R / PRACTICAL HANDBOOK

آموزش نرم‌افزار R

آموزش صفر تا صد

از اولین خط کد تا تحلیل مستقل داده و گزارش نهایی



کدنویسی • داده • آمار • نمودار • پروژه

راهنمای فارسی با مثال‌های قابل اجرا، تمرین و پاسخ
همراه با دادهٔ تمرینی و پروژهٔ کامل تحلیل فروش

ویرایش نخست • شهریور ۱۴۰۵
مناسب خودآموزی و استفاده در کنار کلاس

فهرست تعاملی • بخش 1 از ۲

مسیر یادگیری شما

برای رفتن به هر درس، روی عنوان آن کلیک کنید. شماره‌ها با صفحات PDF یکسان‌اند.

از این کتاب چگونه استفاده کنید؟	4
R چیست و با RStudio چه فرقی دارد؟	5
نصب R و RStudio؛ قدم‌به‌قدم	6
شناخت محیط و نوشتن اولین اسکریپت	7
پروژه، مسیر فایل و ذخیرهٔ درست	8
بسته‌ها، راهنما و خطای نصب	9
اشیا، محاسبات و عملگرها	10
بردار و انتخاب عناصر	11
نوع داده و عامل دسته‌ای	12
جدول، لیست و ماتریس	13
شرط، حلقه و تابع اختصاصی	14
مقدار گمشده، صفر و خطای محاسبه	15
ورود CSV و کنترل ستون‌ها	16
Excel، اعداد فارسی و واحد پول	17
بازرسی و پاک‌سازی با R پایه	18
انتخاب، فیلتر و ساخت ستون با dplyr	19
گروه‌بندی و خلاصه‌سازی	20
متن، تاریخ و مرتب‌سازی زمانی	21
تبدیل جدول عریض و بلند	22
اتصال جدول‌ها بدون تکثیر اشتباه	23
آمار توصیفی؛ عددها چه می‌گویند؟	24
انتخاب نمودار مناسب	25
نمودارهای پایهٔ R	26
ساخت نمودار لایه‌ای با ggplot2	27
ذخیرهٔ نمودار و خروجی فارسی	28
نمونه‌گیری، احتمال و فاصلهٔ اطمینان	29
انتخاب آزمون و خواندن p-value	30
آزمون t، مقایسهٔ زوجی و ویلکاکسون	31
دادهٔ دسته‌ای و مقایسهٔ چند گروه	32
همبستگی و رابطهٔ غیرعلّی	33

فهرست تعاملی • بخش 2 از ۲

مسیر یادگیری شما

برای رفتن به هر درس، روی عنوان آن کلیک کنید. شماره‌ها با صفحات PDF یکسان‌اند.

رگرسیون خطی و تفسیر خروجی	34
بررسی فرض‌ها و نقاط اثرگذار	35
آموزش، آزمون و ارزیابی پیش‌بینی	36
رگرسیون لجستیک برای پاسخ صفر و یک	37
ماتریس خطا و معیارهای طبقه‌بندی	38
خوشه‌بندی و کاهش بُعد	39
سری زمانی و یک پیش‌بینی پایه	40
ساخت گزارش خودکار با Quarto	41
اولین اپ تعاملی با Shiny	42
پایگاه داده و داده‌های بزرگ‌تر	43
خطایابی و بهبود سرعت	44
نسخه‌ها، renv و اجرای دوباره	45
پروژهٔ کامل؛ تحلیل فروش ساختگی	46
پروژه؛ اجرای کنترل کیفیت	47
پروژه؛ ساخت شاخص‌های مالی	48
پروژه؛ مقایسهٔ شهرها و ماه‌ها	49
پروژه؛ نمودار و گزارش نهایی	50
تمرین‌های پایه؛ قبل از دیدن پاسخ	51
تمرین‌های داده و نمودار	52
تمرین‌های مدل‌سازی و پروژه	53
پاسخ تمرین‌های پایه	54
پاسخ تمرین‌های داده	55
پاسخ مدل‌سازی و معیار قبولی پروژه	56
برگهٔ تقلب؛ مبانی و داده	57
برگهٔ تقلب؛ آمار و خروجی	58
خطاهای رایج؛ از پیام به راه‌حل	59
برنامهٔ مطالعه و قدم بعد	60
منابع رسمی؛ مبانی و کار با داده	61
منابع رسمی؛ نمودار، مدل و گزارش	62

راهنمای مطالعه

از این کتاب چگونه استفاده کنید؟

این آموزش برای کسی نوشته شده که می‌خواهد از نقطهٔ شروع به اجرای مستقل یک تحلیل داده برسد: سؤال را مشخص کند، داده را وارد و پاک‌سازی کند، نمودار بسازد، روش آماری مناسب انتخاب کند و نتیجه را گزارش دهد. نیاز نیست از قبل برنامه‌نویس باشید؛ آشنایی با فایل و پوشه و مفهوم میانگین کافی است.

منظور از صفر تا صد در این جزوه

مسیر اصلی کار با R از نصب تا تحویل یک پروژهٔ قابل‌بازتولید پوشش داده می‌شود. مدل‌سازی، یادگیری ماشین و ساخت اپ نیز معرفی می‌شوند؛ تسلط تخصصی بر هرکدام تمرین و مطالعهٔ جداگانه می‌خواهد.

بخش	چه چیزی تحویل می‌گیرید؟
شروع و مبانی	اسکریپت اجراشدنی و درک ساختارهای داده
آماده‌سازی و نمودار	جدول تمیز و شکل قابل ارائه
آمار و مدل‌سازی	تحلیل همراه با محدودیت و ارزیابی
پروژه و تمرین	گزارش فروش، کدها، داده و پاسخ تمرین

هر درس را با این ترتیب بخوانید: توضیح، اجرای کد، تغییر یک ورودی، بررسی خروجی. بلوک‌های یک درس را از بالا به پایین اجرا کنید. علامت‌های ابتدای Console مانند > و + جزو کدی که باید تایپ کنید نیستند.

نمونه‌های پایه به بستهٔ اضافی نیاز ندارند. برای مسیرهای تکمیلی، نصب بسته‌ها اینترنت می‌خواهد. دادهٔ فروشِ همراه کاملاً ساختگی است و همهٔ مبالغ آن تومان‌اند. کدها از عملگر |> استفاده می‌کنند و R نسخهٔ ۴٫۱ یا جدیدتر لازم دارند.

فایل‌های همراه

فایل R-course.Rproj را باز کنید. کدهای مستقل در scripts، داده در data و نتایج در outputs قرار می‌گیرند. نسخهٔ PDF متنی است و کدهای آن را می‌توان انتخاب و کپی کرد.

بخش اول • شروع

R چیست و با RStudio چه فرقی دارد؟

R یک زبان برنامه‌نویسی و محیط محاسبات آماری است. با آن می‌توانید داده‌های جدولی را بخوانید، محاسبه و مدل‌سازی کنید و خروجی تصویری یا گزارشی بسازید. مزیت اصلی برای تحلیل‌گر این است که مراحل کار به کد تبدیل می‌شوند؛ با دادهٔ جدید می‌توان همان مراحل را دوباره اجرا کرد.

ابزار	نقش آن در کار شما
R	موتور اجرای محاسبات و کدها
RStudio	محیطی برای نوشتن کد، دیدن متغیرها، نمودارها و راهنما
Package	مجموعه‌ای از توابع و امکانات افزوده
CRAN	یکی از مخازن اصلی دریافت R و بسته‌های آن

RStudio خودِ زبان R نیست. نصب RStudio به‌تنهایی معمولاً جای نصب R را نمی‌گیرد. می‌توانید R را بدون RStudio نیز اجرا کنید، ولی در این آموزش از RStudio برای راحتی کار استفاده می‌کنیم.

چه نوع کارهایی با R مناسب است؟

- خلاصه‌سازی فروش، پرسش‌نامه، دادهٔ آزمایشگاهی یا اطلاعات آموزشی.
- رسم نمودارهای دقیق، آزمون آماری و ساخت مدل پیش‌بینی.
- تولید گزارش تکرارشونده و اپ تعاملی برای بررسی داده.

برای شروع روی یک سؤال کوچک تمرکز کنید؛ مثلاً «در کدام شهر درآمد سفارش‌های تکمیل‌شده بیشتر است؟». انتخاب ابزار زمانی مفید است که معنای ردیف‌ها، واحدها و تصمیم موردنظر روشن باشد.

> مطالعهٔ تکمیلی: معرفی و راهنماهای رسمی R

بخش اول • شروع

نصب R و RStudio؛ قدم‌به‌قدم

- ابتدا سایت CRAN را باز کنید و بخش دانلود متناسب با Windows، macOS یا Linux را انتخاب کنید.
- در ویندوز از شاخهٔ base، نصب‌کنندهٔ نسخهٔ جاری را دریافت و اجرا کنید. تنظیمات پیش‌فرض برای شروع مناسب‌اند.
- در مک، فایل متناسب با پردازندهٔ دستگاه و نسخهٔ سیستم‌عامل را انتخاب کنید؛ راهنمای همان صفحه ملاک سازگاری است.
- در لینوکس، دستور مخصوص توزیع و نسخهٔ سیستم‌عامل را از راهنمای CRAN بخوانید. دستور توزیع دیگری را کپی نکنید.
- پس از نصب R، نسخهٔ رایگان RStudio Desktop را از سایت Posit بگیرید و نصب کنید.

آزمون نصب

```
R.version.string
2 + 3
sessionInfo()
```

باید نسخهٔ R را ببینید و نتیجهٔ جمع ۵ باشد. اگر RStudio پیام پیدا نشدن R می‌دهد، بسته‌شدن و بازکردن مجدد برنامه و بررسی مسیر نصب R را امتحان کنید. نصب نسخه‌ای که سیستم‌عامل شما پشتیبانی نمی‌کند با تغییر تنظیمات Console حل نمی‌شود.

