\Big(\{\widehat{\Theta}_0 \geqslant \theta_0 + \texttt{dev}\}\Big) \\ &= F_{\widehat{\Theta}_0}(\theta_0 - \texttt{dev}) \;+\; \big(1 - F_{\widehat{\Theta}_0}(\theta_0 + \texttt{dev})\big) \\ &= \texttt{rvTheta0.cdf(th0-dev)} + (1 - \texttt{rvTheta0.cdf(th0+dev)}). \end{aligned}$$

When doing hypothesis testing based on analytical approximations, it is very common to perform a pivotal transformation on the test statistic $\theta_{\mathbf{x}} = g(\mathbf{x})$, then calculate p-values in terms of one of the standard reference distributions, as we'll show next.

Pivotal quantities and standard reference distributions

We can use pivotal transformations to standardize the test statistics we use for hypothesis testing. We obtain a *standardized test statistic*, by starting from the observed value of an estimator $\hat{\theta}_{\mathbf{x}}$, subtracting a location parameter, and dividing by a scale parameter:

$$\text{test statistic} = \frac{\hat{\theta}_{\mathbf{x}} - \texttt{loc}}{\texttt{scale}}.$$

The constants `loc` and `scale` are determined from the properties of the sampling distribution under null hypothesis $\widehat{\Theta}_0$. Recall we saw pivotal transformations earlier in Section 3.1 and again in Section 3.2 when we used them to construct confidence intervals.

The three test statistics (location-scale *pivotal quantities*) that we'll study in this section are:

- The z-statistic: $z_{\mathbf{x}} = \frac{\overline{\mathbf{x}} - \mu_{X_0}}{\mathbf{se}_{\overline{\mathbf{x}},0}}$, where $\mathbf{se}_{\overline{\mathbf{x}},0} = \frac{\sigma_{X_0}}{\sqrt{n}}$.
- The t-statistic: $t_{\mathbf{x}} = \frac{\overline{\mathbf{x}} - \mu_{X_0}}{\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}}$, where $\widehat{\mathbf{se}}_{\overline{\mathbf{x}}} = \frac{s_{\mathbf{x}}}{\sqrt{n}}$.
- The chi-square statistic: $q_{\mathbf{x}} = s_{\mathbf{x}}^2 / \big(\frac{\sigma_{X_0}^2}{n-1}\big)$.

Using the test statistics $z_{\mathbf{x}}$, $t_{\mathbf{x}}$, and $q_{\mathbf{x}}$ simplifies p-value calculations, since their sampling distributions under the null hypothesis are standard reference distributions.

Standard reference distributions The sampling distribution of a standardized test statistic under the null hypothesis is obtained

by applying the same location-scale pivotal transformation to the sampling distribution of the estimator $\widehat{\Theta}_0$:

$$\text{sampling distribution of the test statistic} = \frac{\widehat{\Theta}_0 - \texttt{loc}}{\texttt{scale}}.$$

The sampling distributions of the test statistics $z_{\mathbf{x}}$, $t_{\mathbf{x}}$, and $q_{\mathbf{x}}$ for samples of size n under the null hypothesis are described by the following standard reference distributions:

- $Z_0 = \frac{\overline{\mathbf{X}}_0 - \mu_{X_0}}{\mathbf{se}_{\overline{\mathbf{x}},0}} \sim \mathcal{N}(0,1)$
- $T_0 = \frac{\overline{\mathbf{X}}_0 - \mu_{X_0}}{\widehat{\mathbf{se}}_{\overline{\mathbf{x}}}} \sim \mathcal{T}(\nu = n-1)$
- $Q_0 = S^2_{\mathbf{X}_0} / \left(\frac{\sigma^2_{X_0}}{n-1}\right) \sim \chi^2(\nu = n-1)$

The random variables Z_0, T_0, and Q_0 are distributed according to *standard reference distributions* with location zero and scale one: the standard normal $Z_0 \sim \mathcal{N}(\texttt{loc}=0, \texttt{scale}=1)$, the standard Student t-distribution with $n-1$ degrees of freedom $T_0 \sim \mathcal{T}(\texttt{df}=n-1, \texttt{loc}=0, \texttt{scale}=1)$, and the standard chi-square distribution with $n-1$ degrees of freedom $Q_0 \sim \chi^2(\texttt{df}=n-1, \texttt{loc}=0, \texttt{scale}=1)$. The effect of the location-scale transform is to remove the dependence on the model-specific parameters like μ_{X_0} and σ_{X_0}, and extract a generic model for the "shape" of the sampling distribution.

We use the subscript $_0$ on all sampling distributions as a reminder that these models describe the variability of the test statistic under the null hypothesis.

Calculating p-values for standardized test statistics

Pivotal transformations simplify p-value calculations, since the standard reference distributions are widely available in all kinds of software, including Excel. It is also possible to use lookup tables (you might have to do this on an exam).

Suppose we have obtained the sample $\mathbf{x}$ from an unknown population $X \sim \mathcal{N}(\mu, \sigma)$, and computed the observed statistic $\hat{\theta}_{\mathbf{x}} = g(\mathbf{x})$. Next, we apply the location-scale transform on the observed statistic $\hat{\theta}_{\mathbf{x}}$ in order to convert it to a standardized test statistic:

$$t_{\mathbf{x}} = \frac{\hat{\theta}_{\mathbf{x}} - \texttt{loc}}{\texttt{scale}},$$

where `loc` and `scale` are two constants that depend on the sampling distribution $\widehat{\Theta}_0$. The sampling distribution of the t-statistic under the

3.5 Two-sample hypothesis tests

We now turn our attention to the task of comparing two samples to determine if they come from the same population or from different populations. This is one of the most important topics in this book, because it represents the most common type of analysis that researchers perform on a regular basis.

As a researcher, you want to show that the average outcome for one population (the intervention group) differs from a reference population (the control group). For example, you might be comparing the health outcomes for patients that received some drug, versus patients who received a placebo, and you want to show that the drug has a beneficial effect on the health outcome.

The null hypothesis we use when comparing two populations is a skeptical claim of "no difference" between the populations. In the medical example, this would correspond to the drug having no effect (not better than a placebo). Other examples of scenarios where we compare samples from two different populations include the `debate` and `lecture` curriculum variants and the East End versus West End electricity prices. In each of these scenarios, we want to show that a difference between the two groups exists, which we can do if we can reject the null hypothesis of "no difference" between the two populations.

In this section, we'll focus on the statistical analysis for the difference between means and show two different methods for computing p-values under the "no difference between populations" null hypothesis: the *permutation test* and the *two-sample t-test*. You'll have to learn some new math formulas and computational tricks specific to the statistical comparison of two populations, but the overall hypothesis testing procedure will be exactly the same as we've seen in the previous two sections. Yes, you're about to watch a third rerun of the hypothesis testing show! But no worries, your TV is not broken. I've intentionally chosen this repetitive programming for you, so that you can get used to the complicated hypothesis testing procedure and start to see it as boring and obvious.

3.5.1 Definitions

We'll start with the notation and terminology we use for the statistical comparison of two populations.

- *Populations*. We call the two unknown populations X and Y, and assume they are described by the model family $\mathcal{M}$ with a mean parameter: $X \sim \mathcal{M}(\mu_X)$ and $Y \sim \mathcal{M}(\mu_Y)$.

- *Parameters*. The means of the two populations μ_X and μ_Y are unknown. We want to know if μ_X and μ_Y are the same or different.
- *Samples*. We have obtained two samples $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ using random sampling from the two populations X and Y.
- *Test statistic*. We'll use the difference between sample means estimate $\hat{d} = \bar{\mathbf{x}} - \bar{\mathbf{y}}$ as the test statistic.
- *Estimator* $\mathbf{dmeans}$ = `dmeans`. The function we use to compute the difference-between-means test statistic $\hat{d} = \mathbf{dmeans}(\mathbf{x}, \mathbf{y}) = \bar{\mathbf{x}} - \bar{\mathbf{y}}$, from the two samples $\mathbf{x}$ and $\mathbf{y}$.
- *Sampling distribution under the null* $\widehat{D}_0 \stackrel{\text{def}}{=} \mathbf{dmeans}(\mathbf{X}_0, \mathbf{Y}_0)$. The difference between means computed from two random samples $\mathbf{X}_0$ and $\mathbf{Y}_0$ under the null hypothesis $H_0 : \mu_X = \mu_Y$.

We've already worked with the difference-between-means estimator $\mathbf{dmeans}$ = `dmeans` in Section 3.1 and again when we learned about confidence intervals in Section 3.2, so don't expect any new math. In this section, we'll describe a hypothesis testing procedure for determining if the observed difference between sample means $\hat{d} = \mathbf{dmeans}(\mathbf{x}, \mathbf{y})$ computed from the samples $\mathbf{x}$ and $\mathbf{y}$ could have occurred by chance according to the sampling distribution of the difference between means $\widehat{D}_0$ under the null hypothesis of "no difference" between populations.

3.5.2 Comparing two populations

Our starting point is a *research question* about the difference between two populations. For example, Charlotte wants to show that teaching using the `debate` curriculum format is more effective than using the `lecture` curriculum format. She measured the effectiveness of the two curriculums using the final scores of students who took the two variants of the curriculum.

Charlotte formulates her research question as a *statistical question* about the means of two unknown populations. Specifically, she assumes that the final score distributions for the `debate` and `lecture` curriculums are described by the population models $X_D \sim \mathcal{M}(\mu_D)$ and $X_L \sim \mathcal{M}(\mu_L)$, with unknown means μ_D and μ_L. Charlotte's claim that the `debate` curriculum is better than the `lecture` curriculum translates to the equation $\mu_D > \mu_L$, where μ_D is the average final score for a theoretical population of all students who follow the new `debate` curriculum in her class, and μ_L is the average final score of students who get the usual `lecture` format.

The population parameters μ_D and μ_L are unknown, but we can estimate them by calculating the sample means $\bar{\mathbf{x}}_D$ and $\bar{\mathbf{x}}_L$ computed from samples $\mathbf{x}_D$ and $\mathbf{x}_L$ collected from the two populations.

Hypotheses

The main scientific question we'll want to answer is whether the two populations are the same or different. To answer this question we'll analyze the following two competing statistical hypotheses:

$$H_A : \mu_X \neq \mu_Y \qquad \text{and} \qquad H_0 : \mu_X = \mu_Y.$$

The alternative hypothesis H_A claims there is a difference between the parameters of the two populations, which means either $\mu_X > \mu_Y$ or $\mu_X < \mu_Y$. We use the "not-equal to" symbol $\neq$ (two-sided inequality) since it includes both possibilities. In contrast, the null hypothesis H_0 claims that the two populations have the same mean $\mu_X = \mu_Y$. We'll use the notation μ_0 to represent the unknown, common mean of the two populations: $\mu_0 = \mu_X = \mu_Y$ under the null hypothesis.

For example, Charlotte's comparison between the `debate` and `lecture` curriculum variants can be stated as the hypotheses:

$$H_A : \mu_D \neq \mu_L \qquad \text{and} \qquad H_0 : \mu_D = \mu_L.$$

Charlotte wants to show that students who follow the `debate` curriculum will achieve higher scores (on average) than students who follow the `lecture` curriculum ($\mu_D > \mu_L$), but she's also open to the possibility that the `debate` curriculum actually leads to lower scores ($\mu_D < \mu_L$). This is why she formulates a two-sided alternative hypothesis.

In contrast, the null hypothesis claims there will be "no difference" between students' scores ($\mu_D = \mu_L$). The null hypothesis represents the viewpoint of a skeptical colleague who says the type of curriculum (`debate` or `lecture`) will make no difference in students' final scores.

Probability model under the null hypothesis

The *null probability model* is the probability model that assumes the null hypothesis is true: $H_0 : \mu_X = \mu_Y$. The null hypothesis describes the viewpoint of a skeptical colleague that believes there is **no difference between the two populations**. According to the skeptical colleague, any difference observed between the samples $\mathbf{x}$ and $\mathbf{y}$ is due to chance. In Charlotte's case, the null hypothesis $H_0 : \mu_D = \mu_L$ describes the "no difference" between mean scores,

so whether students followed the `debate` curriculum or the `lecture` curriculum doesn't influence their scores.

The sampling distribution of the estimator **dmeans** under the null model describes the type of differences between means we can expect to observe for two random samples $\mathbf{X}_0$ and $\mathbf{Y}_0$:

$$\widehat{D}_0 = \mathbf{dmeans}(\mathbf{X}_0, \mathbf{Y}_0).$$

We use the subscript $_0$ throughout this section to denote quantities computed under the assumption that the null hypothesis is true.

Calculating *p*-values The *p-value* is the probability of observing the value of the test statistic *at least as extreme* as the observed test statistic $\hat{d} = \mathbf{dmeans}(\mathbf{x}, \mathbf{y})$, according to the sampling distribution of the estimator $\widehat{D}_0$ under the null hypothesis.

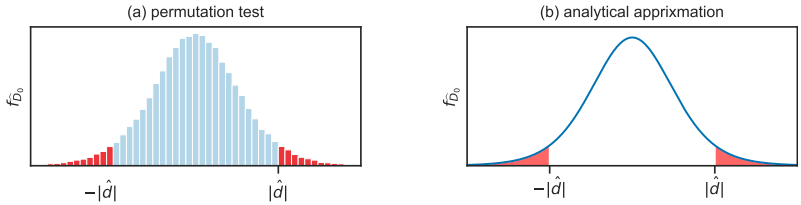


Figure 3.49: Visual representations of the two-tailed p-value calculations associated with the two-sided alternative hypothesis $H_A : \mu_X \neq \mu_Y$. The histogram on the left illustrates the numerical p-value calculation when the sampling distribution $\widehat{D}_0$ is obtained through a computational approach. The plot on the right shows the p-value calculation based on an analytical approximation for the sampling distribution $\widehat{D}_0$.

In this section, we'll show two different methods for computing p-values. In Subsection 3.5.3, we'll describe a clever computational method called the *permutation test*, which gives us a convenient way to simulate observations under the null hypothesis of "no difference" between populations. In Subsection 3.5.5 we'll show an analytical approximation method based on Student's t-distribution. Figure 3.49 shows a preview of the p-value calculations we'll need to perform for the two methods.

Indistinguishability According to the null model, the samples $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ are indistinguishable, since they come from the same distribution $\mathcal{M}(\mu_0)$. This means the grouping into an $\mathbf{x}$-sample and a $\mathbf{y}$-sample is of no importance, and we might as well think of the data as a single long sample with $n + m$ values $(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_m)$. Essentially, we can

ignore whether the values come from sample $\mathbf{x}$ or sample $\mathbf{y}$, since, according to the null hypothesis H_0, both populations have the same distribution: $X \sim \mathcal{M}(\mu_0)$ and $Y \sim \mathcal{M}(\mu_0)$.

Probability model under the alternative hypothesis

The probability model under the alternative hypothesis describes two populations $X \sim \mathcal{M}(\mu_X)$ and $Y \sim \mathcal{M}(\mu_Y)$, whose means are different: $H_A : \mu_X \neq \mu_Y$. In Charlotte's case, the probability model under the alternative hypothesis $H_A : \mu_D \neq \mu_L$ claims that the average score of students who follow the two curriculum variants is different.

Effect size The *effect size*, denoted Δ (the Greek letter delta), describes the magnitude of the difference between the two populations. When doing a hypothesis test for the difference between means, we define the effect size as the difference between the means of the two unknown populations:

$$\Delta = \mu_X - \mu_Y.$$

In Charlotte's case, the difference between the mean scores of the two populations $\Delta \stackrel{\text{def}}{=} \mu_D - \mu_L$ measures the improvement (or decrease) in the scores we can expect to observe as a result of adopting the `debate` curriculum instead of the `lecture` curriculum.

Estimated effect size The observed difference between sample means $\hat{d} = \mathbf{dmeans}(\mathbf{x}, \mathbf{y})$ is a point estimate for the effect size Δ.

Confidence interval for the effect size We can construct a $(1-\alpha)$-*confidence interval* for the effect size $\mathbf{ci}_{\Delta,(1-\alpha)} = [\mathbf{l}_\Delta, \mathbf{u}_\Delta]$, which is a range of numbers that is likely to contain the true effect size Δ. We studied this in Section 3.2. Recall the helper function `ci_dmeans` for constructing confidence intervals (see code block 3.2.21 on page 93). The function `ci_dmeans` will come in handy for the statistical analyses we'll perform in this section.

3.5.3 Permutation tests

The *permutation test* is a computational technique for approximating the sampling distribution under a null hypothesis that claims there is no difference between two populations. By repeatedly "shuffling" the values between the two observed samples $\mathbf{x}$ and $\mathbf{y}$, we can

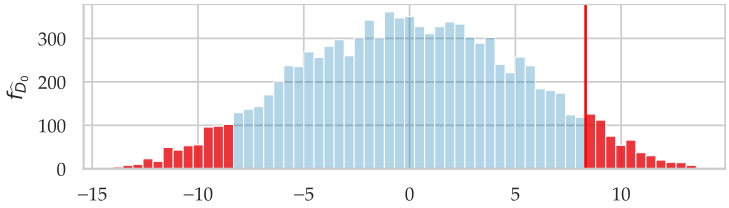


Figure 3.51: The p-value calculation of the observed difference between means $\hat{d}_s = 8.32$ under the sampling distribution $\widehat{D}_0$ assuming H_0 is true.

The p-value is higher than the cutoff $\alpha = 0.05$, which means we don't have enough evidence to reject the null hypothesis.

I know what you're thinking: this must be super disappointing for Charlotte! She put in so much effort to prepare two variants of the lectures for her class, then did "honest work" to perform the hypothesis test, and in the end all she got was a non-significant result. Unfortunately, this is how the world works sometimes, especially when we're working with small sample sizes and the data has a lot of variability.

Would Charlotte have obtained a significant result if she used a larger sample size? In Section 3.6, we'll learn how the sample size influences the *power* of a statistical experiment, which is the probability of detecting a difference between the two populations when it exists.

3.5.4 Exercises

E3.32 Use the permutation test to check for a difference between sleep scores for rural and urban doctors in the doctors dataset.

E3.33 Use the permutation test to compare the `treat` and `ctrl` groups in the `iqs2` dataset.

3.5.5 Analytical approximations

We'll now describe the *two-sample t-test* for the difference between means, which is an analytical approximation method for testing against the no-difference-between-means null hypothesis. We have obtained two samples $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ from two normally distributed populations $X \sim \mathcal{N}(\mu_X, \sigma_X)$ and $Y \sim \mathcal{N}(\mu_Y, \sigma_Y)$. The means μ_X and μ_Y and the standard deviations σ_X and σ_Y are unknown parameters, so we'll have to estimate them from the observed samples.

3.6 Statistical design and error analysis

The outcome of a hypothesis test is a decision: either we *reject* the null hypothesis or we *fail to reject* the null hypothesis. This decision is either *correct* or *incorrect*, depending on whether an effect really exists. The goal of the hypothesis testing procedure is to reject the null hypothesis when an effect exists, and fail to reject the null hypothesis when no effect exists. However, it is possible for the test to produce an incorrect decision: rejecting the null when in fact no effect exists, or failing to reject the null when an effect exists.

Consider the two-sample t-test that we use to compare samples from two populations. The alternative hypothesis states that the samples come from *different* populations, while the null hypothesis states that the samples come from the *same* population. One type of incorrect decision is to reject H_0 when in fact the samples come from the same population. Another type of incorrect decision is failing to reject H_0 when the populations are different.

In this section, we'll analyze the hypothesis testing decision rule and quantify the probabilities of different types of errors that can occur. We'll learn about the *statistical design* calculations that we need to perform during the planning phase of a scientific study. You're already familiar with the technical procedures of hypothesis testing (test statistics and calculations based on sampling distributions), so there will be no new concepts on that front. Our focus will be on describing the "quality guarantees" that we can provide for the results of hypothesis testing procedures.

3.6.1 Definitions

We've already seen the main concepts of hypothesis testing in the previous sections. The *alternative hypothesis* H_A describes the presence of some effect, like a discrepancy between a population and a theoretical model or a difference between two populations. The *null hypothesis* H_0 describes the contrary claim that no effect exists.

We know how to compute a *test statistic* (z, t, q) from the sample and compare it to the *sampling distribution* under the null hypothesis (Z_0, T_0, Q_0) to obtain a *p-value*. The p-value is the probability of observing a value of the test statistic *at least as extreme* as the one you calculated from your sample under the null hypothesis. We then make a decision whether to reject H_0 by comparing the p-value to a predetermined cutoff value α (usually $\alpha = 0.05$). These concepts will also be used as building blocks for statistical design.

In this section, we'll carefully analyze the hypothesis testing decision process. Specifically, we'll describe a new, simplified decision

rule based on test statistics instead of p-values, then analyze the probabilities of making correct and incorrect decisions based on the new decision rule. Let's start with a list of all the concepts and terminology we'll use in this section.

- *False positive rate* α (Type I error rate). This is the probability of rejecting the null hypothesis when no effect exists,

$$\alpha = \Pr\big(\text{reject } H_0 \mid \text{ no effect exists}\big).$$

  A widespread convention is to choose $\alpha = 0.05$ as the Type I error rate, which means there is a 1-in-20 chance that the statistical test will incorrectly report a difference (reject H_0), when in fact no difference exists.
- *Critical value* CV_α. This is the smallest value of the test statistic that will lead us to reject H_0. The critical value CV_α is computed from the cutoff value α and the sampling distribution of the test statistic under the null hypothesis.
- *Rejection region of the test statistic*. This is the set of values of the test statistic that will lead us to reject H_0. The critical value CV_α is the boundary of the rejection region.
- *False negative rate* β (Type II error rate). This is the probability that the test will not reject H_0 even though an effect exists:

$$\beta = \Pr\big(\text{fail to reject } H_0 \mid \text{ effect exists}\big).$$

  Calculating the false negative rate of a statistical test requires making an assumption about the magnitude of the effect size Δ under the alternative hypothesis H_A.
- *Power* $(1 - \beta)$. The *statistical power* of a hypothesis test is the probability of correctly detecting an effect if it exists,

$$\text{power} = (1 - \beta) = \Pr\big(\text{reject } H_0 \mid \text{ effect exists}\big).$$

  The statistical power of the test $(1 - \beta)$ is the complement of the false negative rate β. This is also known as the *sensitivity* of the test—how good it is at detecting an effect when it really exists.
- *Effect size* Δ. The parameter Δ describes the effect size we expect to observe under the alternative hypothesis H_A.
  - ▻ In a one-sample test for comparing the unknown mean μ to the null reference value μ_0, we can use the effect size $\Delta \stackrel{\text{def}}{=} \mu - \mu_0$ as a measure of the deviation from the null.
  - ▻ In a two-sample test for comparing two unknown population means μ_X and μ_Y, we can use the difference-between-means effect size $\Delta \stackrel{\text{def}}{=} \mu_X - \mu_Y$.

The true effect size Δ is unknown, so we usually try to estimate it either by making a guess, by looking at the estimate obtained from previous research, by choosing the smallest effect size of interest, or from a theoretical prediction.

- *Standardized effect size d*. We often express the effect size in terms of Cohen's d, which divides the raw effect size Δ by the population standard deviation σ to obtain a standardized metric.
  - ▻ For a one-sample tests that compares the unknown mean μ to the reference value μ_0, we use Cohen's d effect size $d \stackrel{\text{def}}{=} \frac{\mu - \mu_0}{\sigma}$, where σ is the population standard deviation.
  - ▻ When comparing two population means, we can use Cohen's d effect size $d \stackrel{\text{def}}{=} \frac{\mu_X - \mu_Y}{\sigma}$, which computes the difference between means divided by the common population standard deviation σ.
- *Sample size n*. The size of the sample we use for the analysis influences the critical value and the error rates of the test. The larger the sample, the smaller the variability of the sample statistics, making the sampling distributions narrower (smaller standard errors) and overlap less. For a fixed α, using larger n reduces β and increases the power $(1 - \beta)$.
- *Statistical design*. The process of choosing the four parameters α, β, Δ, and n of a hypothesis testing procedure. Statistical design involves making trade-offs between the parameters we want the test to have, and what we can get.

In the remainder of the section, we'll explain how to use statistical design calculations to prepare hypothesis tests with guarantees on the two types of errors.

3.6.2 Hypothesis testing decision rules

The goal of hypothesis testing is to distinguish between H_0 (no effect) and H_A (an effect exists). The outcome of the hypothesis test is a binary decision: either we reject H_0 or we fail to reject H_0.

Decision rule based on p-values (Fisher)

The procedure that we used in the previous sections to decide whether to reject H_0 involves the following steps:

- Choose the cutoff value α (Type I error rate) for the test.
- Collect the data sample $\mathbf{x}$.
- Calculate the test statistic $t_{\mathbf{x}}$ from the observed sample $\mathbf{x}$.

- Obtain the sampling distribution T_0 of the test statistic under the null hypothesis H_0.
- Calculate the p-value of $t_{\mathbf{x}}$ under T_0.
- Compare the p-value to the cutoff α to make a decision whether to reject H_0 or not.

We reject H_0 if the p-value is smaller than the cutoff value α, which means we deem the observed value of the test statistic is unlikely to occur under the null hypothesis. Otherwise, we fail to reject H_0, which means the observed test statistic is consistent with the null hypothesis. The entire procedure can be summarized as follows:

code 3.6.1
```
>>> alpha = ...  # choose cutoff value in advance
# COLLECT DATA SAMPLE
>>> obst = ...   # calculate test statistic from the sample
>>> rvT0 = ...   # sampling distribution under H0
>>> pvalue = ... # calculated from obst under rvT0
>>> if pvalue <= alpha:
        decision = "Reject H0"
    else:
        decision = "Fail to reject H0"
```

We choose the cutoff value α in advance, then perform all the probability calculations once we have obtained the sample $\mathbf{x}$.

A natural question to ask (for statisticians) is whether we can simplify the decision rule, by performing some of the probability calculations *before* observing the sample $\mathbf{x}$?

Simplified decision rule (Neyman–Pearson)

It turns out that we can do all the probability calculations before observing the sample. This approach leads us to a simplified decision rule that asks us to compare the test statistic to a *critical value* CV_α that is **computed before seeing the data** sample $\mathbf{x}$, based on some choice of the cutoff value α and the sampling distribution of the test statistic under the null hypothesis. The hypothesis test using the simplified decision rule can be summarized by the following code:

code 3.6.2
```
>>> alpha = ...     # chosen in advance
>>> rvT0 = ...      # sampling distribution under H0
>>> CV_alpha = ... # calculated from alpha and rvT0
# COLLECT DATA SAMPLE(S)
>>> obst = ...      # calculated from sample
>>> if obst >= CV_alpha:
        decision = "Reject H0"
    else:
        decision = "Fail to reject H0"
```

The main thing I want you to notice is that we calculate the critical value $\text{CV}_\alpha =$ `CV_alpha` for the test statistic *before* collecting the data sample $\mathbf{x}$. In other words, we're doing all the probability calculations

required to perform the test upfront in a generic way that will allow us to run the hypothesis testing procedure for any sample we might observe $\mathbf{x}$. Once we observe a particular sample $\mathbf{x}$, we'll simply need to calculate the test statistic $t_{\mathbf{x}} = \texttt{obst}$ and compare it to the critical value CV_α to reach a decision.

The critical value CV_α depends on the direction of the alternative hypothesis. For the sake of simplicity, we'll discuss only the "greater than" alternative hypothesis $H_A : \theta > \theta_0$ in the remainder of this section. We defer the discussion of the "less than" and "two-sided" hypothesis testing scenarios until later in this section. To calculate the critical value CV_α, we must answer the following question: "What value of the test statistic $t_{\mathbf{x}} = \texttt{obst}$ would produce a right-tail p-value equal to α?" This corresponds to the equation

$$\alpha \quad = \quad \Pr_{T_0}(\{T_0 \geqslant \text{CV}_\alpha\}) \quad = \quad 1 - F_{T_0}(\text{CV}_\alpha).$$

We can rewrite this as $F_{T_0}(\text{CV}_\alpha) = 1 - \alpha$ and solve for CV_α by applying the inverse cumulative distribution function $F_{T_0}^{-1}$, which gives us

$$\text{CV}_\alpha \quad = \quad F_{T_0}^{-1}(1 - \alpha).$$

The decision to reject H_0 if the p-value is less than α is equivalent to rejecting H_0 by comparing the observed test statistic $t_{\mathbf{x}}$ to the critical value CV_α. We're just moving the decision from the space of p-values to the space of test statistics, but we're essentially making the same comparison.

Figure 3.54 shows a visualization of the decision rule performed based on the test statistic $t_{\mathbf{x}}$. We reject H_0 if $t_{\mathbf{x}}$ is equal or greater than the critical value CV_α, otherwise we fail to reject H_0. The set of values of the test statistic that lead us to reject the null hypothesis is called the *rejection region* for the test.

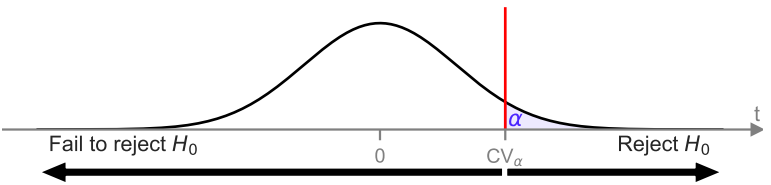


Figure 3.54: Illustration of the *rejection region* used to make a decision as part of the simplified decision rule. If the observed value of the test statistic $t_{\mathbf{x}} = \texttt{obst}$ is above the critical value CV_α, then we reject H_0. If the test statistic is below the critical value, then we fail to reject H_0.

Let's summarize the steps of the simplified procedure for reaching a decision when running a hypothesis test:

Recall we've seen this parameter in the previous sections, when we used the value $\alpha = 0.05$ as the cutoff for observations that we deemed unlikely to have occurred by chance under the null hypothesis. Choosing the cutoff value $\alpha = 0.05$ means we have a 5% chance of rejecting the null hypothesis when it is true.

If we want to reduce the Type I error rate, one obvious solution would be to use a smaller cutoff value α. For example, choosing $\alpha = 0.01$ means we have only 1 chance in 100 to incorrectly reject the null hypothesis. This lower value of α will lead to a higher critical value $\text{CV}_{0.01} > \text{CV}_{0.05}$, which means we'll only reject the null hypothesis if the observed test statistic is very large. We'll be less likely to incorrectly reject the null when it is true, but also less likely to reject the null even when an effect exists.

False negative error rate β and power $(1 - \beta)$. The *Type II* error rate describes the probability that the test will fail to reject the null hypothesis when an effect exists (H_A is true). We use the symbol β for this probability:

$$\beta = \Pr\big(\text{fail to reject } H_0 \mid H_A \text{ is true}\big).$$

Statisticians more commonly talk about the Type II error rate in the form of its complement, the *statistical power*. The *power* of a test, denoted $(1 - \beta)$, describes the ability of the test to correctly detect the effect when it exists:

$$\text{power} \stackrel{\text{def}}{=} (1 - \beta) = \Pr\big(\text{reject } H_0 \mid H_A \text{ is true}\big).$$

For example, the Type II error rate $\beta = 0.2$ corresponds to power of $1 - 0.2 = 0.8 = 80\%$, which means we have 80% chance of correctly rejecting the null hypothesis when H_A is true. The power is sometimes called the *sensitivity* of the test.

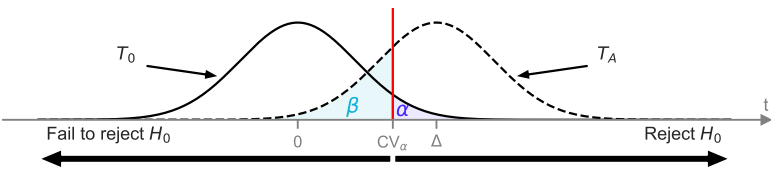


Figure 3.55: The sampling distributions of the test statistic under the null hypothesis (solid line) and the alternative hypothesis (dashed line). The critical value CV_α is the boundary of the rejection region. The tails of the distributions that correspond to Type I and Type II errors are highlighted.

Figure 3.55 illustrates the Type I and Type II error rates of a statistical test. The parameter Δ is an assumption about the effect size under the alternative hypothesis H_A. The critical value CV_α

(determined based on the desired Type I error rate α) affects the Type II error rate β. If we move CV_α to the right, we would decrease α but increase β. If we move CV_α to the left, we would increase the false positive rate α but gain more power $(1 - \beta)$.

Influence of effect size on power Figure 3.56 illustrates the different amounts of "overlap" between the sampling distribution under H_0 and H_A for four different effect sizes. We have chosen $\alpha = 0.05$ to obtain the critical value CV_α in each case. In panel (a), the effect size Δ is small, which means there is a lot of overlap between the sampling distributions under H_0 and H_A, and the probability of making a Type II error is huge (the test has low power). Panel (b) shows a medium effect size, which leads to smaller Type II error rate. In panel (c), we see a large effect size, which leads to a small Type II error. In panel (d), the effect size is very large so there is almost no chance of making a Type II error.

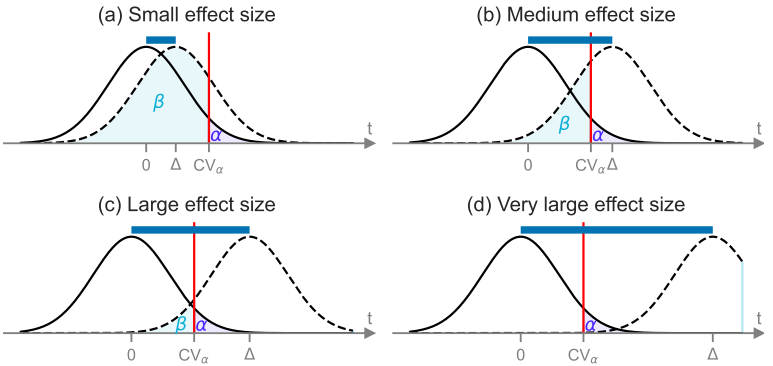


Figure 3.56: Plots of the probability of Type II errors (area labelled β) for different effect sizes Δ. The magnitude of the effect size is visually indicated by the thick line above each plot. The choice of α is kept constant at $\alpha = 0.05$.

Influence of sample size on power The choice of sample size n affects the power of a test. Using large sample sizes reduces the variance of the sampling distributions and makes the distribution under the alternative hypothesis easier to distinguish from the sampling distribution under the null hypothesis.

Figure 3.57 shows the effect of sample size on the Type II error rate of two hypothesis tests designed with the same $\alpha = 0.05$ and the same effect size Δ. Larger sample sizes reduce the widths of sampling distributions (recall the central limit theorem). When the sample size is $n = 30$, there is less overlap between the sampling distributions, and hence a lower Type II error rate, assuming the choice of the Type I error rate α is kept fixed.

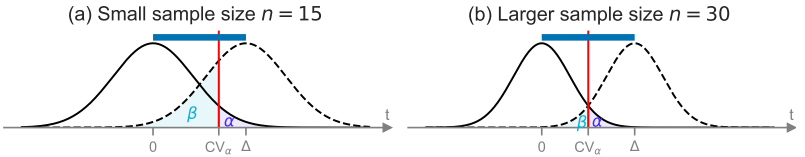


Figure 3.57: Influence of n on the error rates α and β. The larger sample size $n = 30$ makes the sampling distributions narrower and overlap less, which reduces the Type II error rate β for the same choice of α.

Balancing significance and power The parameters α and β play a central role in any statistical analysis based on hypothesis testing. Choosing a small significance level α gives us confidence that, when we detect a pattern (reject H_0), we have really seen some discrepancy from the null. However, choosing a small α tends to reduce the power of the test, which means we might incorrectly fail to reject H_0 when in fact an effect exists. In contrast, using a large α like $\alpha = 0.1$ leads to a higher chance of making a Type I error, but increases the power of the test, as shown in Figure 3.58.

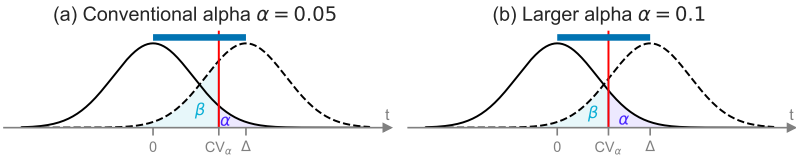


Figure 3.58: Increasing α reduces the Type II error rate β. Using $\alpha = 0.1$ decreases the β relative to the β of a test based on the cutoff value $\alpha = 0.05$. The effect size Δ and the sample size n are the same in both figures.

The takeaway message from all these figures is that every statistical analysis is a balancing act between the choices of the error rates α and β, the sample size n, and the effect size Δ. The process of choosing these parameters is called *statistical design*.

3.6.3 Statistical design

The term *statistical design* refers to the planning calculations we do **before collecting the data** for a study. Statistical design calculations are required to make sure the statistical analysis we perform on the data will have the desired characteristics. For example, if we want the statistical analysis to have Type I error α and Type II error β (assuming the effect size is Δ), then the statistical design process will tell us the size of the sample(s) we need to collect. Statistical design allows us to make different kinds of trade-offs between the parameters α, β, Δ, and n, depending on the specific goals of the statistical analysis.

3.7 Inventory of statistical tests

During the past 100 years, statisticians developed dozens of statistical "recipes" for use in different data analysis scenarios. Each of these recipes is an instance of the hypothesis testing procedure that we studied in the past four sections, so you're already familiar with the general procedure. The only new material is the specifics of the data analysis questions these recipes are used to answer.