ابزارهای ساخت بسته

برای بسیاری از بسته‌ها فایل آمادهٔ نصب وجود دارد. Rtools در ویندوز و ابزارهای توسعه در مک یا لینوکس زمانی لازم می‌شوند که نصب بسته به ساخت از کد منبع نیاز داشته باشد؛ برای نوشتن اولین برنامه الزامی نیستند.

مطالعهٔ تکمیلی: دریافت R از CRAN

مطالعهٔ تکمیلی: دانلودهای رسمی Posit

بخش اول • شروع

شناخت محیط و نوشتن اولین اسکریپت

بخش محیط	کاربرد
Source	نوشتن و ذخیرهٔ کد در فایل با پسوند .R
Console	اجرای دستورها و دیدن پیام‌ها و خطاها
Environment	دیدن اشیای موجود در حافظهٔ نشست فعلی
Files / Plots / Help	فایل‌ها، نمودارها و مستندات توابع

از مسیر File → New File → R Script یک فایل بسازید و آن را با نام 01_start.R ذخیره کنید. در ویندوز و لینوکس، معمولاً Ctrl+Enter خط یا انتخاب فعلی را اجرا می‌کند؛ در مک از Cmd+Enter استفاده کنید. دکمهٔ Source همهٔ فایل را اجرا می‌کند.

```
scores <- c(12, 16, 18, 14)
average_score <- mean(scores)
print(average_score)
length(scores)
```

در این مثال scores چهار نمره را نگه می‌دارد، میانگین ۱۵ است و تعداد عناصر ۴. علامت <- یعنی نتیجهٔ سمت راست را در نام سمت چپ ذخیره کن. اگر پس از اجرای کل اسکریپت خروجی موردنظر دیده نشد، از print() استفاده کنید.

اگر علامت + دیدید

Console منتظر ادامهٔ دستور است؛ معمولاً پرانتز، آکولاد یا کوتیشن بسته نشده است. با Esc اجرای نیمه‌تمام را لغو کنید و خط اصلی را اصلاح کنید.

مطالعهٔ تکمیلی: راهنمای محیط RStudio

بخش اول • شروع

پروژه، مسیر فایل و ذخیرهٔ درست

یک پروژه را به یک پوشهٔ مستقل تبدیل کنید. از File → New Project یک پروژهٔ جدید بسازید؛ برای بستهٔ همراه، فایل R-course.Rproj آماده است. فایل‌های ورودی، کد و خروجی را از هم جدا نگه دارید تا دادهٔ خام هنگام تحلیل تغییر نکند.

مسیر در پروژه	محتوا
data/sales_raw.csv	فایل خام و دست‌نخورده
scripts/	مراحل آماده‌سازی و تحلیل
outputs/	جدول‌ها، گزارش کیفیت و نمودارهای خروجی
report/	متن و کد گزارش Quarto

```
getwd()
list.files()
file.exists("data/sales_raw.csv")
file.path("data", "sales_raw.csv")
dir.create("outputs", showWarnings = FALSE)
```

مسیر نسبی از پوشهٔ کاری فعلی شروع می‌شود. باز کردن پروژه کمک می‌کند مسیرها در کامپیوترهای مختلف یکسان بمانند. برای مسیرهای ویندوز از / یا file.path() استفاده کنید. مسیر کاربر دیگری مانند C:/Users/... را داخل تحلیل ثابت نکنید.

نشست تمیز، نتیجهٔ قابل‌اعتماد

در تنظیمات عمومی RStudio، بازیابی خودکار .RData را غیرفعال و ذخیرهٔ Workspace را روی Never قرار دهید. اشیای لازم باید با اجرای کد ساخته شوند. پیش از تحویل، R را Restart کنید و کل اسکریپت را از اول اجرا کنید. ذخیرهٔ اسکریپت با UTF-8 برای متن فارسی مناسب است.

بخش اول • شروع

بسته‌ها، راهنما و خطای نصب

نصب بسته معمولاً یک‌بار برای یک کتابخانه و نسخهٔ R انجام می‌شود؛ بارگذاری آن در هر نشست جدید لازم است. به‌روزرسانی R یا استفاده از پروژهٔ مستقل ممکن است نصب مجدد بسته‌ها را لازم کند.

```
install.packages("ggplot2",
                 repos = "https://cloud.r-project.org")
library(ggplot2)
packageVersion("ggplot2")

?mean
help("read.csv")
args(mean)
example(mean)
```

تابع install.packages() فایل بسته را دریافت و نصب می‌کند؛ library() توابع آن را به نشست می‌آورد. شکل ggplot2::ggplot() تابع را با نام بسته مشخص می‌کند و برای جلوگیری از اشتباه بین توابع هم‌نام مفید است.

پیام یا وضعیت	اولین بررسی
there is no package called ...	نام بسته و موفقیت مرحلهٔ نصب
package is not available	نسخهٔ R، مخزن و سازگاری اعلام‌شدهٔ بسته
خطای دسترسی به مسیر نصب	استفاده از کتابخانهٔ شخصی پیشنهادی R
دانلود ناموفق	اینترنت، نشانی مخزن و پیام دقیق اتصال

ترتیب رفع مشکل

اولین خطای واقعی را بخوانید؛ خطای آخر ممکن است فقط پیام شکست یک وابستگی باشد. بسته‌های موردنیاز این جزوه در scripts/00_install_optional.R فهرست شده‌اند. همهٔ آن‌ها برای پروژهٔ پایه لازم نیستند.

بخش دوم • مبانی زبان

اشیا، محاسبات و عملگرها

در R تقریباً هر نتیجه در قالب یک شیء قابل نگهداری است: عدد، متن، جدول یا مدل. نام‌گذاری روشن مثل total_sales از نام‌هایی مانند x1 خواناتر است. R به حروف بزرگ و کوچک حساس است؛ sales و Sales دو نام متفاوت‌اند.

```
price <- 200000
quantity <- 3
discount <- 0.10
net <- price * quantity * (1 - discount)
net
# Expected: 540000

7 %% 3
7 %/% 3
2^4
```

نماد	معنا
+ - * / ^	جمع، تفریق، ضرب، تقسیم و توان
%% و %/%	باقی‌مانده و خارج‌قسمت صحیح
== و !=	برابر بودن و نابرابر بودن
> >= < <=	مقایسهٔ عددی
! | &	و، یا و نقیض برای عناصر بردار

برای مقایسه از == استفاده کنید؛ <- برای انتساب است. عملگرهای && و || برای شرط‌های تک‌مقداری و ارزیابی کوتاه مناسب‌اند، نه فیلتر سطرهای جدول. از نام‌های c، mean، T و F برای متغیرهای خود استفاده نکنید.

تفاوت کوچک ولی مهم

برای اعشار در کد از نقطه و برای عددها از رقم انگلیسی استفاده کنید: 0.15. عدد ۱۵٪ در محاسبهٔ تخفیف باید به صورت 0.15 وارد شود.

بخش دوم • مبانی زبان

بردار و انتخاب عناصر

بردار مجموعه‌ای از عناصر هم‌نوع است. تابع c() عناصر را کنار هم می‌گذارد. شماره‌گذاری در R از ۱ شروع می‌شود؛ عنصر اول x[1] است. انتخاب می‌تواند با شماره، نام یا شرط منطقی انجام شود.

```
x <- c(10, 20, 30, 40, 50)
x[1]
x[c(2, 4)]
x[-1]
x[x >= 30]
x[x >= 20 & x <= 40]

names(x) <- c("a", "b", "c", "d", "e")
x["c"]
seq(0, 1, by = 0.25)
rep("R", 3)
```

x[-1] همهٔ عناصر به‌جز اولی را برمی‌گرداند. اندیس منفی را با اندیس مثبت ترکیب نکنید. length(x) طول بردار را می‌دهد؛ sum(x) جمع و mean(x) میانگین را محاسبه می‌کند.

محاسبهٔ برداری

```
prices <- c(100, 200, 300)
prices * 1.1
# 110 220 330
c(1, 2, 3, 4) + c(10, 20)
# 11 22 13 24
```

بازیافت عناصر

R در برخی محاسبات بردار کوتاه‌تر را تکرار می‌کند. اگر طول‌ها سازگار باشند ممکن است هیچ هشداری نبینید؛ بنابراین قبل از محاسبهٔ سطربه‌سطر، برابر بودن طول‌ها و ترتیب سطرها را بررسی کنید.

بخش دوم • مبانی زبان

نوع داده و عامل دسته‌ای

نوع	مثال و کاربرد
numeric / double	مقادیر اعشاری مانند وزن یا قیمت
integer	عدد صحیح مانند 3L
character	متن مانند "Tehran"
logical	دو مقدار TRUE و FALSE
factor	دسته‌ها با مجموعهٔ سطوح مشخص
Date	تاریخ با قابلیت محاسبه و مرتب‌سازی

```
x <- c(10, 20, 30)
class(x)
typeof(x)
is.numeric(x)

level <- factor(c("low", "high", "medium"),
                levels = c("low", "medium", "high"),
                ordered = TRUE)
table(level)
```

تابع class() کلاس شیء و typeof() نوع ذخیره‌سازی داخلی را نشان می‌دهد. یک بردار اتمی هم‌نوع است؛ c(1, "2") به بردار متنی تبدیل می‌شود. قبل از محاسبه روی ستونی که از Excel آمده، نوع آن را ببینید.

```
f <- factor(c("10", "20", "10"))
as.numeric(f)
# Internal level codes, not the original numbers
as.numeric(as.character(f))
```

عامل را مستقیم به عدد تبدیل نکنید

خروجی as.numeric روی factor معمولاً کد سطح‌هاست. اگر برچسب‌ها واقعاً عدد هستند، ابتدا آن‌ها را به متن و سپس به عدد تبدیل کنید. برای دسته‌هایی مثل شهر، تبدیل عددی معنای تحلیلی ندارد.