We don't have the room to cover all recipes in detail, so we'll present the 20 most common statistical analysis recipes in condensed form. For each hypothesis testing recipe, we'll describe the type of data it is suitable to analyze, state the null and alternative hypotheses, specify the assumptions we're making, give the formulas for computing the test statistic, and show the relevant sampling distribution we use to calculate p-values. We'll also discuss the Python helper functions for performing each of these tests.

You can think of this section as a "recipe book" that contains the 20 most common statistical analysis recipes that you need to know about. Basically, I went through a bunch of statistics textbooks and I compiled an "inventory" of statistical testing recipes normally covered in introductory statistics courses (STATS 101). **You're not expected to read this section from end to end**—you just need to skim through the inventory to know what is available, so you can come back to it whenever you need to perform some specific statistical analysis.

3.7.1 Definitions

Let's start with a review of hypothesis testing concepts and a reminder of the data terminology that we introduced in Chapter 1.

Hypothesis testing context

Consider the *sample* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, which consists of n observations from some unknown population $X \sim \mathcal{M}(\theta_X)$. We assume the sample is *representative* of the population that we're interested in, and the observations x_i are *independent* of each other. The statistical analysis of the sample $\mathbf{x}$ allows us to make inferences about the unknown population X. In particular, we'll compare a test statistic computed from the sample $\mathbf{x}$ to the sampling distribution of this test statistic under the null hypothesis H_0.

Variable types

Let's review the *levels of measurement* terminology that we introduced in Section 1.1. See Figure 1.3 on page 19. The type of data we are analyzing dictates what kind of analysis we can perform on it, so the first question to ask is what type of data we're working with?

Here is a list of the types of data we'll discuss in this section:

- *numerical* data $\mathbf{n} = (n_1, n_2, \ldots, n_n)$, where each n_i is an integer or a decimal number. We'll use the symbol $\mathbf{N}$ to denote numerical variables and subdivide them into:
  - ▷ *normally distributed* numerical variables, for which the population model $\mathcal{N}(\mu, \sigma)$ is an accurate representation of the unknown population. We'll use the symbol $\mathbf{N}_{\mathcal{N}}$ to describe this type of numerical variables.
  - ▷ numerical variables that are *not normally distributed*, which includes all other population distributions (e.g. uniform, exponential), and data with skew, long tails, or outliers.
- *categorical* data $\mathbf{c} = (c_1, c_2, \ldots, c_n)$, where each c_i comes from a finite set of values. We'll use $\mathbf{C}$ to denote categorical variables, and further subdivide categorical variables into three subtypes:
  - ▷ *ordinal categorical* data $\mathbf{o} = (o_1, o_2, \ldots, o_n)$ that can be sorted (categorical variables with relative order between values). We'll refer to ordinal categorical variables simply as *ordinal*, and denote them with the symbol $\mathbf{C}_{\mathcal{O}}$.
  - ▷ *binary categorical* data $\mathbf{b} = (b_1, b_2, \ldots, b_n)$, where each b_i is either 1 or 0. We'll refer to binary categorical variables simply as *binary* and denote them $\mathbf{C}_2$.
  - ▷ *nominal categorical* variables are categorical variables that are not binary and not ordinal. We'll refer to nominal categorical variables using the generic symbol $\mathbf{C}$.

A dataset usually consists of multiple variables organized in a data table. We often study bivariate data that consists of pairs of measurements $[\mathbf{c}, \mathbf{n}] = [(c_1, n_1), (c_2, n_2), \ldots, (c_n, n_n)]$, where each observation (c_i, n_i) is a joint measurement of a categorical variable and a numerical variable. For example, in the electricity prices dataset, each row of the data frame `eprices` consists of a `location` measurement (categorical with possible values East and West), and a `price` measurement (numerical).

In previous sections, we described several statistical analyses for normally distributed numerical variables $\mathbf{N}_{\mathcal{N}}$. In this section, we'll introduce other hypothesis tests for analysis of non-normal numerical $\mathbf{N}$, categorical $\mathbf{C}$, ordinal $\mathbf{C}_{\mathcal{O}}$, and binary $\mathbf{C}_2$ data.

Predictor and outcome variables

The purpose of many statistical analyses is to quantify the dependence between two variables in a dataset. Recall the terminology we use to describe variables, depending on their role in the analysis:

- *predictor variable*: the variable that influences or causes the different outcomes. Predictor variables are also called *explanatory variables*, *independent variables*, or *treatment variables*.
- *outcome variable*: the variable of interest that we suspect is influenced or caused by the predictor variable. Outcome variables are also called *dependent variables* or *response variables*.

We'll use this terminology to categorize the different statistical analysis recipes. More on that in Section 3.7.3.

Categorical variables as groups When the predictor variable is categorical, we often refer to the observations of the different possible values of the categorical variable as groups or subpopulations. In other words, we interpret categorical variables as representing group membership. For example, we can describe the analysis of the East and West electricity prices as a comparison of two groups. Using more fancy statistical language, we would describe the same analysis as the study of the effects of a binary predictor variable (`location`) on a numerical outcome variable (`price`), and represent this type of analysis using the notation "$\mathbf{C}_2 \rightarrow \mathbf{N}$."

Statistical test types

We can categorize the different statistical tests into three categories:

- *Parametric tests*: hypothesis tests that make assumptions about the population probability model family. The parametric tests in this section assume the population is normally distributed $X \sim \mathcal{N}(\mu_X, \sigma_X)$, where the mean μ_X and the standard deviation σ_X are unknown parameters. We refer to these tests as *parametric* because the hypotheses compare the unknown population parameters to some expected theoretical parameter value described by the null model. The z-test and t-tests that we saw in sections 3.4 and 3.5 are examples of parametric tests.
- *Nonparametric tests*: hypothesis tests that don't make any assumptions about the population probability model. The calculations we use to perform nonparametric tests are based on pairwise comparisons between data values (*greater than*, *less than*, or *equal to* comparisons) and ranks (positions within a sorted list). The focus on direct comparisons between the

observed data means nonparametric tests work for data that comes from any distribution, so they are the main tool we use for non-normal numerical data (skewed, long tails, outliers). We can also use nonparametric tests to analyze ordinal data.

- *Resampling methods*: hypothesis tests that reuse observed data to simulate the relevant sampling distribution. The *permutation test* that we studied in Section 3.5 is an example of a resampling method, which simulates the no-difference-between-groups null distribution by repeatedly shuffling the observed data samples between the two groups. Resampling methods are very versatile since they can be used for any type of data and allow us to perform tests based on any statistic.

Most of the test recipes we'll cover in this section are parametric tests, which make one or more assumptions about the unknown population model. In contrast, nonparametric tests (Section 3.7.9) don't make assumptions about the population distribution. Indeed, nonparametric tests were designed for use as a "fallback strategy" when the assumptions required for parametric tests don't apply.

Both parametric and nonparametric tests are based on predefined math formulas that statisticians have developed for use in a specific scenario. In contrast, resampling methods (Section 3.7.10) use a direct, data-driven approach that allows us to simulate the sampling distributions of test statistics, which makes them broadly applicable in any scenario.

Trait		Parametric tests	Nonparametric tests	Resampling methods
Conditions/ Assumptions	Population distribution	Assume the population is normally distributed	Applies to any population distribution	Any
	Works for small sample sizes?	Best (when assumptions met)	Okay	Weak
	Data types	Normally distributed numerical	Non-normally distributed numerical or ordinal	Any
	Power	Higher	Lower	High
Results	Confidence intervals	Yes	Not straightforward	Yes (bootstrap)
	Statistical power calculations	Yes	Not straightforward	No

Figure 3.64: Comparison of the traits of three types of statistical tests.

Figure 3.64 summarizes the different characteristics and properties of the three types of tests we'll discuss in this section.

Assumptions of parametric tests

Parametric test recipes are only valid if certain assumptions about the population distribution and the sample size are satisfied. The three most common assumptions are:

- **(NORM)**: the population is normally distributed. The normality assumption is very restrictive, since it means the statistical analysis is only applicable for populations that are normally distributed, which is a subset of all data.
- **(LARGEn)**: the sample size is "large enough" for normal approximations to be valid. The sampling distributions in data analysis scenarios where the population is not normally distributed (sometimes not even numerical) can be approximated using a normal distribution when the sample size n is large. This is a consequence of the central limit theorem (Section 2.8.4), which tells us that the sampling distribution of the mean computed from *any* distribution is approximately normally distributed, for large enough n. We sometimes refer to the "n large enough" assumption as *asymptotic normality*.

  When working with categorical variables, the **(LARGEn)** assumption is sometimes stated in terms of a minimum frequency (count): "at least 5 expected observations" for each of the possible values of the categorical variable.
- **(EQVAR)**: the variance in different groups is the same. For data analysis scenarios that compare two groups (two unknown populations X and Y), the **(EQVAR)** assumption corresponds to $\sigma_X^2 = \sigma_Y^2$. More generally, when comparing I groups, the assumption is $\sigma_1^2 = \sigma_2^2 = \cdots = \sigma_I^2$. The fancy Greek word *homoscedasticity* (same variability) is another way to describe the **(EQVAR)** assumption.

You need to pay careful attention to these assumptions, so you don't end up using parametric tests outside of their validity context. We'll use the eye-catching symbols **(NORM)**, **(LARGEn)**, and **(EQVAR)** when presenting the test recipes in the remainder of this section to make it easy for you to spot the assumptions required for each test recipe. Using a parametric test on data that doesn't satisfy the required assumptions will lead to completely invalid statistical results. Another way to say this, is that **parametric tests are sensitive to assumptions**.

Checking assumptions There are several formal and informal ways to check if a given sample **x** satisfies the **(NORM)** assumption. Given the importance of assumptions for parametric tests, you

should know about these "distribution checks" so that you can verify that the assumptions required for a particular parametric test are satisfied. For example, you can inspect a histogram of the data $\mathbf{x}$ to see if it shows a unimodal distribution with a central peak and symmetric tails.

Another way to check the **(NORM)** assumption is to visually inspect a Q-Q plot of the data $\mathbf{x}$ against a normal distribution. If the data $\mathbf{x}$ is normally distributed, then the points in the Q-Q plot should all be close to the diagonal line. Look back at the examples in Section 2.7 (page 250) for a reminder of what Q-Q plots look like and how to generate them. Minor deviations from the diagonal are okay, but seeing systematic deviations in the tails tells you that the sample is unlikely to come from a normally distributed population.

There are also hypothesis testing procedures designed to check if a given sample $\mathbf{x}$ comes from a particular distribution, which we'll describe in Section 3.7.12. For example, the *Kolmogorov–Smirnov test* (Section 3.7.12.1) and the *Shapiro–Wilk test* (Section 3.7.12.2) are used to check if a sample comes from a normally distributed population, as required by the **(NORM)** assumption. In general, these tests don't perform very well, so instead we recommend you use the visual inspection of the Q-Q plot as your primary tool for assessing if the **(NORM)** assumption is satisfied or not.

Use nonparametric and resampling methods as a fallback strategy If the **(NORM)** and **(LARGEn)** assumptions required for parametric tests are not valid, then you can use nonparametric tests or resampling methods, since they don't require these assumptions to be true.

Judgment calls You will inevitably run into situations when you need to make a judgment call and decide if a given sample $\mathbf{x}$ satisfies the **(NORM)** assumption, or it doesn't. That's how it goes. Making judgment calls is an integral part of doing statistics. Some of you might feel uneasy in such situations, preferring to use a formal statistical test to make the **(NORM)**-or-not-**(NORM)** decision for you. However, if you gain some experience with the Kolmogorov–Smirnov and Shapiro–Wilk test, you'll quickly see their accuracy and sensitivity are not something to write home about, and realize you're better off making the decision using visual inspection (is the Q-Q plot close to the diagonal).

You'll also have to make judgment calls whether the asymptotic normality assumption **(LARGEn)** holds. To perform a parametric test that depends on the **(LARGEn)** assumption, you need to decide if the sample size n is "large enough." I'm intentionally not giving you any rules to follow in this book, because you have to make

these decisions on a case-by-case basis, depending on the amount of skewness in the data and the presence of outliers. Using statistical procedures is inevitably associated with self-doubt feelings, and questions like "Am I allowed to do this?" regarding specific calculations, and "Am I allowed to say this?" regarding the conclusion we draw. Don't feel weird about having to make such subjective decisions, instead, consider yourself welcomed to the club!

3.7.2 Null hypothesis significance testing procedure

Let's review the structure of the *null hypothesis significance testing* (NHST) procedure that is common to all the statistical test recipes.

The starting point is a *research question* about a *population* of interest, which we formulate as two competing statistical hypotheses: H_0 and H_A. Statistical tests usually involve computing the *observed test statistic* $\hat{\theta}_{\mathbf{x}}$ from the sample $\mathbf{x}$, then comparing $\hat{\theta}_{\mathbf{x}}$ to the *sampling distribution* $\hat{\Theta}_0$ under H_0. The p-value is the probability of observing the test statistic $\hat{\theta}_{\mathbf{x}}$ or more extreme under the distribution $\widehat{\Theta}_0$.

The hypothesis testing recipes differ because of the different types of data, different types of statistical analyses, and the assumptions they make. We'll explain more about those differences in the next section, but for now, we start with the common parts.

Statistical design parameters

The primary design parameters we consider when planning the hypothesis testing procedure include the *sample size* n and the *significance level* α, which is also known as the Type I (false positive) error rate. Additionally, if we make an assumption about the *effect size* we want to detect Δ, we can calculate the *power* of the test $(1-\beta)$, where β is the Type II (false negative) error rate. Recall we discussed the different types of statistical designs in Section 3.6.3 (see page 205).

Steps of the hypothesis testing procedure

The steps of the hypothesis testing procedure are as follows:

1. **Modelling step**. Identify the population of interest and clearly state the research question you want to answer. Formulate two competing hypotheses H_0 and H_A and choose a test statistic (estimator) relevant for distinguishing the two hypotheses.

2. **Design step**. Choose the Type I error rate α for the test and determine the other design parameters β, Δ guess, and n.

3. **Data collection step**. Collect the sample $\mathbf{x}$ from the population of interest. Check if the data you collected satisfies the statistical model assumptions.

4. **Calculation step**. Compute the observed value of the test statistic $\hat{\theta}_{\mathbf{x}}$ from the sample $\mathbf{x}$. Obtain $\widehat{\Theta}_0$, the sampling distribution of the test statistic under the null hypothesis. Calculate the p-value by looking at where the observed $\hat{\theta}_{\mathbf{x}}$ fits within $\widehat{\Theta}_0$.

5. **Decision step**. Make a decision whether to reject H_0 or not by comparing the p-value to the cutoff value α.

6. **Reporting step**. State the conclusion you have reached. Report the value of the test statistic and the p-value. Quantify the discrepancy from the null model by calculating an effect size estimate $\widehat{\Delta}$. Construct confidence intervals for the unknown parameter $\mathbf{ci}_{\theta,\gamma}$ and the effect size $\mathbf{ci}_{\Delta,\gamma}$.

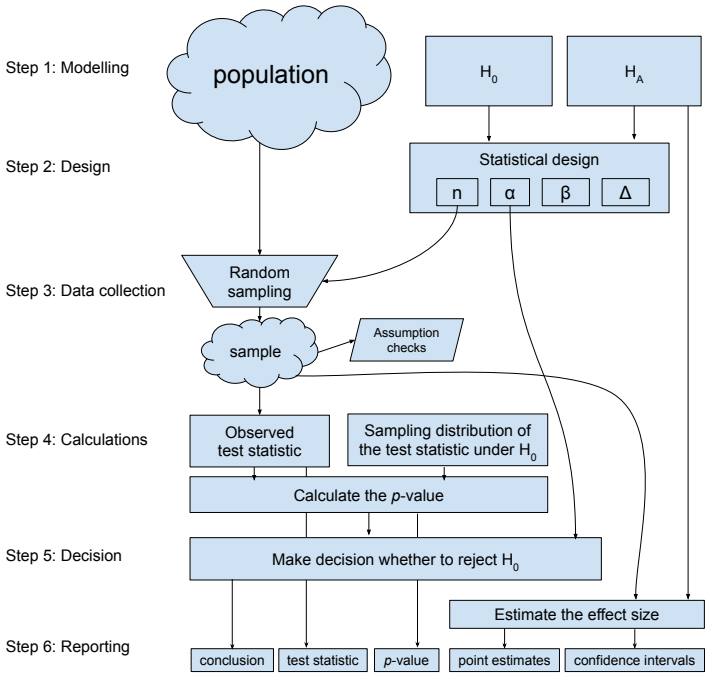


Figure 3.65: The data flows between the six steps of the NHST procedure.

Figure 3.65 shows the data flow between the different steps of the NHST procedure. You've seen these steps several times before, so they should all be familiar to you. The only new component we've

added is the assumptions check sidebar, which we perform to ensure the data we're analyzing satisfies the assumptions of the hypothesis test we want to perform.

The logic of hypothesis testing is based on the calculation of how likely or unlikely the observed test statistic is under H_0. We make the decision whether to reject the null hypothesis by comparing the p-value to the predetermined cutoff value α. Recall the p-value calculation can be based on the left-tail, the right-tail, or both tails of the distribution, depending on the directionality of the alternative hypothesis. For the sake of simplicity, we'll only give the formulas for the two-tailed alternative hypothesis when presenting the recipes in this section.

Hypothesis testing results

The outputs of the reporting step of the hypothesis testing procedure (Step 6) are the following:

- State the conclusion, whether to reject H_0 or fail to reject H_0.
- Report the observed value of the test statistic $\hat{\theta}_{\mathbf{x}}$.
- Report the p-value computed from $\hat{\theta}_{\mathbf{x}}$ under H_0.
- Calculate a point estimate of the effect size $\widehat{\Delta}$.
- Construct a confidence interval of the effect size $\mathbf{ci}_{\Delta,\gamma}$.

Roughly speaking, you want to report all the relevant information that a fellow researcher or colleague might be interested in. The conclusion (reject H_0 or fail to reject H_0) is important, but reporting an estimate of the effect size is even more important, since the effect size quantifies the discrepancy from the null model.

The hypothesis testing recipes we'll present in the remainder of this section all apply in different situations: different data scenarios, different modelling assumptions, different computations, but the overall steps 1 through 6 will be the same for all the recipes.

3.7.3 Categorization of statistical test recipes

A "statistical test recipe" is an instance of the general NHST procedure that applies to a specific type of data, under a specific set of modelling assumptions. We can categorize the statistical testing recipes along several axes: the type of data (numerical, ordinal, or categorical), the statistical analysis type (the relationship between the predictor variable and the outcome variable we're trying to detect), and the probability distribution family of the sampling distribution we use to compute the p-value.

We'll first categorize tests based on **data type** and **analysis type**. We'll then spend the remainder of the section discussing the details of each hypothesis test (data, assumptions, hypotheses, test statistics, etc.), organized into subsections based on the **probability distribution families**.

Categorization based on data types

The type of data we're analyzing determines the kind of statistical analysis recipes that we can use.

- For *normally distributed numerical* variables, denoted $\mathbf{N}_{\mathcal{N}}$, we can use statistical tests based on the sample mean $\bar{\mathbf{x}}$ and the sample standard deviation $s_{\mathbf{x}}$, which are estimates of the population parameters μ_X and σ_X of the unknown population $X \sim \mathcal{N}(\mu_X, \sigma_X)$. We refer to these procedures as *parametric tests*.
- For *numerical* variables $\mathbf{N}$ that are *not normally distributed*, we can't use any of the parametric tests, so instead we must use *nonparametric tests* (Section 3.7.9) or *resampling methods* (Section 3.7.10), which work with all possible distributions, including highly-skewed ones, and datasets that contain outliers.
- When working with *ordinal* data $\mathbf{C}_{\mathcal{O}}$, we can perform greater-than-or-less-than comparison between values, which allows us to compute ranks and medians needed for *nonparametric tests* (Section 3.7.10). We can also treat ordinal data as categorical, and apply the tests used for categorical data.
- For *categorical* data $\mathbf{C}$, the only statistics we can compute are frequencies (counts) of the different observations. The main tool we use for categorical data is the chi-square family of tests (Section 3.7.7), which are based on computing the squared deviations between the observed and expected count data, which can be approximated as a chi-square distribution.
- For *binary categorical* variables $\mathbf{C}_2$, we compute relative frequencies (proportions) and use *proportion tests* (Section 3.7.5).

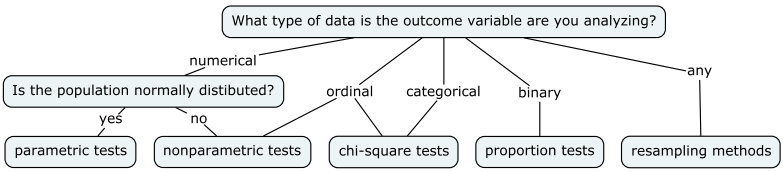


Figure 3.66: The hypothesis tests that we can use for different types of data.

The general rules for choosing which type of test you can use in different situations are summarized in Figure 3.66.

Categorization based on statistical analysis types

We can subdivide the different statistical analysis recipes according to the type of question we want to answer.

- Comparing one group to theory (as seen previously in sections 3.3 and 3.4). The data consists of a single sample $\mathbf{x}$ from an unknown population X, which we want to compare to some known theoretical distribution X_0.
- Comparing two groups (as seen in Section 3.5). The data consists of samples from two groups, and we want to know if the samples come from the same population or from two different populations. For example, we compare the treatment group $\mathbf{x}_t$ and the control group $\mathbf{x}_c$ in a statistical experiment.
- Comparing paired measurements (repeated measures). The data consists of measurements $\mathbf{x}_b$ and $\mathbf{x}_a$ collected from the same set of individuals before and after some intervention.
- Comparing three or more groups. For example, we can compare samples from three groups $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$ to see if they come from the same population or from different populations.
- Distribution checks. Given the sample $\mathbf{x}$, we want to test if it comes from the probability model family $\mathcal{M}$ or not.
- Detecting associations between variables. For example, we can check if two categorical variables are *correlated* or *independent*.

We'll now provide a condensed overview of the statistical analysis recipes available for each of these scenarios and give the exact section reference where you can learn more about each recipe.

Comparing one group to theory Suppose we have obtained one sample $\mathbf{x}$ from an unknown population $X \sim \mathcal{M}(\theta)$ and we want to compare this unknown population to a known theoretical model $X_0 \sim \mathcal{M}(\theta_0)$. This is the original hypothesis testing scenario we studied in Section 3.3, where we introduced *simulation tests* (Section 3.7.10.1), and in Section 3.4, where we studied parametric tests like the *one-sample z-test* (Section 3.7.4.1), the *one-sample t-test* (Section 3.7.6.1), and the *chi-square test for the population variance* (Section 3.7.7.4). All these tests are based on the assumption that the unknown population is normally distributed $X \sim \mathcal{M}(\mu_X, \sigma_X)$.

Figure 3.67 shows the other hypothesis testing recipes that we can use to compare one group to theory. For categorical variables, we can use the *chi-square test for goodness of fit* (Section 3.7.7.1). For binary categorical variables, we can use the *one-sample z-test for proportions* (Section 3.7.5.4). When working with ordinal data or numerical data that doesn't come from a normal population, we

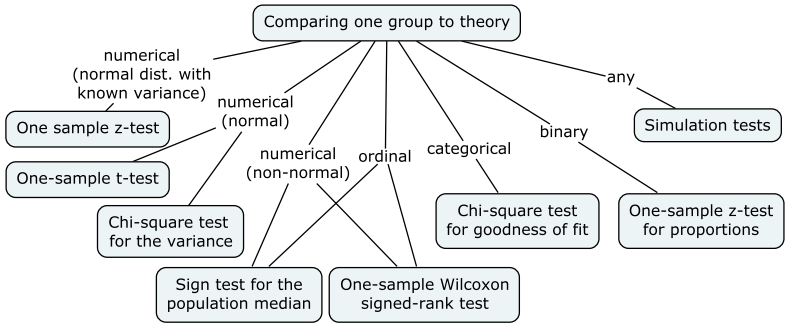


Figure 3.67: Hypothesis tests that compare one group to theory.

can apply nonparametric tests like the *sign test for the population median* (Section 3.7.9.1) and the *one-sample Wilcoxon signed-rank test* (Section 3.7.9.2).

Comparing two groups Suppose now that we have obtained the samples **x** and **y** from two unknown populations $X \sim \mathcal{M}(\theta_X)$ and $Y \sim \mathcal{M}(\theta_Y)$. We want to know if the two populations are the same or different. This is the scenario we discussed in Section 3.5, when we introduced the *two-sample permutation test* (Section 3.7.10.2) and *Welch's two-sample t-test* (Section 3.7.6.2).

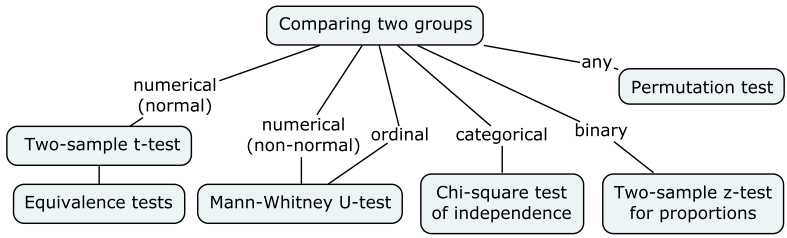


Figure 3.68: Hypothesis tests for comparing two groups.

The *Mann–Whitney U test* (Section 3.7.9.3) is a nonparametric equivalent to the *two-sample t-test*, and it is an important tool for the analysis of numerical data that doesn't satisfy the **(NORM)** or **(LARGEn)** assumptions.

The *two-sample z-test for proportions* (Section 3.7.5.6) is similar to the *two-sample t-test*, but applied to proportions.

When working with two categorical variables, we can use the *chi-square test of independence* (Section 3.7.7.2) to check if an association between the variables exists, or if they're independent.

Comparing three or more groups Consider now the scenario in which we want to compare samples from three groups $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$. The ANalysis Of VAriance (ANOVA) family of tests (Section 3.7.8) are used to compare the means of three or more groups against the null hypothesis of "all groups are the same."

The *one-way analysis of variance test* (Section 3.7.8.1) studies the effect of the grouping variable with three or more levels on a normally distributed outcome variable. The *Kruskal–Wallis analysis of variance by ranks* (Section 3.7.9.4) is the equivalent nonparametric test, that doesn't require the normality assumption. The *permutation ANOVA* test (Section 3.7.10.3) is another alternative that works for any type of data.

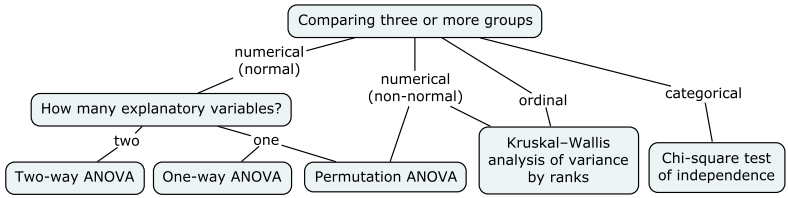


Figure 3.69: Hypothesis tests for comparing three or more groups.

The *two-way ANOVA test* (Section 3.7.8.2) is a generalization of the *one-way ANOVA* that studies the effect of *two* predictor variables on a numerical outcome variable.

For categorical data, we can use the *chi-square test of independence* (Section 3.7.7.2) to check for an association between variables.

Paired and repeated measurements We sometimes collect multiple measurements from individuals over time. For example, the *paired measurements* statistical analysis scenario involves comparison of before and after measurements of one group of individuals. The *paired t-test* (Section 3.7.6.3) can be used in this scenario.

Another scenario is *repeated measures*, where multiple measurements for an individual are collected and must be aggregated together. The *repeated measures ANOVA* and its nonparametric counterpart, the *Friedman test*, are normally used for this kind of repeated measures scenarios, but we will not discuss them in this book.

Distribution checks Another class of tests involve checking the properties of a distribution. These tests are sometimes used to check the assumptions required for another test, before proceeding with the analysis (see Step 3 in Figure 3.65).

The *Kolmogorov–Smirnov test* (Section 3.7.12.1) can be used to check if an arbitrary sample comes from a given distribution, or

if two samples come from the same distribution. The *Shapiro–Wilk normality test* (Section 3.7.12.2) is specially designed to check if a data sample is normally distributed. The *Levene's test* is used to check whether two samples have the same variance (which is sometimes called *homogeneity* of variance or *homoscedasticity*).

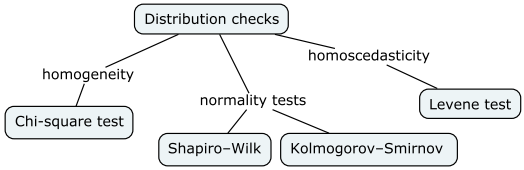


Figure 3.70: Distribution checks.

For categorical variables, we can use the *chi-square test for goodness of fit* to check if some count data matches an expected distribution. We can also use the *chi-square test for homogeneity* (Section 3.7.7.3) to check if the distribution of a categorical variable is the same across samples from different populations.

Numerical predictor variables All the tests discussed above study the effect of a categorical predictor variable on outcome variables of different types, which we denote as "$\mathbf{C} \to *$." When the predictor variable is numerical, we require different techniques to model the relationship. In Chapter 4, we'll discuss *linear models* and learn about hypothesis tests for the overall model and individual coefficients of the model. See Section 4.3.4 if you want to get a preview of the hypothesis tests for "$\mathbf{N} \to *$" statistical analyses.

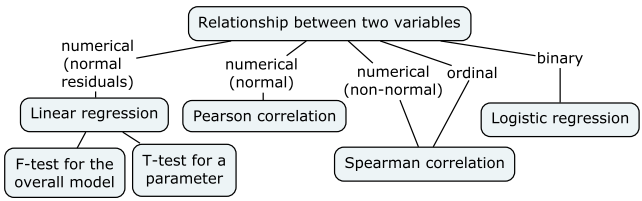


Figure 3.71: Tests for relations between variables.

* * *

Okay, let's get started with the inventory! Remember, you're not expected to read the descriptions of the test recipes in detail. I just want you to skim through the next two dozen pages so you'll know what's available.

3.7.4 Z-tests

The standard normal distribution $Z \sim \mathcal{N}(0,1)$ is an important model for the sampling distribution of the mean. Recall the central limit theorem tells us that the sampling distribution of the mean of samples from *any* distribution will be normally distributed, provided the sample size n is large enough.

3.7.4.1 One-sample z-test

We discussed this hypothesis test in Section 3.4.2. The goal of the one-sample z-test is to check if the mean μ_X of an unknown population $X \sim \mathcal{N}(\mu_X, \sigma_0)$ equals the mean μ_0 of a theoretical distribution $X_0 \sim \mathcal{N}(\mu_0, \sigma_0)$. Note we assume that the standard deviation of the unknown population X is known and equal to the standard deviation of the theoretical population σ_0.

Data One sample of numerical observations $\mathbf{x} = [x_1, x_2, \ldots, x_n]$.

Assumptions We assume the population is normally distributed **(NORM)**, or the sample is large enough **(LARGEn)**. We also assume that the population variance σ_0^2 is known.

Hypotheses $H_0 : \mu_X = \mu_0$ and $H_A : \mu_X \neq \mu_0$, where μ_X is the unknown population mean and μ_0 is the expected theoretical mean.

Statistical design We can use the formula $n = \frac{(F_Z^{-1}(1-\alpha) - F_Z^{-1}(\beta))^2 \sigma^2}{\Delta^2}$ to find the minimum sample size, given α, β, and the effect size Δ.

Test statistic Compute $z = \frac{\overline{\mathbf{x}} - \mu_0}{\sigma_0/\sqrt{n}}$, where $\overline{\mathbf{x}}$ is the sample mean, μ_0 is the theoretical population mean, σ_0 is the known standard deviation.

Sampling distribution Standard normal distribution $Z \sim \mathcal{N}(0,1)$.

Effect size Observed deviation $\hat{\Delta} = \overline{\mathbf{x}} - \mu_0$ or Cohen's $d = \frac{\overline{\mathbf{x}} - \mu_0}{\sigma_0}$.

Confidence interval for the unknown mean A γ-confidence interval for the population mean is $\mathbf{ci}_{\mu,\gamma} = [\overline{\mathbf{x}} - z_{\gamma/2}\frac{\sigma_0}{\sqrt{n}},\ \overline{\mathbf{x}} + z_{\gamma/2}\frac{\sigma_0}{\sqrt{n}}]$, where $z_{\gamma/2} = F_Z^{-1}(1-\gamma/2)$ is the $(1-\gamma/2)$-quantile of $Z \sim \mathcal{N}(0,1)$.

Examples See the `examples/one_sample_z-test.ipynb` notebook.

The *one-sample t-test* (Section 3.7.6.1) is a more general test that uses the sample standard deviation as an approximation to the population standard deviation.

3.8 Statistical practice

Statistical inference is a complicated topic and, like most complicated topics, it is rife with misconceptions. Well-meaning researchers often use statistical procedures inappropriately and end up shooting themselves in the foot. The complicated procedures required for statistical inference also leave room for less well-meaning researchers to use questionable research practices that allow them to bias the outcome of tests toward their desired result.

In this section, we'll review some common traps and misconceptions about statistical analysis procedures, which will help you recognize them in your own research and in other people's work. We'll discuss the possible statistical misconceptions you need to watch out for (unintentional mistakes), then list some of the most common questionable research practices (intentional mistakes). We'll also give a short guide to the standard practices for communicating statistical results. Due to space limitations, we'll cover all these topics only briefly, but we'll provide links to external resources where you can learn more about statistical practice topics.

3.8.1 Avoiding statistical misconceptions

It's easy to misinterpret or over-interpret the results of a hypothesis test. In this section, we'll look at some examples of statistical misconceptions and other problems you might run into. The problems we'll describe apply to the null hypothesis significance testing (NHST) procedure in general, which means they apply to all the statistical analysis recipes we studied in Section 3.7.

The correct interpretation of hypothesis test results

Before we describe the *wrong* ways to interpret hypothesis testing results, let's briefly review the *right* way to interpret them.

Calculating the p-value We compute the p-value based on the probability model under the null hypothesis H_0, which describes the hypothetical "no effect" or "no difference" scenario. The p-value is the conditional probability of the observed data under the null model, which we denote $\Pr(\textbf{data}|H_0)$. Note the p-value calculation *assumes* the null hypothesis is true and doesn't involve the alternative hypothesis H_A in any way. Indeed, the NHST procedure is not actually considering an alternative theory, but just performing a basic sanity check to see if there is some pattern in the data that H_0 can't explain.

Decision rule based on a cutoff value The NHST procedure dictates that we should reject the null hypothesis if the p-value is smaller than a predetermined cutoff value α, which represents a threshold of "surprisingness" we're using for the test. The NHST procedure gives us a simple way to make a decision between "reject H_0" and "fail to reject H_0." The cost of this simplicity is rigidity—our decision is based on an arbitrary cutoff. Is $p = 0.049$ any different from $p = 0.051$? Not really, but if we're using the conventional cutoff $\alpha = 0.05$, we would make opposite decisions in those two cases. Using $\alpha = 0.05$ leads to a 5% Type I error rate by definition.

The logic behind the NHST procedure is that observing a small p-value provides indirect evidence against H_0. It's essential that we do not over-interpret the strength of this evidence by using words like "shown that..." or "proved that..." A more appropriate language to describe our conclusion after observing a small p-value is to say we'll "act as if H_0 is false," without claiming that it truly is. Here is an example of an elaborate, precise statement we can make when we reject H_0.

> We claim there is a non-zero effect, while acknowledging that if scientists make claims using this methodological procedure, they will be misled, in the long run, at most α% of the time, which we deem acceptable. We will, for the foreseeable future, and until new data or information emerges that proves us wrong, assume this claim is correct.
>
> —Daniël Lakens, [Lak22, Sec 1.6]

Of course, I don't expect you to write all of that every time you report the results of a hypothesis test, but this is what you should be thinking in your head.

Note I didn't use the phrase "statistically significant" anywhere in the above statements (not even in my thoughts!). We're banishing that phrase, because it makes things sound more impressive than they actually are, which is a dangerous footgun!

Reporting effect size estimates A small p-value suggests that the observed data shows a pattern that is inconsistent with the probability model under the null hypothesis. In order to quantify this deviation from the null hypothesis, we compute an *effect size* estimate, which is calculated based on the probability model under the alternative hypothesis H_A. The effect size estimate is often the most interesting part of any study, since it tells us what we really want to know: the amount of discrepancy from a theoretical model, or the size of the observed difference between two groups. Knowing the effect size allows us to interpret the practical importance of the

result. We usually report both a point estimate for the effect size and a confidence interval to communicate the uncertainty in the point estimate.

Misconceptions about p-values

Let's now discuss the numerous "wrong" ways to think about the conclusion of the NHST procedure. The fundamental difficulty in interpreting NHST results is that researchers are interested in knowing whether the alternative hypothesis H_A is true or the null hypothesis H_0 is true. Using math notation, we could say we're interested in the probabilities of the two hypotheses given the observed data:

$$\Pr(H_A|\textbf{data}) \qquad \text{and} \qquad \Pr(H_0|\textbf{data}).$$

Unfortunately, the NHST procedure quantifies the probability of the data, assuming that the null hypothesis is true, $\Pr(\textbf{data}|H_0)$, which is different from both of the above desired probabilities. The mismatch between the probabilities we want to know and the probability that the NHST procedure gives us is the source of many of the misconceptions we'll describe below.

Probability of hypotheses The most commonly encountered misconception about the p-value is interpreting it as the probability that H_0 is true. This is simply not the case: $p = \Pr(\textbf{data}|H_0)$, which is not the same as $\Pr(H_0|\textbf{data})$. A similar misconception is that $(1 - p)$ corresponds to the probability that H_A is true, which is also incorrect. In fact, H_A was not considered at all in the p-value calculation! In general, any conclusion that makes claims about the "probability of" or our "confidence" or "belief" in H_0 or H_A is not going to be correct.

Viewing results as absolute proof We interpret a small p-value ($p < \alpha$) as *evidence* against H_0, but it is wrong to claim that we have definitely shown that the null hypothesis is false. For example, statements like "$p < \alpha$, therefore there is an effect" or "there is a difference between the two groups" are not valid conclusions. Similarly, if the p-value is large ($p > \alpha$), this doesn't mean that the null hypothesis is true. It's wrong to make statements like "there is no effect" or "there is no difference" in those situations. The logic of hypothesis testing is probabilistic and not absolute.

Probability that we made a mistake Another misconception is that observing a p-value less than $\alpha = 0.05$, and thus rejecting H_0, implies that the probability we have made a Type I error is guaranteed to be

less than 5%. This is not the correct interpretation. Recall that α is the *long-term average* error rate of the procedure, and not a specific probability of error for this test. We would obtain the 5% guarantee on Type I errors if we were to use the NHST procedure repeatedly for many hypothesis tests on different datasets.

Probability of replication A related misconception is that $(1 - p)$ is the probability that the observed result will hold up in future studies. Nope. This misconception seems to stem from a confusion with the wording used to define α in terms of the long-term average error rate that we would observe when we use the hypothesis testing procedure for several repeated analyses.

Interpreting the p-value as an effect size The size of the p-value is not an indication of the strength or magnitude of the effect size. Indeed, the p-value is calculated assuming H_0 is true, so it can't tell us anything about the effect size under H_A. Unfortunately, many people misinterpret observing a small p-value as if a large effect has been discovered, which makes no sense.

It's also possible to make the mistake in the opposite direction: interpreting a large p-value as indicating that the effect size is small. A large p-value can also occur even when the effect size is large, depending on the power of the test. If you want an estimate of the effect size, compute an estimate of the effect size.

Misconceptions about confidence intervals Many of the misconceptions we described above are analogous to the misconceptions about confidence intervals that we saw previously in Section 3.2 (see page 98). Correct interpretation of statistical results is very tricky!

* * *

I know some of the misconceptions I described above may seem like "technicalities" to you, but I assure you the formal statements are important, and using them wrong leads to real-world problems. The interpretation of statistical results is not the time for imprecise, hand-waving explanations. You can't just say "you know what I mean" when making statistical conclusions. Mathematically precise statements are important because they describe the exact statistical procedure we performed and keep our expectations realistic. This is why we spent so many pages discussing the math machinery of hypothesis testing (estimators and sampling distributions) in this chapter. I want you to have the solid foundation and know

3.9 Conclusion

This chapter introduced the main techniques of classical statistics. These are the most widely used methods of statistical inference. The most important theoretical tool we developed in this chapter is the *sampling distribution* of an estimator, which we used to construct confidence intervals and hypothesis tests.

Error control

The main feature of the null hypothesis significance testing (NHST) procedure is the control over the Type I (false-positive) errors by choosing the design parameter α. We can't give a guarantee about any particular decision, but if we apply a hypothesis test repeatedly across many datasets, we have a guarantee that the proportion of false positives will be at most α. The parameter α is chosen by the researcher in advance depending on the needs of the statistical analysis.

If we know the expected effect size Δ and the sample size n, we can also calculate the Type II error rate β and the power $(1-\beta)$ of the testing procedure. Alternatively, we can start with a desired power $(1-\beta)$ and solve for the sample size n required to achieve it.

The key advantage of the classical NHST approach is that it allows us to design procedures with desired error rates specified *in advance* of seeing the data $\mathbf{x}$.

Frequentist paradigm

The techniques we developed in this chapter are sometimes called *frequentist statistics*, because they measure the performance of procedures in terms of error frequencies over repeated use on multiple datasets. If we were to use a given procedure (confidence intervals or hypothesis test) repeatedly for a bunch of samples $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, we know the count (frequency) of incorrect results will be at most αN, where the design parameter α is chosen in advance.

In frequentist statistics, we split the responsibilities between two roles: statisticians and scientists. The role of statisticians is to invent general procedures that can be used on any dataset, assuming the dataset satisfies some assumptions. The role of scientists is to choose appropriate procedures and apply them to the specific datasets they are working on. The frequentist quality guarantees are the interface between these two roles: the statistician promises to the scientist that if they use the procedure on N datasets, it will work as expected for about $(1-\alpha)N$ of them.

Limitations of frequentist inference

In Section 3.8, we discussed the most common misconceptions about p-values and confidence intervals. I hope you'll watch out for these traps when reporting results and avoid making interpretation mistakes. The key thing to remember is that the quality guarantees we give concern the correctness of the *procedure* and not any particular result of the procedure. I know this is kind of weird, but this is the kind of guarantee we can give when we rely on procedures prepared in advance of seeing the data.

Next up

In the next chapter, we'll learn about linear models, which are a broad family of models where the outcome variable depends on one or more predictors through a linear relationship. After that, we'll learn about the *Bayesian paradigm* for doing statistics in Chapter 5.

3.10 Statistical inference problems

P3.1 Your friend Alex has come up with the estimator $\tilde{s}^2_{\mathbf{x}} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2$, which uses the normalizing factor $\frac{1}{n}$ instead of the $\frac{1}{n-1}$ that is used in the sample variance $s^2_{\mathbf{x}}$. Alex claims that estimates $\tilde{s}^2_{\mathbf{x}}$ computed from a sample $\mathbf{x}$ is "just as good" as the estimate $s^2_{\mathbf{x}}$ for estimating the population variance σ^2_X. You've read somewhere that $\tilde{s}^2_{\mathbf{x}}$ produces biased estimates (specifically underestimates of the population variance), and you want to show that to Alex. Unfortunately you're at the remote location with no access to the internet, so you can't just find a page that shows this, you have to show that using only pen and paper. Show that the expected value of the sampling distribution of Alex's estimator is not equal to the population variance: $\mathbb{E}[\tilde{S}^2] \neq \sigma^2_X$.

Hint: You'll need to use the parallel axis formula $\mathbf{Var}[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2$ twice in this derivation.

Hint: Use $\mathbf{Var}[X] = \mathbb{E}[X^2] - \mu^2_X$ to obtain an expression for $\mathbb{E}[X^2]$.

Hint: Use $\mathbf{Var}[\overline{X}] = \mathbb{E}[\overline{X}^2] - \mathbb{E}[\overline{X}]^2$ to obtain an expression for $\mathbb{E}[\overline{X}^2]$.

Hint: Use the LLN: $\mathbf{Var}[\overline{X}] = \frac{\sigma^2_X}{n}$ for large n.

Hint: Use the central limit theorem: $\mathbf{Var}[\overline{X}] = \frac{\sigma^2_X}{n}$ for large n.

P3.2 Prove that the sample variance estimator S^2 is unbiased by showing that its expected value of the $\mathbb{E}[S^2]$ is equal to the population variance σ^2_X.

Hint: Try to reproduce the derivation step in P3.1 from memory.

Chapter 4

Linear models

Linear models form a broad family of statistical models for describing relationships between one or more predictor variables and an outcome variable (usually denoted Y). Specifically, we model the mean of the outcome variable μ_Y as a *linear function* of the predictor variables, hence the term *linear model*.

Linear models have applications across science, engineering, medicine, and many other domains. The reason for the widespread use of linear models is that they embody the simple but powerful idea of describing changes in the mean of the outcome variable as *proportional* to changes in the predictor variables. This is the simplest kind of dependence between variables, yet it is general enough to allow modelling all kinds of phenomena.

This chapter introduces the core ideas behind linear models. In Section 4.1, we'll start with *simple linear regression*, which is a linear model with one predictor variable. Then in sections 4.2 and 4.3, we'll study *multiple linear regression* models, which have several predictor variables. In the remainder of the chapter, we'll extend linear models to different contexts, including categorical predictors (Section 4.4), and different distributions of the outcome variable (Section 4.6).

In this chapter, I'm going to show you how to ...

- **fit** a linear model $Y = \beta_0 + \beta_1 x + \mathcal{E}$ to the bivariate dataset $[\mathbf{x}, \mathbf{y}]$
- **fit** a multiple linear regression model $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \mathcal{E}$ to the multivariate dataset with p predictors $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p, \mathbf{y}]$
- **interpret** the parameters of linear models
- **perform** hypothesis tests for linear models and their parameters
- **predict** the outcomes of new observations given the predictors
- **check** model assumptions using diagnostic plots
- **preprocess** categorical predictors for use in linear models (dummy coding)
- **understand** the strengths and limitations of linear models

4.1 Simple linear regression

Consider the bivariate dataset $[\mathbf{x}, \mathbf{y}] = [(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$ that consists of n pairs of observations (x_i, y_i). We want to study the relationship between the *predictor* variable x and the *outcome* variable y. Specifically, we'll model the outcome as a random variable Y whose mean is a *linear* function of x, $\mu_Y(x) = \beta_0 + \beta_1 x$, where β_0 and β_1 are constants.

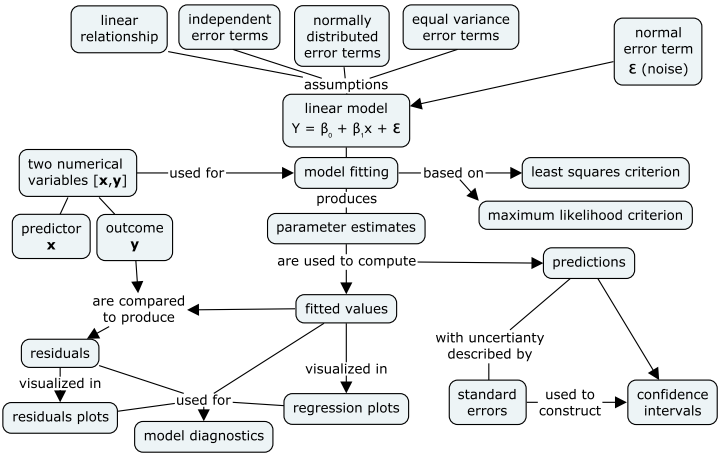


Figure 4.1: Concepts and ideas related to simple linear regression models.

Figure 4.1 shows an overview of the new concepts we'll learn in this section. We'll learn about linear models with one numerical predictor variable, which is called *simple linear regression*.

4.1.1 Definitions, notation, and terminology

Let's now define the notation and terminology that we'll use in this section and in the remainder of the chapter.

- *Outcome variable*: the variable of interest whose values we want to model and predict. Another name for the outcome variable is *endogenous* variable, which means "determined by the model." We'll see the outcome variable in several contexts:
  - ▻ The random variable Y specified by the theoretical model.
  - ▻ An observation y_i from the $\mathbf{y}$ column in the dataset $[\mathbf{x}, \mathbf{y}]$.
  - ▻ The random variable $\widehat{Y}$ described by the estimated model.
  - ▻ A prediction $\widehat{y}$ from the estimated model. For example, the predicted outcome $\widehat{y}_i$ calculated from the predictor x_i.

- *Predictor variable*: a variable that we believe influences the outcome variable. We can also refer to predictors as *independent*, *explanatory*, or *exogenous* (determined outside the model) variables. We'll see the predictor variable in the several contexts:
  - ▷ The variable x used to define the theoretical model.
  - ▷ An observation x_i from the $\mathbf{x}$ column in the dataset $[\mathbf{x}, \mathbf{y}]$.
  - ▷ A generic variable x that varies over a range that we use to plot the model's predictions.
  - ▷ A new observation x_{new} that is not part of the dataset $[\mathbf{x}, \mathbf{y}]$, for which we want to generate a prediction $\hat{y}_{\text{new}}$.
- *Linear function*: a general math concept that describes relationships where the changes in the output variable is *proportional* to the changes in input variable(s). For example, the functions $f(x) = mx$ and $g(x_1, x_2) = m_1x_1 + m_2x_2$ are linear, while the functions $q(x) = 1 + x + x^2$ and $p(x_1, x_2) = x_1x_2$ are not linear. For a linear function, scaling the input by a factor k produces an output that is k times larger: $f(kx) = kf(x)$. For multivariable linear functions, linearity tells us the contributions of the different variables are additive: $g(k_1, k_2) = g(k_1, 0) + g(0, k_2) = k_1g(1, 0) + k_2g(0, 1)$.
- *Linear model for the mean* $\mu_Y(x) = \beta_0 + \beta_1 x$: an equation that describes an idealized relationship between the predictor variable x and the mean of the outcome variable Y. The function $\mu_Y(x)$ is also called the *regression function* and is the core of the simple linear regression model that we'll define in the next section.
- *Model parameters* β_0 and β_1. The parameter β_0 is called the *intercept*, while β_1 is called the *slope*. Note, the Greek letter *beta* in this chapter refers to model parameters, and has no relation to the Type II error rate we discussed in the previous chapter.
- *Error term* $\mathcal{E} \sim \mathcal{N}(0, \sigma)$: a normally distributed random variable with mean zero and standard deviation σ that represents the inherent variability of the outcome variable. We use the Greek letter $\mathcal{E}$ (*epsilon*) for the error term, following a widespread convention for describing linear models.
- *Estimated linear model for the mean* $\mu_{\widehat{Y}}(x) = \widehat{\beta}_0 + \widehat{\beta}_1 x$: the equation of the best-fitting regression line for the dataset $[\mathbf{x}, \mathbf{y}]$.
- *Estimated model parameters* $\widehat{\beta}_0$ and $\widehat{\beta}_1$: the parameters of the best-fitting linear model. The parameters $\widehat{\beta}_0$ and $\widehat{\beta}_1$ are estimates for unknown model parameters β_0 and β_1.
- *Model predictions* $\widehat{y}$: values of the outcome variable predicted by the estimated linear model. We'll discuss predictions in two different contexts:

  - ▻ Fitted outcome values $\widehat{y}_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i$, calculated for each observed values x_i in the data $\mathbf{x}$.
  - ▻ A prediction $\widehat{y}_{\text{new}}$ for a previously unseen values x_{new} that is not part of the dataset $[\mathbf{x}, \mathbf{y}]$.
- *Residuals* $r_i = y_i - \widehat{y}_i$: the difference between the actual value of the outcome variable y_i and the fitted value $\widehat{y}_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i$ predicted by the model. Another term to describe the residuals is *prediction errors*.

4.1.2 Linear model equations

We'll now describe the equation of the linear model that we'll use in the rest of the section. The conditional distribution of the outcome variable Y given the predictor x is described by the following normal distribution:

$$Y|x \sim \mathcal{N}(\beta_0 + \beta_1 x,\ \sigma).$$

The mean of Y is described by the equation $\mu_Y(x) = \beta_0 + \beta_1 x$, and its standard deviation is σ. Another way to describe this model is using the equation

$$Y|x = \beta_0 + \beta_1 x + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma)$ is a normally distributed *error term*. Compare the above two equations and convince yourself that they describe the same conditional distribution. Later in this section, we'll also use the *formula notation* "`y ~ 1 + x`" to describe this model in code examples.

The linear model has two parts:

1. The *regression function* $\mu_Y(x) = \beta_0 + \beta_1 x$ describes how the mean of Y varies as a function of x.
2. A random error term $\mathcal{E} \sim \mathcal{N}(0, \sigma)$, which we assume is normally distributed with unknown standard deviation σ.

Figure 4.2 shows the visualization of the theoretical linear model. Fundamentally, we're modelling the conditional distribution $Y|X$ and not a joint distribution (X, Y). The regression line describes the predominant "trend" of how the mean of the ys varies with x, and it is determined by the parameters β_0 and β_1, which have the following geometric interpretation:

- β_0: the *intercept term* describes the mean of Y when x is zero.
- β_1: the *slope* describes the average change in Y that we expect to observe for a unit increase in x.

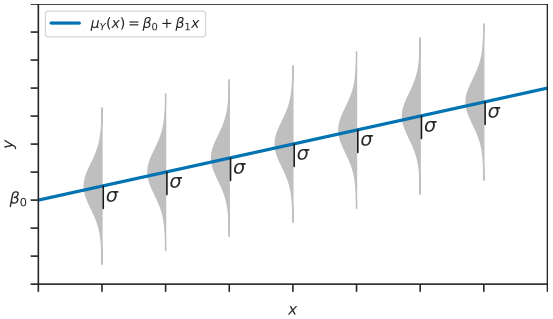


Figure 4.2: The regression line $\mu_Y(x) = \beta_0 + \beta_1 x$ describes the expected value of the outcome variable Y for different values of x. The actual outcomes are normally distributed around this line with standard deviation σ.

The regression line describes the centre of the conditional distribution $Y|x$. We expect the actual outcomes of the variable Y to be randomly distributed above and below this line, and we assume these deviations are normally distributed with scale parameter σ.

Using the linear model to describe bivariate data

So far we described a theoretical model for the distribution of the random variables Y conditional on the predictor x. How can we use this model to describe the relationship between the observations in the bivariate dataset $[\mathbf{x}, \mathbf{y}] = [(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$?

According to the linear model we defined above, we can explain the relationship between the variables in the observation (x_i, y_i) through the equation

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i,$$

where ε_i is a realization of the random variable $\mathcal{E} \sim \mathcal{N}(0, \sigma)$. The model parameters β_0, β_1, and σ are unknown quantities in this equation, while x_i and y_i are known. We have n such equations, one for each observation (x_i, y_i) in the dataset $[\mathbf{x}, \mathbf{y}]$.

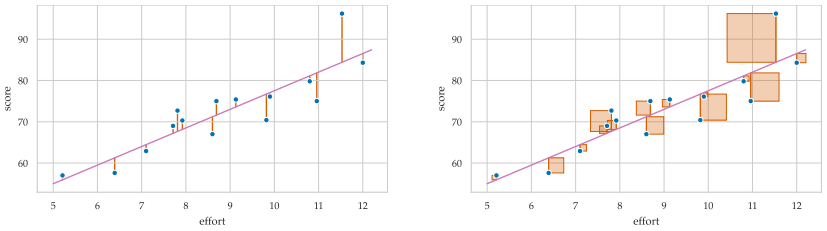
Estimating the model parameters The statistical inference task is to find the estimates $\widehat{\beta}_0$, $\widehat{\beta}_1$, and $\widehat{\sigma}$ for the unknown model parameters β_0, β_1, and σ that best describe the dataset $[\mathbf{x}, \mathbf{y}]$. This notation follows the convention we introduced in Chapter 3 of using hats to denote estimates (quantities computed from the data).

There are several different approaches we can use to obtain the estimates $\widehat{\beta}_0$, $\widehat{\beta}_1$, and $\widehat{\sigma}$. Estimating the linear model is equivalent to choosing the regression line that best "fits" the data, which is why

the dataset $[\mathbf{x}, \mathbf{y}] = [(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$. The residuals are the differences between the observed outcomes y_i and the predicted outcomes $\widehat{y}_i$:

$$r_i \stackrel{\text{def}}{=} y_i - \widehat{y}_i = y_i - (\widehat{\beta}_0 + \widehat{\beta}_1 x_i).$$

Graphically speaking, the residual r_i is the vertical distance between observation (x_i, y_i) and the regression line. See Figure 4.5 (a) for an illustration. Each residual r_i could be positive or negative, and we expect a line that passes through the "middle" of the scatter plot to have roughly equal amounts of positive residuals (model underestimates) and negative residuals (model overestimates).



(a) Illustration of the residuals $r_i = y_i - \widehat{y}_i$.

(b) Illustration of the squared residuals r_i^2.

Figure 4.5: Scatter plots of the students dataset and the best-fitting linear model. The residuals (left) and squared residuals (right) are illustrated.

The *ordinary least squares* (OLS) procedure chooses the parameters $\widehat{\beta}_0$ and $\widehat{\beta}_1$ that minimize the *sum of squared residuals* (**SSR**) quantity, which is defined as:

$$\mathbf{SSR}(\widehat{\beta}_0, \widehat{\beta}_1) \stackrel{\text{def}}{=} \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2 = \sum_{i=1}^{n} \Big(y_i - (\widehat{\beta}_0 + \widehat{\beta}_1 x_i)\Big)^2.$$

Intuitively, a small $\mathbf{SSR}(\widehat{\beta}_0, \widehat{\beta}_1)$ means we have obtained a line $\widehat{y} = \widehat{\beta}_0 + \widehat{\beta}_1 x$ that is a good fit to the data $[\mathbf{x}, \mathbf{y}]$. Figure 4.5 (b) illustrates the squared residuals of the best-fitting linear model for the student scores. The quantity $\mathbf{SSR}(\widehat{\beta}_0, \widehat{\beta}_1)$ corresponds to the sum of the areas of the 15 squares. The best-fitting line is the one that makes the combined area of the squares as small as possible.

Estimating the standard deviation parameter

We can estimate the variance of the error term σ^2 by calculating the sum of squared residuals divided by $n - 2$:

$$\widehat{\sigma}^2 \stackrel{\text{def}}{=} \frac{\mathbf{SSR}}{n-2} = \frac{\sum_{i=1}^{n} r_i^2}{n-2}.$$

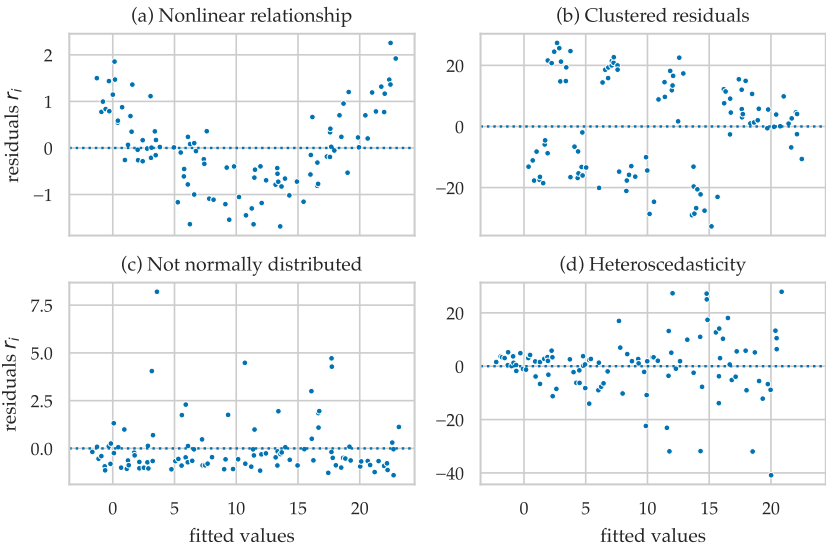


Figure 4.8: Residual plots showing violations of the model assumptions.

Normality check using the Q-Q plot

It can be difficult to see if the residuals are normally distributed by looking at the residual plot. A better way to check the **(NORMε)** assumption is to generate a Q-Q plot of the residuals using the `qqplot` function from `statsmodels`. A Q-Q plot compares the quantiles of the residuals to the quantiles of the normal distribution. If the residuals are normally distributed, they should fall close to the diagonal line in the Q-Q plot.

code 4.1.8
```
>>> from statsmodels.graphics.api import qqplot
>>> qqplot(residuals, line="s")
See Figure 4.9.
```

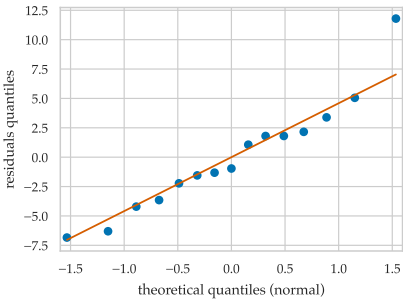


Figure 4.9: Q-Q plot of the residuals $r_i = s_i - \hat{s}_i$ of the linear model.

The Q-Q plot in Figure 4.9 shows most of the points are close to the

Knowing $\widehat{\mathbf{se}}_{\mu_{\widehat{S}}} = 1.27$, we can now construct the confidence interval based on the quantiles of Student's t-distribution with $n - 2 = 13$ degrees of freedom:

code 4.1.12

```
>>> from scipy.stats import t as tdist
>>> alpha = 0.1
>>> t_l, t_u = tdist(df=n-2).ppf([alpha/2, 1-alpha/2])
>>> [scorehat+t_l*se_meanscore, scorehat+t_u*se_meanscore]
[70.74303643371016, 75.25696356628984]
```

The confidence interval $\mathbf{ci}_{\mu_S,0.9}(9) = [70.7, 75.3]$ tells us the range of average scores that we can expect for students who invest 9 hours of effort. We can repeat this calculation for different values of effort to visualize the model uncertainty over the whole range of predictor values.

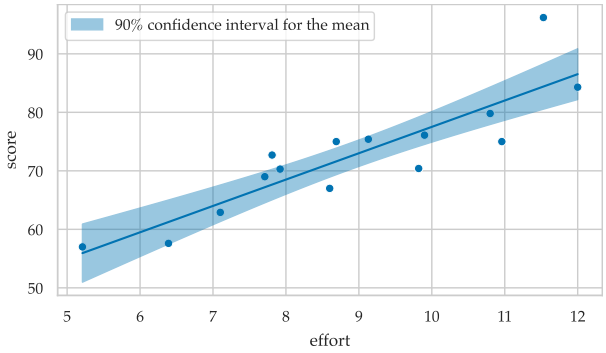


Figure 4.10: Plot of the mean $\mu_{\widehat{S}}(e) = \widehat{\beta}_0 + \widehat{\beta}_1 e$ and 90% CI for $\mu_S(e)$.

The plot in Figure 4.10 shows the 90% confidence interval for the model estimates for the mean. Note the confidence intervals get wider as we move further from the middle of the effort axis.

Predicting the outcome values

Suppose we want to now predict the possible outcomes $\widehat{Y}(x)$ we might observe for a given value of the predictor variable x. The estimated model for the outcome variable is

$$\widehat{Y}(x) \sim \mathcal{N}\big(\mu_{\widehat{Y}}(x), \widehat{\sigma}\big) = \mathcal{N}\big(\widehat{\beta}_0 + \widehat{\beta}_1 x, \widehat{\sigma}\big) = \widehat{\beta}_0 + \widehat{\beta}_1 x + \mathcal{N}(0, \widehat{\sigma}).$$

In other words, to obtain value predictions, we first predict the mean $\mu_{\widehat{Y}}(x)$ = `predict(x,b0,b1)` using the linear model, then add the additional error term with standard deviation $\widehat{\sigma} = \sqrt{\frac{\mathbf{SSR}}{n-2}}$. We'll denote a particular prediction as $\widehat{y}(x)$ or simply as $\widehat{y}$. The standard

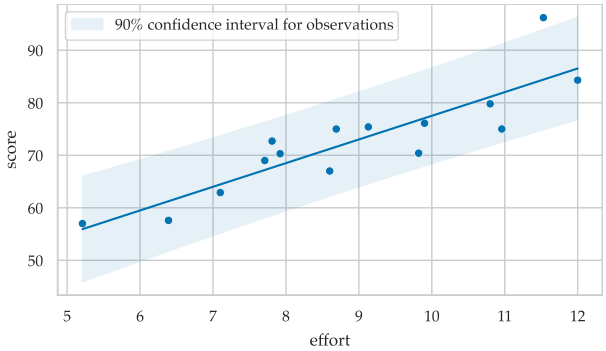


Figure 4.11: A 90% confidence interval for the predicted student scores $S(e)$.

prediction $\mathbf{ci}_{\mu_S,0.9}(9) = [70.7, 75.3]$, since in describes the uncertainty of the actual predictions, not their mean.

We can repeat the above calculations for the uncertainty of the predictions for all plausible effort values to obtain a *prediction band*, as shown in Figure 4.11. The confidence band for the predicted observations (Figure 4.11) is wider than the confidence band for the mean predictions (Figure 4.10), since we're predicting actual observations that include the error term $\mathcal{N}(0, \widehat{\sigma})$, not their mean.

Prediction caveats

We'll now discuss the limitations of the predictions we obtain from linear models. The range of the `effort` variable in the students dataset is

code 4.1.15
```
>>> efforts.min(), efforts.max()
(5.21, 12.0)
```

The process of making score predictions for effort values between 5.21 and 12 is called *interpolation*. It makes sense to use the model to make predictions over the range of values where we have observed data.

In contrast, making predictions outside this range of data we have observed is called *extrapolation*, and is not something we're allowed to do, because we have no guarantee that the model will be valid outside the range of observed values. For example, if we use the model to predict the `score` for a student who invests 20 hours of `effort` per week, we obtain:

code 4.1.16
```
>>> predict(20, b0=32.5, b1=4.5)
122.5
```

The model predicts the grade 122.5, which is an impossible value, since it is above the maximum possible grade of 100.

4.2 Multiple linear regression

We'll now learn about linear models with multiple predictor variables. Consider the multivariate dataset $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p, \mathbf{y}]$, where each observation consists of p predictor values $(x_1, x_2, \ldots, x_p)$ and the corresponding outcome y. The multiple linear regression model for the outcome variable conditional on the p predictor variables is

$$Y|x_1, x_2, \ldots, x_p \quad \sim \quad \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma)$ is a Gaussian error term with mean zero and unknown standard deviation σ. Another way to write the model is $Y \sim \mathcal{N}(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p, \sigma)$, which makes it clear that the mean of the outcome variable is a linear function of the predictors: $\mu_Y(x_1, x_2, \ldots, x_p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$.

Each "slope" parameter β_k quantifies the strength of the influence of the predictor x_k on the outcome variable. To *fit* a linear model is to find the values of the parameter estimates $\widehat{\beta}_0, \widehat{\beta}_1, \widehat{\beta}_2, \ldots, \widehat{\beta}_p$ that best describe the relationship observed in the data.

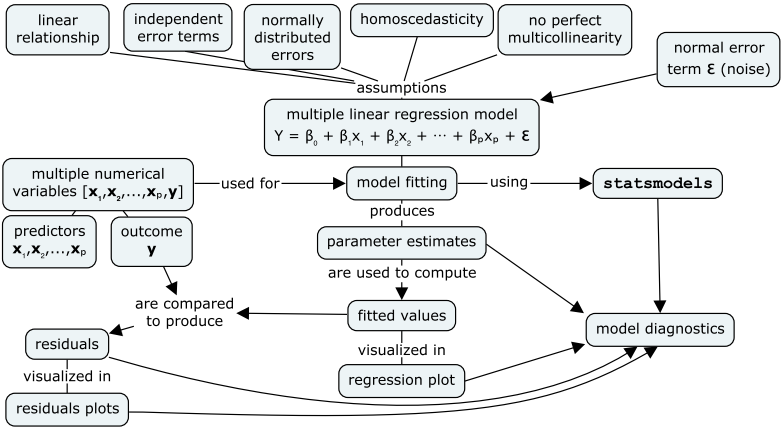


Figure 4.14: Concepts and assumptions used for multiple linear regression.

Figure 4.14 shows an overview of the new concepts we'll discuss in this section. You're already familiar with the main ideas of linear models from the previous section. In this section we'll show more general formulas that apply when there are multiple predictors.

4.2.1 Doctors sleep study dataset

We'll now introduce a new dataset that is suitable for analysis using multiple linear regression. The doctors dataset (`doctors.csv`)

contains data about the life habits of 156 doctors selected at random from the population of doctors in the country. The data was collected as part of an observational study of the factors that influence sleep quality. We'll use the doctors dataset for all the example calculations in the next few sections.

Let's start by loading the data file `datasets/doctors.csv` and looking at the first few rows to see what the data looks like.

code 4.2.1

```
>>> doctors = pd.read_csv("../datasets/doctors.csv")
>>> doctors.shape
(156, 12)
>>> doctors.head()
   permit  loc work  hours  caf  alc  weed  exrc  score
0   93273  rur  hos     21    2    0   5.0   0.0     63
1   90852  urb  cli     74   26   20   0.0   4.5     16
2   92744  urb  hos     63   25    1   0.0   7.0     58
3   73553  urb  eld     77   36    4   0.0   2.0     55
4   82441  rur  cli     36   22    9   0.0   7.5     47
```

The columns contain the following information:

- `permit`: a unique identifier for each doctor (not shown above)
- `loc`: location where the doctor lives (`rur` = rural or `urb` = urban)
- `work`: workplace type (`hos`pital, `cli`nic, or `eld`erly home)
- `hours`: number of work hours per week
- `caf`: caffeine consumption (cups per week)
- `alc`: alcohol consumption (standard drinks per week)
- `weed`: marijuana consumption (grams per week)
- `exrc`: exercise (hours per week)
- `score`: the sleep score (out of 100)

The sleep `score` is the outcome variable we want to study. We're interested in the influence of the other variables (`loc`, `work`, `caf`, etc.) on the doctors' sleep scores. Each of these variables could potentially influence the sleep `score` either positively or negatively, so we'll build a regression model that includes multiple predictor variables.

4.2.2 Multiple linear regression model

We'll now describe the math formulas for a linear model with $p = 3$ predictors, although the formulas apply generally for any value of p. A multiple regression model for the outcome variable Y based on the three predictors x_1, x_2, and x_3 is defined as follows:

$$Y \quad \sim \quad \mathcal{N}(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3, \sigma).$$

The linear structure of the model captures the combined effects of the predictors x_1, x_2, and x_3 on the outcome Y. To be more precise,

we could use the notation $Y|x_1, x_2, x_3$ or $Y(x_1, x_2, x_3)$ to make it clear the outcome variable Y depends on the predictors x_1, x_2, and x_3.

An alternative description for the same model is

$$Y \quad \sim \quad \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma)$ is a normally distributed *error term*. The `statsmodels` formula for this model is `y ~ 1 + x1 + x2 + x3`.

The parameters of this model consist of the intercept β_0, the three slope parameters β_1, β_2, and β_3, and the standard deviation of the error term σ. The β-parameters determine the *systematic* part of the model, which describes the mean of the outcome variable $\mu_Y(x_1, x_2, x_3) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$. The noise term $\mathcal{E} \sim \mathcal{N}(0, \sigma)$ models the variability in the outcome Y that remains after we have subtracted the systematic part. The structure of the model is pretty much the same as the simple linear regression model that we studied in the previous section, but there are three predictors influencing the outcome variable simultaneously.

Interpretation of the slope parameters

The model $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \mathcal{E}$ takes into account the combined effect of the three predictors x_1, x_2, and x_3. This means we have to interpret the slope parameters within this joint structure. The slope parameter β_1 measures the influence of a unit change in x_1 on the mean of Y *when* x_2 *and* x_3 *are held constant*. The expression *controlling for* x_2 and x_3 is another way of describing this situation. Similarly, the parameter β_2 measures the effect of x_2 when controlling for x_1 and x_3, while β_3 measures the effects of x_3 when controlling for x_1 and x_2.

Essentially, the statistical model assumes that (the mean of) the outcome variable is determined by a mix of influences from the three predictors. Using statistical inference allows us to "disentangle" the individual contributions of the three predictors. This is a very powerful, almost magical, ability of linear models which makes them a popular tool in many areas of science and research.

Model assumptions

Multiple linear regression models make the same assumptions as the simple linear regression models (see page 294), as well as one new assumption specific to the multivariable case.

The model equation $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \mathcal{E}$ is equivalent to n individual equations $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i$, one for each observation $(x_{i1}, x_{i2}, x_{i3}, y_i)$, where the error terms ε_i are

```
>>> from ministats import plot_resid
>>> plot_resid(lm2)
See Figure 4.17.
```

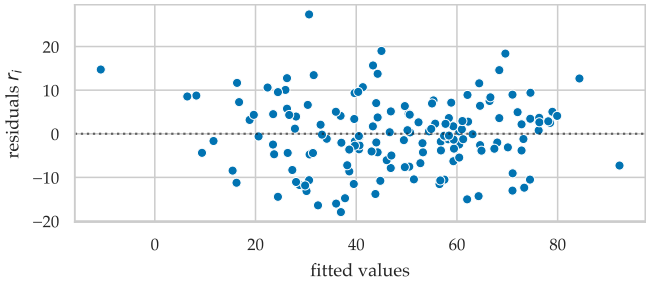


Figure 4.17: Residual plot of the residuals r_i versus the fitted values $\widehat{s}_i$.

Inspecting the residual plot in Figure 4.17 will allow us to visually check the model assumptions like the independence **(INDEPε)**, normality **(NORMε)**, and homoscedasticity **(EQVARε)** of the residuals.

We can also plot the residuals against each of the predictors. To do this, we pass the name of the predictor to the `pred` option when calling the function `plot_resid`.

code 4.2.7
```
>>> fig, (ax1,ax2,ax3) = plt.subplots(1, 3, sharey=True)
>>> plot_resid(lm2, pred="alc",  ax=ax1)
>>> plot_resid(lm2, pred="weed", ax=ax2)
>>> plot_resid(lm2, pred="exrc", ax=ax3)
The three residual plots are shown Figure 4.18.
```

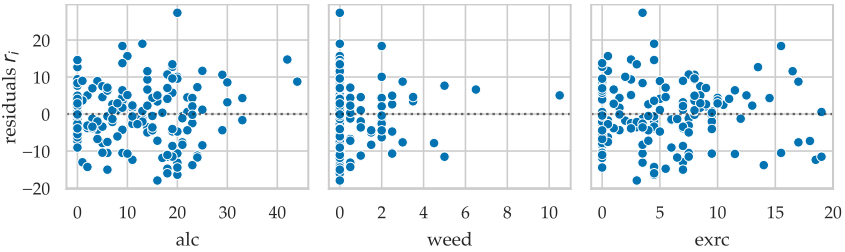


Figure 4.18: Plots of the model residuals against the three predictors.

Inspecting the residual plots in Figure 4.18 allows us to rule out any associations between residuals and individual predictors, which is another way to check the independence assumption **(INDEPε)**. We defer the detailed discussion about model diagnostics and assumption checks until Section 4.3.