بخش دوم • مبانی زبان

جدول، لیست و ماتریس

برای تحلیل جدولی بیشتر با data.frame یا نوع پیشرفته‌تر آن، tibble، کار می‌کنید. هر ستون یک متغیر و هر ردیف یک مشاهده است. ستون‌های جدول می‌توانند نوع‌های متفاوت داشته باشند، ولی طول آن‌ها باید با تعداد ردیف‌ها سازگار باشد.

```
students <- data.frame(
  name = c("Sara", "Ali", "Nima"),
  score = c(18, 14, 16),
  city = c("Tehran", "Shiraz", "Tehran")
)
students$score
students[1:2, c("name", "score")]
students[students$score >= 16, , drop = FALSE]
students$passed <- students$score >= 15
str(students)
```

در table[row, column]، بخش اول برای ردیف و بخش دوم برای ستون است. استفاده از drop = FALSE کمک می‌کند نتیجه حتی با انتخاب یک ستون همچنان جدول باقی بماند.

```
m <- matrix(1:6, nrow = 2)
m[1, 2]
result <- list(data = students, average = mean(students$score))
result[["average"]]
result["average"]
```

ماتریس یک آرایهٔ دوبعدی هم‌نوع است و در محاسبات عددی کاربرد دارد. لیست می‌تواند جدول، عدد و مدل را هم‌زمان نگه دارد. [[ ]] خودِ عضو و [ ] یک زیرلیست را برمی‌گرداند. برای تحلیل‌های روزمره ابتدا روی جدول مسلط شوید.

بخش دوم • مبانی زبان

شرط، حلقه و تابع اختصاصی

وقتی یک محاسبه را چندبار تکرار می‌کنید، تابع بنویسید. تابع باید ورودی روشن، خروجی مشخص و رفتار قابل‌پیش‌بینی داشته باشد. شرط if یک مقدار منطقی می‌خواهد؛ برای برچسب‌زدن به تک‌تک عناصر می‌توانید از ifelse() استفاده کنید.

```
net_price <- function(price, discount = 0) {
  stopifnot(is.numeric(price), all(is.finite(price)),
            all(price >= 0), length(discount) == 1,
            is.finite(discount), discount >= 0, discount <= 1)
  price * (1 - discount)
}
net_price(c(100000, 200000), 0.1)

scores <- c(12, 18, 16)
ifelse(scores >= 15, "pass", "review")
```

حلقهٔ خوانا و از پیش تخصیص‌یافته

```
x <- c(2, 4, 6)
out <- numeric(length(x))
for (i in seq_along(x)) {
  out[i] <- x[i]^2
}
out
vapply(list(a = 1:3, b = 4:6), mean, numeric(1))
```

برای ورودی خالی، seq_along(x) از الگوی 1:length(x) امن‌تر است. vapply() نوع خروجی مورد انتظار را بررسی می‌کند. وجود حلقه اشتباه نیست؛ ابتدا کد درست و قابل‌فهم بنویسید و تنها در صورت کند بودن واقعی آن را بهینه کنید.

خروجی تابع

آخرین عبارت ارزیابی‌شده به‌طور خودکار برگردانده می‌شود. استفاده از return نیز ممکن است. متغیرهای داخل تابع معمولاً محلی‌اند؛ وابستگی پنهان به اشیای بیرون تابع را کم کنید.

بخش سوم • آماده‌سازی داده

مقدار گمشده، صفر و خطای محاسبه

مقدار گمشده یعنی اطلاعات نداریم؛ صفر یعنی مقدار اندازه‌گیری‌شده صفر است. جایگزینی همهٔ مقدارهای گمشده با صفر، میانگین و نسبت‌ها را تغییر می‌دهد. ابتدا علت گم‌شدن و سهم آن را بررسی کنید.

```
x <- c(10, 20, NA, 40)
is.na(x)
mean(x)
mean(x, na.rm = TRUE)
sum(is.na(x))
x[!is.na(x) & x > 15]

is.nan(0 / 0)
is.infinite(1 / 0)
is.finite(c(1, Inf, NA))
```

مقدار	برداشت درست
NA	مقدار ناموجود
NaN	نتیجهٔ عددی تعریف‌نشده؛ مانند صفر تقسیم بر صفر
Inf	بی‌نهایت در محاسبه
NULL	نبود یک شیء یا مؤلفه؛ با NA یکسان نیست

برای شناسایی گمشده از is.na() استفاده کنید؛ مقایسهٔ x == NA درست نیست. گزینهٔ na.rm = TRUE فقط مقادیر گمشده را از همان محاسبه کنار می‌گذارد و دادهٔ اصلی را اصلاح نمی‌کند.

مخرج را گزارش کنید

اگر از ۱۰۰ ردیف فقط ۸۰ ردیف در محاسبه حضور دارند، نتیجه مربوط به همان ۸۰ مورد است. میانگین ستونی که همهٔ مقادیرش گمشده‌اند، با حذف NA همچنان مقدار قابل‌تفسیر تولید نمی‌کند.

بخش سوم • آماده‌سازی داده

ورود CSV و کنترل ستون‌ها

پس از باز کردن پروژهٔ همراه، فایل CSV را بخوانید. انتخاب جداکننده، نوع ستون و کدگذاری به اندازهٔ خود دستور مهم است. اگر همهٔ داده‌ها در یک ستون افتادند، ابتدا جداکنندهٔ فایل را بررسی کنید.

```
sales <- read.csv(
  "data/sales_raw.csv",
  fileEncoding = "UTF-8",
  na.strings = c("", "NA"),
  stringsAsFactors = FALSE
)
head(sales)
dim(sales)
str(sales)
colSums(is.na(sales))
```

روش جایگزین با readr

```
sales2 <- readr::read_csv(
  "data/sales_raw.csv",
  na = c("", "NA"),
  col_types = readr::cols(
    order_id = readr::col_character(),
    date = readr::col_date(),
    .default = readr::col_guess()
  )
)
readr::problems(sales2)
```

شناسه را متن نگه دارید تا صفر ابتدای آن حذف نشود. در روش دوم، problems() خطاهای تبدیل را نشان می‌دهد. حدس نوع ستون برای شروع مفید است، ولی برای تحلیل تکرارشونده بهتر است نوع ستون‌های مهم را صریح تعیین کنید.

> مطالعهٔ تکمیلی: مستندات readr

بخش سوم • آماده‌سازی داده

Excel، اعداد فارسی و واحد پول

```
readxl::excel_sheets("data/report.xlsx")
df <- readxl::read_excel("data/report.xlsx", sheet = 1)
str(df)
writexl::write_xlsx(df, "outputs/report_copy.xlsx")
```

کد بالا الگوی کار با فایل Excel خودتان است؛ report.xlsx در دادهٔ همراه وجود ندارد. مسیر و نام شیت را مطابق فایل واقعی تغییر دهید. سلول‌های ادغام‌شده، عنوان‌های چندردیفی و جمع کل انتهای جدول را پیش از تحلیل مدیریت کنید.

تبدیل نمونهٔ عدد فارسی به عدد قابل‌محاسبه

رشته‌های زیر نمایش یونیکدِ ۱٬۲۵۰، ۲٬۴۰۰ و ۳٫۵ هستند؛ R کدهای \u را هنگام اجرا به نویسهٔ اصلی تبدیل می‌کند.

```
txt <- c("\u06F1\u066C\u06F2\u06F5\u06F0",
         "\u06F2\u066C\u06F4\u06F0\u06F0", "\u06F3\u066B\u06F5")
fa_digits <- intToUtf8(0x06F0:0x06F9)
ar_digits <- intToUtf8(0x0660:0x0669)
latin <- chartr(fa_digits, "0123456789", txt)
latin <- chartr(ar_digits, "0123456789", latin)
latin <- gsub("\u066C", "", latin, fixed = TRUE)
latin <- gsub("\u066B", ".", latin, fixed = TRUE)
as.numeric(latin)
```

این تبدیل برای قالب مشخص همین مثال است. علامت منفی، فاصله، درصد، نماد پول و جداکننده‌های دیگر به بررسی جداگانه نیاز دارند. حذف بی‌قاعدهٔ همهٔ نشانه‌ها ممکن است ۳٫۵ را به ۳۵ تبدیل کند. بعد از تبدیل، مقدارهای تازهٔ NA را بررسی کنید.

تومان و ریال

واحد همهٔ ستون‌های مالی را در فرهنگ داده بنویسید. فقط وقتی داده واقعاً ریال است آن را بر ۱۰ تقسیم کنید. در پروژهٔ این کتاب قیمت‌ها از ابتدا تومان‌اند.

مطالعهٔ تکمیلی: راهنمای رسمی readxl

بخش سوم • آماده‌سازی داده

بازرسی و پاک‌سازی با R پایه

پیش از حذف یا تغییر داده، معنای هر ردیف را مشخص کنید. تکراری‌بودن شناسه همیشه خطا نیست؛ ممکن است هر سفارش چند قلم داشته باشد. در پروژهٔ ما هر ردیف یک سفارش تک‌دسته‌ای است و شناسه باید یکتا باشد.

```
sales <- read.csv("data/sales_raw.csv",
                  na.strings = c("", "NA"))
sales$city <- trimws(sales$city)
exact_duplicate <- duplicated(sales)
sales <- sales[!exact_duplicate, ]
anyDuplicated(sales$order_id)
summary(sales)

bad <- is.na(sales$quantity) | sales$quantity <= 0 |
      is.na(sales$unit_price) | sales$unit_price < 0
rejected <- sales[bad, ]
clean <- sales[!bad, ]
```

duplicated(sales) ردیف‌های کاملاً یکسان را علامت می‌زند. اگر دو ردیف یک شناسه ولی مبلغ متفاوت دارند، انتخاب بی‌دلیل اولین ردیف راه‌حل نیست؛ باید قاعدهٔ نسخهٔ معتبر یا منبع اختلاف معلوم شود.