4.3 Interpreting linear models

The linear regression result objects we obtain when we use the `smf.ols` function from `statsmodels` provide *a lot* of useful information and diagnostics. Here is a reminder for the code we used to obtain the linear model `lm2` for the doctors' sleep scores in the previous section.

code 4.3.1
```
>>> doctors = pd.read_csv("../datasets/doctors.csv")
>>> n = doctors.shape[0]  # number of observations
>>> formula = "score ~ 1 + alc + weed + exrc"
>>> lm2 = smf.ols(formula, data=doctors).fit()
```

We save the number of predictors ($p = 3$) in the variable `p` for use in later code blocks, then call the method `lm2.summary()` to print the model summary table shown in Figure 4.20.

code 4.3.2
```
>>> p = lm2.df_model  # = number of predictors
>>> lm2.summary()
```

OLS Regression Results

Dep. Variable:	score	R-squared:	0.842		model fit information
Model:	OLS	Adj. R-squared:	0.839		
Method:	Least Squares	F-statistic:	270.3		
Date:	Thu, 21 Mar 2024	Prob (F-statistic):	1.05e-60		
Time:	13:22:08	Log-Likelihood:	-547.63		
No. Observations:	156	AIC:	1103.		
Df Residuals:	152	BIC:	1115.		
Df Model:	3				
Covariance Type:	nonrobust				

	coef	std err	t	P>|t|	[0.025	0.975]	
Intercept	60.4529	1.289	46.885	0.000	57.905	63.000	parameter inference
alc	-1.8001	0.070	-25.726	0.000	-1.938	-1.662	
weed	-1.0216	0.476	-2.145	0.034	-1.962	-0.081	
exrc	1.7683	0.138	12.809	0.000	1.496	2.041	

Omnibus:	1.140	Durbin-Watson:	1.828	assumption checks
Prob(Omnibus):	0.565	Jarque-Bera (JB):	0.900	
Skew:	0.182	Prob(JB):	0.638	
Kurtosis:	3.075	Cond. No.	31.2	

Figure 4.20: Summary table for the regression results obtained from the formula `score ~ 1 + alc + weed + exrc` fitted to the `doctors` dataset.

The summary table has three sections: the top section contains model fit metrics, the middle section shows statistical inference results for the parameters β_0, $\beta_{\texttt{alc}}$, $\beta_{\texttt{weed}}$, and $\beta_{\texttt{exrc}}$, while the bottom section displays the results of various diagnostic assumption checks. In the remainder of this section, we'll learn to interpret and use all the information displayed in this model fit summary table.

4.3.1 Model fit metrics

Let's start with the top section of the model fit summary table (Figure 4.20), which contains the information about model fit metrics.

The metrics reported in this section tell us how well the linear regression model fits the data.

Model info The first column tells us the outcome (dependent, `Dep.`) variable is `score` and parameter estimates were obtained using the ordinary least squares (`OLS`) method. The number of observations is $n = 156$, which is the number of rows in the `doctors` data frame. The model has three predictors `alc`, `weed`, and `exrc`, so the degrees of freedom of the model are $\nu_{\text{model}} = p = 3$. The degrees of freedom of the residuals are defined as $\nu_{\text{resid}} = n - p - 1 = 152$. We'll use the degrees of freedom parameters ν_{model} and ν_{resid} to determine the uncertainty in parameter estimates and for other inference tasks.

Coefficients of determination The second column contains several goodness of fit metrics. You already know about the *coefficient of determination* $R^2 = 1 - \frac{\mathbf{SSR}}{\mathbf{TSS}}$, which measures the proportion of variance in the outcome variable that is explained by the model.

The table also shows the *adjusted* R^2, which is computed using a formula that includes a "penalty factor" based on the number of parameters:

$$\text{adjusted-}R^2 \stackrel{\text{def}}{=} 1 - \frac{\mathbf{SSR}/\nu_{\text{resid}}}{\mathbf{TSS}/(n-1)} = 1 - \frac{\mathbf{SSR}/(n-p-1)}{\mathbf{TSS}/(n-1)}.$$

Here is the code for obtaining the R^2 and the adjusted R^2 directly from the regression result object `lm2`.

code 4.3.3
```
>>> lm2.rsquared,         lm2.rsquared_adj
(   0.8421649167873537,   0.8390497506713147)
```

F-test for the overall model The table reports the F-statistic and its associated p-value `Prob (F-statistic)`, which are the results of a hypothesis test comparing the model `lm2` to a null model where all the slope parameters are zero: $\beta_{\texttt{alc}} = 0$, $\beta_{\texttt{weed}} = 0$, and $\beta_{\texttt{exrc}} = 0$. We'll discuss hypothesis tests later in this section.

Log-likelihood The *log-likelihood* metric tells us the logarithm of the likelihood of the best-fit parameters $\widehat{\beta}_0, \widehat{\beta}_{\texttt{alc}}, \widehat{\beta}_{\texttt{weed}}, \widehat{\beta}_{\texttt{exrc}}$, given the doctors dataset. The log-likelihood can also be obtained from the `lm2.llf` attribute of the regression result object `lm2`.

code 4.3.4
```
>>> lm2.llf
-547.6259042117637
```

Information criteria The Akaike Information Criterion (AIC) is computed using the formula AIC $= 2(p+1) - 2\log L$, where $p+1$ is the number of parameters and $\log L$ is the model's log-likelihood (see above). The Bayesian Information Criterion (BIC) is computed using the formula BIC $= \log(n)(p+1) - 2\log L$. The lower the AIC and BIC measures, the better the model fit. BIC penalizes the model complexity more heavily than AIC because usually $\log n > 2$.

code 4.3.5
```
>>> lm2.aic,       lm2.bic
(   1103.2518,     1115.4512)
```

* * *

Of all the model fit metrics, only R^2 has an intuitive interpretation as the proportion of variance explained by the model. The adjusted R^2, log-likelihood, F-statistic, AIC, and BIC metrics don't have an intuitive interpretation and are primarily used for model comparisons.

4.3.2 Parameter estimates

The middle section of the summary table (Figure 4.20) shows the estimated model parameters and additional statistical inference calculations for each parameter. The information for each parameter is presented on a separate row. The first column (heading `coef`) contains the estimates of the best-fitting model parameters $\widehat{\beta}_0$, $\widehat{\beta}_{\texttt{alc}}$, $\widehat{\beta}_{\texttt{weed}}$, and $\widehat{\beta}_{\texttt{exrc}}$. We can also obtain the parameters from the `.params` attribute on the regression result object `lm2`.

code 4.3.6
```
>>> lm2.params
Intercept    60.452901
alc          -1.800101
weed         -1.021552
exrc          1.768289
```

The summary table doesn't report an estimate of the standard deviation parameter σ, but we can compute the estimated standard error of the model $\widehat{\sigma} = \sqrt{\frac{\textbf{SSR}}{n-p-1}}$ by taking the square root of `lm2.scale`.

code 4.3.7
```
>>> sigmahat = np.sqrt(lm2.scale)
>>> sigmahat
8.202768119825624
```

We can combine the information from `lm2.params` and `np.sqrt(lm2.scale)` to write the complete estimated linear model for the sleep score as a function of `alc`, `weed`, and `exrc`:

$$\begin{aligned}\widehat{S} &\sim \widehat{\beta}_0 + \widehat{\beta}_{\texttt{alc}}\,\texttt{alc} + \widehat{\beta}_{\texttt{weed}}\,\texttt{weed} + \widehat{\beta}_{\texttt{exrc}}\,\texttt{exrc} + \mathcal{N}(0, \widehat{\sigma}) \\ &= 60.45 - 1.8\,\texttt{alc} - 1.02\,\texttt{weed} + 1.77\,\texttt{exrc} + \mathcal{N}(0, 8.2).\end{aligned}$$

Recall that each $\widehat{\beta}_k$ is an estimate of the unkonwn model parameter β_k. The standard error of this estimate is displayed in the second column (heading `std err`) of the middle section of the summary table (see Figure 4.20). We can also obtain the standard errors of the parameter estimates from the attribute `lm2.bse`, as shown below.

code 4.3.8
```
>>> lm2.bse
Intercept    1.289380
alc          0.069973
weed         0.476166
exrc         0.138056
```

Knowing the standard errors $\widehat{\mathbf{se}}_{\widehat{\beta}_{\texttt{alc}}}$, $\widehat{\mathbf{se}}_{\widehat{\beta}_{\texttt{weed}}}$, and $\widehat{\mathbf{se}}_{\widehat{\beta}_{\texttt{exrc}}}$ allows us construct confidence intervals and perform hypothesis tests for the unknown parameters $\beta_{\texttt{alc}}$, $\beta_{\texttt{weed}}$, and $\beta_{\texttt{exrc}}$, which is what we'll discuss next.

4.3.3 Confidence intervals for model parameters

The sampling distribution of the linear model parameter estimates is Student's t-distribution with $n - p - 1$ degrees of freedom. Knowing the point estimate $\widehat{\beta}_k$ and the standard error $\widehat{\mathbf{se}}_{\widehat{\beta}_k}$ allows us to construct a $(1 - \alpha)$-confidence interval for the unknown β_k parameter:

$$\mathbf{ci}_{\beta_k,(1-\alpha)} = \left[\widehat{\beta}_k + t_\ell \cdot \widehat{\mathbf{se}}_{\widehat{\beta}_k}\,,\;\; \widehat{\beta}_k + t_u \cdot \widehat{\mathbf{se}}_{\widehat{\beta}_k}\right],$$

where $t_\ell = F_T^{-1}(\frac{\alpha}{2})$ and $t_u = F_T^{-1}(1 - \frac{\alpha}{2})$ are the $\frac{\alpha}{2}$ and $1 - \frac{\alpha}{2}$ quantiles of Student's t-distribution with $n - p - 1$ degrees of freedom. The last two columns in middle section of the summary table show the limits of the 95% confidence intervals for the unknown parameters β_0, $\beta_{\texttt{alc}}$, $\beta_{\texttt{weed}}$, and $\beta_{\texttt{exrc}}$. We can also obtain the confidence intervals for the parameters by calling the method `lm2.conf_int` as shown below.

code 4.3.9
```
>>> lm2.conf_int(alpha=0.05)
               [0.025     0.975]
Intercept   57.905480  63.000321
alc         -1.938347  -1.661856
weed        -1.962309  -0.080794
exrc         1.495533   2.041044
```

The limits of the confidence intervals tell us a more complete picture about the "quality" of the three point estimates $\widehat{\beta}_{\texttt{alc}} = -1.80$, $\widehat{\beta}_{\texttt{weed}} = -1.02$, and $\widehat{\beta}_{\texttt{exrc}} = 1.77$. We see `alc` clearly has a negative influence on sleep scores, and `exrc` has a clear positive influence. The uncertainty in the `weed` predictor is much higher, so the influence of this predictor is not as clear.

4.3.4 Hypothesis testing for linear models

We can use the hypothesis testing machinery we developed in Chapter 3 to answer certain questions about the estimated linear model we obtained. Specifically, we can perform t-tests for individual parameters β_k and an F-test for the overall model. In fact, when we called the method `lm2.summary()`, we already performed all of these tests, and their results are displayed in the summary table shown in Figure 4.20. Let's now explain the logic behind these tests and describe how to interpret their results.

T-tests for individual parameters

The goal of this type of test is to check if the slope parameter β_k is zero or non-zero. The two competing hypotheses are as follows:

$$H_0 : \beta_k = 0, \qquad H_A : \beta_k \neq 0.$$

We can perform this test for each of the parameters in the model. To make things more concrete, let's focus on the parameter $\beta_{\texttt{weed}}$ in the linear model `lm2`. The null hypothesis is $H_0 : \beta_{\texttt{weed}} = 0$, which corresponds to a skeptical claim that the variable `weed` has *no effect* on sleep scores. According to H_0, we could remove variable `weed` from the model altogether, so the model under the null hypothesis is

$$H_0 : \quad S \sim \mathcal{N}(\beta_0 + \beta_{\texttt{alc}}\cdot\texttt{alc} + \beta_{\texttt{exrc}}\cdot\texttt{exrc},\ \sigma).$$

Note the term $\beta_{\texttt{weed}}\cdot\texttt{weed}$ is missing from the model. The alternative hypothesis states that `weed` has an effect on sleep scores, so it must be included in the model:

$$H_A : \quad S \sim \mathcal{N}(\beta_0 + \beta_{\texttt{alc}}\cdot\texttt{alc} + \beta_{\texttt{weed}}\cdot\texttt{weed} + \beta_{\texttt{exrc}}\cdot\texttt{exrc},\ \sigma).$$

To distinguish between these two hypotheses, we'll use the point estimate $\widehat{\beta}_{\texttt{weed}} = -1.02$, the estimated standard error $\widehat{\mathbf{se}}_{\widehat{\beta}_{\texttt{weed}}} = 0.476$, and the knowledge that the sampling distribution of parameter estimates is a t-distribution with ν_{resid} degrees of freedom. We first compute the t-statistic $t = (\widehat{\beta}_{\texttt{weed}} - 0)/\widehat{\mathbf{se}}_{\widehat{\beta}_{\texttt{weed}}}$, then obtain the p-value from the t-distribution with $\nu_{\text{resid}} = n - p - 1$ degrees of freedom.

code 4.3.10

```
>>> obst_weed = (lm2.params["weed"] - 0) / lm2.bse["weed"]
>>> obst_weed    # = lm2.tvalues["weed"]
-2.145370545436854
>>> from scipy.stats import t as tdist
>>> pleft = tdist(df=n-p-1).cdf(obst_weed)
>>> pright = 1 - tdist(df=n-p-1).cdf(obst_weed)
>>> pvalue_weed = 2 * min(pleft, pright)
>>> pvalue_weed  # = lm2.pvalues["weed"]
0.033511561813423775
```

Definitions and notation

Let's start with some definitions of the new concepts.

- An **outlier** is an observation for which the observed value y_i is far from the model prediction $\widehat{y}_i$. Outliers increase the standard deviation estimate $\widehat{\sigma}$ and lower the coefficient of determination R^2. Outliers may or may not also affect the slope parameter estimates.
- A **high-leverage point** is an observation for which the predictor x_k has an extreme value relative to its mean $\overline{\mathbf{x}}_k$. High leverage points have the potential to "pull" the best-fit line toward them more strongly than observations close to the mean $\overline{\mathbf{x}}_k$.
- An **influential observation** is an observation that is both an outlier and high leverage. These points strongly influence the slope parameter estimates. If we were to exclude these observations from the analysis, the estimated model parameters will change.

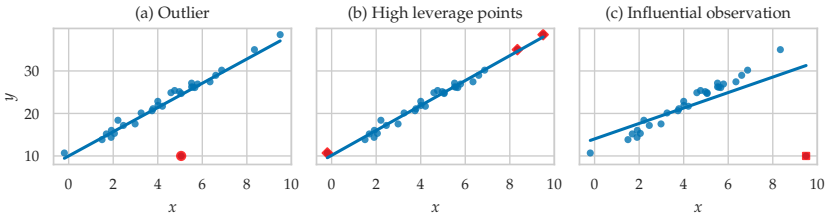


Figure 4.24: Examples of linear model fits affected by: (a) the presence of an outlier (marked as a larger circle), (b) high-leverage points (marked as a diamonds), and (c) an influential observation (marked as a square).

We previously discussed outliers in Section 1.2 (see page 52), and defined them as points that are much larger or smaller than the others. In the context of a linear models, outliers are not always a problem. For example, outlier that appears near the middle of the dataset don't affect the best-fitting linear model too much, as we see in Figure 4.24 (a). Similarly, the high leverage points shown in Figure 4.24 (b) don't affect the slope because they follow the overall trend. We only need to worry about *influential observations* that are both outliers *and* have high leverage, because these points tend to exercise an undue influence on the best-fit regression line, as we see in Figure 4.24 (c).

- The *Jarque–Bera hypothesis test* is another normality test based on the skewness and kurtosis of the residuals. The summary table shows the value of the `Jarque-Bera(JB)` statistic and the associated p-value `Prob(JB)=0.638`.

The p-values of the both the omnibus and the Jarque–Bera tests are high, so there are are no reasons to suspect there is a violation of the **(NORMε)** assumption.

I mention these tests only because they appear in the summary table, but you don't need to think about these tests too much in your statistical analyses. A much more important skill you should develop is looking at the various diagnostic plots (partial regression plots, residual plots, location-scale plot) to detect violations of model assumptions.

4.3.9 Discussion

I know this has been a long section and you must be tired from all the new material. The good news is we're finished now, and you made it! Here is a concept map that summarizes the connections between all the concepts we discussed in this section.

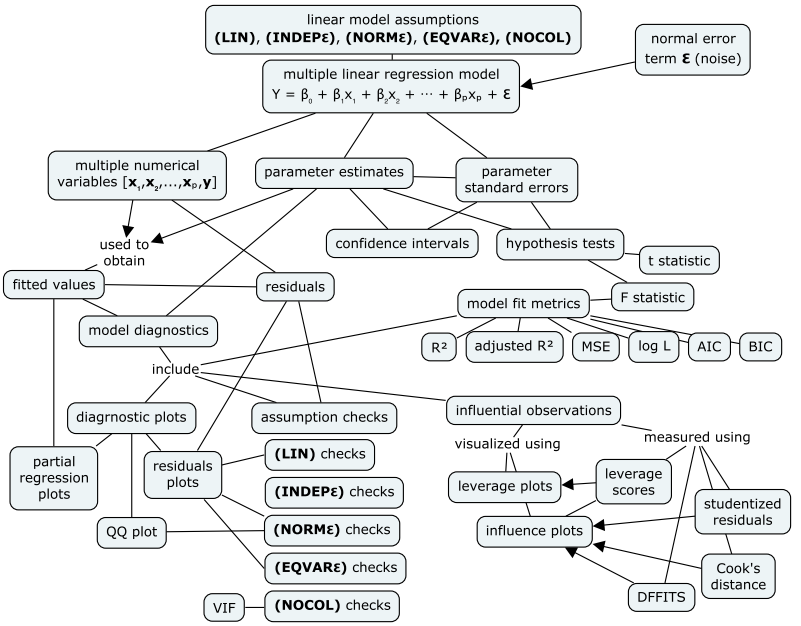


Figure 4.27: Concept map showing the model fit metrics and diagnostics used for interpreting the results of linear models.

4.4 Regression with categorical predictors

We include categorical predictor variables in linear models by using a clever encoding strategy that is called *dummy coding*. In this section, we'll continue working with the doctors dataset, and learn how sleep scores are influenced by the two categorical variables `loc` (location with values `rur` = rural or `urb` = urban) and `work` (workplace type with values `hospital`, `clinic`, or `elderly` home). We'll also describe an interesting correspondence between linear models and some of the statistical tests that we saw in Chapter 3 like the two-sample t-test and the one-way ANOVA test.

4.4.1 Definitions

Let's start by defining the new concepts that you need to know about to understand the techniques presented in this section.

- *Design matrix*: a matrix (rectangular array of numbers) whose rows correspond to different observations, and whose columns correspond to different predictors.
- *Indicator variable* $\texttt{var}_{\texttt{val}}$: a variable that is equal to 1 when the categorical variable `var` has the value `val`, else is equal to 0.
- *Dummy coding*: the process of encoding a categorical variable with K possible values in terms of a reference category and $K-1$ indicator variables for the other categories.

For example, the dummy coding of a categorical variable `cat` that can take on three possible values `A`, `B`, and `C` can be done by choosing the value `A` as the reference value, and defining indicator variables $\texttt{cat}_{\texttt{B}}$ and $\texttt{cat}_{\texttt{C}}$ for the other two possible values. We'll explain dummy coding through several examples in the rest of this section, but first I need to tell you about design matrices.

Design matrices for linear models

In previous sections, we discussed linear models without using the language of linear algebra, because this allowed us to skip some non-essential complications. I was trying to protect you from unnecessary details and notation about matrices and vectors. In order to understand the material in this section, however, we need to use the language of linear algebra, so we have some catching up to do. In particular, the concept of a *design matrix* is essential to understanding how to encode categorical variables for use in linear models.

Consider the multiple linear regression model $Y \sim \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \mathcal{E}$ that we want to fit based on the dataset $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p, \mathbf{y}]$. The dataset contains n observations. The linear model formula corresponds to a *design matrix* that includes the p predictor variables as columns:

$$X = \underbrace{\begin{bmatrix} 1 & | & | & & | \\ \vdots & \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_p \\ 1 & | & | & & | \end{bmatrix}}_{p+1 \text{ columns}}.$$

The first column contains 1s and corresponds to the constant term β_0. The dimension of the design matrix is $n \times (p+1)$, since it contains the data from n observations, and the model has $p+1$ terms.

The design matrix allows us to express the model using compact linear algebra notation $\mathbf{y} = X\boldsymbol{\beta} + \mathcal{E}$, where $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2, \ldots, \beta_p)$ is the vector of unknown parameters. The linear algebra approach for finding the parameter estimates vector $\widehat{\boldsymbol{\beta}}$ is to use the Moore–Penrose pseudoinverse formula $\widehat{\boldsymbol{\beta}} = (X^\mathsf{T}X)^{-1}X^\mathsf{T}\mathbf{y}$, which is a standard linear algebra technique for solving equations. We can then write the model predictions (fitted values) using matrix-vector product $\widehat{\mathbf{y}} = X\widehat{\boldsymbol{\beta}}$.

If you're not familiar with linear algebra concepts, you can safely ignore the formulas in the previous paragraph. The only thing you need to know is that each linear model formula is equivalent to a design matrix whose columns correspond to the predictor variables.

Let's look at the design matrix of the simple linear regression model `lm1` that we studied in Section 4.1. We can inspect the design matrix via the attribute `.model.exog`. The code below shows the first three rows of the design matrix $X_{\texttt{lm1}}$.

code 4.4.1

```
>>> import pandas as pd
>>> import statsmodels.formula.api as smf
>>> students = pd.read_csv("../datasets/students.csv")
>>> lm1 = smf.ols("score ~ 1 + effort", data=students).fit()
>>> lm1.model.exog[0:3]  # first 3 rows of the design matrix
array([[ 1.  , 10.96],
       [ 1.  ,  8.69],
       [ 1.  ,  8.6 ]])
>>> students["effort"].values[0:3]   # first 3 effort values
array([10.96,  8.69,  8.6 ])
```

The first column of the design matrix contains 1s (the constant terms), while the second column contains the values of the `effort` variable.

We can also look at the design matrix $X_{\texttt{lm2}}$ for the multiple linear

regression model `lm2` that we discussed in sections 4.2 and 4.3. Once again, we only show the first three rows.

code 4.4.2

```
>>> doctors = pd.read_csv("../datasets/doctors.csv")
>>> formula = "score ~ 1 + alc + weed + exrc"
>>> lm2  = smf.ols(formula, data=doctors).fit()
>>> lm2.model.exog[0:3]
array([[ 1. ,  0. ,  5. ,  0. ],
       [ 1. , 20. ,  0. ,  4.5],
       [ 1. ,  1. ,  0. ,  7. ]])
```

The first column contains 1s, while the second, third, and fourth columns contain the values of the `alc`, `weed`, and `exrc` variables.

The formulas we provide to the `statsmodels` function `smf.ols` are a user-friendly way to describe the linear model. Under the hood, `statsmodels` converts the formula into the appropriate design matrix, then uses linear algebra to obtain the vector of model parameter estimates $\widehat{\boldsymbol{\beta}}$. You don't need to build design matrices by hand, but you need to know about design matrices to understand how to encode categorical variables, which is what we'll discuss next.

4.4.2 Example 1: binary predictor variable

The doctors dataset contains the variable `loc` that encodes the location where the doctors live. The `loc` variable can take on one of two possible values: `rur` for rural doctors and `urb` for urban doctors.

Suppose we want to build a simple linear regression model for the sleep `score` based on the `loc` variable as a predictor. We can't use the `loc` variable directly because it is not a numerical quantity. Instead, we define an *indicator variable* that takes on the value 1 when the `loc` has the value `"urb"` and zero otherwise:

$$\texttt{loc}_{\texttt{urb}} = \begin{cases} 1 & \text{if } \texttt{loc = "urb"}, \\ 0 & \text{otherwise.} \end{cases}$$

We can now define a linear model for the sleep score S as follows:

$$S \sim \beta_0 + \beta_{\texttt{urb}} \cdot \texttt{loc}_{\texttt{urb}} + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma)$ is a normally distributed error term. The linear model equation is a condensed way to describe the models for the two groups of doctors:

$$S_{\texttt{rur}} = \mathcal{N}(\beta_0, \sigma) \qquad \text{and} \qquad S_{\texttt{urb}} = \mathcal{N}(\beta_0 + \beta_{\texttt{urb}}, \sigma).$$

The unknown mean for the rural doctors corresponds to β_0, while the mean for urban doctors is $\beta_0 + \beta_{\texttt{urb}}$, and we assume the variability in both groups is the same (the **(EQVAR)** assumption). Recall

we previously discussed the comparison of two groups in Section 3.5 using a different set of parameters: the group means $\mu_{\texttt{rur}}$ and $\mu_{\texttt{urb}}$, and the difference between them $\Delta = \mu_{\texttt{urb}} - \mu_{\texttt{rur}}$. We'll discuss the equivalence between linear models with categorical predictors and what we studied in Chapter 3 later in this section. For now, let's continue with the discussion, and see how we can use the machinery of linear models for comparing two groups.

The `statsmodels` formula that corresponds to the above math equation is `"score ~ 1 + C(loc)"`, where `C(...)` is the syntax we use to include categorical predictor variables in a linear model. Let's fit this model and see the parameter estimates we obtain.

code 4.4.3
```
>>> lmloc= smf.ols("score ~ 1 + C(loc)",data=doctors).fit()
>>> lmloc.params
Intercept         52.956522
C(loc)[T.urb]     -6.992885
```

The `Intercept` term corresponds to the estimate of the mean score for rural doctors $\beta_0 = \mu_{\texttt{rur}}$. The second parameter is labelled `C(loc)[T.urb]` and corresponds to the best-fit estimate $\widehat{\beta}_{\texttt{urb}}$ for the slope parameter $\beta_{\texttt{urb}}$. The slope parameter encodes the difference between mean scores for urban doctors relative to the rural scores: $\beta_{\texttt{urb}} = \Delta = \mu_{\texttt{urb}} - \mu_{\texttt{rur}}$.

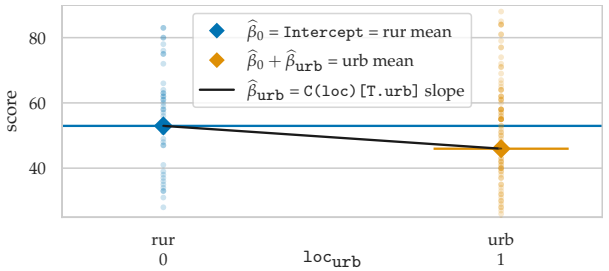


Figure 4.28: Results of the linear model fit `score ~ 1 + C(loc)`.

Figure 4.28 shows a visual summary of the `lmloc` fitted model. The slope of the line corresponds to the difference between group means. The slope is negative, which means the sleep scores of urban doctors are worse than those of rural doctors. Specifically, the estimate $\widehat{\beta}_{\texttt{urb}} = \texttt{C(loc)[T.urb]} = -6.99$ tells us that urban doctors score about 7 points lower on average.

Note that including "`C(loc)`" in the `statsmodels` formula automatically converted the categorical variable `loc` into the indicator variable $\texttt{loc}_{\texttt{urb}}$. We can see this by looking at the first few rows of the `loc` column in the `doctors` data frame, and comparing them to the values in the model's design matrix $X_{\texttt{lmloc}} = \texttt{lmloc.model.exog}$.

code 4.4.4

4.5 Causal inference using linear models

Linear models allow us to study the effects of one or more predictor variables on an outcome variable of interest. In previous sections, we learned how to fit linear models and how to run model diagnostic checks. In this section, we'll describe some additional steps we must take, if we want to interpret the parameter estimate $\widehat{\beta}_{\mathtt{x}}$ as a *causal association* between the predictor X and the outcome variable Y.

Causal inference is a tricky business! In particular, inferring causal associations from observational data is a challenging task because of the possible presence of *confounding variables* that might influence the apparent association between the predictor X and the outcome Y. If confounders are not handled correctly, the parameter estimates we obtain will be misleading about the strength of the causal association between X and Y. The situation is not totally hopeless, though: tools like causal graphs and techniques like the statistical control of confounders can allow us to infer causal associations even from observational data. Specifically, the focus of this section will be on the rules for choosing which variables to include as predictors in a linear model.

4.5.1 Causal inference from observational data

We'll start with a high-level overview of the statistical context we're working in, namely fitting linear models to observational data. We'll describe the various difficulties that we face when trying to obtain causal associations to motivate the need for the tools and techniques that we'll discuss in the remainder of the section.

Different use cases for linear models

There are multiple reasons why we might want to fit a linear model to a particular dataset: *descriptive* modelling, *predictive* modelling, and modelling for *causal inference*. In each case, the mechanics of fitting the linear model are the same, but the uses and interpretation of the results are different.

When our goal is descriptive modelling, we try to capture all patterns in the data and draw a regression line (or curve) that most closely resembles the data. We don't pay any particular attention to the model parameters, treating them as "control knobs" that we tweak to produce the model shape that matches the observations.

In predictive modelling, our goal is to learn the pattern of association between the predictor(s) and the outcome variable from a given dataset, so that we can later predict the outcome variable

in new observations. We measure the quality of the model by its predictive accuracy, as described in Section 4.3 (see page 354). We don't care about the values of the fitted parameters, seeing them as levers inside a machine for making predictions. Linear models are often used for prediction in industry and in applied research.

When building a linear model for causal inference, we want to find the *causal effect* of the predictor X on the outcome Y, in the presence of other variables. We want the parameter estimate $\widehat{\beta}_x$ we obtain from the fitted model to correctly reflect the strength of the *causal association* between X and Y. You can think of causal inference as a special case of prediction, where we're interested in predicting what happens if we increase the predictor X by one unit. The quantification of cause-and-effect relationships between variables is the primary goal of research in science, engineering, medicine, and the social sciences.

Statistical experiments versus observational studies

We previously discussed causal inference in the context of statistical experiments, where we manipulate (choose the value of) the predictor variable X, then observe its effect on the outcome variable Y. For example, we can randomly assign participants to an intervention and control groups, apply the treatment only to the intervention group, then look for differences between the outcomes in the two groups. The hope is that random assignment creates two groups that are roughly identical (*ceteris paribus*), so that we can measure the *causal effect* of the intervention.

In observational studies, we're also interested in the causal effect of the predictor X on the outcome Y, but we don't have control of the predictor X, we're only observing it. The maxim "correlation is not causation" applies here, and prevents us from drawing causal conclusions from observational data. The fact is, any association that we observe between X and Y could be a direct causal link $X \rightarrow Y$, or due to some *confounder* variable W that is a common cause for both X and Y, which we can visualize as $X \leftarrow W \rightarrow Y$.

The goal of causal inference from observational data is to *control for* the influence of confounding variables so that we can estimate the true strength of the *causal association* between X and Y, which we'll denote $\beta_{\mathbf{x}}$. Causal inference from observational data requires careful analysis of the relationships between predictors and the outcome variable, and controlling for the right set of variables in order to remove confounding. We'll refer to the results we obtain from observational data after controlling for confounding as *causal associations*, to differentiate from the stronger language of *causal*

effects used to describe the results of statistical experiments.

Statistical control of confounders

When studying the causal influence of the predictor X on the outcome Y, we may want to *control for* a third variable W to avoid its confounding effect. The most common approach for statistical control of confounders is to use *multivariable adjustment*, which means including the variable W as a predictor in the linear model. Multivariable adjustment allows us to control for several confounders simultaneously W_1, W_2, W_3, by including them as predictors in the model. We refer to the variables we include in the model to control for confounding as the *adjustment set* and the resulting linear model as the *adjusted model*.

Consider a scenario in which a confounder variable W influences both the predictor variable X and the outcome variable Y. If we fit a linear model based on the `statsmodels` formula `y ~ 1 + x`, we would obtain an inaccurate estimate of the causal association between X and Y, because of the confounding effect of W. We can remove this confounding effect by including the variable W in the model, which means fitting a linear model based on the `statsmodels` formula `y ~ 1 + x + w`. Intuitively, including the confounding variable W in the adjusted model has the effect of subtracting (regressing out) the effects of W from the outcome variable Y and the predictor X. The parameter estimate $\hat{\beta}_{\texttt{x}}$ that we obtain from the adjusted model `y ~ 1 + x + w` is equivalent to a linear model for the residuals of the model `y ~ 1 + w` (the part of Y that is not explained by W) against the residuals of the model `x ~ 1 + w` (the part of X that can't be explained by W). This is exactly what we need to estimate the true causal effect of X on Y, without the confounding effect of the variable W on X and Y getting in the way.

There are other means of statistical control like *matching*, *restriction*, and *stratification*, but we'll focus on the multivariable adjustment approach for statistical control, since it is the most common approach and requires only the linear modelling techniques you're already familiar with.

The opposite of controlling for a variable is to *marginalize* over it, which is what happens when we don't include this variable in the model. By not including the variable in the model, we're letting the variable vary freely, without taking its values into account in the model equation. You can think of marginalization as "leaving the variable alone" which is the correct strategy to use for variables that are not confounders, as we'll see next.

Overadjustment Controlling for variables by including them in the model is not always a good thing. In some cases, including an additional variable Z in the regression model can lead to a biased estimate of the causal association between the variables X and Y. We refer to this phenomenon as *overadjustment* or, more informally, as *bad controls*.

We can classify the variables that are candidates for inclusion in a linear model into two categories:

- **Good controls**: variables whose inclusion in the model eliminates or reduces confounding effects.
- **Bad controls** (overadjustment): variables whose inclusion in the model introduces new confounding effects.

The fundamental task of causal inference from observational data is to decide which variables to include in the model (control for) and which variables we leave out of the model (marginalize over). We want to include all the variables needed to prevent confounding effects, but avoid the overadjustment problems caused by introducing too many controls. We'll refer to this task as *variable selection*.

Naive strategies such as including all available variables in the model, or selecting variables based on model fit metrics are not suitable solutions, since they are just as likely to choose bad controls or omit good controls. Instead, we need to carefully consider the causal structure between variables, and make our decision based on the role each variable plays.

The main objective of causal inference In summary, the goal of statistical control is to control for all confounding variables. If we make the right choices, the parameter estimate $\widehat{\beta}_{\mathtt{x}}$ we obtain from the linear regression model will be an unbiased estimate of $\beta_{\mathtt{x}}$, the true causal association between the predictor X and the outcome Y.

Next, we'll describe a powerful graphical tool (causal graphs) that can help us choose the variables we must control for in various situations. We'll then look at several examples that show what happens when we include both good and bad controls.

4.5.2 Causal graphs

We'll now introduce a visual tool for thinking about the causal influences between variables. A *causal graph* consists of *nodes* connected by *arrows*. The nodes represent variables that are relevant for the current statistical analysis. The arrows represent causal influences between variables. An arrow from node A to node B indicates that changes in the variable A will cause changes in the variable

B. The arrows don't specify the direction (positive or negative) or the probability distribution of the relationship between the variables. Causal graphs are sometimes referred to as *directed acyclic graphs* (DAGs).

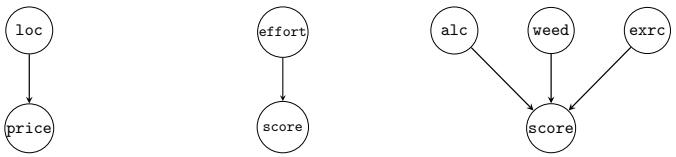
Applications of causal graphs

Causal graphs allow us to visualize key concepts like causal chains and confounding variables, which makes them a useful tool for doing causal inference. By drawing a causal graph that shows the connections between all variables of interest, we make our assumptions about causal links explicit. This intuitive graphical representation facilitates discussions between domain experts and statisticians.

We can use causal graphs to identify which variables we must *control for* to avoid confounding effects and thus obtain the causal associations we're interested in measuring, as we'll explain in the remainder of this section.

Simple graphs

Let's start by looking at some simple examples of causal graphs for the statistical relationships that we studied previously in the book.



(a) Electricity prices. (b) Student scores (`lm1`). (c) Sleep scores (`lm2`).

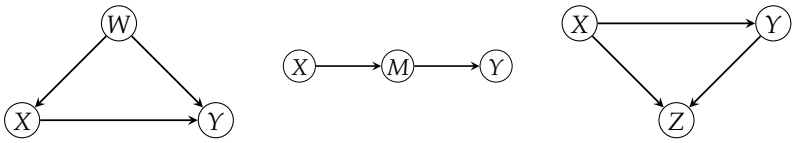
Figure 4.31: Causal graphs from statistical analyses we've previous discussed: (a) the electricity prices `price ~ 1 + C(loc)`, (b) student scores as a function of effort `score ~ 1 + effort`, and (c) the doctor's sleep score dependence on three predictors `score ~ 1 + alc + weed + exrc`.

The arrows in each of the examples shown in Figure 4.31 indicate the direction of causal influence that we believe exists. The graph in (a) shows our belief that the location of the charging station influences the price, which we studied in Example 1 in Section 4.4. The graph in (b) indicates we believe that student's effort influences their scores, which we studied in Section 4.1. The graph in (c) displays the three predictors that we believe influence the doctor's sleep scores, which we studied in sections 4.2 and 4.3. The absence of arrows between the predictors `alc`, `weed`, and `exrc` represents our assumption that the three predictors are independent and don't influence each other.

Causal graphs with influences between predictors

Causal graphs become particularly useful when there are more complicated relationships between the predictors and the outcome variable. Figure 4.32 shows three common causal graph patterns that are important to know about. In each pattern, we're interested in estimating the causal association between the predictor X and the outcome Y. We'll denote the true causal effect of X on Y as $\beta_{\mathtt{x}}$. This is the quantity we would obtain if we were able to manipulate X like in an experiment.

The presence of extra variables W, M, and Z can potentially influence the strength of the apparent causal association estimate $\widehat{\beta}_{\mathtt{x}}$ that we obtain when we fit a linear model. If we want the estimate $\widehat{\beta}_{\mathtt{x}}$ that we obtain from our model to be close to the true $\beta_{\mathtt{x}}$, we must handle the extra variable correctly in each scenario: either by controlling for it (including it in the model) or by marginalizing over it (not including it in the model).



(a) The fork pattern. (b) The pipe pattern. (c) The collider pattern.

Figure 4.32: Three common patterns that occur in causal graphs: (a) the *fork* pattern occurs when a common cause W influences both the predictor and the outcome, (b) the *pipe* pattern occurs when a mediator M acts as a conduit of the influence from predictor to outcome, and (c) the *collider* pattern occurs when both the predictor and the outcome influence a third variable Z.

We use the following terminology to describe the extra variables in the causal graphs shown in Figure 4.32.

- A *confounder* is a variable that influences both the predictor and the outcome, like the variable W in the middle of the *fork* pattern in Figure 4.32 (a). The common cause W influences the apparent association between X and Y, and is the canonical example of the *confounding* problem.
- A *mediator* is a variable on the causal path between the predictor and the outcome variable, like the variable M in the middle of the *pipe* pattern shown in Figure 4.32 (b). The flow of causal association between X and Y passes through the intermediate variable M. The pipe pattern is sometimes called a *causal chain*.
- A *collider* is a variable that is simultaneously influenced by two variables in the graph, like the variable Z in Figure 4.32 (c). The collider variable Z is influenced by both the predictor X

and the outcome variable Y, but it doesn't influence the causal association between X and Y. The collider pattern is sometimes called an *inverted fork*.

The correct choice of variables to *control for* (include in the model) in each pattern is different, and depends on the role the variable plays in the causal graph.

- We **must control for the confounder** W to avoid its confounding effect on the association between X and Y.
- We **must not control for the mediator** M to avoid blocking the causal association that flows though the pipe from X to Y.
- We **must not control for the collider** Z to avoid inducing a noncausal association between X and Y.

If we follow the above prescriptions for statistical controls, we'll obtain a parameter estimate $\widehat{\beta}_{\mathtt{x}}$ from the linear model that approximates $\beta_{\mathtt{x}}$, the true causal association between X and Y.

We'll now introduce some additional terminology for describing the flow of association in causal graphs.

Causal and noncausal paths

Within a causal graph, a *path* is a set of nodes connected by arrows. The arrows along the path don't need to point in the same direction. We can use the language of paths to discuss the long-distance connections and flows of association between variables. We can subdivide paths into two categories:

- **Causal paths** are paths starting at the predictor X and ending at the outcome Y that consists of arrows pointing in the same direction. Causal paths include the *direct causal effect*, represented by an arrow from X to Y, and *indirect causal effects* that pass through mediators, like the pipe pattern $X \rightarrow M \rightarrow Y$ in Figure 4.32 (b). We sometimes refer to the sum of direct and indirect causal paths as the *total causal effect* of X on Y.
- **Noncausal paths** are paths connecting the predictor X to the outcome Y that include one or more backward arrows. The fork and the collider patterns in Figure 4.32 are examples of noncausal paths. In particular, we care about *backdoor paths*, which are noncausal paths with an arrow pointing into the predictor X. The fork pattern in Figure 4.32 (a) is an example of a backdoor path. Backdoor paths produce confounding effects between the predictor and outcome variables, and so we must control for them if we want our model to give us an estimate of the true causal effect.

In general, we will have to construct separate models to study the causal influence for each predictor, since evaluating the causal influence of each predictor requires different controls. You can't just fit one model and obtain all the causal association estimates at once. For each causal association you want to study, you have to start with the causal graph to find which variables you need to control for and build a different model with those specific controls.

If we're also interested in the causal association influence of the predictor W on the outcome Y, we would need to build a model that includes the appropriate statistical controls. For example, to estimate the total causal influence $W \to Y$ in the causal graph of Example 1 (see Figure 4.33 on page 388), we would need the model based on the formula `y ~ w`, which doesn't include X, since X acts as a mediator between W and Y.

4.5.7 Case study: smoking and lung function in teens

We'll now look at a complete example of a statistical analysis scenario to show how causal graphs and the backdoor criterion are used in realistic situations. The forced expiratory volume (FEV) is a measure of healthy lung function that is calculated by measuring the volume expelled during one second of constant effort. The dataset `smokefev.csv` contains FEV measurements of 654 children and teens aged between 3 and 19, that were collected as part of a longitudinal study of smoking and lung function[TWM+83, TMR+85]. The dataset was later used in statistics education[R+06] and in particular for causal inference[CAP+20]. Let's load the dataset and look at the last rows:

code 4.5.24
```
>>> smokefev = pd.read_csv("../datasets/smokefev.csv")
>>> smokefev.tail(3)
     age    fev  height sex smoke
651   18  2.853    60.0   F    NS
652   16  2.795    63.0   F    SM
653   15  3.211    66.5   F    NS
```

The dataset includes measurements of the following variables:

- `age`: the age of the subject in years.
- `fev`: forced expiratory volume (measured in litres).
- `height`: the height of the subject in inches.
- `sex`: biological sex of the subject (`F` or `M`).
- `smoke`: if the subject had ever smoked (`SM`) or not (`NS`).

We're interested in estimating the effect of smoking (`smoke`) has on the forced expiratory volume (`fev`). The other variables have been collected so that we can control for possible confounding effects.

Descriptive statistics

We start by looking at the descriptive statistics of the dataset.

code 4.5.25
```
>>> smokefev.describe()
             age       fev    height
mean       9.931     2.637    61.144
std        2.954     0.867     5.704
min        3.000     0.791    46.000
median    10.000     2.548    61.500
max       19.000     5.793    74.000
```

Next, we calculate the mean `fev` for the smokers (`smoke="SM"`) and nonsmokers (`smoke="NS"`) groups.

code 4.5.26
```
>>> smokefev.groupby("smoke")["fev"].mean()
smoke
NS     2.566143
SM     3.276862
```

The difference between the two groups is $\overline{\texttt{fev}}_{\texttt{SM}} - \overline{\texttt{fev}}_{\texttt{NS}} = 0.7107$, which tell us the smokers in our dataset had higher FEV on average. We can further split the results by `sex`, as shown in Figure 4.47.

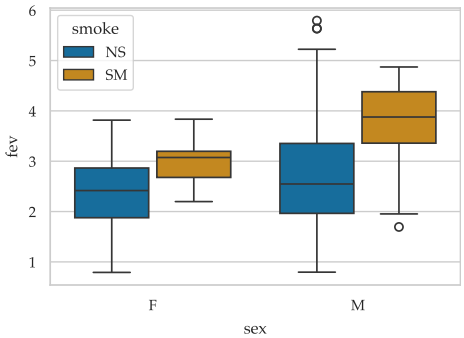


Figure 4.47: Comparison of the FEV for smokers and non smokers.

This is very counterintuitive result: it seems smoking is actually associated with higher values of forced expiratory volume, and thus better overall lung function. Give them kids a pack of cigs per day, and watch them collect all the Olympic medals in a few years!

The counterintuitive observation is easily explained when we consider the `age` variable. Figure 4.48 shows a scatterplot of `fev` versus `age`, with smokers and nonsmokers indicated using different markers. We can clearly see that smokers tend to be older individuals, with larger lung capacity. In other words, `age` is a confounding variable in this analysis.

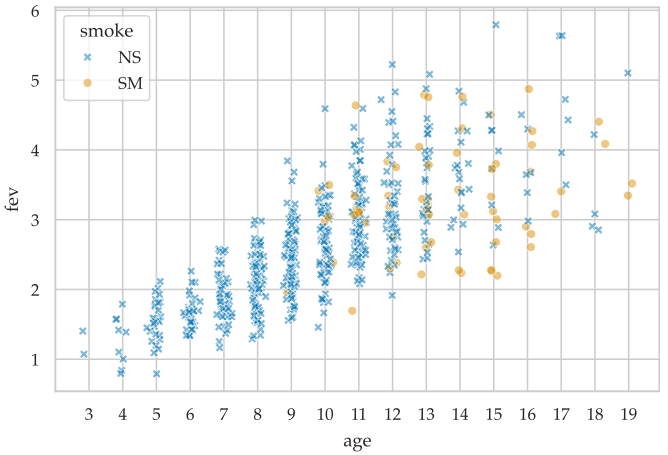


Figure 4.48: Scatter plot of `dev` versus `age`. Smokers and nonsmokers are indicated by different markers. Note most of the smokers are older students.

Fit an unadjusted model

If we fit a linear model without controlling for any of the confounders, we obtain a misleading result.

code 4.5.27

```
>>> formula_unadj = "fev ~ 1 + C(smoke)"
>>> lmfev_unadj =smf.ols(formula_unadj,data=smokefev).fit()
>>> lmfev_unadj.params
Intercept          2.566143
C(smoke)[T.SM]     0.710719
```

The parameter estimate $\widehat{\beta}_{\texttt{smoke}} = 0.7107$ we obtain from the unadjusted model echoes the simple difference in means we calculated earlier. Any negative effect of `smoke` on `fev` is drowned by the confounding variable `age`, which is positively associated with both `smoke` and `fev`.

Clearly, we need to adjust for confounders if we want to obtain the true causal association between `smoke` and `fev`.

Drawing the causal graph

Since we're interested in the causal association `smoke` $\rightarrow$ `fev`, the first step is to draw a causal graph between all the variables in the dataset. The general strategy when building the causal graph is to add all links that could conceivably exist—we need to be inclusive and add the arrow, except for relationships that are impossible. For example, the variable `sex` can influence all other variables except `age`. Similarly, `age` likely influences all other variables other than `sex`. It is also fair to assume smoking (`smoke`) is influenced by `sex`

and `age`, but doesn't influence these variables. We include an arrow `smoke` → `fev`, since this is the direct causal effect we're interested in measuring. We also add an arrow from `smoke` → `height` since it is plausible that smoking affects growth. We add an arrow `height` → `fev` because tall people tend to have larger lungs. Figure 4.49 shows the completed causal graph, which includes all the causal effects that we believe exist in this situation. The construction of the causal graph is subject to debate and updating.

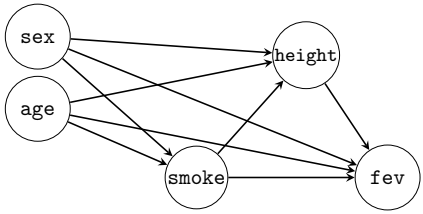


Figure 4.49: Causal diagram of links between the variables in the dataset.

In the ideal case, you should consult with a pulmonologist to confirm that each arrow $A \rightarrow B$ in the causal graph makes sense physiologically. Specifically, there should be some plausible mechanism that explains *how* the variable at the beginning of the arrow causes the variable at the end of the arrow. The specific mechanism (theory) will depend on different domains, and vary from just guess, random idea, abstract theories, or testable theories, etc. In other words, we would like all arrows (causal links) to be backed by some theory. You should also ask the domain expert about other variables you should include in the causal graph. Even if they tell you about variables that would be difficult to measure, it's still useful to include these variables in the causal graph so that you'll know about possible confounding paths.

In the worst case scenario, you have no expertise or theory about any of the relationships between the variables in the study, other than the fact you want to estimate the strength of the causal association between the predictor X and the outcome Y. In such situations, you should start by assuming all possible arrows in the graph are plausible, then "prune" the causal graph using common sense to decide which arrows couldn't possibly exist (no plausible mechanism).

If you're feeling a little uneasy about the arbitrariness of the process we used to draw the causal graph, you're not alone. I've commented many times in the previous chapters about the *ad hoc* nature of statistical analysis, and this example reinforces that point. More importantly, this case study highlights the overall importance of statistical thinking, and the comparative unimportance of statis-

tical calculations (number crunching). The real statistics happens in the design phase, when we think about formulating the problem correctly, the causal associations, confounding effects, and measurements. The actual numerical calculations like calculating means and fitting models is of negligible importance once you know what you're doing. Besides, `numpy`, `scipy`, and `statsmodels` do all the heavy lifting for you anyway.

But I digress... Let's get back on task and continue our analysis and estimation of the `smoke` → `fev` causal association.

Applying the backdoor path procedure

Using the causal graph in Figure 4.49, we can apply the backdoor path criterion to figure our which variables we need to control for in order to obtain the correct estimate of the causal association between the predictor `smoke` and the outcome variable `fev`. We'll follow the three step procedure outlined in Section 4.5.6 (see page 400).

Step 1 is to identify all the backdoor paths that connect `smoke` and `fev`. We can identify four such noncausal paths starting with an arrow pointing into `smoke`:

- `smoke` ← `age` → `fev`
- `smoke` ← `age` → `height` → `fev`
- `smoke` ← `sex` → `fev`
- `smoke` ← `sex` → `height` → `fev`

Step 2 tells us we need to focus only on `open` backdoor paths. Each of the four backdoor paths listed above is open, since they don't contain colliders.

Step 3 tells us we need to control for one variable along each path to block the flow of causal association along that path. The variables `age` and `sex` are part of all four backdoor paths, which means adding these variables to the linear model will block all four paths. The *adjustment set* we need for this scenario is {`age`, `sex`}.

Fit the adjusted model

The remaining steps are straightforward. We fit the adjusted model based on the formula `fev ~ 1 + C(smoke) + C(sex) + age`, which includes the variables `sex` and `age` as controls.

code 4.5.28

```
>>> formula_adj = "fev ~ 1 + C(smoke) + C(sex) + age"
>>> lmfev_adj = smf.ols(formula_adj, data=smokefev).fit()
>>> lmfev_adj.params
Intercept         0.237771
C(smoke)[T.SM]   -0.153974
C(sex)[T.M]       0.315273
```

```
age                0.226794
```

The parameter estimate $\widehat{\beta}_{\texttt{smoke}} = -0.154$ we obtain from the adjusted model tells us that smoking is associated with a 0.154 decrease in forced expiratory volume, on average.

Interpreting the results

To better understand the real-world effect of smoking, let's compare two 15 year old females: one who smokes and the other who doesn't. We can calculate the expected mean FEV for these two individuals, and calculate the relative reduction in FEV.

code 4.5.29

```
>>> nonsmoker15F = {"age":15, "sex":"F", "smoke":"NS"}
>>> fevNS = lmfev_adj.predict(nonsmoker15F)[0]
>>> fevNS
3.6396839416862896
>>> smoker15F = {"age":15, "sex":"F", "smoke":"SM"}
>>> fevSM = lmfev_adj.predict(smoker15F)[0]
>>> fevSM
3.485709826691059
>>> (fevNS - fevSM) / fevNS
0.04230425428750093
```

A 4.2% reduction in FEV is practically significant, and definitely concerning finding.

Limitations In this study, we controlled for two confounders `age` and `sex`, which closed the four backdoor paths that we identified in the causal graph. It's important to keep an open mind about the limitation of our findings: there are many other possible confounding variables could influence our results, which we haven't measured and therefore haven't controlled for. Some examples include lifestyle, socioeconomic background, and nutrition, which are factors that could influence `smoke` and `fev` variables.

4.5.8 Discussion

The examples and adjustment strategies we discussed in this section only scratched the surface of the topic of causal inference from observational data. There are a lot of other interesting topics and techniques we could discuss, but we don't have the room for that, so instead I'll close with a collection of brief mentions of notable ideas, and general conclusions.

4.6 Generalized linear models

All the linear models we discussed so far in this chapter assumed the error term was normally distributed $\mathcal{E} \sim \mathcal{N}(0, \sigma)$, which is a reasonable assumption when analyzing continuous outcome variables, but not appropriate for discrete outcome variables.

It turns out we can use the structure of a linear regression model (linear combination of predictors) to build models for other types of outcome variables. In this section, we'll describe two such models: *logistic regression* for binary data, and *Poisson regression* for count data, which are both instances of the *generalized linear model*.

4.6.1 Definitions

Generalized linear models

A generalized linear model (GLM) has a linear combination of predictors at its core, $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$, but uses a different probability distribution family (denoted $\mathcal{M}$) to model the outcome variable. Let's look at the standard linear regression and the GLM equations side by side to highlight the parallels between the two types of model:

Linear regression	Generalized linear model
$Y \vert x_1, \ldots, x_p \sim \mathcal{N}(\mu, \sigma)$, where	$Y \vert x_1, \ldots, x_p \sim \mathcal{M}(\mu)$, where
$\mu = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$.	$\mu = g^{-1}(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)$.

Both models describe the distribution of the outcome variable Y conditional on the predictors $x_1, x_2, \ldots, x_p$. Specifically, we assume the predictors influence the mean parameter μ of the distribution. In linear regression, the outcome variable is normally distributed, whereas in generalized linear models we use another distribution family $\mathcal{M}$, which is what enables GLMs to model different types of outcome variables. The *inverse link function* g^{-1} is an "adapter" that convert between the space of all possible linear combinations of predictors and the parameter space of the probability distribution family $\mathcal{M}$ (more on this shortly).

It can be helpful to think about the generalized linear model as a "template" that you fill with particular choices of the distribution family $\mathcal{M}$ and the inverse link function g^{-1} to obtain different models. In this section, we'll study the following two instances of the GLM template:

- Logistic regression for binary data, which is based on the Bernoulli distribution ($\mathcal{M}$ = Bernoulli) and the logistic inverse link function $g^{-1}(x) = \texttt{expit}(x) = \frac{1}{1+e^{-x}}$.

- Poisson regression for count data, which is based on the Poisson distribution ($\mathcal{M} = \text{Pois}$) and the exponential inverse link function $g^{-1}(x) = \exp(x) = e^x$.

You're already familiar with the Bernoulli distribution from Section 2.3.2 and the Poisson distribution from Section 2.3.5. We've also been dealing with linear combinations of predictors $\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ this whole chapter, so that's nothing new as well. The only new math concepts in this section are link functions and their inverses, which are the "adapters" we use to convert real numbers to the specific probability representation required for different probability distribution families.

Probability representations and link functions

Probabilities, odds, and log-odds The *odds* of an event that has probability of success p is defined as:

$$\text{odds} = \frac{p}{1-p}.$$

Odds are numbers that vary between 0 and ∞. For example, if $p = 0.5$ the odds are $\frac{0.5}{1-0.5} = \frac{0.5}{0.5} = 1$. When $p = 0.9$, the odds are $\frac{0.9}{1-0.9} = \frac{0.9}{0.1} = 9$. In words, the odds 9 tells us that the probability of success is nine times higher than the probability of failure.

When the odds are less than 1, it tells us the probability of failure is higher than the probability of success. For example, $p = 0.2$ corresponds to odds of $\frac{0.2}{1-0.2} = \frac{0.2}{0.8} = \frac{1}{4}$. In words, the probability of failure is four times higher than the probability of success.

The *log-odds* of an event is the logarithm of its odds:

$$\text{log-odds} = \log\left(\frac{p}{1-p}\right).$$

The *log-odds* of an event is a number between $-\infty$ and ∞. Here are some example calculations of log-odds.

code 4.6.1

```
>>> np.log(0.5 / (1-0.5))
0.0
>>> np.log(0.9 / (1-0.9))
2.1972245773362196
>>> np.log(0.2 / (1-0.2))
-1.3862943611198906
```

The log-odds scale is a useful representation for probabilities because it varies from $-\infty$ for very small probabilities to ∞ for probabilities near 1. The log-odds of zero correspond to odds $= 1$ and $p = 0.5$. Log-odds are symmetric around zero: the log-odds of success is the negative of the log-odds of the failure, since $\log\left(\frac{p}{1-p}\right) = -\log\left(\frac{1-p}{p}\right)$.

The key thing to remember is that probabilities $p \in [0,1]$, odds $\in (0,\infty)$, and log-odds $\in (-\infty,\infty)$ are all equivalent ways of describing the probability of an event, and you can easily convert between these representations. See the exercises E4.26 and E4.27 for more on this.

The logit function and its inverse We define the *logit function* as the math transformation that converts probabilities to log-odds:

$$\texttt{logit}(p) \stackrel{\text{def}}{=} \log\left(\frac{p}{1-p}\right).$$

Here is the Python implementation of this function.

code 4.6.2
```
>>> def logit(p):
        x = np.log(p / (1-p))
        return x
```

Let's look at some example calculations. When $p = 0.5$, we have odds $= 1$, and the log-odds are $\texttt{logit}(0.5) = \log(1) = 0$. When $p = 0.9$, the log-odds are $\texttt{logit}(0.9) = \log(\frac{0.9}{1-0.9}) = \log(9) \approx 2.197$, and $\texttt{logit}(0.2) = \log(\frac{0.2}{1-0.2}) = \log(\frac{1}{4}) = -\log(4) \approx -1.386$.

code 4.6.3
```
>>> logit(0.5),    logit(0.9),      logit(0.2)
(   0.0,           2.197224577,     -1.38629436)
```

The function `logit` takes numbers in the interval $(0,1)$ as inputs and produces numbers in the interval $(-\infty,\infty)$ as outputs (real numbers). The input $p = 0$ is mapped to $-\infty$, while $p = 1$ is mapped to $+\infty$. See Figure 4.51 (b) for the function's graph.

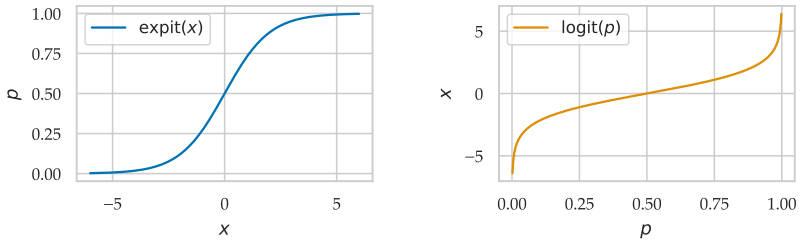


Figure 4.51: Graphs of the *logistic* function `expit`, which is a map from $\mathbb{R}$ to the interval $(0,1)$, and its inverse, the *logit function*, which maps $(0,1)$ to $\mathbb{R}$.

The *logistic* function is the inverse of the `logit` function, and it is defined as follows:

$$\texttt{expit}(x) \stackrel{\text{def}}{=} \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}.$$

Let's define the Python function `expit` and look at some example calculations of converting log-odds to probabilities.

code 4.6.4

```
>>> def expit(x):
        p = 1 / (1 + np.exp(-x))
        return p
>>> expit(0),  expit(2),   expit(-2)
(   0.5,       0.880797,   0.1192029)
```

We choose the notation $\texttt{expit}(x)$ for the logistic function instead of the more common $\sigma(x)$, because we're already using the Greek letter *sigma* to denote standard deviations. The `expit` function takes real numbers as inputs and produces outputs in the interval $(0, 1)$.

The inverse relationship between the `logit` function and the `expit` function means $\texttt{expit}(\texttt{logit}(p)) = p$ for any $p \in (0, 1)$, as well as $\texttt{logit}(\texttt{expit}(x)) = x$, for any $x \in \mathbb{R}$. For example, $\texttt{expit}(\texttt{logit}(0.5)) = 0.5$ and $\texttt{logit}(\texttt{expit}(3)) = 3$.

Link functions for logistic and Poisson regression

Each generalized linear model (GLM) is associated with a particular *link function* g. The *link function* g and the *inverse link function* g^{-1} are "adapters" that convert between the space of all possible linear combinations of predictors, and the parameter space for the mean of the probability distribution family $\mathcal{M}$. The linear combination $\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ is usually a real number ranging between $-\infty$ and ∞, but the parameter space for the means of the Bernoulli and Poisson distributions are restricted: the mean parameter p of the Bernoulli distribution must be in the interval $[0, 1]$, while the mean parameter λ for the Poisson distribution must be in the interval $(0, \infty)$. Hence, if we want to build Bernoulli or Poisson models that include linear combinations of predictors, we need to use adapter functions that map real numbers to the parameter spaces $[0, 1]$ and $(0, \infty)$, as we'll describe next.

Logistic regression The logistic function $\texttt{expit}(x)$ has precisely the properties we need for the inverse link function g^{-1} for the logistic regression model. We obtain the mean parameter $\mu_Y = p$ of the Bernoulli distribution by applying the logistic function to the linear combination of predictors:

$$\mu_Y(x) = p(x) = \texttt{expit}(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p).$$

Since the outputs of $\texttt{expit}(x)$ are always bounded between 0 and 1, they are valid parameters for the mean of the Bernoulli distribution.

Poisson regression The inverse link function for Poisson regression is the exponential function $\exp(x) = e^x$, which maps between

real numbers $\mathbb{R}$ and the positive real numbers $(0, \infty)$. The mean parameter for Poisson regression is given by the equation

$$\mu_Y(x) = \lambda(x) = \exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p).$$

The outputs of the exponential function are always positive numbers, which is what we need for the λ parameter of the Poisson distribution.

Summary table Table 4.1 summarizes the probability distribution families and link functions for the generalized linear models. The names of the `statsmodels` functions we use to fit different models are also shown.

	Linear	Logistic	Poisson
outcome Y	numerical	binary data	count data
model family $\mathcal{M}$	$\mathcal{N}$	Bernoulli	Pois
mean parameter μ	μ	p	λ
link function g	`id`	`logit`	log
link inverse g^{-1}	`id`	`expit`	exp
`smf` function	`smf.ols`	`smf.logit`	`smf.poisson`

Table 4.1: Summary of the components of three GLM models: linear regression, logistic regression, and Poisson regression.

The linear model $Y|x_1, \ldots, x_p \sim \mathcal{N}(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p, \sigma)$ is also shown for comparison. The mean parameter for linear regression is $\mu_Y(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$, which means the link function g is the identity function $\texttt{id}(x) = x$. Basically, we don't need an adapter because the linear combination $\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ can be used directly as the mean parameter of the normal distribution $\mathcal{N}$.

I hope looking at Table 4.1 will help you to see the parallels between linear regression, logistic regression, and Poisson regression models. Clearly these three models are all instances of the common GLM template $Y|x_1, \ldots, x_p \sim \mathcal{M}(g^{-1}(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p))$.

4.6.2 Logistic regression

Consider the dataset $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p, \mathbf{y}]$, where $\mathbf{y}$ is a binary outcome variable (1 or 0, success or failure, pass or fail, etc.). The logistic regression model predicts the probability of the occurrence of an event, then feeds this probability into the Bernoulli distribution to produce a binary outcome:

$$Y \sim \text{Bernoulli}(p), \quad \text{where} \quad p = \texttt{expit}(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p).$$

Chapter 5

Bayesian statistics

In this chapter, we'll learn about statistical inference based on the Bayesian interpretation of probability theory, which uses probability distributions as representations of the uncertainty in our state of knowledge. Instead of treating population parameters θ as fixed unknown quantities, in Bayesian statistics we model our knowledge about the parameters θ as probability distributions. We refer to our state of knowledge before observing the data sample $\mathbf{x}$ as the *prior* distribution, and our state of knowledge after observing the data $\mathbf{x}$ as the *posterior* distribution. This branch of statistics is called "Bayesian" because Bayes' rule is central to the procedure we use to update our state of knowledge about the parameters θ.

This chapter is a crash course on Bayesian statistics that leverages your expertise from previous chapters. We'll revisit statistical inference tasks like parameter estimation, uncertainty quantification, and hypothesis testing using a Bayesian approach. We'll start by presenting the main ideas of Bayesian statistics in Section 5.1. In Section 5.2, we'll introduce the Python library Bambi, which will allow us to build more advanced Bayesian models, such as Bayesian linear models (Section 5.3), Bayesian models for comparing two groups (Section 5.4), and Bayesian hierarchical models (Section 5.5).

In this chapter, I'm going to show you how to ...

- **model** uncertainty about parameters using probability distributions
- **interpret** graphical model diagrams for Bayesian models
- **use** Bayes' rule to update the probability distribution of a parameter
- **fit** Bayesian models using the grid approximation
- **fit** Bayesian models using the Bambi library for Bayesian inference
- **visualize** posterior distributions using ArviZ helper functions
- **summarize** posterior distributions with point and interval estimates
- **fit** Bayesian linear models using Bambi
- **fit** Bayesian models for comparing two groups using Bambi
- **fit** Bayesian hierarchical models using Bambi

5.1 Introduction to Bayesian statistics

Suppose we have a sample of n observations $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ from an unknown population. We can model the population as a random variable X. The goal of all statistical inference, Bayesian or other, is to learn about the unknown population X based on the sample $\mathbf{x}$. We can use different types of models to describe the population X. Each model tells a "story" about how the data was generated.

In previous chapters, we used probability models of the form $X \sim \mathcal{M}(\theta)$, where $\mathcal{M}$ is the model family, and θ are the unknown model parameters. The lowercase θ indicates that we're looking for some *fixed* unknown value of the parameters θ.

In this chapter, we'll learn about Bayesian models of the form $X \sim \mathcal{M}(\Theta)$, where we consider the parameters to be an unknown random variable Θ. In a Bayesian analysis, we start with some prior knowledge about the parameters (the *prior* distribution), then use Bayesian inference to *update* our knowledge about the parameters to the *posterior* distribution, which combines the information from both the prior distribution and the observed data $\mathbf{x}$.

In this section, we'll describe the Bayesian update procedure using visual explanations, math formulas, and hands-on code calculations. The triply redundant explanations are provided to maximize the understandability of the Bayesian calculations. I want to make sure you get the basic idea, because once you learn how the Bayesian update procedure works, the rest of the chapter will be smooth sailing! Are you ready for this? Let's go!

5.1.1 Definitions

Let's start by introducing the Bayesian statistics terminology that we'll use in this section and in the rest of the chapter. The good news is that you're already familiar with the main building blocks: random variables and their probability distributions. The bad news is there will be lots of new notation to get used to, which is required to distinguish between the various probability distributions we're dealing with. Pay close attention to the various subscripts, and remember the convention of using capital letters for random variables, and lowercase letters for fixed quantities.

Data model and likelihood function

Given fixed values of the parameters θ, we can describe the variability in the population as a random variable X with probability distribution $f_{X|\theta}$. We refer to $f_{X|\theta}$ as the *data model* for the population.

The data model tells us the probability of observing the event $\{X = x\}$ under the model with parameters θ:

$$\Pr(X = x|\theta) = f_{X|\theta}(x|\theta).$$

The notation "$|\theta$" (read "given theta") makes it clear that the distribution of X depends on the parameters θ. We previously used the notation f_X to describe population data models, but the notation $f_{X|\theta}$ is better, because it shows explicitly that the probability model depends on the unknown population parameters θ.

Next, we define two key concepts based on the data model $f_{X|\theta}$:

- $f_{\mathbf{X}|\theta}$: the probability distribution for an independent, identically distributed (i.i.d.) random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ from the population X. The i.i.d. assumption tells us that $f_{\mathbf{X}|\theta}$ is the n-fold product of $f_{X|\theta}$s:

$$f_{\mathbf{X}|\theta}(\mathbf{x}|\theta) = f_{X|\theta}(x_1|\theta) f_{X|\theta}(x_2|\theta) \cdots f_{X|\theta}(x_n|\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta).$$

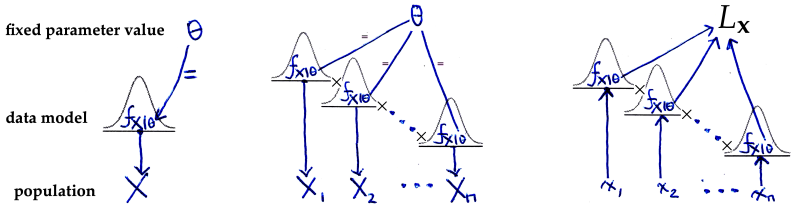
- $L_{\mathbf{x}}$: the *likelihood function* (or simply *likelihood*) corresponds to the same expression as the probability distribution $f_{\mathbf{X}|\theta}$, but viewed as a function of θ with $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ fixed:

$$L_{\mathbf{x}}(\theta) = f_{X|\theta}(x_1|\theta) f_{X|\theta}(x_2|\theta) \cdots f_{X|\theta}(x_n|\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta).$$

  The subscript $\mathbf{x}$ indicates that the likelihood function $L_{\mathbf{x}}$ depends on the observed data sample $\mathbf{x}$. The likelihood function $L_{\mathbf{x}}$ tells us the "plausibility" of different values of θ based on the observed data $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and the data model $f_{X|\theta}$.

Figure 5.1 shows a visual representation of the data model and the two quantities defined above. Part (a) of the figure shows the data model $f_{X|\theta} = \mathcal{M}(\theta)$ is a distribution (pictured as some Gaussian-like curve), whose shape depends on the fixed parameter θ. Part (b) shows that we obtain the probability model for a random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ by multiplying together n-copies of the data model $f_{X|\theta}$. Part (c) shows the "backward" use of the probability model $f_{\mathbf{X}|\theta}$ to calculate the likelihood $L_{\mathbf{x}}$ for different values of the parameters θ, based on the observations in a particular data sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

I know it's weird to define two concepts that correspond to the same mathematical expression, but you have to trust me that it is useful to have different names and notation depending on whether



(a) Data model. (b) Probability of the sample **X**. (c) Likelihood function.

Figure 5.1: Illustrations of calculations based on the data model for a given value of θ. (a) shows an individual observation $X \sim f_{X|\theta}$. (b) shows the probability of a random sample of size n. (c) shows the evaluation of the likelihood function $L_{\mathbf{x}}$ given the observed data $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

θ is fixed and **X** variable, or Θ is variable and **x** is fixed. It might be helpful to think of the probability distribution $f_{\mathbf{X}|\theta}$ and the likelihood function $L_{\mathbf{x}}$ as different use cases of the same math expression. The arrows in parts (b) and (c) of Figure 5.1 show the "information flows" for the two use cases:

- We use the function $f_{\mathbf{X}|\theta}$ when θ is known, and we want to evaluate the probability of different possible realizations of the random sample **X**. The domain of $f_{\mathbf{X}|\theta}$ is the set of all possible samples of size n, which we can denote $\mathcal{X}^n$, where $\mathcal{X}$ is the sample space for the random variable X.
- We use the likelihood function $L_{\mathbf{x}}$ when we have a given observed sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and we ask **what value of θ could have produced this sample**? The domain of the function $L_{\mathbf{x}}$ is the θ parameter space (all possible values of θ).

The likelihood function $L_{\mathbf{x}}$ is a useful concept for any statistical analysis of the data **x** based on the model $f_{X|\theta} = \mathcal{M}(\theta)$. If some parameter value θ has a very low probability of producing data like **x**, then the likelihood $L_{\mathbf{x}}(\theta)$ will be a small number. In contrast, the likelihood function is high for parameters θ that have a high probability of producing data like **x**. You can think of the likelihood function as representing the "plausibility" we attribute to the different values of θ, after observing the data $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

The integral of the likelihood over all possible values of θ doesn't have to equal one, $\int_\theta L_{\mathbf{x}}(\theta)\, d\theta \neq 1$. The values of the likelihood function are usually small positive numbers. We often don't care about the specific values of the likelihood function, but use the *relative likelihoods* of different parameters.

The data model makes up half of every Bayesian model. The other half is a distribution over the model parameters, which is what we'll

define next.

Parameter distributions

In Bayesian statistics, we represent the model parameters as a random variable Θ. There are two different distributions for Θ, representing our state of knowledge *before* and *after* observing the data sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$:

- f_Θ: the *prior* distribution represents our knowledge about the parameters Θ **before observing the data x**.
- $f_{\Theta|\mathbf{x}}$: the *posterior* distribution represents our knowledge about the parameters Θ **after observing the data x**.

The prior and posterior distributions are the main new concepts in Bayesian statistics. We'll spend the rest of this chapter getting to know them really well. Roughly speaking, you can think of the prior distribution as the input to the Bayesian analysis, and the posterior as the output of the analysis, from which we calculate all the results.

Bayesian model

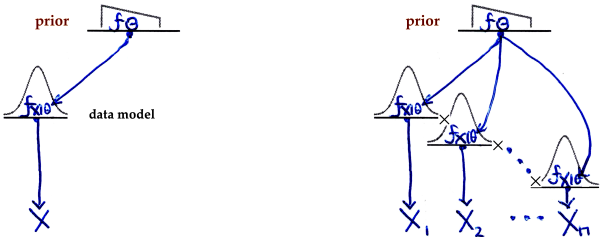
A Bayesian model consists of two parts: the data model $f_{X|\theta}$ for a fixed choice of parameters θ and the prior distribution of the parameters f_Θ. We multiply these two distributions to obtain the *joint distribution* of the two variables $(X, \Theta) \sim f_{X,\Theta} = f_{X|\theta} f_\Theta$. The joint probability of observing a particular pair (x, θ) is given by:

$$\Pr(X = x, \Theta = \theta) \;=\; f_{X,\Theta}(x, \theta) \;=\; f_{X|\theta}(x|\theta) f_\Theta(\theta).$$

The joint distribution describes the *data generating process*, which is the complete story for how observations from the population X are generated. Figure 5.2 (a) provides a visual representation of the Bayesian data generating process. Graphical model diagrams like this one allow us to see the structure of the Bayesian model. Reading the diagram from top to bottom, we see that the parameters θ come from the prior distribution f_Θ, then we plug the value θ into the data model distribution $f_{X|\theta}$ to draw one observation X.