سه خروجی لازم از پاک‌سازی

- جدول قابل‌تحلیل با قواعد روشن و نوع ستون درست.
- ردیف‌های کنارگذاشته‌شده برای بررسی و اصلاح در منبع.
- گزارش تعداد ردیف‌ها قبل و بعد از هر مرحله.

> **حذف خودکار ممنوع نیست؛ حذف بی‌دلیل مشکل است**
>
> برای خطای روشن مانند تعداد منفی، قاعده تعیین کنید. برای مقدار پرت واقعی مانند یک خرید بسیار بزرگ، ابتدا منشأ را بررسی کنید؛ بزرگ‌بودن به‌تنهایی دلیل حذف نیست.

بخش سوم • آماده‌سازی داده

انتخاب، فیلتر و ساخت ستون با dplyr

در کدهای dplyr، هر فعل یک کار مشخص انجام می‌دهد. عملگر |> خروجی سمت چپ را به ورودی تابع سمت راست می‌دهد؛ بنابراین زنجیره را از بالا به پایین بخوانید. ابتدا بسته را بارگذاری کنید.

```
library(dplyr)
sales <- read.csv("data/sales_raw.csv",
                  na.strings = c("", "NA"))
result <- sales |>
  filter(status == "completed", !is.na(quantity),
         quantity > 0, unit_price >= 0) |>
  mutate(net = quantity * unit_price * (1 - discount)) |>
  select(order_id, city, category, net) |>
  arrange(desc(net))
head(result)
```

تابع	عمل
select()	انتخاب ستون‌ها
filter()	انتخاب ردیف‌های مطابق شرط
mutate()	ایجاد یا تغییر ستون
arrange()	مرتب‌کردن ردیف‌ها
rename(new = old)	تغییر نام ستون
distinct()	نگه‌داشتن ترکیب‌های یکتا

در این درس هدف شناخت فعل‌هاست؛ پروژهٔ نهایی کنترل کامل تکرار و اعتبار داده را نیز انجام می‌دهد. filter() فقط ردیف‌هایی را نگه می‌دارد که شرطشان TRUE باشد؛ ردیفی با نتیجهٔ NA کنار گذاشته می‌شود.

مطالعهٔ تکمیلی: فهرست توابع dplyr

بخش سوم • آماده‌سازی داده

گروه‌بندی و خلاصه‌سازی

سؤال «فروش کل هر شهر چقدر است؟» به گروه‌بندی نیاز دارد. سؤال «میانگین قیمت هر قلم چقدر است؟» ممکن است میانگین وزنی بخواهد. ابتدا تعریف شاخص را بنویسید؛ نام یک تابع نمی‌تواند ابهام شاخص را حل کند.

```
library(dplyr)
source("scripts/02_clean_base.R")
by_city <- orders |>
  group_by(city) |>
  summarise(
    orders_n = n(),
    revenue = sum(net),
    average_order = mean(net),
    .groups = "drop"
  ) |>
  arrange(desc(revenue))
by_city
```

n() تعداد ردیف‌ها را می‌شمارد. این تعداد فقط وقتی تعداد سفارش است که هر ردیف یک سفارش باشد. در جدول چندقلمی از n_distinct(order_id) استفاده کنید. گزینهٔ .groups = "drop" گروه‌بندی خروجی را برمی‌دارد تا محاسبات بعدی ناخواسته داخل هر گروه اجرا نشوند.

```
orders |>
  summarise(
    units = sum(quantity),
    revenue = sum(net),
    net_per_unit = sum(net) / sum(quantity),
    weighted_discount = 1 - sum(net) / sum(gross)
  )
```

میانگینِ میانگین‌ها

میانگین سادهٔ میانگین شهرها، میانگین کل سفارش‌ها نیست؛ مگر وزن گروه‌ها برابر باشد. برای شاخص کلی از دادهٔ ردیفی یا وزن صحیح هر گروه استفاده کنید.

بخش سوم • آماده‌سازی داده

متن، تاریخ و مرتب‌سازی زمانی

```
city <- c(" Tehran ", "Shiraz", "Tabriz")
trimws(city)
tolower(trimws(city))
grepl("^T", trimws(city))

x <- c("2026-01-05", "2026-02-10")
d <- as.Date(x, format = "%Y-%m-%d")
format(d, "%Y-%m")
as.integer(d[2] - d[1])
# 36 days
```

تاریخ متنی را پیش از فیلتر و محاسبه به Date تبدیل کنید. قالب %Y-%m-%d یعنی سال چهاررقمی، ماه و روز. برای زمانِ ساعت‌دار از as.POSIXct() و منطقهٔ زمانی مشخص استفاده کنید. حذف منطقهٔ زمانی می‌تواند مرز روزها را در دادهٔ چندکشوری جابه‌جا کند.

```
time <- as.POSIXct("2026-01-05 09:30:00",
                   format = "%Y-%m-%d %H:%M:%S",
                   tz = "Asia/Tehran")
format(time, tz = "UTC", usetz = TRUE)
```

تاریخ شمسی

تبدیل رقم فارسی به انگلیسی، تبدیل تقویم نیست. متنی مانند 1405/06/15 با as.Date به‌طور خودکار از شمسی به میلادی تبدیل نمی‌شود. تقویم منبع را ثبت کنید، از روش تبدیل معتبر استفاده کنید و چند تاریخ معلوم را جداگانه کنترل کنید.

در متن فارسی، «ی/ي» و «ک/ك» یا فاصله و نیم‌فاصله می‌توانند دسته‌های ظاهراً یکسان ولی متفاوت بسازند. استانداردسازی را در یک ستون جدید انجام دهید و مقدار خام را برای ردیابی نگه دارید.

بخش سوم • آماده‌سازی داده

تبدیل جدول عریض و بلند

در دادهٔ مرتب، هر ستون یک متغیر است. اگر ماه‌های مختلف در چند ستون جدا هستند، معمولاً برای رسم نمودار یا گروه‌بندی باید آن‌ها را به یک ستون ماه و یک ستون مقدار تبدیل کنید.

```
library(tidyr)
wide <- data.frame(city = c("Tehran", "Shiraz"),
                   Jan = c(120, 80), Feb = c(150, 90))
long <- wide |>
  pivot_longer(cols = c(Jan, Feb),
               names_to = "month", values_to = "sales")
long

back <- long |>
  pivot_wider(names_from = month, values_from = sales)
back
```

شهر	ماه	فروش نمونه
Tehran	Jan	120
Tehran	Feb	150
Shiraz	Jan	80
Shiraz	Feb	90

pivot_longer() تعداد ستون‌ها را کمتر و ردیف‌ها را بیشتر می‌کند؛ pivot_wider() مسیر برعکس است. قبل از عریض‌کردن مطمئن شوید ترکیب کلیدها، در اینجا شهر و ماه، یکتا است. اگر یکتا نیست باید معنای تجمیع مشخص باشد.

پرکردن مقدار خالی

صفرگذاری برای ترکیبی که در داده وجود ندارد فقط وقتی درست است که نبود آن واقعاً به معنی صفر باشد. نبود گزارش فروش با فروش صفر متفاوت است.

مطالعهٔ تکمیلی: تبدیل شکل داده با tidyr

بخش سوم • آماده‌سازی داده

اتصال جدول‌ها بدون تکثیر اشتباه

برای افزودن مشخصات شهر یا مشتری به سفارش‌ها، دو جدول را با یک کلید مشترک وصل می‌کنیم. جدول سمت راست اگر برای هر کلید چند ردیف داشته باشد، ممکن است تعداد ردیف‌های خروجی چند برابر شود و درآمد اشتباه جمع زده شود.

```
library(dplyr)
orders_small <- data.frame(id = 1:3,
                           city = c("Tehran", "Shiraz", "Tehran"))
cities <- data.frame(city = c("Tehran", "Shiraz"),
                     region = c("central", "south"))
stopifnot(!anyDuplicated(cities$city))
joined <- left_join(orders_small, cities, by = "city",
                    relationship = "many-to-one")
stopifnot(nrow(joined) == nrow(orders_small))
anti_join(orders_small, cities, by = "city")
```

تابع	چه ردیف‌هایی می‌مانند؟
left_join()	همهٔ ردیف‌های جدول چپ
inner_join()	فقط تطبیق‌های دو طرف
full_join()	همهٔ کلیدهای دو طرف
anti_join()	ردیف‌های چپ که تطبیقی ندارند

آرگومان relationship در نسخه‌های جدید dplyr موجود است؛ برای اجرای همین کد dplyr نسخهٔ ۱٫۱٫۱ یا جدیدتر داشته باشید. برای کلید چندستونی، هر دو ستون را در by مشخص کنید. تعداد ردیف‌ها و جمع شاخص اصلی را پیش و پس از اتصال کنترل کنید.

> مطالعهٔ تکمیلی: راهنمای رسمی اتصال‌ها

بخش چهارم • شناخت داده

آمار توصیفی؛ عددها چه می‌گویند؟

قبل از آزمون و مدل، شکل داده را بشناسید: تعداد، مرکز، پراکندگی و مقادیر نامعمول. میانگین به مقادیر بزرگ حساس است؛ میانه مقدار وسط توزیع را توصیف می‌کند. هیچ‌کدام در همهٔ مسئله‌ها جای دیگری را نمی‌گیرد.

```
x <- c(10, 12, 13, 14, 100)
c(n = length(x), mean = mean(x), median = median(x),
  sd = sd(x), min = min(x), max = max(x), IQR = IQR(x))
quantile(x, probs = c(0.25, 0.5, 0.75))

counts <- table(iris$Species)
counts
round(100 * prop.table(counts), 1)
```

میانگین مثال ۲۹٫۸ و میانه ۱۳ است. مقدار ۱۰۰ علت اختلاف است؛ ممکن است خطای ثبت یا یک مورد واقعی باشد. انحراف معیار پراکندگی حول میانگین را نشان می‌دهد و با خودِ متغیر هم‌واحد است. IQR فاصلهٔ چارک اول تا سوم است و به دُم‌های توزیع کمتر حساس است.