Figure 5.2 (b) illustrates the data generating process for a random sample of n observations, $\mathbf{X} = (X_1, X_2, \ldots, X_n)$. The joint distribution of $\mathbf{X}$ and Θ is the product of the probability distribution for the random sample $f_{\mathbf{X}|\theta}$ and the prior f_Θ, which gives us $(\mathbf{X}, \Theta) \sim f_{\mathbf{X},\Theta} = f_{\mathbf{X}|\theta} f_\Theta$. The joint probability of observing a particular pair $(\mathbf{x}, \theta)$ is given by:

$$\Pr(\mathbf{X} = \mathbf{x}, \Theta = \theta) \;=\; f_{\mathbf{X},\Theta}(\mathbf{x}, \theta) \;=\; f_{\mathbf{X}|\theta}(\mathbf{x}|\theta) f_\Theta(\theta).$$



(a) Bayesian model for X. (b) Bayesian model for $\mathbf{X}$.

Figure 5.2: The Bayesian model for the population X combines the prior distribution f_Θ and the data model $f_{X|\theta}$. The Bayesian model for the random sample $\mathbf{X}$ of size n consists of n copies of the model for one observation X.

Recall that $f_{\mathbf{X}|\theta}(\mathbf{x}|\theta)$ is defined as $\prod_{i=1}^{n} f_{X|\theta}(x_i|\theta)$. Note the same θ is used to generate all the observations in the sample.

The joint probability distribution $f_{\mathbf{X},\Theta}$ describes the Bayesian model for all possible samples $\mathbf{X}$ and all possible values of the parameters Θ. When doing a Bayesian analysis of a particular data sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, we're interested in the particular "slice" through the joint distribution $f_{\mathbf{X},\Theta}$ that corresponds to $\{\mathbf{X} = \mathbf{x}\}$. We want to know the *conditional distribution* of Θ given we have observed the event $\{\mathbf{X} = \mathbf{x}\}$, which is called the *posterior* distribution:

$$f_{\Theta|\mathbf{x}}(\theta|\mathbf{x}) = \Pr(\Theta = \theta|\mathbf{X} = \mathbf{x}) = \frac{\Pr(\mathbf{X} = \mathbf{x}, \Theta = \theta)}{\Pr(\mathbf{X} = \mathbf{x})}.$$

You can think of $f_{\Theta|\mathbf{x}}$ as a shorthand notation for $f_{\Theta|\mathbf{X}=\mathbf{x}}$. Calculating the posterior distribution is the main task in Bayesian inference.

Calculating the posterior distribution

The posterior distribution $f_{\Theta|\mathbf{x}}$ is the output of the Bayesian update procedure, and describes our knowledge about the parameters Θ after we have observed the data sample $\mathbf{x}$. Bayes' rule from probability theory gives us the following formula for computing the posterior:

$$\underbrace{f_{\Theta|\mathbf{x}}(\theta|\mathbf{x})}_{\text{posterior}} = \frac{1}{C} \underbrace{L_{\mathbf{x}}(\theta)}_{\text{likelihood}} \times \underbrace{f_\Theta(\theta)}_{\text{prior}},$$

where C is a normalization constant required to make $f_{\Theta|\mathbf{x}}$ a valid probability distribution. In words, this formula tells us that we can compute the posterior distribution by multiplying together the likelihood function and the prior distribution. We'll spend the next hundred pages or so applying this formula to various statistical analyses, so you'll have plenty of time to get used to it. This

is, essentially, the *only* formula you need to know to do Bayesian statistics.

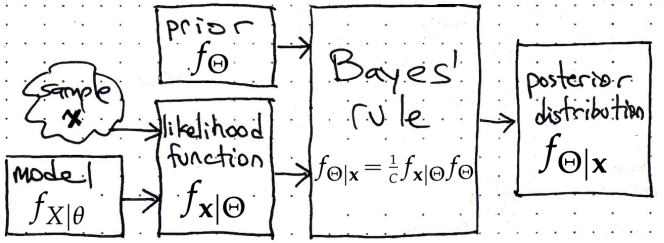


Figure 5.3: Diagram showing the procedure for calculating the posterior.

Figure 5.3 shows how to compute the posterior distribution $f_{\Theta|\mathbf{x}}$. The inputs to the procedure are the prior distribution f_Θ, the observed data sample $\mathbf{x}$, and the data model $f_{X|\theta}$. The likelihood function $L_\mathbf{x}(\theta)$ is the product of the probabilities of individual observations: $L_\mathbf{x}(\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta)$. We then multiply the likelihood function and the prior to compute the posterior: $f_{\Theta|\mathbf{x}} = \frac{1}{C} L_\mathbf{x} f_\Theta$.

The posterior distribution is the "main actor" in Bayesian statistics. Most Bayesian analysis results are calculated from the posterior distribution $f_{\Theta|\mathbf{x}}$. Indeed, it wouldn't be wrong to refer to Bayesian statistics as *posterior distribution statistics*, since the main inference task is to calculate the posterior distribution $f_{\Theta|\mathbf{x}}$, and all the remaining tasks involve visualizing and summarizing the posterior.

Bayesian inference results

The posterior $f_{\Theta|\mathbf{x}}$ describes our knowledge about the parameters Θ after we have observed the data $\mathbf{x}$. Figure 5.4 shows an example of a probability density plot of a posterior distribution, with additional markers for the summary statistics computed from the posterior.

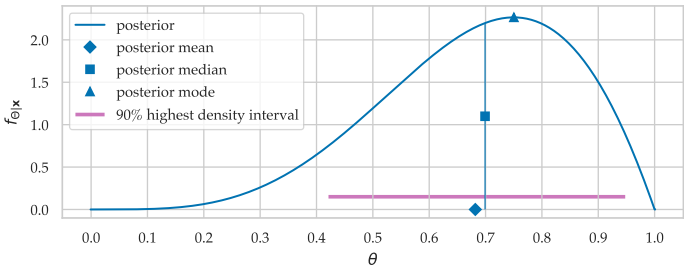


Figure 5.4: Plot of the posterior distribution with extra annotations for its mean $\mu_{\Theta|\mathbf{x}}$, median $\mathbf{med}_{\Theta|\mathbf{x}}$, mode $\widehat{\theta}_{\text{MAP}}$, and a 90% credible interval $\mathbf{hdi}_{\theta,0.9}$.

Bayesian point estimates We can summarize the properties of the posterior distribution using the descriptive statistics concepts that we learned in Section 1.3. The following are common numerical summaries (point estimates) for posterior distributions:

- **Posterior mean** $\mu_{\Theta|\mathbf{x}}$. The posterior mean is the centre of mass of the posterior distribution $f_{\Theta|\mathbf{x}}$.
- **Posterior median** $\mathbf{med}_{\Theta|\mathbf{x}}$. The median splits the probability mass of the posterior distribution into two equal parts.
- **Posterior mode** $\widehat{\theta}_{\text{MAP}}$. The mode of the posterior is the most credible parameter value. Another name for the mode is the *maximum a posteriori* (MAP) estimate. It is defined as the maximum of the posterior distribution: $\widehat{\theta}_{\text{MAP}} \stackrel{\text{def}}{=} \text{argmax}_\theta \, f_{\Theta|\mathbf{x}}(\theta|\mathbf{x})$.

Point summaries like the mean, the median, and the mode of the posterior don't tell us about the uncertainty about the parameters. For this, we turn to interval estimates.

Bayesian credible intervals A Bayesian *credible interval* is a region of the parameter space that contains a certain proportion of the posterior distribution $f_{\Theta|\mathbf{x}}$. We can construct a Bayesian credible interval by looking for a subset of the parameter space where the posterior distribution $f_{\Theta|\mathbf{x}}$ has the highest density, which is called a *highest density interval* (HDI). For example, a 90% Bayesian credible interval for the parameter θ is denoted $\mathbf{hdi}_{\theta,0.9} = [\mathbf{l}_\theta, \mathbf{u}_\theta]$, and contains 90% of the probability mass of the posterior: $\int_{\mathbf{l}_\theta}^{\mathbf{u}_\theta} f_{\Theta|\mathbf{x}}(\theta|\mathbf{x})d\theta = 0.9$. The horizontal line in Figure 5.4 corresponds to the highest density interval $\mathbf{hdi}_{\theta,0.9}$. The width of this credible interval is a measure of the uncertainty in our estimates about the parameter θ.

Bayesian predictions We can use the posterior distribution to make predictions about future observations from the model. The *posterior predictive distribution*, denoted $f_{\tilde{X}|\mathbf{x}}$, describes the observations $\tilde{X}$ that we expect to see in the future, based on our knowledge of the posterior distribution $f_{\Theta|\mathbf{x}}$. We'll discuss Bayesian predictions in Section 5.1.6.

Bayesian hypothesis testing We can also use the Bayesian inference machinery to test hypotheses. Bayesian hypothesis testing is fundamentally different from classical null hypothesis significance testing (NHST), since it allows us to model the probabilities of different hypotheses directly. We compare competing hypotheses using their likelihoods and posterior distributions. We'll describe some approaches to Bayesian hypothesis testing in Section 5.1.7.

Contrast with frequentist results We'll use the term *frequentist statistics* to describe the classical statistical analysis techniques that we studied in Chapter 3. The name *frequentist* comes from the type of "quality guarantees" attached to inference results. Frequentist results come with a guarantee about the **performance over repeated uses of the procedure**: if we were to use the confidence interval or hypothesis testing procedure repeatedly for a bunch of samples $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, we know the count (frequency) of correct results will be roughly $(1-\alpha)N$, where α is a design parameter of the procedure that we can choose in advance.

For example, let's compare a Bayesian credible interval $\mathbf{hdi}_{\theta,0.9}$ constructed from the posterior $f_{\Theta|\mathbf{x}}$, and a frequentist confidence interval $\mathbf{ci}_{\theta,0.9}$ constructed using the technique we studied in Section 3.2. The frequentist result comes with a quality guarantee for the long-term average **reliability of the procedure** that we used to construct the confidence interval, but gives no guarantee about the specific confidence interval $\mathbf{ci}_{\theta,0.9}$ computed from the particular sample $\mathbf{x}$ we are currently analyzing. In contrast, the Bayesian highest density interval $\mathbf{hdi}_{\theta,0.9}$ has a direct interpretation as the probability that this particular interval will contain the true unknown parameter, $\Pr(\Theta \in \mathbf{hdi}_{\theta,0.9}) = 0.9$.

Frequentist methods don't provide any guarantee for the results of any specific dataset $\mathbf{x}$. They only guarantee that the procedure works with probability $(1-\alpha)$ *on average*, for random samples $\mathbf{X}$. We *hope* that the specific $\mathbf{x}$ in our analysis is one of the ones where the procedure works, but we have no guarantee. With probability α, we could get one of the samples $\mathbf{x}$ where the procedure doesn't work. I know this is a weird kind of guarantee, but it makes sense if you believe **your job as a statistician is to specify a procedure in advance of seeing the data**, which was the dominant paradigm for statistics during the last 100 years. Frequentist guarantees are the best we can get if we insist on creating data-independent procedures (recipes), that researchers can use on many datasets.

In contrast, Bayesian statistics is all about the data sample $\mathbf{x}$. Bayesian inference is a data-dependent procedure that focuses on what we can say about Θ after we have observed the data $\mathbf{x}$, assuming our prior knowledge about Θ is described by the prior f_Θ. The guarantees we get are direct probability statements derived from the information we learned from the data sample $\mathbf{x}$ and the prior f_Θ, based on the Bayesian model we used for the analysis.

to combine our prior knowledge f_Θ and the likelihood $L_\mathbf{x}$ to obtain the posterior distribution $f_{\Theta|\mathbf{x}}$:

$$\overbrace{f_{\Theta|\mathbf{x}}(\theta|\mathbf{x})}^{\text{posterior}} = \frac{\overbrace{L_\mathbf{x}(\theta)}^{\text{likelihood}} \cdot \overbrace{f_\Theta(\theta)}^{\text{prior}}}{\underbrace{f_\mathbf{X}(\mathbf{x})}_{C}} = \frac{1}{C} \cdot \overbrace{L_\mathbf{x}(\theta)}^{\text{likelihood}} \cdot \overbrace{f_\Theta(\theta)}^{\text{prior}} \quad \text{for all } \theta.$$

The denominator in the formula, $f_\mathbf{X}(\mathbf{x})$, doesn't depend on θ, so we can replace it with a constant C. Replacing $f_\mathbf{X}(\mathbf{x})$ with the constant C allows us to focus our attention on the important parts of the equation: the multiplication of the likelihood and the prior.

Similar to the combined Bayes' rule formula for $f_{A|B}$ in the example from the previous section, Bayes' rule of statistical inference is a formula that applies for all values of the parameters θ. In other words, Bayes' rule describes a relationship between three functions $f_{\Theta|\mathbf{x}}$, $L_\mathbf{x}$, and f_Θ, which are all functions of θ. We can express Bayes' rule compactly as a product of functions $f_{\Theta|\mathbf{x}} = \frac{1}{C} L_\mathbf{x} f_\Theta$, keeping in mind this equation applies for all θ. Figure 5.5 illustrates the complete story of the Bayesian update procedure.

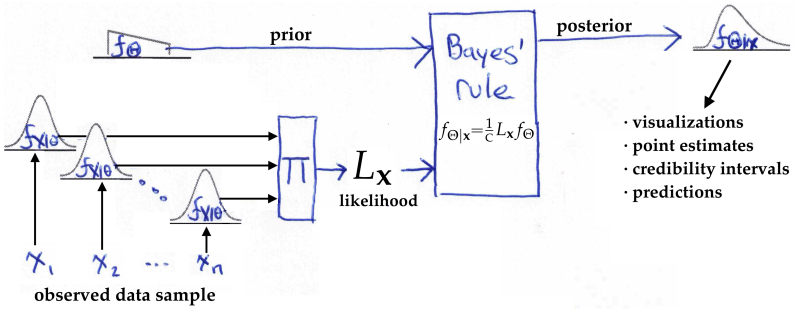


Figure 5.5: The calculations used to obtain the posterior distribution $f_{\Theta|\mathbf{x}}$ using Bayesian inference. The two inputs that go into the Bayes' rule formula are the prior distribution f_Θ, and the likelihood function $L_\mathbf{x}$, which is the product of the probabilities of the observations, $L_\mathbf{x}(\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta)$.

Visualizing the Bayesian update procedure

The plots in Figure 5.6 show the three quantities that appear in the Bayesian update calculation: $f_{\Theta|\mathbf{x}}(\theta|\mathbf{x}) = \frac{1}{C} L_\mathbf{x}(\theta) f_\Theta(\theta)$, for all θ. The unknown parameter θ takes on values between 0 and 1.

Suppose we initially know that θ values close to 0 are slightly more likely than values close to 1. We can express this knowledge as a prior distribution f_Θ that assigns higher density to values close

to $\theta = 0$, and lower density to values close to $\theta = 1$. An example of such a "sloped" prior is shown in the top plot in Figure 5.6.

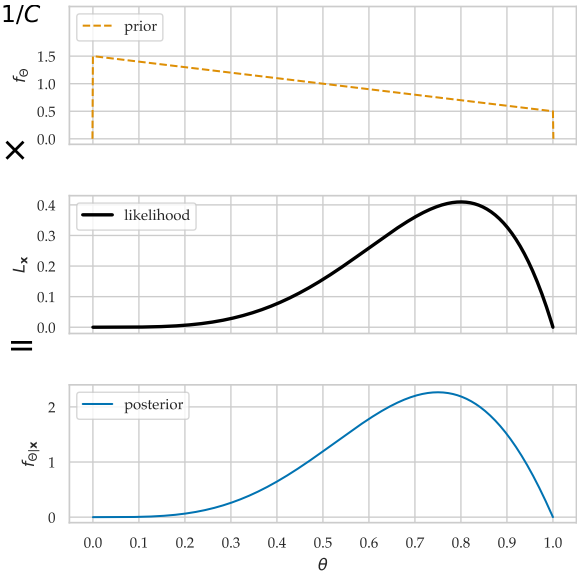


Figure 5.6: The Bayes update formula $f_{\Theta|\mathbf{x}}(\theta|\mathbf{x}) = \frac{1}{C} L_\mathbf{x}(\theta) f_\Theta(\theta)$ applied for all values of θ. The prior f_Θ (top) times the likelihood $L_\mathbf{x}$ (middle) gives the posterior distribution $f_{\Theta|\mathbf{x}}$ (bottom), up to a normalizing constant C.

Given the observed data $\mathbf{x}$, we can calculate the likelihood $L_\mathbf{x}(\theta)$ for each value of θ, which is shown in the middle plot of Figure 5.6. The likelihood is not a probability distribution (the area under the curve is not 1). The likelihood is a function that tells us the "relative plausibility" of the different values of θ.

We obtain the shape of the posterior distribution $f_{\Theta|\mathbf{x}}(\theta|\mathbf{x})$ by multiplying the prior distribution $f_\Theta(\theta)$ and the likelihood $L_\mathbf{x}(\theta)$ at each value of θ. To calculate the actual posterior distribution, we need to normalize the result of this calculation to ensure the integral of the posterior is equal to one. The normalization step corresponds to the factor $1/C$ in the figure. The normalized posterior $f_{\Theta|\mathbf{x}}$ is shown in the bottom plot in Figure 5.6.

Note the curve of the posterior distribution is similar to the likelihood function, but slightly shifted to the left due to the effect of the prior, which favours values close to $\theta = 0$ over values close to $\theta = 1$. The posterior distribution contains a "mix" of two influences: our prior knowledge f_Θ, and the new evidence obtained from the data observations $\mathbf{x}$ via the likelihood $L_\mathbf{x}$. Specifically, we mix these two sources of information by multiplying them together, which

- **Step 5**: Calculate `numerator/sum(numerator)` to normalize the result of Step 4 and obtain the posterior distribution for each point in the grid: $[f_{\Theta|\mathbf{x}}(\theta_0|\mathbf{x}),\ f_{\Theta|\mathbf{x}}(\theta_1|\mathbf{x}),\ \ldots,\ f_{\Theta|\mathbf{x}}(\theta_K|\mathbf{x})]$.

Figure 5.7 illustrates the grid approximation calculations for grid of 26 points that cover the parameter space from 0 to 1. We use stem plots to indicate that we only know the values of the prior, the likelihood, and the posterior for the points of the grid.

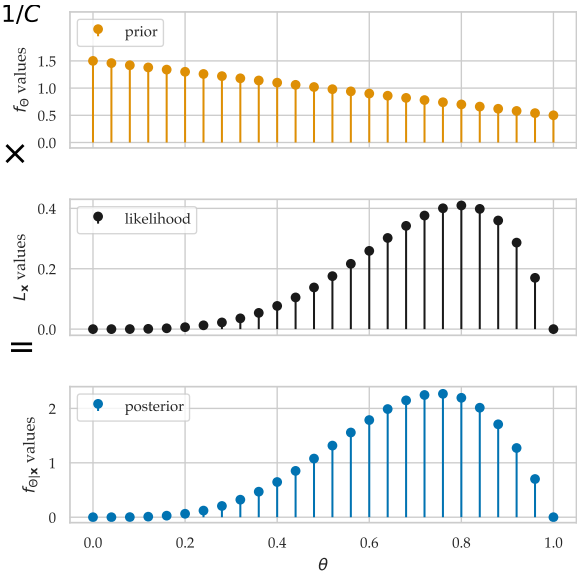


Figure 5.7: We evaluate the prior (top) and the likelihood (middle) over a grid of 26 evenly spaced points. We then multiply the prior times the likelihood and normalize to obtain the posterior distribution (bottom).

The stem plots we see in Figure 5.7 are approximations to the continuous functions we saw earlier in Figure 5.6. The approximation we obtain from a grid of 26 points allows us to see the overall shape of the posterior. We can obtain a more accurate approximation by using a denser grid with hundreds or thousands of points.

The grid approximation method works well for small models with a few parameters, like the examples we'll discuss next.

5.1.3 Example 1: estimating the bias of a coin

We'll now look at a complete worked example of a Bayesian analysis, which includes choosing an appropriate prior, calculating the posterior distribution, and reporting the results of the Bayesian analysis by summarizing the posterior distribution.

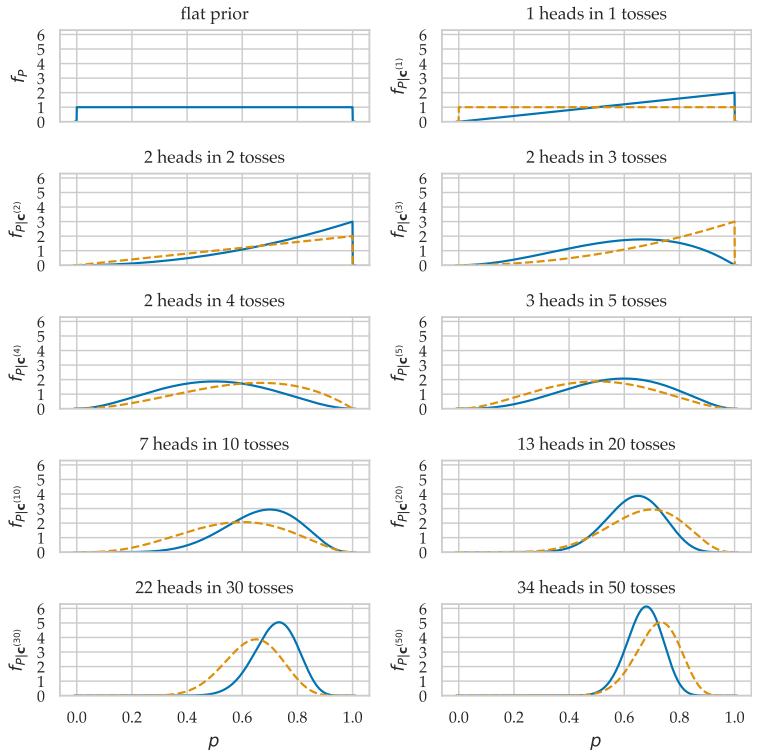


Figure 5.10: Illustrations of our knowledge about the parameter P (solid line) after observing k coin tosses. The first plot shows the prior distribution f_P before making any observations. The other plots show the posterior distributions after observing 1, 2, 3, 4, 5, 10, 20, 30, and 50 tosses. The dashed lines show the prior distribution (the posterior from the previous plot).

$f_{P|\mathbf{c}^{(2)}}$ changes shape to favour values close to $p = 1$ even more strongly. See exercise EXX for the calculation of the exact density function of the distribution $f_{P|\mathbf{c}^{(2)}}$ as the product of the prior and the likelihood. On the third toss, we observe the outcome 0 (`tails`), which changes the shape of the posterior significantly. Since we've observed both heads and tails outcomes, we know the parameter values $p = 0$ and $p = 1$ are impossible. The Bayesian updating procedure automagically produces a probability distribution that slopes down toward zero for the two values that we know are logically impossible: $p = 0$ and $p = 1$. On the fourth toss, we observe 0 (`tails`) again, which means we have observed an equal number of `heads` and `tails` outcomes. The posterior $f_{P|\mathbf{c}^{(4)}}$ attributes highest probability to the values close to $p = 0.5$. The shape of the distribution is very wide, to indicate a high level of uncertainty, since

Prediction intervals are better than point estimates

By using the information from the entire posterior distribution $f_{\Theta|\mathbf{x}}$, the posterior predictive distribution $f_{\tilde{X}|\mathbf{x}}$ takes into account our uncertainty about the parameters θ. This leads to more realistic predictions, as compared to using a point estimate of the parameters.

For example, suppose we decide to use the posterior mean $\mu_{\Theta|\mathbf{x}}$ as a summary for the posterior distribution $f_{\Theta|\mathbf{x}}$. The predictions we obtain from the distribution $f_{X|\theta=\mu_{\Theta|\mathbf{x}}}$ will have unrealistically low variance, because using a point estimate doesn't take into account the uncertainty about the parameters θ in the posterior. If we want realistic predictions, we need the posterior predictive distribution $f_{\tilde{X}|\mathbf{x}}$, which takes into account all possible values under the posterior.

The posterior predictive distribution calculations are illustrative of the general Bayesian approach to calculations based on statistical results. The power of the Bayesian analysis comes when we use the entire posterior distribution for all "downstream" calculations, which takes into account the uncertainty about the parameters.

5.1.7 Bayesian hypothesis testing

We use hypothesis testing procedures to detect "deviations" from a hypothetical model, which we call the *null hypothesis* and denote H_0. The null hypothesis corresponds to the point of view of a skeptical colleague that contradicts whatever claim we're trying to make. The result of a hypothesis test is a forced yes-or-no decision if we have seen enough evidence to reject the null hypothesis, or if the observed data is consistent with the null hypothesis.

For example, in the analysis of the IQ scores dataset, we want to show that the smart drug has an effect. To do so, we must be able to argue against a skeptical colleague who says that the drug has no effect. The distribution of IQ scores in the general population is known to have mean $\mu_0 = 100$. According to the skeptical claim (the null hypothesis), the drug does nothing, so the IQ scores dataset `iqs` comes from a population with mean $\mu = 100 = \mu_0$. A hypothesis testing procedure is a strategy that uses probability calculations to show that the skeptical claim is wrong.

In Chapter 3, we studied various frequentist hypothesis tests based on the *null hypothesis significance testing* (NHST) procedure. We can also use Bayesian statistics for hypothesis testing, however there is no single equivalent to the NHST procedure. Instead, there are several different approaches based on Bayesian models. Below, we'll briefly describe some of these approaches.

Using credible intervals for hypothesis testing

We can use Bayesian credible intervals to make decisions about the plausibility of the point null hypothesis $H_0 : \theta = \theta_0$. The credible interval $\mathbf{hdi}_{\theta,1-\alpha}$ tells us a range of values within which the parameter θ is likely to lie. If the parameter value under the null hypothesis θ_0 falls outside the Bayesian credible interval $\mathbf{hdi}_{\theta,1-\alpha}$, then we can reject the null hypothesis (up to a possible error of α).

For example, suppose we want to make a yes-or-no decision about the effectiveness of the smart drug based on the Bayesian data analysis we did in Example 2. The "no-effects" skeptical colleague claims that the `iqs` data sample could have come from the general population, which has mean $\mu_0 = 100$. We can use the `ministats` helper function `hdi_from_grid` to construct a 95% ($\alpha = 0.05$) highest density interval for the posterior of the unknown mean M.

code 5.1.25

```
>>> from ministats import hdi_from_grid
>>> hdi95 = hdi_from_grid(mus, posterior2, hdi_prob=0.95)
>>> hdi95
[102.32, 113.04]
```

The 95% credible interval $\mathbf{hdi}_{\mu,0.95} = [102.32, 113.04]$ doesn't contain the null value $\mu_0 = 100$, which tells us that the skeptical colleague is at least 95% wrong. In other words, we have shown evidence against the null hypothesis H_0. Using frequentist language, we can say that we can reject the null hypothesis H_0 at the "95% significance level."

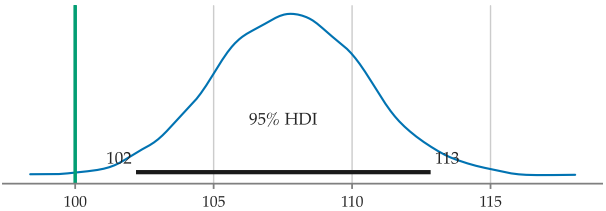


Figure 5.15: Bayesian hypothesis test using the credible interval $\mathbf{hdi}_{\mu,0.95}$.

The is-the-null-value-inside-the-**hdi** approach to hypothesis testing is very similar in nature to what we did in Chapter 3. It also inherits the same one-sided informational weakness as the classical NHST procedure: it is possible to reject the null hypothesis, but we can never "accept" the null hypothesis. When the null value θ_0 falls inside the Bayesian credible interval $\mathbf{hdi}_{\theta,0.95}$, our decision is "fail to reject H_0," which has no informational value. We looked for a pattern different from H_0, and we didn't find evidence that such a pattern exists, but that doesn't mean we've shown that H_0 is true.

Region of practical equivalence

A better way to describe the "no effect" hypothesis is to define a *region of practical equivalence* (ROPE) around the null value, instead of the point null. Hypothesis testing using a ROPE is the Bayesian analogue of the frequentist *equivalence tests*, like the *two one-sided tests* (TOST) procedure that we saw in Section 3.7 (see page 262).

We'll make a decision whether to accept or reject the null hypothesis based on the proportion of the posterior probability that is within the region of practical equivalence (ROPE). The three possible outcomes of the test are illustrated in Figure 5.16. Using the conventional significance level of $\alpha = 0.05$, we use the following rules to make our decision:

- If 95% or more of the posterior is within the ROPE, then we accept H_0.
- If less than 5% of the posterior is within the ROPE, then we reject H_0.
- Otherwise the test is inconclusive.

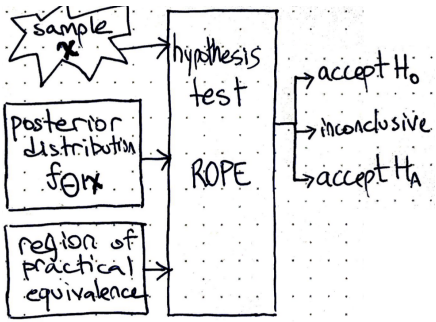


Figure 5.16: The three possible outcomes of a Bayesian hypothesis test based on a region of practical equivalence (ROPE) approach.

For example, in our analysis of the smart drug, we could say that IQ scores within 3 points of the national average $\mu_0 = 100$ are not of practical significance. Instead of the point null, we represent the null hypothesis as the *region of practical equivalence* $[97, 103]$, which describes a range of values that we're going to count as "no effect." See Figure 5.17 for an illustration.

The proportion of the posterior probability that falls inside the ROPE is defined as the following integral $\Pr(\{M \in [97, 103]\}) = \int_{\mu=97}^{\mu=103} f_{M|\mathbf{x}}(\mu|\mathbf{x})\, d\mu$. The code below shows how we can calculate the proportion of the posterior distribution $f_{M|\mathbf{x}} \approx$ `posterior2` that lies inside the region of practical equivalence $[97, 103]$.

code
5.1.26

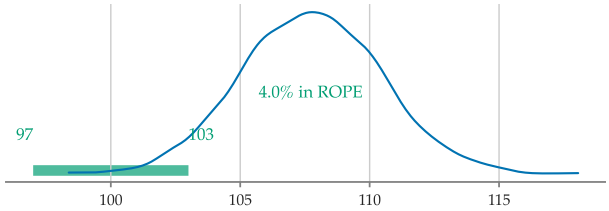


Figure 5.17: Percentage of the posterior distribution $f_{M|\mathbf{x}}$ that falls inside the region of practical equivalence (ROPE).

```
>>> mu97 = mus.searchsorted(97)
>>> mu103 = mus.searchsorted(103)
>>> np.sum(posterior2[mu97:mu103+1])
0.040481237454239115
```

Based on the conventional cutoff of 5%, our decision is to reject H_0. The skeptical colleague is wrong: the smart drug must be doing something.

Note that using a ROPE decision rule also allows us to accept H_0 when 95% or more of the posterior density falls within the ROPE. This is a market improvement from classical NHST procedure, which only allows us to reject H_0, but never accept it. The benefits of hypothesis testing using a ROPE are similar to the benefits we get from running a frequentist equivalence test like the TOST (page 262).

Bayesian model comparison

We can also think about Bayesian hypothesis testing as a comparison between two models: the model under the null hypothesis H_0, and the model under the alternative hypothesis H_A. We'll use the subscripts $_0$ and $_A$ to denote the parameters and the probability distributions under the two hypotheses. The Bayesian models under the two hypotheses are constructed from separate data models, $f_{X|\theta_0}$ and $f_{X|\theta_A}$, likelihood functions, $L^0_{\mathbf{x}}$ and $L^A_{\mathbf{x}}$, and priors, f_{Θ_0} and f_{Θ_A}. Additionally, we specify our prior belief about which hypothesis is more likely to be true, which we'll denote as $\Pr(H_0)$ and $\Pr(H_A)$.

The likelihood of the data according to the Bayesian models under the two hypotheses is computed as the weighted average of likelihoods and the priors:

$$\Pr(\mathbf{x}|H_0) = \int_{\theta_0} L^0_{\mathbf{x}}(\theta_0) f_{\Theta_0}(\theta_0)\, d\theta_0 \quad \text{and} \quad \Pr(\mathbf{x}|H_A) = \int_{\theta_A} L^A_{\mathbf{x}}(\theta_A) f_{\Theta_A}(\theta_A)\, d\theta_A.$$

The likelihoods $\Pr(\mathbf{x}|H_0)$ and $\Pr(\mathbf{x}|H_A)$ are new beasts that we haven't seen before. We've previously used the concept of likelihood for different *parameters* θ under the data model $f_{X|\theta}$, and defined the

likelihood function $L_{\mathbf{x}} = \prod_{i=1}^{n} f_{X|\theta}$. Now we define $\Pr(H_0|\mathbf{x})$ and $\Pr(H_A|\mathbf{x})$ which are the likelihoods of different *hypotheses* H_0 and H_A.

We referred to $\Pr(H_0|\mathbf{x})$ and $\Pr(H_A|\mathbf{x})$ as "beasts" for dramatic effect, but if we look at the formulas for a minute, you'll see they make sense at an intuitive level. They are just likelihoods at a different level of abstraction (hypotheses instead parameters). Since we're interested in comparing the two hypotheses, it makes sense to consider all possible values of the parameters θ_0 and θ_A. Specifically, we obtain the model-level likelihood functions $\Pr(H_0|\mathbf{x})$ and $\Pr(H_A|\mathbf{x})$ by *marginalizing* over the models' internal degrees of freedom (the random variables Θ_0 and Θ_A).

Hypotheses as parameters We can model our knowledge about which hypothesis is true as a binary random variable H that can take on one of two possible values, $H = \texttt{"0"}$ for the null hypothesis and $H = \texttt{"A"}$ for the alternative hypothesis. The Bayesian models under H_0 and H_A describe two different possible worlds. We model these two worlds as a parameter H in a higher-level Bayesian model.

The prior probabilities $\Pr(H_0)$ and $\Pr(H_A)$ about which hypothesis is true correspond to the values of the prior distribution of the random variable H in the higher-level model, $\Pr(H_0)$ and $\Pr(H_A)$. The model-level likelihood functions $\Pr(\mathbf{x}|H_0)$ and $\Pr(\mathbf{x}|H_A)$ correspond to the two branches of a likelihood function $\Pr(\mathbf{x}|H)$ for the higher-level model.

Okay, so we have a prior for H and a likelihood function $\Pr(\mathbf{x}|H)$, what do you think will be the next step?

Posteriors over hypotheses We can calculate the posterior probabilities of two hypotheses using Bayes' rule:

$$\Pr(H_0|\mathbf{x}) = \frac{\Pr(\mathbf{x}|H_0)\Pr(H_0)}{\Pr(\mathbf{x})} \quad \text{and} \quad \Pr(H_A|\mathbf{x}) = \frac{\Pr(\mathbf{x}|H_A)\Pr(H_A)}{\Pr(\mathbf{x})}.$$

This is really great, since we finally get what we wanted! Recall the goal of hypothesis testing is to decide between two competing models of reality: the probability models under H_0 and H_A. Bayes' rule allows us to compute the posterior probabilities $\Pr(H_0|\mathbf{x})$ and $\Pr(H_A|\mathbf{x})$, which combine our prior beliefs about the relative of the to models $\Pr(H_0)$ and $\Pr(H_A)$ and the model-level likelihoods $\Pr(\mathbf{x}|H_0)$ and $\Pr(\mathbf{x}|H_A)$, to tell us which model of reality better describes the observed data $\mathbf{x}$.

We usually only care about the *relative* posterior probabilities of

5.2 Bayesian inference computations

Bayesian inference is computationally "expensive" because calculating the posterior distribution $f_{\Theta|\mathbf{x}}$ requires integrating over all possible parameter values. The grid approximation that we used in the previous section doesn't scale well for models with many parameters, so we need to find a different approach for calculating the posterior distribution.

Markov Chain Monte Carlo (MCMC) is a computational technique for generating samples from the posterior distribution. MCMC samples provide us with an *empirical approximation* to the posterior distribution $f_{\Theta|\mathbf{x}}$. We can use the samples from the posterior to obtain all the desired results of a Bayesian analysis (visualizations, point estimates, interval estimates, and predictions).

We won't be introducing any new Bayesian statistics concepts in this section, and will instead focus on getting to know the new MCMC computational machinery. Specifically, we'll learn about two powerful Python libraries for Bayesian analysis: Bambi and ArviZ.

Bambi The Python library Bambi provides a user-friendly way to define and fit Bayesian models. The name Bambi is not deer-related, but an acronym for *BAyesian Model-Building Interface*. We build a Bambi model by specifying the data model and the prior distribution(s), then use this model to generate samples from the posterior $f_{\Theta|\mathbf{x}}$.

ArviZ The Python library ArviZ (pronounced *AR-vees*) provides helper functions for manipulating, visualizing, and summarizing MCMC samples obtained from Bayesian inference. ArviZ is like a Swiss Army knife for processing and analyzing Bayesian inference results.

In this section, we'll revisit the Bayesian statistical analyses we saw in the previous section, but this time we'll calculate all results by working with samples from the posterior distributions, instead of using the grid approximation. We'll use Bambi to build and fit the Bayesian models for the biased coin (Example 1) and the IQ scores (Example 2), then visualize the results using ArviZ functions. Spoiler alert: we'll obtain the same results as in Section 5.1. This makes sense because the grid approximation and the samples-from-the-posterior representation are two equivalent ways to work with probability distributions, so it makes sense that we obtain the same results.

Name	Hyperparameters	Probability distribution
`Uniform`	`lower`, `upper`	$\mathcal{U}(\alpha = \texttt{lower}, \beta = \texttt{upper})$
`Normal`	`mu`, `sigma`	$\mathcal{N}(\mu = \texttt{mu}, \sigma = \texttt{sigma})$
`HalfNormal`	`sigma`	$\mathcal{N}^+(\mu = 0, \sigma = \texttt{sigma})$
`StudentT`	`mu`, `sigma`, `nu`	$\mathcal{T}(\mu = \texttt{mu}, \sigma = \texttt{sigma}, \nu = \texttt{nu})$
`HalfStudentT`	`sigma`, `nu`	$\mathcal{T}^+(\mu = 0, \sigma = \texttt{sigma}, \nu = \texttt{nu})$
`Beta`	`alpha`, `beta`	$\text{Beta}(\alpha = \texttt{alpha}, \beta = \texttt{beta})$
`Gamma`	`alpha`, `beta`	$\text{Gamma}(\alpha = \texttt{alpha}, \beta = \texttt{beta})$
`Exponential`	`lam`	$\text{Expon}(\lambda = \texttt{lam})$
`Cauchy`	`alpha`, `beta`	$\text{Cauchy}(\alpha = \texttt{alpha}, \beta = \texttt{beta})$

Table 5.3: Prior distributions we can use when building Bambi models.

distribution $\mathcal{T}$, but they are defined only for nonnegative values. We often use these distributions as priors for scale parameters that can't be negative, like the standard deviation σ in a normal data model.

Building a Bambi model

We combine all the components that we discussed above and provide them as arguments when creating a Bambi `Model` object.

code 5.2.1
```
>>> import bambi as bmb
>>> mod = bmb.Model(formula="<formula>",
                    family="<family name>",
                    link="<link name>",
                    priors={"param": <prior for param>},
                    data=df)
```

The priors are specified as a dictionary that maps each parameter name to a Bambi `Prior` object.

Fitting the model

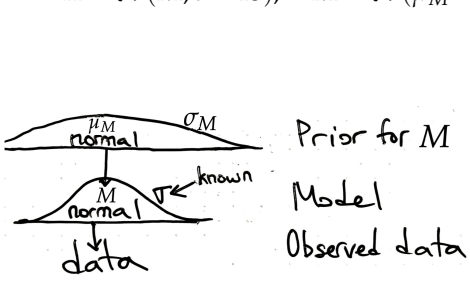
Once we've created the Bambi model `mod`, we can call `mod.fit()` to run the Bayesian inference procedure and generate thousands of samples from the posterior. Fitting simple models takes a few seconds, while complicated models can take minutes or hours.

Fitting the model involves doing a lot of work behind the scenes, including choosing which sampling algorithm to use, initializing multiple inference chains, letting the chains "warm up" for T tuning iterations, then finally collecting samples from the posterior. All the complexity of Bayesian inference calculations is completely hidden from us.

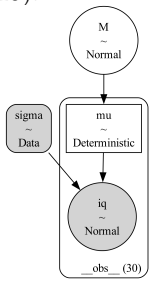
Bayesian model for the IQ scores

The Bayesian model for this analysis consists of a normal data model with standard deviation $\sigma = 15$ and a normal prior on the mean:

$$X \sim \mathcal{N}(M, \sigma = 15), \quad M \sim \mathcal{N}(\mu_M = 100, \sigma_M = 40).$$



(a) Graphical diagram for the Bayesian model.



(b) Output of `mod2.graph()`.

Figure 5.22: Graphical model diagrams for the Bayesian model that show the data model $X \sim \mathcal{N}(M, \sigma)$ and the prior distribution $M \sim \mathcal{N}(\mu_M, \sigma_M)$.

Figure 5.22 (a) shows two graphical model diagrams for the Bayesian model we'll use to analyze the IQ scores data. The prior distribution $f_M = \mathcal{N}(\mu_M = 100, \sigma_M = 40)$ is very broad, which means any value of M in the interval $[60, 140]$ is plausible.

We must now translate the model specification into the format that Bambi expects. We start by rewriting the model definition using the notation for generalized linear models:

$$X \sim \mathcal{N}(\mu, \sigma = 15), \qquad \mu = \beta_0, \qquad \beta_0 \sim \mathcal{N}(\mu_M = 100, \sigma_M = 40).$$

We see that the model `family` is Gaussian, the `link` function is the identity, and the model formula is `"iq ~ 1"`, since we're modelling the mean using only an intercept term, β_0 = `Intercept`. The prior on the intercept term is $\mathcal{N}(100, 40)$. We can now create the Bambi model object that corresponds to the above math description.

code 5.2.18

```
>>> import bambi as bmb
>>> priors2 = {
        "Intercept": bmb.Prior("Normal",mu=100,sigma=40),
        "sigma": 15,
    }
>>> mod2 = bmb.Model(formula="iq ~ 1",
                     family="gaussian",
                     link="identity",
                     priors=priors2,
                     data=iqs)
>>> mod2.set_alias({"Intercept": "M"})
```

Note we set the fixed value `15` as the parameter `sigma`, instead of a prior distribution. We also defined the alias `M` for the variable β_0 = `Intercept`, so that we can use the label `M` when analyzing the results.

We can print the model `mod2` to make sure Bambi correctly interpreted the model definition we provided, and use the method `mod2.graph()` to generate a graphical model diagram for it.

code 5.2.19
```
>>> mod2
       Formula: iq ~ 1
        Family: gaussian
          Link: mu = identity
  Observations: 30
        Priors:
    target = mu
        Common-level effects
            Intercept ~ Normal(mu: 100.0, sigma: 40.0)
        Auxiliary parameters
            sigma ~ 15.0
>>> mod2.build()  # need to call .build() before .graph()
>>> mod2.graph()
See the model graph in Figure 5.22 (b).
```

Fitting the model

We're now ready to fit the model by calling its `.fit()` method. This time, we'll instruct Bambi to generate 2000 draws from each chain by passing the option `draws=2000` to the `.fit()` method.

code 5.2.20
```
>>> idata2 = mod2.fit(draws=2000)
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [M]
Sampling 4 chains, 0 divergences ---------------- 100%
Sampling 1000 tune and 2000 draw iterations (8000 total)
```

The MCMC inference procedure took a little longer than the inference in Example 1, since we're running the chains for twice as many draws. The usual verbose MCMC information was printed. Note the total number of samples we generated is 4 `chains` $\times$ 2000 `draws` $=$ 8000.

The next step is to extract the samples from the posterior. Specifically, we'll access the `posterior` group within the `idata2` object, select the variable `M` within the posterior group, then use `.values.flatten()` to obtain a one-dimensional array of samples.

code 5.2.21
```
>>> postM = idata2["posterior"]["M"].values.flatten()
>>> len(postM)
8000
>>> postM
array([109.72, 108.78, 108.42, ..., 113.23, 109.71, 102.97])
```

The array `postM` contains 8000 samples from the posterior $f_{M|\mathbf{x}}$.

Visualizing the posterior

We can plot the posterior distribution using either `sns.histplot` or `sns.kdeplot`, with the results producing essentially the same shape.

code 5.2.22
```
>>> sns.histplot(x=postM, stat="density")
See Figure 5.23 (a).
>>> sns.kdeplot(x=postM)
See Figure 5.23 (b).
```

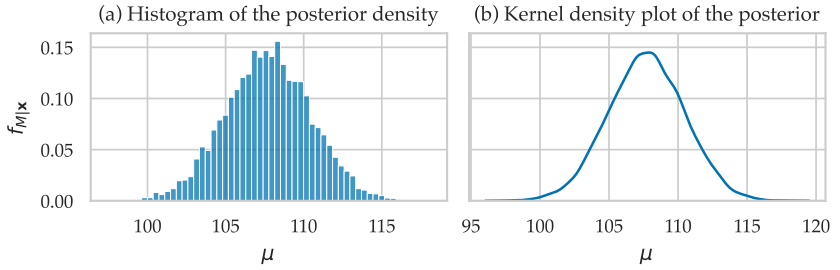


Figure 5.23: Visualization of the 8000 samples from posterior $f_{M|\mathbf{x}}$.

We see the posterior distribution is centred around $\mu = 107.5$, and its half-width (standard deviation) is around three IQ points.

Summarizing the posterior

We can make the informal visual estimates more precise by computing posterior summary statistics like the mean, standard deviation, median, and the quartiles of the posterior distribution.

code 5.2.23
```
>>> np.mean(postM)     # posterior mean
107.70792049114165
>>> np.std(postM)      # posterior standard deviation
2.747782246509056
>>> np.median(postM)   # posterior median
107.71896436263302
>>> np.quantile(postM, [0.25, 0.5, 0.75])  # Q1, Q2, Q3
array([105.8404028 , 107.71896436, 109.58680961])
```

Constructing a credible interval

To construct a 90% credible interval for the unknown μ, we can use the function `hdi_from_samples` from the `ministats` library.

code 5.2.24
```
>>> from ministats import hdi_from_samples
>>> hdi_from_samples(postM, hdi_prob=0.9)
[103.26580730069273, 112.284577803755]
```

The highest density interval $\mathbf{hdi}_{\mu,0.9} = [103.26, 112.28]$ contains 90% of the probability mass of the posterior distribution $f_{M|\mathbf{x}}$.

* * *

Compare the results we obtained here to the Example 2 results we obtained in Section 5.1 using the grid approximation (see page 465). Same stuff, right? I hope you'll agree with me that outsourcing the Bayesian inference calculations to the Bambi library simplifies our life as compared to the grid approximation. Speaking of outsourcing parts of the Bayesian analysis, let's see how we can outsource some of the summarization and visualization steps.

5.2.6 Visualizing and interpreting posterior distributions

The Python library ArviZ is a toolkit for visualizing Bayesian inference data. The name ArviZ, pronounced "AR-vees," plays on the acronym "RVs," which is short for "random variables." The spelling "viZ" is a hint that the library provides visualizations tools.

ArviZ complements Bambi by providing helper functions for summarizing and visualizing the samples from the posterior that we obtain from the MCMC procedure. For example, instead of computing the posterior mean using `np.mean` and the standard deviation using `np.std`, we can call the ArviZ `summary` function to perform these calculations at once.

The `arviz` Python module is usually imported under the alias `az`. Here is a list of some of the most commonly used ArviZ functions:

- `az.summary`: calculates posterior summary statistics like the mean, the standard deviation, and the highest density interval.
- `az.plot_posterior`: generates density plots of the posterior distribution with additional annotations like point estimates (mean, median, mode), highest density intervals, and probabilities within a specified region of practical equivalence (ROPE).
- `az.plot_forest`: plots the posterior summary statistics including the quartiles and the highest density interval.
- `az.extract`: extracts samples from `InferenceData` objects.
- `az.plot_ppc` : plots simulated samples from the prior predictive distribution or the posterior predictive distribution.
- `az.plot_trace`: generates diagnostic plots, which can help us identify problems with the MCMC inference procedure.

The common feature of all these functions is that they are designed to work with `InferenceData` objects, and they "know" how to access different groups, variables, and chains inside them. This means we can use ArviZ functions directly with the `InferenceData` objects that we obtain when we fit Bambi models.

```
>>> az.summary(idata2, kind="stats", hdi_prob=0.9)
      mean         sd       hdi_5%  hdi_95%
M  107.708      2.748      103.266  112.285
```

The results are exactly the same as the values we computed earlier using `np.mean(postM)` and `np.std(postM)`.

Plotting the posterior Let's now plot the posterior $f_{M|\mathbf{x}}$ using the `az.plot_posterior` function. We'll show both a kernel density plot and a histogram of the posterior.

code 5.2.29

```
>>> az.plot_posterior(idata2, var_names="M", hdi_prob=0.9)
>>> az.plot_posterior(idata2, var_names="M", hdi_prob=0.9,
                      kind="hist", bins=70)
```

The results of the two commands are shown in Figure 5.26.

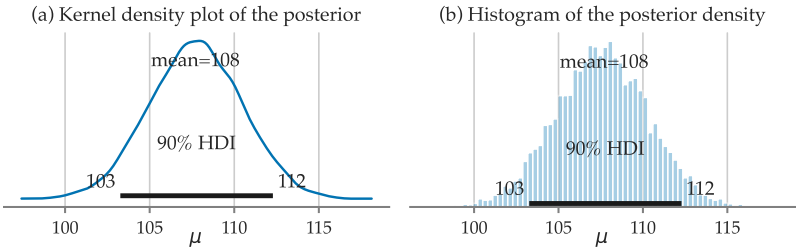


Figure 5.26: Plots of the 8000 samples from the posterior distribution $f_{M|\mathbf{x}}$. Each plot is annotated with the value of the posterior mean $\mu_{M|\mathbf{x}} = 108$ and a 90% highest density interval $\mathbf{hdi}_{\mu,0.9} = [103, 112]$.

The plots in Figure 5.26 are identical to the plots we saw earlier in Figure 5.21, but provide additional useful annotations of the posterior mean $\mu_{M|\mathbf{x}}$ and the 90% highest density interval $\mathbf{hdi}_{\mu,0.9}$.

We can also generate a forest plot for the parameter `M` using the function `az.plot_forest`.

code 5.2.30

```
>>> az.plot_forest(idata2, combined=True, hdi_prob=0.9)
See Figure 5.27.
```

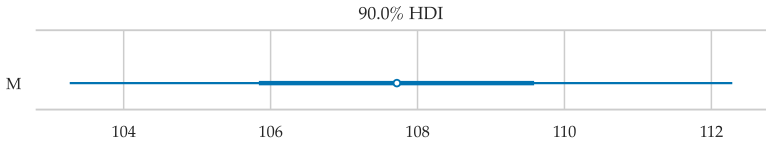


Figure 5.27: Forest plot of the posterior distribution $f_{M|\mathbf{x}}$. The summary statistics are calculated from the `combined` samples from all four chains.

We used the option `combined=True` to generate the forest plot in Figure 5.27, which means the highest density interval is computed from the combined 8000 samples from all four chains.

Bayesian hypothesis testing plots We can use the ArviZ function `az.plot_posterior` for Bayesian hypothesis testing. Let's revisit the two hypothesis testing approaches that we discussed in Section 5.1. The first approach considers the point null hypothesis $H_0 : \mu = 100$. We can construct a 95% credible interval for the unknown mean M using the option `hdi_prob=0.95` when generating the posterior plot.

```
>>> az.plot_posterior(idata2, hdi_prob=0.95)
The result is shown in Figure 5.28 (a).
```
code 5.2.31

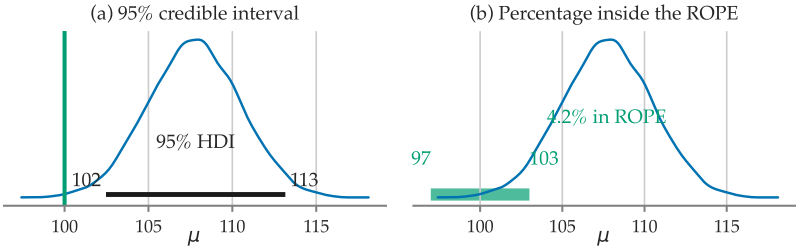


Figure 5.28: Posterior distribution $f_{M|\mathbf{x}}$ from Example 2 with hypothesis testing annotations. The 95% credible interval $\mathbf{hdi}_{\mu,0.95} = [102, 113]$ doesn't include the null value $\mu_0 = 100$, which allows us to reject H_0. The region of practical equivalence $[97, 103]$ contains only 4.2% of the posterior distribution, which is another way to show evidence against the null hypothesis.

We see the 95% highest density interval $\mathbf{hdi}_{\mu,0.95} = [102, 113]$ doesn't include the null value $\mu_0 = 100$, which means we can reject H_0 at the 5% ($\alpha = 0.05$) significance level.

The second approach to hypothesis testing uses an interval null hypothesis, specified as a *region of practical equivalence* (ROPE) around the null value $\mu_0 = 100$. We consider IQ scores within 3 points of the null value $\mu_0 = 100$ not to be practically significant, which corresponds to the ROPE $[97, 103]$. We can pass the option `rope=[97,103]` when generating the posterior plot to calculate and report the proportion of the posterior that falls within the ROPE.

```
>>> az.plot_posterior(idata2,hdi_prob="hide",rope=[97,103])
The result is shown in Figure 5.28 (b).
```
code 5.2.32

The plot in Figure 5.28 (b) shows the results of the probability calculation $\Pr(\{M \in [97, 103]\}) = \int_{\mu=97}^{\mu=103} f_{M|\mathbf{x}}(\mu|\mathbf{x})\, d\mu = 0.042$. The overlap between the posterior distribution and the region of practical equivalence is less than the threshold 5% ($\alpha = 0.05$), which tells us we can reject the null hypothesis at the 5% level.

5.2.7 Explanations

We'll now fill in some details that we skipped in earlier sections.

Intermezzo

So far, we've studied simple Bayesian models with a single parameter. We started simple to make it easy to learn the Bayesian concepts (Section 5.1) and computations (Section 5.2).

In the remainder of the chapter, we'll learn about more realistic Bayesian models with multiple parameters and richer structure. It's not going to be all new material, though, since we'll revisit some of the data analysis scenarios from previous chapters. Learning Bayesian statistics normally takes a whole textbook with thousands of pages of explanations. However, your accumulated knowledge about data, probability theory, and statistical inference will allow us to do a speed run of several Bayesian models.

Here is a quick overview of what is coming in the next three sections. In Section 5.3, we'll revisit linear models from a Bayesian perspective. In Section 5.4, we'll build Bayesian models for comparing two groups. We'll then close the chapter in Section 5.5, where we'll discuss Bayesian hierarchical models.

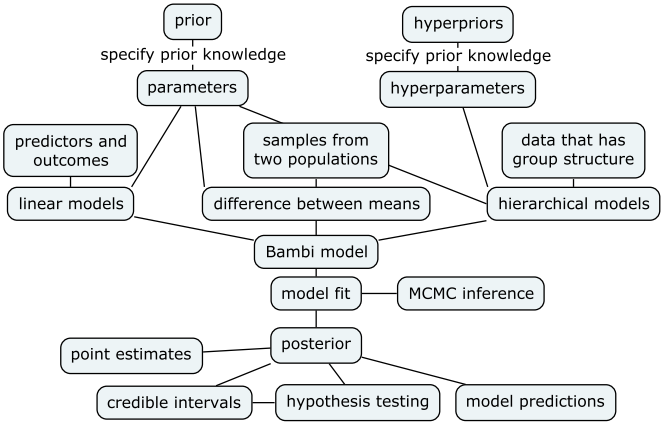


Figure 5.38: Bayesian models and concepts from the next three sections.

5.3 Bayesian linear models

Linear models are a powerful tool for modelling relationships between variables. The simple linear regression model that we studied in Section 4.1 is described by the formula $Y|x \sim \mathcal{N}(\beta_0 + \beta_x \cdot x, \sigma)$, where β_0 is the intercept term and β_x is the slope associated with the predictor variable x. Using least squares fitting (or maximum likelihood), we can find the best-fit point estimates $\widehat{\beta}_0$, $\widehat{\beta}_x$, and $\widehat{\sigma}$ for the unknown parameters β_0, β_x, and σ.

A Bayesian simple linear regression model is described by a similar formula, $Y|x \sim \mathcal{N}(B_0 + B_x \cdot x, \Sigma)$, where B_0, B_x, and Σ are unknown random variables. When working with this Bayesian linear model, we start with prior distributions f_{B_0}, f_{B_x}, and f_Σ for the unknown parameters, then use Bayesian inference (thank you Bambi!) to obtain the posterior distributions $f_{B_0|\mathbf{x},\mathbf{y}}$, $f_{B_x|\mathbf{x},\mathbf{y}}$, and $f_{\Sigma|\mathbf{x},\mathbf{y}}$. We can then summarize the posteriors by computing point estimates and credible intervals, or using them to make predictions and decisions.

In this section, we'll revisit the `students`, `doctors`, and `interns` datasets, and fit Bayesian models for them. We'll show two examples of linear models based on the Gaussian data model (`family="gaussian"`), and one example of a logistic regression model (`family="bernoulli"` with `link="logit"`). The results we obtain from these Bayesian models will be similar to the results we obtained from the frequentist models in Chapter 4, so don't expect any big surprises. However, Bayesian models provide a more natural way to describe the uncertainty in our estimates (posterior distributions and credible intervals) as opposed to the frequentist results (confidence intervals).

5.3.1 Bayesian model for simple linear regression

Consider the bivariate dataset $[\mathbf{x}, \mathbf{y}]$, which consist of the outcome variable y and a single predictor x. The Bayesian simple linear regression model for this dataset is described by the equations:

$$
\begin{aligned}
Y &\sim \mathcal{N}(M_Y, \Sigma), && \text{[data model]} \\
M_Y &= B_0 + B_x \cdot x, && \text{[linear predictor for the mean]} \\
B_0 &\sim \mathcal{N}(\mu_{B_0}, \sigma_{B_0}), && \text{[prior for the intercept term]} \\
B_x &\sim \mathcal{N}(\mu_{B_x}, \sigma_{B_x}), && \text{[prior for the slope parameter]} \\
\Sigma &\sim \mathcal{T}^+(\nu_\Sigma, \sigma_\Sigma). && \text{[prior for the standard deviation]}
\end{aligned}
$$

Starting in this section, we'll use this "vertical" format when presenting the equations that represent the data generative process of

the Bayesian models we use for analysis. These equations provide the complete "story" about how we believe the data was generated. Figure 5.39 shows the corresponding graphical model diagram. I know such large blocks of math symbols look intimidating at first sight, but if you look carefully, you'll see that you're already familiar with all the components of this model. Indeed, we're trying to fit the same model that we studied in Section 4.1, but we're now modelling the parameters as random variables, and we've set priors on them.

Let's look at the equations line by line. The first line defines the Gaussian data model with mean M_Y and standard deviation Σ. The second line specifies how the mean M_Y depends on the predictor x and introduces two new parameters: the intercept B_0 and the slope B_x. The remaining equations define the priors for three model parameters B_0, B_x, and Σ. The prior for the standard deviation Σ is a half-t-distribution $\mathcal{T}^+$, which has the same shape as the regular t-distribution $\mathcal{T}$, but is defined only for nonnegative values. The half-t distribution $\mathcal{T}^+$ is a convenience choice for scale parameters like σ that can't take on negative values. This model definition has six hyperparameters μ_{B_0}, σ_{B_0}, μ_{B_x}, σ_{B_x}, ν_Σ, and σ_Σ. These are the *design parameters* of the model and must be adapted to different data analysis scenarios. We'll describe the general strategy for choosing these hyperparameters in the next section.

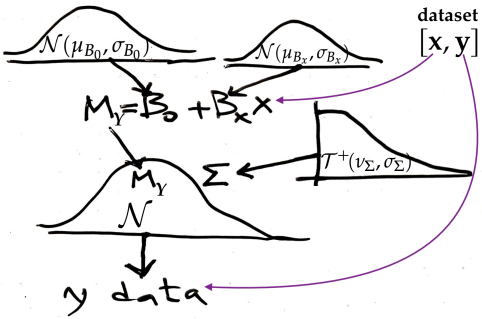


Figure 5.39: Graphical model diagram of the Bayesian simple linear regression model for the dataset $[\mathbf{x}, \mathbf{y}]$.

The graphical model diagram in Figure 5.39 is a summary of the model equations and shows the structure of the model more concisely. It's important that you learn to read these graphical model diagrams, since they make the model structure easier to see than the equations. The outcome variable usually appears at the bottom of the diagram, and we read the diagram starting from the bottom and moving up. Take a moment to identify the correspondences between the elements in Figure 5.39 and the model equations. You should be

able to reconstruct the equations from the graphical model, and vice versa, since both descriptions contain the same information.

Choosing priors and hyperparameters

There is a lot of flexibility in choosing the prior distributions. We'll use normal priors for the intercept B_0 and slope B_x for simplicity. We can choose the location μ_{B_0} and the scale σ_{B_0} hyperparameters for the intercept term based on the sample mean $\overline{\mathbf{y}}$ and standard deviation $s_{\mathbf{y}}$ of the outcome variable. For example, choosing $B_0 \sim \mathcal{N}(\overline{\mathbf{y}}, 2.5s_{\mathbf{y}})$ produces a weakly informative prior.

For the slope parameter, we usually choose a prior centred at zero ($\mu_{B_x} = 0$), since the influence of the predictor x could be either positive or negative. We choose the scale hyperparameter σ_{B_x} based on the sample standard deviations $s_{\mathbf{x}}$ and $s_{\mathbf{y}}$. Specifically, we can set σ_{B_x} as a multiple of the ratio $s_{\mathbf{y}}/s_{\mathbf{x}}$.

Finally, for the standard deviation Σ parameter, we'll use a half-t distribution with $\nu_\Sigma = 4$ degrees of freedom, which produces a heavy-tailed distribution. We set the scale parameter σ_Σ as a multiple of $s_{\mathbf{y}}$.

We'll describe the specific choices of hyperparameters when we look at the hands-on examples in the upcoming sections.

Bayesian modelling advantage

Before we look at the examples, it's worth describing clearly the benefits we expect to get from adopting the Bayesian perspective. Let's be honest, fitting a Bayesian model introduces a lot of additional complexity and computation, so there better be benefits! The four key advantages that the Bayesian models give us relative to frequentist linear models we studied in Chapter 4 are listed below.

1. Bayesian prior distributions allow us to incorporate prior knowledge into the analysis. Using informative priors is particularly useful when working with small sample sizes.

2. Bayesian linear models provide a more direct way to describe uncertainty. Instead of point estimates and confidence intervals, we get posterior distributions $f_{B_0|\mathbf{x},\mathbf{y}}$, $f_{B_x|\mathbf{x},\mathbf{y}}$, and $f_{\Sigma|\mathbf{x},\mathbf{y}}$ that provide more detailed information about the parameters.

3. Bayesian predictions (posterior predictive distributions) provide more realistic predictions, since they account for uncertainty in the parameter estimates.

That is not to say that Bayesian linear models are without problems. Fitting Bayesian models is more computationally expensive and requires us to think carefully about the priors we want to use. The price we pay for the flexibility of Bayesian models, is that we have to perform prior and posterior predictive checks to make sure the Bayesian inference procedure is working as expected.

5.3.2 Example 1: students' scores

Recall Charlotte's students' dataset that we studied previously in Section 1.2 (page 44) and in Section 4.1.3 (page 295). We're interested in the relationship between the predictor variable `effort` and the outcome variable `score`. Do students' scores improve when they invest more effort?

We start by loading the students dataset and calculating the descriptive statistics for the variables `score` and `effort`.

code 5.3.1
```
>>> students = pd.read_csv("../datasets/students.csv")
>>> students[["effort","score"]].describe().T
        count   mean   std   min    25%   50%    75%   max
effort  15.0    8.90   1.95   5.21   7.76  8.69 10.35  12.0
score   15.0   72.58   9.98  57.00  68.00 72.70 75.75  96.2
```

We see that students invest between 5.21 and 12 hours of effort, and the average score is 72.58 points with a standard deviation of 10 points. We'll use these sample statistics as a reference when choosing the model hyperparameters.

Bayesian simple linear regression model

The Bayesian model we'll use for this analysis is the following:

$$
\begin{aligned}
S &\sim \mathcal{N}(M_S, \Sigma), && \text{[data model for the student scores]}\\
M_S &= B_0 + B_e \cdot e, && \text{[linear model for the mean score]}\\
B_0 &\sim \mathcal{N}(\mu_{B_0} = 70, \sigma_{B_0} = 20), && \text{[prior for the intercept term]}\\
B_e &\sim \mathcal{N}(\mu_{B_e} = 0, \sigma_{B_e} = 10), && \text{[prior for the slope]}\\
\Sigma &\sim \mathcal{T}^{+}(\nu_\Sigma = 4, \sigma_\Sigma = 20). && \text{[prior for the standard deviation]}
\end{aligned}
$$

See Figure 5.40 (a) for the graphical model diagram that corresponds to these equations.

Choice of priors Let's discuss how we chose the priors for the model based on the descriptive statistics of the students dataset.

- For the intercept term, we pick $\mu_{B_0} = 70$, which is close to the average score 72.58, and set the scale $\sigma_{B_0} = 20$, which is roughly

twice the sample standard deviation of the `score` variable. We choose round numbers for simplicity.

- For the slope parameter of the effort variable, we choose a normal prior centred at zero. We set the scale σ_{B_e} based on the ratio of the `score` and `effort` sample standard deviations, which is $9.98/1.95 \approx 5$. We double this value to get $\sigma_{B_e} = 10$.
- We choose the prior for the standard deviation to be a half-t distribution with four degrees of freedom to get a distribution with heavy tails. We set the scale parameter $\sigma_\Sigma = 20$, which is twice the sample standard deviation of the `score` variable.

The hyperparameters we chose above produce weakly informative priors, which means the Bayesian model will prioritize the information in the likelihood. We're letting the data speak for itself.

Bambi model

We'll now translate the mathematical equations that describe the Bayesian model into a code description specified in terms of the building blocks that Bambi provides. We start by creating the dictionary of priors `priors1` that specifies the prior distributions we want to use for each parameter. We then create the Bambi model object `mod1` based on the formula `"score ~ 1 + effort"`.

code 5.3.2

```
>>> import bambi as bmb
>>> priors1 = {
      "Intercept": bmb.Prior("Normal",mu=70,sigma=20),
         "effort": bmb.Prior("Normal",mu=0,sigma=10),
          "sigma": bmb.Prior("HalfStudentT",nu=4,sigma=20),
    }
>>> mod1 = bmb.Model("score ~ 1 + effort",
                     family="gaussian",
                     link="identity",
                     priors=priors1,
                     data=students)
```

I assume you're familiar with concepts in the above code block, since we discussed them in Section 5.2. The main new thing is the formula `"score ~ 1 + effort"`, which includes the predictor `effort`.

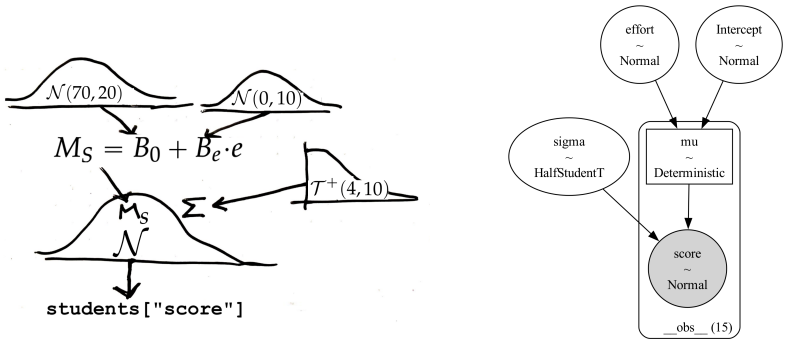
Inspecting the Bambi model

It's always a good idea to check that the "spec" we gave to Bambi produced the correct model. We can print the model object to see a detailed textual representation, or use the `.graph()` method to see the graphical model diagram.

code 5.3.3

```
>>> mod1
       Formula: score ~ 1 + effort
```

```
        Family: gaussian
          Link: mu = identity
  Observations: 15
        Priors:
    target = mu
        Common-level effects
          Intercept ~ Normal(mu: 70.0, sigma: 20.0)
             effort ~ Normal(mu: 0.0, sigma: 10.0)
        Auxiliary parameters
              sigma ~ HalfStudentT(nu: 4.0, sigma: 20.0)
>>> mod1.graph()
The graph is shown in Figure 5.40 (b).
```



(a) Bayesian model for the students dataset.

(b) Output of `mod1.graph()`.

Figure 5.40: Graphical diagram for the Bayesian linear model we use to describe how student scores, S = `students["score"]`, depend on effort, e = `students["effort"]`. The model consists of the familiar deterministic formula $M_S = B_0 + B_e \cdot e$ sandwiched by "probabilistic machinery" above it (priors) and below it (data model). The data model $\mathcal{N}(M_S, \Sigma)$ describes the variability of the observed scores for a given choice of the parameters $M_S = \mu_S$ and $\Sigma = \sigma$. The priors $f_{B_0} = \mathcal{N}(70, 20)$, $f_{B_e} = \mathcal{N}(0, 10)$, and $f_\Sigma = \mathcal{T}^+(4, 20)$ describe our *a priori* knowledge about what parameter values we might observe.

The graphical model diagram shows the structure of the Bambi model we defined. The textual description we obtain when we print the model `mod1` shows the details about the hyperparameters of the prior distribution. It's always a good idea to inspect and compare the graphical model diagram, the textual description, and the model equations to make sure the model is correctly specified.

We won't repeat these "inspection" steps for all the examples in this section, but I can attest to their usefulness in preparing the notebooks for this book. Some of the common mistakes I was able to detect by printing the model include missing priors, misspelled parameter values, and using the wrong syntax for priors.

5.4 Bayesian difference between means

We'll now build a Bayesian model for comparing two samples that come from unknown populations. We want to make a decision whether the samples come from the same population or from different populations. This is a common data analysis scenario, which we studied previously in Section 3.5, where we compared the electricity prices in the East and the West parts of a city to determine if there is a difference. Another example of this type of analysis is to determine if a drug is effective, by comparing the results of a group of patients that received the drug to a control group that received a placebo.

In this section, we'll define a Bayesian model for comparing the samples $\mathbf{x}$ and $\mathbf{y}$ that come from the unknown populations X and Y. We'll model the population means as random variables M_X and M_Y, and analyze the difference between means, defined as $D_M = M_X - M_Y$. The model we construct will be robust to variations in the standard deviations in the two populations, and to the presence of outliers.

5.4.1 Bayesian model for comparing two populations

The starting point of our analysis are two samples of observations $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ that come from two unknown populations X and Y. We want to know if the two populations are the same or different. We'll approach this analysis scenario by building independent models for the two populations, then comparing the estimated population means. Our model can accommodate differences in the standard deviations of the two populations, and the presence of outliers.

The Bayesian model we'll use for all the analyses in this section is shown in Figure 5.49. This model is inspired by the *Bayesian estimation supersedes the t-test* (BEST) model proposed by John Kruschke [Kru13], however we'll make different choices for some prior distributions.

We'll now describe the components of the Bayesian model step by step, starting with the choice of the t-distribution as the data model.

Data model

We expect the two populations to be roughly normally distributed, but we want to build a model that is robust to the presence of outliers. We'll therefore use Student's t-distribution as the data model for the two populations:

$$X \sim \mathcal{T}(\nu, M_X, \Sigma_X) \qquad \text{and} \qquad Y \sim \mathcal{T}(\nu, M_Y, \Sigma_Y).$$

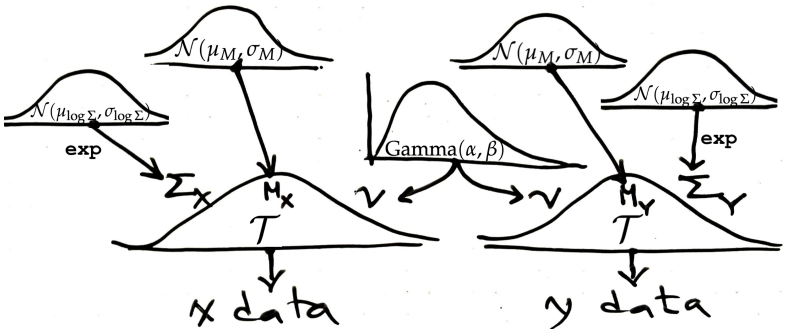


Figure 5.49: Graphical diagram of the Bayesian model for comparing the samples of observations **x** and **y** from two unknown populations.

The parameters M_X and M_Y describe the unknown means of the two populations. We assume the standard deviations for the two populations can be different, and model them as two independent scale parameters Σ_X and Σ_Y. The degrees of freedom parameter ν (pronounced "niu") of Student's t-distribution controls the "heaviness" of the distribution's tails. Since this is a Bayesian model, we treat the degrees of freedom parameter as a random variable ν, and let the Bayesian inference procedure find the range of values that best fits the observed data.

Priors for the means

We set normal priors for the means of the two populations:

$$M_X \sim \mathcal{N}(\mu_M, \sigma_M) \qquad \text{and} \qquad M_Y \sim \mathcal{N}(\mu_M, \sigma_M).$$

To obtain weakly informative priors, we choose μ_M in the same ballpark as the sample means $\overline{\mathbf{x}}$ and $\overline{\mathbf{y}}$, and set the scale hyperparameter σ_M to be a multiple of the sample standard deviations $s_{\mathbf{x}}$ and $s_{\mathbf{y}}$.

Priors for the scale parameters

The story for the priors on the scale parameters Σ_X and Σ_Y is a little more complicated. When using the t-distribution as the data model in Bambi (`family="t"`), we have to use the log-transformed internal representation of the scale parameters. This means we have to set priors on the log-transformed parameters $\log(\Sigma_X)$ and $\log(\Sigma_Y)$ instead of directly on Σ_X and Σ_Y.