نوع سؤال	شاخص شروع
مقدار معمول در توزیع نامتقارن	میانه و چارک‌ها
مجموع فعالیت	جمع با واحد و بازهٔ مشخص
سهم یک دسته	فراوانی و درصد با مخرج روشن
مقایسهٔ گروه‌ها	تعداد هر گروه همراه مرکز و پراکندگی

محدودهٔ اعتبار

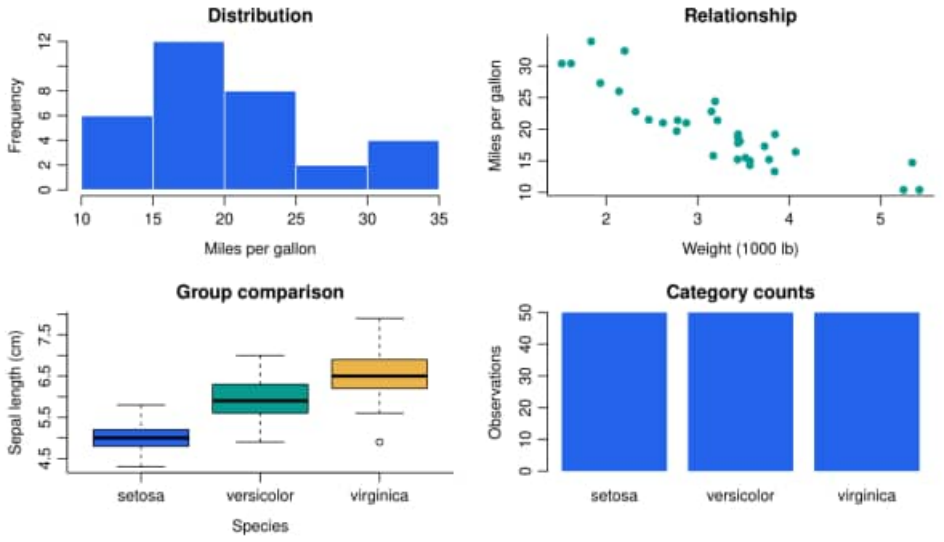
دادهٔ موجود لزوماً نمایندهٔ جامعهٔ هدف نیست. میانگین پاسخ‌دهندگان یک نظرسنجی آنلاین را بدون بررسی روش نمونه‌گیری، میانگین همهٔ مشتریان معرفی نکنید.

بخش چهارم • شناخت داده

انتخاب نمودار مناسب

می‌خواهید چه ببینید؟	نمودار مناسب	نکتهٔ خوانایی
مقایسهٔ دسته‌ها	میله‌ای	محور مقدار از صفر شروع شود
توزیع یک متغیر عددی	هیستوگرام	تعداد بازه‌ها بر شکل اثر می‌گذارد
توزیع در چند گروه	جعبه‌ای و نقاط	اندازهٔ گروه‌ها را فراموش نکنید
رابطهٔ دو متغیر عددی	پراکندگی	هم‌پوشانی نقاط را مدیریت کنید
تغییر در زمان	خطی	ابتدا داده را بر اساس زمان مرتب کنید
رابطه در گروه‌های مختلف	پنل‌های کوچک	مقیاس‌ها را آگاهانه انتخاب کنید

رسم نمودار را با سؤال شروع کنید، نه با افکت. یک نمودار خوب عنوان روشن، نام محور، واحد، بازهٔ زمانی و رنگ‌های قابل‌تمایز دارد. برای مقایسهٔ دقیق، نمودارهای سه‌بعدی و حجم‌های تزئینی معمولاً مانع خواندن مقدارها می‌شوند.



چهار نمودار واقعی تولیدشده با R؛ داده‌های آموزشی داخلی mtcars و iris.

شکل‌های این جزوه برای تحلیل از کد R ساخته شده‌اند. تصویر جلد صرفاً تصویر مفهومی موضوع است و دادهٔ تحلیلی محسوب نمی‌شود.

بخش چهارم • شناخت داده

نمودارهای پایهٔ R

برای اولین نمودار به نصب بسته نیاز ندارید. دادهٔ mtcars همراه R در دسترس است. هر ردیف آن یک مدل خودرو است؛ mpg مسافت به‌ازای گالن و wt وزن برحسب هزار پوند را نشان می‌دهد. این مجموعهٔ کوچک برای آموزش است، نه نمایندهٔ خودروهای امروز.

```
hist(mtcars$mpg, breaks = 8, col = "#2463EB",
     border = "white", main = "Fuel economy",
     xlab = "Miles per gallon")

plot(mtcars$wt, mtcars$mpg, pch = 19,
     col = "#079C91", xlab = "Weight (1000 lb)",
     ylab = "Miles per gallon")
abline(lm(mpg ~ wt, data = mtcars), col = "#D85B35", lwd = 2)

boxplot(Sepal.Length ~ Species, data = iris,
        col = c("#2463EB", "#079C91", "#EDB44A"),
        xlab = "Species", ylab = "Sepal length (cm)")
```

pch شکل نقطه، col رنگ، lwd ضخامت خط و xlab و ylab عنوان محور را تعیین می‌کنند. رابطهٔ منفی در نمودار وزن و مسافت مصرف سوخت دیده می‌شود، ولی یک خط روی شکل به‌تنهایی رابطهٔ علّی را ثابت نمی‌کند.

خروجی خوانا

اگر برچسب‌ها بریده شدند، اندازهٔ پنجرهٔ Plots یا حاشیهٔ شکل را افزایش دهید. ذخیرهٔ تصویر با ابعاد بزرگ‌تر معمولاً از ریزکردن بیش از حد نوشته‌ها بهتر است.

بخش چهارم • شناخت داده

ساخت نمودار لایه‌ای با ggplot2

در ggplot2 ابتدا جدول و نگاشت متغیرها را معرفی می‌کنید و سپس لایه می‌افزایید. داخل aes() می‌گویید کدام ستون به محور یا رنگ وصل شود. رنگ ثابت را بیرون aes() بنویسید تا دستهٔ ساختگی ایجاد نشود.

```
library(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point(aes(color = Species), size = 2.4, alpha = 0.8) +
  labs(title = "Iris measurements",
       x = "Sepal length (cm)", y = "Sepal width (cm)",
       color = "Species") +
  theme_minimal(base_size = 12)
p

p + facet_wrap(~ Species)
```

alpha شفافیت را کنترل می‌کند و برای نقاط روی‌هم مفید است. facet_wrap() برای هر گروه یک پنل می‌سازد. اگر مقیاس پنل‌ها را آزاد کنید، مقایسهٔ مستقیم اندازه‌ها دشوارتر می‌شود؛ این انتخاب را با هدف تحلیل هماهنگ کنید.

افزودن رابطهٔ خطی

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(color = "#2463EB") +
  geom_smooth(method = "lm", formula = y ~ x, se = TRUE) +
  theme_minimal()
```

در این کد، نوار پیرامون خط بیانگر عدم‌قطعیت برآورد میانگین در مدل خطی است؛ آن را محدودهٔ پیش‌بینی تک‌مشاهده تلقی نکنید.

> مطالعهٔ تکمیلی: مستندات ggplot2

بخش چهارم • شناخت داده

ذخیرهٔ نمودار و خروجی فارسی

```
dir.create("outputs", showWarnings = FALSE)
library(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point(aes(color = Species)) + theme_minimal()
ggsave("outputs/iris.png", plot = p,
       width = 8, height = 5, units = "in", dpi = 300)
ggsave("outputs/iris.pdf", plot = p,
       width = 8, height = 5, units = "in")
```

PNG برای درج در بسیاری از صفحات وب و ارائه‌ها مناسب است. PDF می‌تواند اجزای برداری نمودار را نگه دارد تا در بزرگ‌نمایی واضح بمانند. نام شیء نمودار را صریح به plot بدهید تا تصادفاً آخرین شکلِ دیگری ذخیره نشود.

متن فارسی در جدول و نمودار

برای خروجی CSV قابل‌خواندن در بسیاری از نسخه‌های Excel، از readr::write_excel_csv() استفاده کنید؛ این خروجی UTF-8 همراه نشانگر BOM می‌نویسد. جداکنندهٔ مورد انتظار Excel نیز به تنظیمات منطقه‌ای وابسته است؛ در صورت نیاز فایل را از بخش Import با انتخاب UTF-8 وارد کنید.

```
cities <- c("\u062A\u0647\u0631\u0627\u0646",
            "\u0634\u06CC\u0631\u0627\u0632")
readr::write_excel_csv(
  data.frame(city = cities, sales = c(12, 9)),
  "outputs/persian.csv"
)
```

دو رشتهٔ cities نام تهران و شیراز با نمایش یونیکد هستند؛ در فایل خروجی، نام‌ها به فارسی نوشته می‌شوند.

فونت به‌تنهایی همیشه کافی نیست

برای نمودار فارسی، فونت دارای حروف فارسی و دستگاه خروجی سازگار با شکل‌دهی متن لازم است. PNG و PDF نهایی را جداگانه باز کنید؛ درست دیده‌شدن در پنجرهٔ RStudio تضمین نمی‌کند حروف فارسی در هر خروجی درست باشند.

مطالعهٔ تکمیلی: ذخیرهٔ نمودار

مطالعهٔ تکمیلی: نوشتن CSV مناسب Excel

بخش پنجم • آمار و استنباط

نمونه‌گیری، احتمال و فاصلهٔ اطمینان

آمار توصیفی دربارهٔ دادهٔ دیده‌شده حرف می‌زند؛ استنباط آماری دربارهٔ جامعه یا فرایند گسترده‌تر، با فرض‌های مشخص. نمونهٔ بزرگ سوگیری انتخاب را خودکار برطرف نمی‌کند. اگر فقط مشتریان راضی پاسخ دهند، محاسبهٔ دقیق هم پاسخ‌ندادن بقیه را جبران نمی‌کند.