We choose to place normal priors on the logarithms of the standard deviations:

$$\log(\Sigma_X) \sim \mathcal{N}(\mu_{\log\Sigma}, \sigma_{\log\Sigma}) \quad \text{and} \quad \log(\Sigma_Y) \sim \mathcal{N}(\mu_{\log\Sigma}, \sigma_{\log\Sigma}).$$

5.5 Hierarchical models

Data observations often come in groups. For example, suppose we have collected a sample of $n = 60$ student grades $\mathbf{x} = [x_1, x_2, \ldots, x_{60}]$ that come from three different classes. Each class had 20 students and was taught by a different teacher, let's call them A, B, and C. We might reasonably expect to observe similarities between the first 20 grades $\mathbf{x}_A = [x_1, x_2, \ldots, x_{20}]$, since they come from students who were taught by the same teacher. We might also expect to see differences between the grades $\mathbf{x}_A$ and the grades $\mathbf{x}_B = [x_{21}, x_{22}, \ldots, x_{40}]$ of the students taught by the second teacher. **We need some way to take into account the existence of the group structure** when analyzing the students' grades $\mathbf{x} = [x_1, x_2, \ldots, x_{60}]$.

Hierarchical models provide a natural way to model datasets with group dependence. The structure of a hierarchical model allows us to describe the commonality and variability among groups (teacher effects), as well as the variability of individual observations within each group (students' grades within classes). To motivate the need for hierarchical models, it's worth considering them in comparison to two other modelling strategies that make simplifying assumptions. We can classify these modelling strategies by how they "pool together" the information in the dataset $\mathbf{x}$.

- A *complete-pooling* modelling strategy ignores the existence of the groups and considers all observations in the sample $\mathbf{x}$ to come from the same population f_X.
- A *no-pooling* modelling strategy assumes that each group is independent of the other groups, and sees the sample $\mathbf{x}$ as three independent chunks $\mathbf{x}_A$, $\mathbf{x}_B$, and $\mathbf{x}_C$, where each chunk comes from a different population: $f_{X_A}, f_{X_B}, f_{X_C}$.
- Hierarchical models use a *partial-pooling* modelling strategy, which assumes each group comes from a different distribution, but models the parameters of the group-specific distributions as if they come from a common, higher-level distribution.

Each modelling strategy has different advantages and limitations, but the partial-pooling strategy provides the most flexibility.

In this section, we'll learn about Bayesian hierarchical models by starting with definitions and formulas, then looking at three examples of analyses performed on the same dataset. We'll look at a complete-pooling model in Example 1, a no-pooling model in Example 2, and finally a partial-pooling (hierarchical) model in Example 3.

5.5.1 Definitions

Let's start by introducing the new terminology and notation for describing hierarchical datasets and hierarchical models.

Hierarchical datasets

The students-in-classes scenario we discussed above is an example of a hierarchical dataset. Other examples of datasets with hierarchical structure include: patients within hospitals, houses within neighbourhoods, and repeated measurements from different individuals. In each case, we want to model the variability of individual observations within groups, as well as the commonalities and differences between groups.

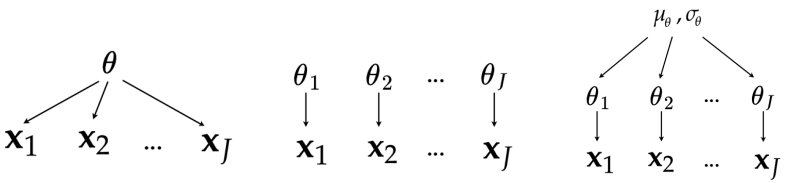
We'll assume the dataset $\mathbf{x}$ contains J groups:

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_J].$$

We denote the number of observations in each group as $n_1, n_2, \ldots, n_J$, and the total sample size as n, where $n = n_1 + n_2 + \cdots + n_J$. We'll use the index $j \in \{1, 2, \ldots, J\}$ to refer to the different groups, and the index $i \in \{1, 2, \ldots, n\}$ when referring to individual observations.

Pooling, no-pooling, and partial-pooling strategies

We'll now describe the three modelling strategies for working with hierarchical datasets. Figure 5.57 illustrates the *complete-pooling*, *no-pooling*, and *partial-pooling* strategies for analyzing a dataset that consists of J groups, $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_J]$.



(a) Complete pooling. (b) No pooling. (c) Partial pooling.

Figure 5.57: Different types of pooling strategies when analyzing the samples from J groups $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_J$.

The *complete-pooling* strategy illustrated in Figure 5.57 (a) assumes that all observations come from the same population, and tries to infer the common parameter θ of that population. In a Bayesian context, we would model the unknown parameter as a random variable Θ and specify a prior distribution f_Θ on it, but we don't show the priors to keep the figure simple and focused on the

differences between the three pooling strategies. The complete-pooling model would be appropriate to use if we don't care about the differences between the different groups, or if the groups are very similar so we can ignore the differences between them. The problem with the complete-pooling strategy is that the uncertainty estimates we obtain will be inflated, since we're treating intergroup variability the same way as the variability of individual observations.

In the *no-pooling* strategy, each group is modelled by its own parameter θ_j, which is completely independent of the parameters of the other groups. We're treating the analysis as J separate inference problems. The no-pooling model is appropriate if the differences between groups are very large, and what we learn from one group is not useful when fitting the inference model for the other groups. The problem with this strategy is that we're ignoring the similarities between the groups.

In the *partial-pooling* strategy, we assume the parameters $\theta_1, \theta_2, \ldots, \theta_J$ of the different groups come from a common distribution, whose hyperparameters are μ_θ and σ_θ. The location hyperparameter μ_θ represents the common central tendency among the groups' parameters $\theta_1, \theta_2, \ldots, \theta_J$. The scale hyperparameter σ_θ measures the variability between groups. The partial-pooling strategy combines aspects of the complete-pooling and no-pooling strategies. Each group gets its own parameter θ_j, but we use the data from the other groups to learn a common population distribution $\mathcal{N}(\mu_\theta, \sigma_\theta)$, which has the effect of "sharing" information between groups.

The partial-pooling strategy includes the complete-pooling and the no-pooling strategies as special cases. For example, if we want the partial-pooling model to behave like the complete-pooling model (a), we can set $\sigma_\theta = 0$, which forces all the θ_js to be the same. On the other hand, if we set $\sigma_\theta = \infty$ in model (c), this corresponds to a completely uninformative common distribution, meaning there is no information worth sharing between the different θ_js, which is equivalent to the no-pooling model (b).

New terminology: hyperpriors and their parameters

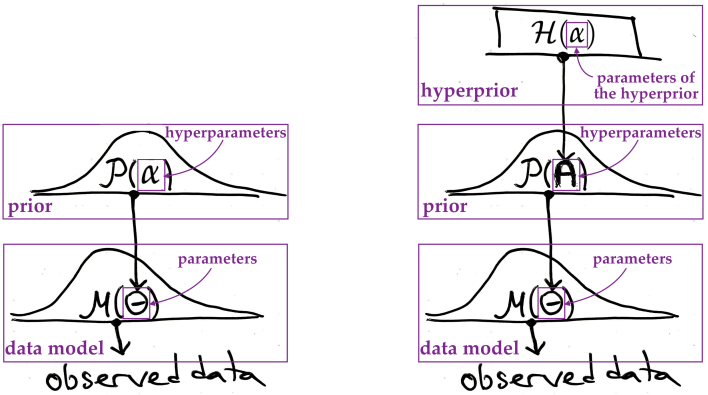
All Bayesian models are constructed from a hierarchy of probability distributions. The main characteristic of Bayesian hierarchical models is that they have multiple "levels" of parameters. This is why hierarchical models are also called *multilevel models*.

The Bayesian models we studied previously in this chapter had one level of parameters (denoted Θ), sandwiched by probability distributions from above (the prior f_Θ) and below (the data

model $\mathcal{M}$). We refer to the parameters of the prior distribution $f_\Theta = \mathcal{P}$ as *hyperparameters*. In Bayesian non-hierarchical models, hyperparameters are fixed constants that determine the shape of the prior distribution, as illustrated in Figure 5.58 (a).

In Bayesian hierarchical models, hyperparameters are random variables (level 2 parameters) and we specify prior distributions for them. We use the following terminology to describe the variables at different levels of a hierarchical model, as illustrated in Figure 5.58 (b):

- *Data*: these are the observations in the dataset $\mathbf{x}$. We assume the observations come from the *data model* $X \sim \mathcal{M}(\Theta)$, where Θ are the parameters of the data model.
- *Parameters* (level 1): the "control knobs" of the data model. We assume the parameters Θ are random variables described by the prior probability distribution $\Theta \sim \mathcal{P}(A)$, where $\mathcal{P}$ is the prior distribution family, and A are its *hyperparameters*.
- *Hyperparameters* (level 2): the parameters of the prior distribution. In Figure 5.58 (b), the hyperparameters are random variables A distributed according to the *hyperprior* $A \sim \mathcal{H}(\alpha)$, where $\mathcal{H}$ is the hyperprior family, and α are its parameters.



(a) Nonhierarchical model. (b) Hierarchical model.

Figure 5.58: Examples of a nonhierarchical model (one level of parameters) and a hierarchical model (two levels of parameters).

The nonhierarchical model in Figure 5.58 (a) is a bit like an Oreo cookie, where the parameters Θ play the role of the white filling. The hierarchical model in Figure 5.58 (b) contains two layers of parameters Θ and A, like a *Double Stuf* Oreo cookie with double the filling.

We'll now look at an example of a hierarchical model to illustrate the new terminology.

Random-intercepts model

Let's look at the Bayesian *random-intercepts* model for a dataset with J groups, $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_J]$. This is a simple hierarchical model that has an intercept term and no predictors. We model the mean of each group as a group-specific intercept term M_j, which is the sum of two parts $M_j = B_0 + B_{0j}$, where B_0 is a common mean for the population as a whole, and B_{0j} is the group-specific deviation of group j from the population mean B_0. The Bayesian hierarchical model is described by the following equations:

$$
\begin{aligned}
X_j &\sim \mathcal{N}(M_j, \Sigma_X), && \text{[data model for each group]} \\
M_j &= B_0 + B_{0j}, && \text{[components of the intercept term]} \\
B_0 &\sim \mathcal{N}(\mu_{B_0}, \sigma_{B_0}), && \text{[prior for the population-level mean]} \\
B_{0j} &\sim \mathcal{N}(0, \Sigma_J), && \text{[prior for the group-specific deviations]} \\
\Sigma_X &\sim \mathcal{T}^+(\nu_{\Sigma_X}, \sigma_{\Sigma_X}), && \text{[prior for the standard deviation]} \\
\Sigma_J &\sim \text{Expon}(\lambda_{\Sigma_J}). && \text{[hyperprior for the scale of the deviations]}
\end{aligned}
$$

Sorry for all the Greek letters and fancy subscripts, but we need some way to refer to all the model's parameters and hyperparameters. Recall that capital letters refer to model parameters that we'll estimate using the MCMC inference procedure, while lowercase letters refer to the model's "control knobs," which we specify in advance.

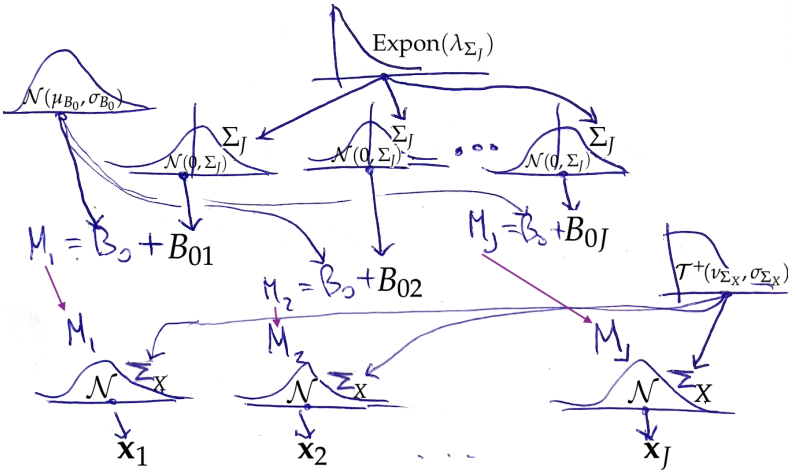


Figure 5.59: Graphical model diagram for the random-intercepts model.

The graphical model diagram in Figure 5.59 illustrates the links between the probability distributions at different levels of the model. Let's walk through the parameters and distributions in the figure step by step, starting from the data model and following the connections upward through the higher levels of the model.

Data model The data model for each group is a normal distribution. Each group is described by a separate mean parameter M_j and a common scale parameter Σ_X. The scale parameter Σ_X represents the inherent variability of individual observations within each group. We place a half-t distribution prior on the scale parameter Σ_X, setting the hyperparameters $\nu_{\Sigma_X} = 4$ and σ_{Σ_X} on the same scale as the sample standard deviation $s_{\mathbf{x}}$ to get a weakly informative prior with heavy tails.

Group-specific parameters We model the mean for group j as a sum of two terms: $M_j = B_0 + B_{0j}$. The term B_0 is the common population-level mean for all groups. The term B_{0j} is a group-specific deviation from the population-level mean B_0. We're using the notation B_0 and B_{0j} for these model parameters because, together, they form the intercept term of the model.

We assume that the group-specific deviations B_{0j} come from a zero-centred normal distribution, $B_{0j} \sim \mathcal{N}(0, \Sigma_J)$. We set the prior mean to zero without loss of generality, because the mean of B_{0j} is equivalent to the common population-level mean parameter B_0. The scale of the group-specific deviations B_{0j} is described by the standard deviation parameter Σ_J.

Population-level parameters The top level of the model hierarchy describes the characteristics of the different groups. The parameter B_0 describes a population-level mean that is common to all groups. We set a normal prior for this variable $B_0 \sim \mathcal{N}(\mu_{B_0}, \sigma_{B_0})$, choosing the value of the hyperparameter μ_{B_0} based on the sample mean $\bar{\mathbf{x}}$, and setting the scale hyperparameter σ_{B_0} to a multiple of the sample standard deviation $s_{\mathbf{x}}$.

The standard deviation parameter Σ_J describes the variability between different groups. The exponential hyperprior $\text{Expon}(\lambda_{\Sigma_J})$ is a common distribution choice. To obtain a weakly informative prior, we can set the parameter λ_{Σ_J} to be in the same ballpark as $1/s_{\mathbf{x}}$.

The levels of the hierarchy The key takeaway I want you to observe from Figure 5.59 and the model equations is that the model has multiple levels of variability:

Figure 5.60: Statisticians like priors and hyperparameters so much, so they decided to put priors on the hyperparameters!

Bayesian modelling advantage

Hierarchical models, in particular Bayesian hierarchical models, have the following benefits:

- Hierarchical models are adapted to the structure of hierarchical datasets. We obtain a model that describes the population-level variability in terms of the parameters B_0 and Σ_J, and, separately, describes the deviation of each group from the population mean through the parameters B_{0j}.
- Hierarchical models allow us to split up the total variability we observe in the dataset $\mathbf{x}$ into two parts: Σ_X describes the variability of observations within groups, while Σ_J describes the variability between groups.
- The group-specific parameters B_{0j} are "tied together" through the common hyperprior. This tie-in allows hierarchical models to share information among groups: the inferences we obtain from one group inform the inferences for the other groups. This aspect is described as "borrowing strength" across groups.
- The sharing of information enabled by the hierarchical structure is particularly important for datasets with unbalanced groups. Without the hierarchical structure, groups with few observations would produce estimates with huge uncertainty. However, in a hierarchical model, we can still obtain useful estimates since the population prior regularizes estimates.
- The population level parameters we infer from a given dataset allow us to predict the data we might observe for a new group.

Overall, hierarchical models are an important and useful part of the modern statistic toolbox, because grouped data is very common.

Hierarchical models are not unique to Bayesian statistics. Later in this section, we'll show how to define and fit a frequentist hierarchical model using the `statsmodels` library. However, the methodology and software used in Bayesian statistics makes it especially easy and straightforward to construct hierarchical models. Basically, in

End matter

Conclusion

The topics we studied in this book are the essence of statistical inference, which is the art of working backward from observed data to parameter estimates. We use probability models to write a "story" about how the data was generated (the data generating process). Based on this story, we work backward to *infer* the parameters that produced the observed data. This is the most important idea that I want you to take away from this book. Everything else is details.

Statistical inference was the common theme in all the chapters in Part 2 of the book. Each chapter discussed different ways to guess the parameters θ that generated the data $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ based on different models: frequentist models like $Y \sim \mathcal{M}(\theta)$ in Chapter 3, linear models like $Y \sim \mathcal{N}(\beta_0 + \beta_1 \cdot x, \sigma)$ in Chapter 4, and Bayesian models like $Y \sim \mathcal{M}(\Theta)$ in Chapter 5. I hope the explanations of the multiple kinds of models helped you see the commonalities and differences between them.

Three big ideas

Each chapter in Part 2 of the book corresponds to one big idea in probability theory. Let's revisit these three big ideas now in the light of what we learned about statistical inference.

Big idea 1: Sampling distributions In Section 3.1 we defined the *sampling distribution* of the estimator g as the probability distribution of the random variable $\widehat{\Theta} \stackrel{\text{def}}{=} g(\mathbf{X})$, where $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ is a random sample of size n from the population X. Knowing the sampling distribution $g(\mathbf{X})$ tells us the variability of the estimates we might expect to observe from different samples of size n taken from that population.

Recall we learned both computational and analytical methods for obtaining sampling distributions. The simplest computational ap-

proach was to use direct simulation: generate thousands of samples from the population $[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N]$, then compute the estimates from each sample $[g(\mathbf{x}_1), g(\mathbf{x}_2), g(\mathbf{x}_3), \ldots, g(\mathbf{x}_N)]$, which provide an empirical approximation to the sampling distribution $g(\mathbf{X})$.

We also learned an important analytical approximation to the sampling distribution that is specific to the sample mean estimator **mean**. The *central limit theorem* tells us that the sampling distribution of the mean for random samples of size n from the population X with mean μ_X and standard deviation σ_X is described by a normal distribution $\overline{\mathbf{X}} \sim \mathcal{N}(\mu_X, \frac{\sigma_X}{\sqrt{n}})$.

We used sampling distributions to construct confidence intervals and to compute p-values for hypothesis tests. Indeed, most of the results in Chapter 3 are obtained from calculations based on sampling distributions.

Big idea 2: Maximum likelihood estimation In Chapter 4, we obtained "best fit" model parameters by maximizing likelihood functions. For example, to model the influence of the predictor x on the outcome variable Y, we can define a generalized linear model $Y \sim \mathcal{M}(g^{-1}(\beta_0 + \beta_1 \cdot x))$, where $\mathcal{M}$ is the data model family and g^{-1} is the inverse link function. We can write the likelihood function $L(b_0, b_1)$ for this model, then use optimization to find the maximum likelihood estimates: $\widehat{\beta}_0^{\text{MLE}}, \widehat{\beta}_1^{\text{MLE}} = \text{argmax}_{b_0, b_e} L(b_0, b_e)$.

The maximum likelihood estimation approach is a general way to find the "best fit" parameters for all the models we studied in Chapter 4, but also applies more widely to fit any model. Likelihood functions also play a key role in Bayesian models.

Big idea 3: Bayesian updates In Chapter 5 we learned about the Bayesian paradigm for statistical inference, where our uncertainty about the unknown parameters is represented as a probability distribution. Bayesian statistical inference boils down to applying the Bayes update that combines likelihood function $L_{\mathbf{x}}$ with prior distribution f_Θ to obtain the posterior distribution $f_{\Theta|\mathbf{x}}$.

The Bayes update is an application of the *Bayes' rule* we learned in Chapter 2 in Part 1 of the book (see page 146 and page 210). Bayesian updates have a surprising number of applications in many domains.

I hope you will remember these big ideas and be able to spot them when they appear in your future studies. It's remarkable how many techniques in statistics and machine learning boil down to applications of these three ideas.

Ethical considerations in statistical practice

Statistics may seem like a purely technical subject like mathematics, but unlike math, doing statistics often has an immediate impact in the real world. You must therefore always consider the implications of the data collection and statistical analyses you perform. How might your statistics results affect the "population" you're studying?

The concept of *informed consent* is essential if you're collecting the data from human subjects. You need to prepare a *consent form* that clearly specifies what data you will be collecting, why you're collecting it, what you plan to do with it, and how you'll safeguard it. That part about safeguarding is extremely important. Whatever data you collect, your primary responsibility is to the individuals: you must do everything in your power to ensure *data privacy*, which means not giving the data to "business partners" without informed consent.

Always report the truth, or the closest thing to it, in your results. Don't selectively report the "good news" and hide the "bad news" results. It takes strength to report results that don't point in the direction where you wanted them to point, but truth always prevails in the long term. Remember, every lie you tell requires "maintenance" over time, since you have to remember to keep telling that lie in the future. It's like a service contract you have to maintain for the rest of your life!

Try to think about the downstream consequences of your research. Could your results be used to justify some undesirable actions against certain individuals or groups? If so, consider not publishing your results, or take appropriate measures to mitigate the negative consequences.

Stats wars

The more you learn about statistics, the more you'll be exposed to the ideological battle that exists between frequentist and Bayesian statisticians. This book doesn't take sides, which is why it includes both approaches. Both the frequentist and Bayesian perspectives provide useful techniques for statistical analysis, so you need to know both.

Ultimately, the choice of statistical methods is not as important as the *statistical thinking* that goes into each analysis. Statistical thinking is required to transform research questions into statistical questions, to collect data that can answer these questions, and to evaluate if you have a "match" between the data and the modelling assumptions you're making. You also have to think carefully about causal

Appendix A

Answers and solutions

Chapter 3 solutions

Answers to exercises

E3.7 $\bar{\mathbf{a}} \pm \widehat{\mathbf{se}}_{\bar{\mathbf{a}}} = 202.6 \pm 3.4$. **E3.17** a) $\mathbf{CI}_{\mu,0.9} = [1001.7, 1005.9]$; b) $\mathbf{CI}_{\mu,0.9} = [1001.8, 1005.9]$. **E3.18** a) $\mathbf{CI}_{\sigma^2_{\mathbf{x}},0.9} = [30.69, 65.18]$; b) $\mathbf{CI}_{\sigma^2_{\mathbf{x}},0.9} = [20, 70]$. **E3.19** a) $\mathbf{CI}_{\Delta,0.95} = [-0.1, 14.1]$; b) $\mathbf{CI}_{\Delta,0.95} = [0.093, 13.8]$. **E3.20** a) $\mathbf{CI}_{\Delta,0.8} = [1.9165, 14.7228]$; b) $\mathbf{CI}_{\Delta,0.8} = [2.7, 14]$. **E3.21** a) $\mathbf{CI}_{\mu,0.95} = [86.42, 92.78]$; b) $\mathbf{CI}_{\mu,0.95} = [86.36, 92.84]$. **E3.22** Not statistically significant difference, so we fail to reject H_0. **E3.23** Fail to reject H_0. **E3.35** $p = 0.10917$; fail to reject H_0. **E3.37** Fail to reject H_0. **E3.40** Effect size of $d = 0.85$ or larger is required for 80% power. **E3.41** The power when $n = m = 17$ is 81%. **E3.42** The power of Bob's t-test is 81%.

Solutions to selected exercises

E3.21 We have observed the sample mean $\bar{\mathbf{x}} = 89.60$ and sample standard deviation $s_{\mathbf{x}} = 12.96$ from this sample of size $n = 64$. The estimated standard error is $\widehat{\mathbf{se}}_{\bar{\mathbf{x}}} = \frac{12.96}{\sqrt{64}} = 1.62$. To answer **a)** we'll use the pivotal quantity $z = \frac{\bar{\mathbf{x}}-\mu}{\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}} \sim \mathcal{N}(0,1)$. To find the 95% confidence interval, we need to find the values z_ℓ and z_u such that $\Pr(\{z_\ell \leqslant Z \leqslant z_u\}) = 0.95$. We can use the inverse CDF of the standard normal F_Z^{-1} to find 2.5$^{\text{th}}$ percentile $z_\ell = F_Z^{-1}(0.025) = -1.96 =$ `norm.ppf(0.025)` and the 97.5$^{\text{th}}$ percentile $z_u = F_Z^{-1}(0.975) = 1.96$. The confidence interval expressed in terms of the pivotal quantity is $-1.96 \leqslant \frac{\bar{\mathbf{x}}-\mu}{\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}} \leqslant 1.96$, which is equivalent to $\bar{\mathbf{x}} - 1.96\,\widehat{\mathbf{se}}_{\bar{\mathbf{x}}} \leqslant \mu \leqslant \bar{\mathbf{x}} + 1.96\,\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}$. Plugging in the numbers we obtain $\mathbf{CI}_{\mu,0.95} = \{86.42 \leqslant \mu \leqslant 92.78\}$, or using interval notation $\mathbf{CI}_{\mu,0.95} = [86.42, 92.78]$. To answer **b)** we'll use the pivotal quantity $t = \frac{\bar{\mathbf{x}}-\mu}{\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}} \sim \mathcal{T}(n-1)$, where $\mathcal{T}(n-1)$ is Student's t-distribution with $\nu = n-1 = 63$ degrees of freedom. Using the inverse CDF of the Student's t-distribution F_T^{-1}, we find 2.5$^{\text{th}}$ percentile $t_\ell = F_{T_\nu}^{-1}(0.025) = -1.998 =$ `tdist.ppf(0.025,df=63)` and the 97.5$^{\text{th}}$ percentile $t_u = F_{T_\nu}^{-1}(0.975) = 1.998 =$ `tdist.ppf(0.975,df=63)`, which then leads us to $\mathbf{CI}_{\mu,0.95} = [\bar{\mathbf{x}} - 1.998\,\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}, \bar{\mathbf{x}} + 1.998\,\widehat{\mathbf{se}}_{\bar{\mathbf{x}}}] = [86.36, 92.84]$. Note the answers in **a)** and **b)** are very similar, since Student's t-distribution with $\nu = 63$ is very similar to the normal distribution (less need for heavy tails when sample size is large).

Appendix B

Notation

This appendix contains a summary of the math notation used in this book. The tables are provided for easy reference: whenever you encounter some math symbol whose meaning you don't know, you can look it up here to learn what it is called and what it means.

Statistics notation

Expression	Denotes
f_X	*probability mass function* of a discrete r.v. X, or *probability density function* of a continuous r.v. X.
n	sample size
$\mathbf{x} \stackrel{\text{def}}{=} (x_1, x_2, \ldots, x_n)$	a particular sample of size n from the r.v. X
$\mathbf{X} \stackrel{\text{def}}{=} (X_1, X_2, \ldots, X_n)$	random sample of size n. Each $X_i \sim f_X$
$\hat{\theta} = g(\mathbf{x})$	particular value of the estimator g computed from the sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$
$\widehat{\Theta} = g(\mathbf{X})$	sampling distribution of the estimator g computed from the random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$
$\overline{\mathbf{x}}$	sample mean
$s^2_{\mathbf{x}}$	sample variance
$s_{\mathbf{x}}$	sample standard deviation
$\mathbf{se}_g$	standard error of the estimator g
$\widehat{\mathbf{se}}_g$	estimated standard error of the estimator g
θ	a parameter of the probability distribution
$\hat{\theta}$	an estimate of the parameter θ
μ	population mean
σ^2	population variance
σ	population standard deviation
ν	degrees of freedom parameter
CV_α	cutoff value for a hypothesis testing decision

Linear models notation

Expression	Denotes
n	number of observations
$[\mathbf{x}, \mathbf{y}]$	bivariate dataset = $[(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$
(x_i, y_i)	the i^{th} observation in the dataset
$\mu_Y(x) = \beta_0 + \beta_1 x$	regression equation of the mean
$\mathcal{E} \sim \mathcal{N}(0, \sigma)$	normally-distributed error term
ε_i	a particular realization of the error term
$Y(x) \sim \beta_0 + \beta_1 x + \mathcal{E}$	linear model (simple linear regression)
β_0	intercept parameter
β_1	slope parameter
σ	standard deviation parameter
$\widehat{\beta}_0$	estimated intercept parameter (`b0` in code)
$\widehat{\beta}_1$	estimated slope parameter (`b1` in code)
$\widehat{\sigma}$	estimated standard deviation parameter
$\mu_{\widehat{Y}}(x) = \widehat{\beta}_0 + \widehat{\beta}_1 x$	estimated regression equation of the mean
$\widehat{Y}(x) \sim \widehat{\beta}_0 + \widehat{\beta}_1 x + \mathcal{N}(0, \widehat{\sigma})$	estimated linear model
$\widehat{y}_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i$	fitted values = predictions on the dataset $[\mathbf{x}, \mathbf{y}]$
$r_i = y_i - \widehat{y}_i$	residuals = prediction errors on the dataset $[\mathbf{x}, \mathbf{y}]$
x_{new}	new, previously unseen predictor value
$\widehat{y}_{\text{new}} = \widehat{\beta}_0 + \widehat{\beta}_1 x_{\text{new}}$	predicted outcome for the value x_{new}
$[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_p, \mathbf{y}]$	multivariate dataset
$(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$	the i^{th} observation in multiple regression model
x_{ik}	value of the k^{th} predictor in the i^{th} observation
β_k	the slope associated with the predictor x_k
$\mathbf{SSR}$	sum of squared residuals $\mathbf{SSR} = \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2$
$\mathbf{ESS}$	explained sum of squares $\mathbf{ESS} = \sum_{i=1}^{n} (\widehat{y}_i - \overline{\mathbf{y}})^2$
$\mathbf{TSS}$	total sum of squares $\mathbf{TSS} = \sum_{i=1}^{n} (y_i - \overline{\mathbf{y}})^2$
R^2	coefficient of determination $R^2 = \frac{\mathbf{ESS}}{\mathbf{TSS}} = 1 - \frac{\mathbf{SSR}}{\mathbf{TSS}}$
$\widehat{\mathbf{se}}_{\mu_{\widehat{Y}}}(x)$	uncertainty in the prediction of the mean μ_Y
$\mathbf{ci}_{\mu_{Y,\gamma}}(x)$	γ-confidence interval for the mean
$\widehat{\mathbf{se}}_{\widehat{y}}(x)$	uncertainty in the prediction of the value Y
$\mathbf{ci}_{Y,\gamma}(x)$	γ-confidence interval for observations

Bayesian statistics notation

Symbol	Denotes
Θ	parameters (random variable). Fixed values are denoted θ.
$f_{X\|\theta}$	population data model given fixed parameters θ
$f_{\mathbf{X}\|\theta}$	probability distribution of a random sample $\mathbf{X}$ given a fixed θ
$L_{\mathbf{x}}$	the likelihood function of θ given the data $\mathbf{x}$
f_{Θ}	prior distribution of the parameters Θ
$f_{\Theta\|\mathbf{x}}$	posterior distribution of Θ after observing data $\mathbf{x}$
$f_{X,\Theta}$	joint distribution of the data X and the parameter Θ
$f_{\mathbf{X}}$	marginal likelihood of the data $\mathbf{X}$
$\mu_{\Theta\|\mathbf{x}}$	posterior mean (expected value of $f_{\Theta\|\mathbf{x}}$)
$\mathbf{med}_{\Theta\|\mathbf{x}}$	posterior median
$\widehat{\theta}_{\text{MAP}}$	posterior mode $\widehat{\theta}_{\text{MAP}} \stackrel{\text{def}}{=} \text{argmax}_{\theta}\ f_{\Theta\|\mathbf{x}}(\theta\|\mathbf{x})$
$\mathbf{hdi}_{\theta,1-\alpha}$	$(1-\alpha)$-Bayesian highest density interval for θ

The notation we use for Bayesian statistics requires careful attention because we're working with conditional distributions and there are multiple random variables to consider. We use the subscripts to indicate which arguments of the functions are fixed (denoted by lowercase letters) and which are variable (uppercase letters).

Prior The *prior* is the distribution of the random variable Θ before observing the data $\mathbf{x}$. We use the notation f_{Θ} to describe the prior's probability density function. Note that f_{Θ} is a function $\mathbb{R} \to \mathbb{R}$, while $f_{\Theta}(\theta)$ is a number.

Probability model The probability of X given the parameter θ is described by the conditional distribution $f_{X|\theta}$, which we can also denote as the data model $\mathcal{M}(\theta)$. Note the argument that varies is X, while θ is fixed. The value $f_{X|\theta}(x|\theta)$ describes the probability density of the outcome $\{X = x\}$ when the parameter is θ. The probability of the i.i.d. random sample $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ is the n-fold product of data model $f_{\mathbf{X}|\theta}(\mathbf{x}|\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta)$.

Likelihood The likelihood function $L_{\mathbf{x}}$ is the probability of the sample $\mathbf{x}$, considered as a function the parameters θ. The likelihood function is defined as $L_{\mathbf{x}}(\theta) = \prod_{i=1}^{n} f_{X|\theta}(x_i|\theta)$. Note θ is the variable of the likelihood function, while the data sample $\mathbf{x}$ is fixed.

Posterior The *posterior* distribution $f_{\Theta|\mathbf{x}}$ describes our updated knowledge about the parameter Θ after observing the data $\mathbf{x}$. We obtain the posterior using Bayes' rule formula: $f_{\Theta|\mathbf{x}}(\theta|\mathbf{x}) = \frac{1}{C} L_{\mathbf{x}}(\theta) f_{\Theta}(\theta)$. In words, this formula tells us that the posterior is the product of the likelihood function $L_{\mathbf{x}}$ and the prior f_{Θ}.

Bibliography

[CAP+20] Kevin Cummiskey, Bryan Adams, James Pleuss, Dusty Turner, Nicholas Clark, and Krista Watts. Causal inference in introductory statistics courses. *Journal of Statistics Education*, 28(1):2–8, 2020.

[CFP22] Carlos Cinelli, Andrew Forney, and Judea Pearl. A crash course in good and bad controls. *Sociological Methods & Research*, page 00491241221099552, 2022.

[CPK+22] Tomás Capretto, Camen Piho, Ravin Kumar, Jacob Westfall, Tal Yarkoni, and Osvaldo A Martin. Bambi: A simple interface for fitting bayesian linear models in python. *Journal of Statistical Software*, 103(15):1–29, 2022.

[DLL17] Marie Delacre, Daniël Lakens, and Christophe Leys. Why psychologists should by default use welch's t-test instead of student's t-test. *International Review of Social Psychology*, 30(1):92–101, 2017. `https://pure.tue.nl/ws/portalfiles/portal/80459772/82_534_3_PB.pdf`.

[DSC+20] Alexander Decruyenaere, Johan Steen, Kirsten Colpaert, Dominique D Benoit, Johan Decruyenaere, and Stijn Vansteelandt. The obesity paradox in critically ill patients: a causal learning approach to a casual finding. *Critical Care*, 24:1–11, 2020.

[DVdS17] Sarah Depaoli and Rens Van de Schoot. Improving transparency and replication in bayesian statistics: The wambs-checklist. *Psychological methods*, 22(2):240, 2017.

[Fey98] Richard P Feynman. Cargo cult science. In *The art and science of analog circuit design*, pages 55–61. Elsevier, 1998. See `https://calteches.library.caltech.edu/51/2/CargoCult.htm`.

[Fis25] R. A. Fisher. *Statistical methods for research workers*. Oliver and Boyd, 1925.

[Fis35] R. A. Fisher. *The Design of Experiments*. Oliver and Boyd, 1935.

[Gel11] Andrew Gelman. The statistical significance filter. `https://tinyurl.com/stat-sig-filter`, 2011. Accessed: 2024-06-25.

[GH06] Andrew Gelman and Jennifer Hill. Data analysis using regression and multilevel/hierarchical models. 2006.

[GVS+20] Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. Bayesian workflow. *arXiv preprint arXiv:2011.01808*, 2020.

[HDSH06] Sonia Hernández-Díaz, Enrique F Schisterman, and Miguel A Hernán. The birth weight "paradox" uncovered? *American journal of epidemiology*, 164(11):1115–1120, 2006.

[Jef98] Harold Jeffreys. *The theory of probability*. OuP Oxford, 1998.

[Kan21] Hyun Kang. Sample size determination and power analysis using the G*Power software. *Journal of educational evaluation for health professions*, 18, 2021.

[Kru13] John K Kruschke. Bayesian estimation supersedes the t test. *Journal of Experimental Psychology: General*, 142(2):573, 2013.

[Kru21] John K Kruschke. Bayesian analysis reporting guidelines. *Nature human behaviour*, 5(10):1282–1291, 2021.

[Lak22] Daniël Lakens. Improving your statistical inferences (textbook). 2022. `https://lakens.github.io/statistical_inferences/`.

[MAB+20] Riana Minocher, Silke Atmaca, Claudia Bavero, Richard McElreath, and Bret Beheim. Reproducibility improves exponentially over 63 years of social learning research. 2020.

[R+06] Bernard A Rosner et al. *Fundamentals of biostatistics*, volume 6. Thomson-Brooks/Cole Belmont, CA, 2006.

[Rit20] Stuart Ritchie. *Science fictions: How fraud, bias, negligence, and hype undermine the search for truth*. Metropolitan Books, 2020.

[Stu08] Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

[TMR+85] Ira B Tager, Alvaro Muñoz, Bernard Rosner, Scott T Weiss, Vincent Carey, and Frank E Speizer. Effect of cigarette smoking on the pulmonary function of children and adolescents. *American review of respiratory disease*, 131(5):752–759, 1985.

[TWM+83] Ira B Tager, Scott T Weiss, Alvaro Muñoz, Bernard Rosner, and Frank E Speizer. Longitudinal study of the effects of maternal smoking on pulmonary function in children. *New England Journal of Medicine*, 309(12):699–703, 1983.

[WWBV11] Eric-Jan Wagenmakers, Ruud Wetzels, Denny Borsboom, and Han LJ VanDerMaas. Why psychologists must change the way they analyze their data: the case of psi: comment on bem (2011). 2011.

No Bullshit Guide to Statistics Part 2: Statistical Inference by Ivan Savov (Minireference Publishing, v0.91, October 2025, ISBN 9781777438937) is available as a digital download from gumroad: gum.co/noBSstats. For more info, visit the book's website: noBSstats.com.