```
set.seed(42)
sample(1:100, size = 10, replace = FALSE)
x <- rnorm(80, mean = 50, sd = 10)
mean(x)
t.test(x)$conf.int

pnorm(1.96)
qnorm(0.975)
```

set.seed() وضعیت آغاز مولد تصادفی را تنظیم می‌کند تا در محیط مشابه نتیجه تکرار شود. rnorm() نمونهٔ تصادفی نرمال می‌سازد، pnorm() احتمال تجمعی و qnorm() چندک را می‌دهد. دادهٔ این مثال شبیه‌سازی‌شده است.

تفسیر فاصلهٔ اطمینان

در تعبیر فراوانی‌گرا، روش ساخت فاصلهٔ اطمینان ۹۵٪، در تکرارهای بسیار نمونه‌گیری و با برقرار بودن فرض‌ها، حدود ۹۵٪ مواقع پارامتر ثابت جامعه را می‌پوشاند. پس از محاسبه، نگویید ۹۵٪ افراد داخل این بازه‌اند؛ بازه دربارهٔ پارامتر، مثلاً میانگین، است.

> **سه چیز را کنار هم گزارش کنید**
>
> برآورد نقطه‌ای، فاصلهٔ اطمینان و تعداد مشاهدات. فقط نوشتن یک میانگین، میزان عدم‌قطعیت و کیفیت نمونه‌گیری را نشان نمی‌دهد.

بخش پنجم • آمار و استنباط

انتخاب آزمون و خواندن p-value

مسئله	نقطهٔ شروع	چه چیزی بررسی شود؟
میانگین دو گروه مستقل	آزمون t ولچ	استقلال، توزیع و موارد اثرگذار
قبل و بعدِ همان افراد	آزمون t زوجی	جفت‌سازی درست و توزیع اختلاف‌ها
دو متغیر دسته‌ای	کای‌دو یا آزمون دقیق	فراوانی مورد انتظار و طرح نمونه
میانگین بیش از دو گروه	ANOVA یا نسخهٔ ولچ	طرح مطالعه، واریانس و باقیمانده‌ها
رابطهٔ دو متغیر عددی	پیرسون یا اسپیرمن	نوع رابطه، پرت‌ها و استقلال

این جدول نقطهٔ شروع است. داده‌های تکراریِ یک فرد، خوشه‌بندی‌شده یا زمانی ممکن است روش دیگری بخواهند. انتخاب آزمون فقط از روی تعداد ستون‌ها کافی نیست؛ واحد مشاهده و طرح جمع‌آوری تعیین‌کننده‌اند.

p-value چه نیست؟

p-value احتمال درست‌بودن فرض صفر یا احتمال تصادفی‌بودن کل نتیجه نیست. این عدد، در چارچوب فرض صفر و فرض‌های آزمون، میزان ناسازگاری دادهٔ مشاهده‌شده را با آن چارچوب نشان می‌دهد. مقدار کوچک، اندازهٔ اثر یا اهمیت عملی را مشخص نمی‌کند.

قبل از دیدن نتیجه تصمیم بگیرید

سؤال، مقایسهٔ اصلی و آستانهٔ تصمیم را از قبل مشخص کنید. ده‌ها آزمون انجام ندهید تا یکی از آن‌ها معنی‌دار شود. برای چند مقایسه، از روش تصحیح مناسب مانند Holm و گزارش شفاف استفاده کنید.

```
p.adjust(c(0.01, 0.03, 0.2, 0.04), method = "holm")
```

بخش پنجم • آمار و استنباط

آزمون t، مقایسهٔ زوجی و ویلکاکسون

```
tt <- t.test(len ~ supp, data = ToothGrowth)
tt$estimate
tt$conf.int
tt$p.value

before <- c(52, 49, 61, 55, 58, 63, 47, 56)
after  <- c(55, 51, 63, 58, 59, 65, 50, 57)
t.test(after, before, paired = TRUE)

wilcox.test(len ~ supp, data = ToothGrowth,
            exact = FALSE)
```

در آزمون مستقل، t.test() به‌طور پیش‌فرض از روش ولچ استفاده می‌کند و برابری واریانس دو گروه را تحمیل نمی‌کند. در مثال زوجی، هر مقدار before و after در یک موقعیت بردار باید متعلق به همان فرد باشد. مرتب‌سازی جداگانهٔ دو بردار جفت‌ها را خراب می‌کند.

در مثال ToothGrowth، مقدار p تقریباً ۰٫۰۶۱ و فاصلهٔ اطمینان اختلاف میانگین تقریباً از ۰٫۱۷- تا ۷٫۵۷ است. نتیجه در آستانهٔ ۰٫۰۵ شواهد کافی برای اختلاف نمی‌دهد؛ این به معنی اثبات برابری نیست. این داده دوزهای مختلف دارد و تحلیل علمی اثر مکمل باید دوز و طرح مطالعه را هم در نظر بگیرد.

ویلکاکسون چه می‌سنجد؟

ویلکاکسون آزمون عمومیِ اختلاف میانگین نیست. تفسیر آن به شکل توزیع‌ها و فرض‌ها وابسته است؛ برای تعبیر صرفاً اختلاف میانه، شرایط بیشتری لازم است. فقط به دلیل ناموفق بودن آزمون نرمال‌بودن، روش را مکانیکی عوض نکنید.

مطالعهٔ تکمیلی: راهنمای t.test

بخش پنجم • آمار و استنباط

دادهٔ دسته‌ای و مقایسهٔ چند گروه

جدول توافقی و آزمون استقلال

```
tab <- matrix(c(30, 20, 15, 35), nrow = 2, byrow = TRUE,
              dimnames = list(group = c("A", "B"),
                              bought = c("yes", "no")))
chi <- chisq.test(tab)
chi$expected
chi$p.value
fisher.test(tab)$p.value
```

برای جدول شمارش، درصد و مخرج هر گروه را هم نشان دهید. اگر فراوانی‌های مورد انتظار کم باشند، تقریب کای‌دو ممکن است مناسب نباشد و آزمون دقیق یا روش دیگری لازم شود. دادهٔ هر سلول تعداد مشاهده است، نه درصد از پیش محاسبه‌شده.

ANOVA و مقایسهٔ پس از آن

```
fit_aov <- aov(Sepal.Length ~ Species, data = iris)
summary(fit_aov)
TukeyHSD(fit_aov)
oneway.test(Sepal.Length ~ Species, data = iris,
            var.equal = FALSE)
```

آزمون کلی ANOVA مشخص نمی‌کند کدام گروه‌ها متفاوت‌اند. روش Tukey برای مقایسه‌های زوجی در چارچوب مدل مناسب کاربرد دارد. نسخهٔ ولچ می‌تواند برابری واریانس‌ها را لازم نداند، ولی استقلال مشاهدات و مناسب‌بودن سایر فرض‌ها همچنان مهم‌اند.

گزارش نتیجه

نام آزمون، اندازهٔ گروه‌ها، شاخص توصیفی، برآورد اختلاف و محدودیت طرح مطالعه را بنویسید. ستاره‌های خروجی R جای توضیح علمی را نمی‌گیرند.

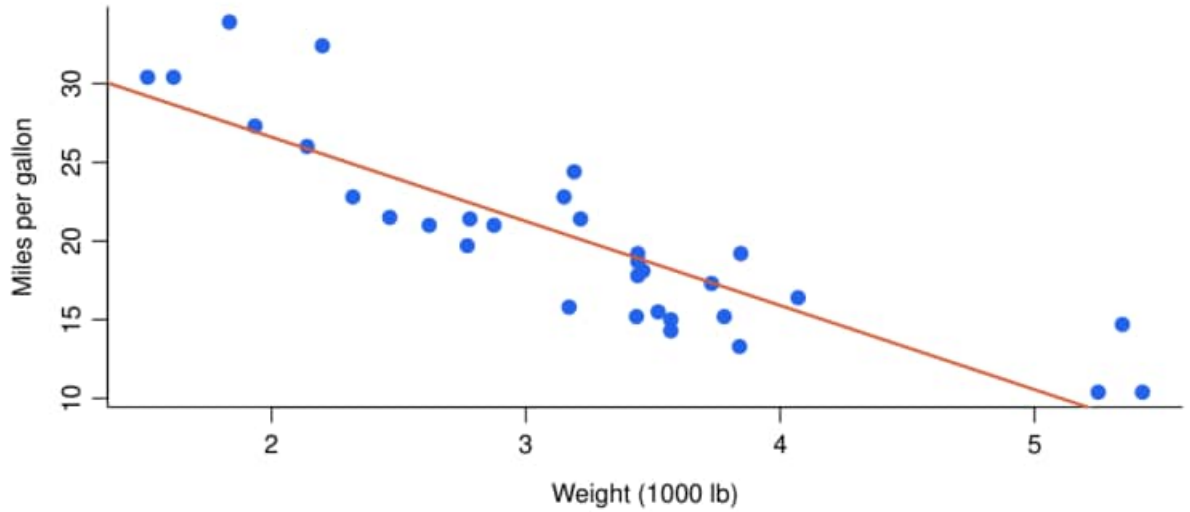
بخش پنجم • آمار و استنباط

همبستگی و رابطهٔ غیرعلّی

```
cor(mtcars$wt, mtcars$mpg, method = "pearson")
cor.test(mtcars$wt, mtcars$mpg, method = "pearson")
cor.test(mtcars$wt, mtcars$mpg, method = "spearman",
         exact = FALSE)
plot(mtcars$wt, mtcars$mpg,
     xlab = "Weight (1000 lb)", ylab = "Miles per gallon")
```

ضریب پیرسون در این داده حدود ۰٫۸۶۸- است و رابطهٔ خطی منفیِ قوی را توصیف می‌کند. ضریب در بازهٔ ۱- تا ۱ قرار دارد. نزدیک‌بودن به صفر لزوماً نبود هر نوع رابطه نیست؛ رابطهٔ خمیده ممکن است ضریب خطی کوچکی داشته باشد.

اسپیرمن بر رتبه‌ها تکیه می‌کند و برای بررسی رابطهٔ یکنواخت مفید است. پیش از انتخاب هرکدام نمودار را ببینید؛ یک مشاهدهٔ اثرگذار، دامنهٔ محدود داده یا ترکیب گروه‌ها می‌تواند برداشت را عوض کند.



رابطهٔ وزن و mpg در دادهٔ آموزشی mtcars؛ خط، برازش ساده است و علت را ثابت نمی‌کند.

همبستگی و علیت

تغییر هم‌زمان دو متغیر به‌تنهایی نشان نمی‌دهد تغییر یکی باعث تغییر دیگری می‌شود. متغیر مخدوش‌کننده، جهت معکوس رابطه و روش انتخاب نمونه را بررسی کنید.

بخش ششم • مدل‌سازی

رگرسیون خطی و تفسیر خروجی

مدل خطی، رابطهٔ میانگین پاسخ با متغیرهای توضیحی را توصیف می‌کند. عبارت mpg ~ wt + hp می‌گوید mpg با وزن و توان مدل شود. علامت ~ نشانهٔ فرمول مدل است، نه برابری عددی.

```
fit <- lm(mpg ~ wt + hp, data = mtcars)
summary(fit)
coef(fit)
confint(fit)

new_car <- data.frame(wt = 3, hp = 110)
predict(fit, newdata = new_car, interval = "prediction")
predict(fit, newdata = new_car, interval = "confidence")
```

جزء خروجی	معنا
Estimate	ضریب برآوردشدهٔ متغیر
Std. Error	عدم‌قطعیت برآورد ضریب در مدل
Residuals	اختلاف مقدار واقعی و برازش‌شده
R-squared	سهم توضیح‌داده‌شدهٔ تغییرپذیری پاسخ در دادهٔ برازش
Adjusted R-squared	نسخهٔ تعدیل‌شده با درنظرگرفتن تعداد متغیرها

ضریب wt در این برازش حدود ۳٫۸۸- است: در چارچوب همین مدل و با ثابت نگه‌داشتن hp، افزایش یک واحد wt، یعنی هزار پوند، با کاهش حدود ۳٫۸۸ واحد mpg همراه است. این جمله یک ارتباط شرطی در مدل است و ادعای علّی نیست.

دو بازهٔ متفاوت

بازهٔ confidence برای میانگین پاسخ در ورودی مشخص است؛ prediction برای مشاهدهٔ جدید نیز تغییرپذیری فردی را لحاظ می‌کند و معمولاً پهن‌تر است.

مطالعهٔ تکمیلی: مستندات lm

بخش ششم • مدل‌سازی

بررسی فرض‌ها و نقاط اثرگذار

```
fit <- lm(mpg ~ wt + hp, data = mtcars)
old <- par(mfrow = c(2, 2))
plot(fit)
par(old)

sort(cooks.distance(fit), decreasing = TRUE)[1:5]
residuals(fit)
fitted(fit)
```

نمودار باقیمانده در برابر مقدار برازش‌شده را برای الگوی خمیده یا تغییر پراکندگی ببینید. نمودار Q-Q دربارهٔ شکل توزیع باقیمانده‌ها اطلاعات می‌دهد. فاصلهٔ کوک به شناسایی نقاطی کمک می‌کند که ممکن است اثر زیادی بر برازش داشته باشند.

نشانه	اقدام بررسی
الگوی خمیدهٔ باقیمانده	رابطهٔ غیرخطی یا متغیر جاافتاده
پراکندگی بادبزنی	ناهمسانی واریانس و مناسب‌بودن مدل
نقطهٔ بسیار اثرگذار	صحت ثبت و حساسیت نتیجه به آن مشاهده
وابستگی زمانی یا فردی	روش سازگار با طرح نمونه و وابستگی

نرمال‌بودن همهٔ متغیرهای ورودی شرط عمومی رگرسیون خطی نیست. برای استنباط کلاسیک در نمونه‌های کوچک، رفتار خطاها و فرض‌های مدل اهمیت دارد. همچنین هم‌خطی شدید می‌تواند ضرایب ناپایدار ایجاد کند، حتی وقتی R-squared بزرگ است.

تحلیل حساسیت

اگر نتیجه با حذف یک مشاهده عوض می‌شود، این حساسیت را گزارش کنید. مشاهده را فقط برای بهترشدن p-value حذف نکنید. مدل را بر اساس سؤال و ماهیت داده اصلاح کنید.

بخش ششم • مدل‌سازی

آموزش، آزمون و ارزیابی پیش‌بینی

برای سنجش پیش‌بینی، بخشی از داده را تا پایان آموزش کنار بگذارید. امتیاز روی دادهٔ آموزش به‌تنهایی توان تعمیم را نشان نمی‌دهد. مثال زیر کوچک و آموزشی است؛ نمرهٔ یک تقسیم روی ۳۲ خودرو را معیار پایدار عملکرد واقعی ندانید.

```
set.seed(42)
id <- sample(seq_len(nrow(mtcars)),
             size = floor(0.75 * nrow(mtcars)))
train <- mtcars[id, ]
test <- mtcars[-id, ]
fit <- lm(mpg ~ wt + hp, data = train)
pred <- predict(fit, newdata = test)
rmse <- sqrt(mean((test$mpg - pred)^2))
mae <- mean(abs(test$mpg - pred))
baseline <- sqrt(mean((test$mpg - mean(train$mpg))^2))
c(RMSE = rmse, MAE = mae, baseline_RMSE = baseline)
```

RMSE و MAE در واحد پاسخ‌اند و مقدار کمتر معمولاً بهتر است. RMSE به خطاهای بزرگ وزن بیشتری می‌دهد. مدل را با یک خط مبنای ساده، مانند پیش‌بینی میانگین آموزش، مقایسه کنید. بهتر بودن از مدل پیچیدهٔ دیگر، بدون مقایسه با خط مبنا، کافی نیست.

نشت اطلاعات

میانگین جایگزینی مقادیر گمشده، استانداردسازی و انتخاب ویژگی باید فقط از دادهٔ آموزش یاد گرفته شوند. سپس همان پارامترها روی آزمون اعمال شوند. استفاده از کل داده پیش از تقسیم، نتیجه را خوش‌بینانه می‌کند.

برای دادهٔ زمانی، آموزش باید پیش از آزمون باشد. برای چند ردیف از یک مشتری، تقسیم را در سطح مشتری انجام دهید. برای انتخاب مدل، اعتبارسنجی متقاطع در بخش آموزش مفید است؛ مجموعهٔ آزمون نهایی را مرتب برای تنظیم مدل مصرف نکنید.

بخش ششم • مدل‌سازی

رگرسیون لجستیک برای پاسخ صفر و یک

اگر پاسخ «خرید کرد یا نکرد» است، مدل لجستیک می‌تواند احتمال خرید را برآورد کند. در این نمونه، تمام داده‌ها ساختگی‌اند؛ هدف یادگیری فرایند است. بخش اول داده را شبیه‌سازی و بخش دوم مدل را روی آموزش برازش می‌کند.

```
set.seed(7)
d <- data.frame(visits = rpois(240, 4),
                minutes = runif(240, 1, 15))
p <- plogis(-3 + 0.35 * d$visits + 0.1 * d$minutes)
d$bought <- rbinom(nrow(d), 1, p)
id <- sample(seq_len(nrow(d)), 180)
train <- d[id, ]
test <- d[-id, ]
g <- glm(bought ~ visits + minutes, data = train,
         family = binomial())
prob <- predict(g, newdata = test, type = "response")
head(prob)
yhat <- as.integer(prob >= 0.5)
```

با type = "response" احتمال دریافت می‌کنید. ضرایب خام مدل روی مقیاس log-odds هستند؛ آن‌ها را افزایش مستقیم احتمال نخوانید. exp(coef(g)) ضرایب را به نسبت شانس تبدیل می‌کند، با تفسیر شرطی بر دیگر متغیرهای مدل.

آستانهٔ ۰٫۵ قانون همیشگی نیست

آستانه به هزینهٔ مثبت کاذب و منفی کاذب و هدف کاربرد وابسته است. آن را با دادهٔ اعتبارسنجی انتخاب کنید، نه با نگاه مکرر به نتیجهٔ آزمون نهایی.

مطالعهٔ تکمیلی: راهنمای glm

بخش ششم • مدل‌سازی

ماتریس خطا و معیارهای طبقه‌بندی

این درس ادامهٔ مثال لجستیک صفحهٔ قبل است. ردیف ماتریس را مقدار واقعی و ستون آن را پیش‌بینی می‌گیریم. ترتیب سطح‌ها را صریح تعیین کنید تا حتی اگر یک دسته در خروجی نبود، جای سلول‌ها عوض نشود.

```
cm <- table(actual = factor(test$bought, levels = 0:1),
            predicted = factor(yhat, levels = 0:1))
cm
accuracy <- mean(yhat == test$bought)
precision <- if (sum(cm[, 2]) > 0) {
  cm[2, 2] / sum(cm[, 2])
} else NA_real_
recall <- if (sum(cm[2, ]) > 0) {
  cm[2, 2] / sum(cm[2, ])
} else NA_real_
c(accuracy = accuracy, precision = precision, recall = recall)
```

معیار	پرسش
Accuracy	چه سهمی از همهٔ پیش‌بینی‌ها درست است؟
Precision	از موارد پیش‌بینی‌شدهٔ مثبت، چند مورد واقعاً مثبت‌اند؟
Recall	از همهٔ مثبت‌های واقعی، چند مورد پیدا شده‌اند؟

اگر فقط ۱٪ موارد مثبت باشند، مدل همیشه‌منفی ۹۹٪ Accuracy دارد ولی هیچ مثبت واقعی را پیدا نمی‌کند. در چنین مسئله‌ای، ترکیب معیارها و هزینهٔ خطا اهمیت دارد. امتیاز احتمال را نیز می‌توان از نظر کالیبراسیون بررسی کرد: آیا گروهی با احتمال ۰٫۷ واقعاً نزدیک ۷۰٪ رخداد دارد؟

گزارش کاربردی

ماتریس خطا، آستانه، تعداد آزمون و نرخ پایهٔ رخداد را همراه معیارها بنویسید. یک عدد بدون این زمینه‌ها برای تصمیم‌گیری کافی نیست.

بخش ششم • مدل‌سازی

خوشه‌بندی و کاهش بُعد

در خوشه‌بندی، برچسب هدف از قبل وجود ندارد و نمونه‌ها بر اساس شباهت کنار هم قرار می‌گیرند. استانداردسازی مهم است؛ وگرنه متغیری با مقیاس بزرگ‌تر ممکن است فاصله‌ها را تحت‌تأثیر قرار دهد. این مثال فقط چهار ستون عددی iris را استفاده می‌کند.

```
x <- scale(iris[, 1:4])
set.seed(42)
km <- kmeans(x, centers = 3, nstart = 20)
km$size
table(cluster = km$cluster, species = iris$Species)

pc <- prcomp(iris[, 1:4], scale. = TRUE)
summary(pc)
plot(pc$x[, 1:2], col = km$cluster, pch = 19,
     xlab = "PC1", ylab = "PC2")
```

تعداد خوشهٔ ۳ برای نمایش انتخاب شده، نه اینکه با این کد بهترین تعداد خوشه اثبات شود. پایداری، تفسیرپذیری و معیارهایی مانند silhouette می‌توانند به انتخاب کمک کنند. شمارهٔ خوشه‌ها قراردادی است؛ خوشهٔ ۱ ذاتاً بهتر یا کوچک‌تر از خوشهٔ ۲ نیست.

PCA محورهای جدیدی می‌سازد که تغییرپذیری دادهٔ عددی را خلاصه کنند. مؤلفهٔ اول بیشترین واریانس را در بر می‌گیرد، اما بیشترین واریانس لزوماً مهم‌ترین ویژگی کسب‌وکار نیست. بارهای مؤلفه‌ها و واحدها را بررسی کنید.

اگر هدف پیش‌بینی است

استانداردسازی و PCA را فقط روی آموزش برازش دهید و همان تبدیل را به آزمون اعمال کنید. مثال این صفحه اکتشافی است و از کل iris برای مشاهدهٔ ساختار استفاده می‌کند.

مطالعهٔ تکمیلی: راهنمای prcomp

بخش ششم • مدل‌سازی

سری زمانی و یک پیش‌بینی پایه

دادهٔ زمانی علاوه بر مقدار، ترتیب و فاصلهٔ زمانی دارد. روند و فصل‌مندی را پیش از پیش‌بینی بررسی کنید. مجموعهٔ آموزشی AirPassengers تعداد ماهانهٔ مسافران را نشان می‌دهد و به صورت شیء ts در R موجود است.

```
train_ts <- window(AirPassengers, end = c(1959, 12))
test_ts <- window(AirPassengers, start = c(1960, 1))
fit_ts <- HoltWinters(train_ts, seasonal = "multiplicative")
forecast <- predict(fit_ts, n.ahead = length(test_ts))
seasonal_naive <- rep(tail(as.numeric(train_ts), 12),
                      length.out = length(test_ts))
c(model_RMSE = sqrt(mean((test_ts - forecast)^2)),
  naive_RMSE = sqrt(mean((test_ts - seasonal_naive)^2)))
plot(fit_ts, forecast)
```

مدل فقط تا پایان ۱۹۵۹ آموزش دیده و سال ۱۹۶۰ برای آزمون کنار گذاشته شده است. خط مبنای فصلی مقدار همان ماه سال قبل را تکرار می‌کند. مدل ضربی به دادهٔ مثبت نیاز دارد؛ وجود صفر، مقدار منفی یا فصل‌مندی نامتناسب انتخاب مدل را تغییر می‌دهد.

ماه گمشده با فروش صفر فرق دارد

پیش از ساخت ts، فاصلهٔ زمانی را منظم کنید. حذف یک ماه و چسباندن بقیهٔ ردیف‌ها، تقویم را جابه‌جا می‌کند. علت نبود داده را مشخص کنید و هر روش تکمیل را ثبت کنید.

برای ارزیابی جدی، چند مبدأ زمانی و بازه‌های آزمون بعد از آن‌ها را بررسی کنید. تعطیلات، تغییر قیمت و رویدادهای بیرونی ممکن است الگوهای گذشته را عوض کنند؛ پیش‌بینی باید با عدم‌قطعیت و محدودیت گزارش شود.

بخش هفتم • تحویل و بازتولید

ساخت گزارش خودکار با Quarto

گزارش قابل‌بازتولید، متن و کد را در یک فایل نگه می‌دارد. با تغییر داده، جدول‌ها و نمودارها دوباره ساخته می‌شوند. Quarto یک برنامهٔ جداگانه است و برای اجرای کد R به بسته‌هایی مانند knitr و rmarkdown نیاز دارد. فایل نمونهٔ همراه با پسوند .qmd آماده شده است.

نمونهٔ حداقلی یک فایل qmd

```
---
title: "My R report"
format: html
---

## Summary

```{r}
summary(iris)
```

Plot

```{r}
plot(iris$Sepal.Length, iris$Sepal.Width)
```
```

در RStudio فایل را باز و Render را انتخاب کنید. برای بستهٔ همراه، از Terminal در ریشهٔ پروژه دستور quarto render report/report.qmd را اجرا کنید. تنظیم execute-dir: project در فایل _quarto.yml باعث می‌شود مسیرهای داده از ریشهٔ پروژه تفسیر شوند.

گزارش فارسی

برای HTML از lang: fa و dir: rtl استفاده کنید. ساخت PDF با Quarto ممکن است به موتور TeX و تنظیم فونت مناسب فارسی نیاز داشته باشد؛ HTML معمولاً نقطهٔ شروع ساده‌تری برای گزارش تعاملی و راست‌به‌چپ است.

مطالعهٔ تکمیلی: اجرای R در Quarto

بخش هفتم • تحویل و بازتولید

اولین اپ تعاملی با Shiny

Shiny اجازه می‌دهد ورودی کاربر به خروجی جدول یا نمودار وصل شود. اپ کوچک زیر با تغییر تعداد بازه‌ها، هیستوگرام را دوباره رسم می‌کند. آن را در فایل app.R ذخیره کنید؛ در بستهٔ همراه، پوشهٔ app آماده است.

```
library(shiny)
ui <- fluidPage(
  titlePanel("Iris explorer"),
  sliderInput("bins", "Histogram bins", 5, 30, 12),
  plotOutput("dist")
)
server <- function(input, output, session) {
  output$dist <- renderPlot({
    hist(iris$Sepal.Length, breaks = input$bins,
         col = "#2463EB", border = "white",
         main = "Sepal length", xlab = "cm")
  })
}
shinyApp(ui, server)
```

ui ظاهر و ورودی‌ها را تعریف می‌کند. server منطق محاسبه را نگه می‌دارد. شناسهٔ dist در plotOutput() باید با output$dist یکی باشد. خواندن input$bins داخل بخش واکنشی باعث می‌شود با تغییر اسلایدر خروجی به‌روز شود.

```
shiny::runApp("app")
```

اجرا و انتشار دو مرحله‌اند

این دستور اپ را روی دستگاه خودتان اجرا می‌کند. با Stop آن را متوقف کنید. انتشار برای دیگران به محل میزبانی و تنظیم دسترسی نیاز دارد؛ قبل از انتشار، دادهٔ شخصی و کلیدهای اتصال را از کد و خروجی حذف کنید.

مطالعهٔ تکمیلی: آموزش رسمی Shiny

بخش هفتم • تحویل و بازتولید

پایگاه داده و داده‌های بزرگ‌تر

وقتی داده در پایگاه داده است، لازم نیست همیشه همهٔ آن را به حافظه بیاورید. با DBI می‌توان اتصال ساخت و فقط ستون‌ها و ردیف‌های لازم را خواند. مثال زیر یک پایگاه SQLite موقت در حافظه می‌سازد و به حساب یا سرور واقعی وصل نمی‌شود.

```
# Install DBI and RSQLite once before this example.
con <- DBI::dbConnect(RSQLite::SQLite(), ":memory:")
DBI::dbWriteTable(con, "cars", mtcars)
small <- DBI::dbGetQuery(
  con,
  "SELECT mpg, wt, hp FROM cars WHERE mpg > 20"
)
DBI::dbDisconnect(con)
head(small)
```

در پروژهٔ واقعی، اتصال را حتی در صورت خطا ببندید؛ قرار دادن on.exit(DBI::dbDisconnect(con)) داخل تابع یک الگوی مناسب است. رمز عبور را در فایل عمومی قرار ندهید. مقدار ورودی کاربر را با چسباندن متن به SQL وارد نکنید؛ از پارامترگذاری پشتیبانی‌شدهٔ درایور استفاده کنید.

پیش از خرید سخت‌افزار بیشتر

- فقط ستون‌های لازم را بخوانید و نوع داده را درست تعیین کنید.
- فیلتر و تجمیع را در پایگاه داده انجام دهید و نتیجهٔ کوچک را منتقل کنید.
- زمان اجرای مرحله‌ها و اندازهٔ اشیا را اندازه بگیرید.
- برای دادهٔ ستونی یا خارج از حافظه، ابزارهای تخصصی را متناسب با مسئله بررسی کنید.

محدودیت حافظه

اجرای بسیاری از تحلیل‌های R به حافظهٔ در دسترس وابسته است. بزرگ‌بودن فایل روی دیسک تنها عامل نیست؛ تبدیل‌ها و کپی‌های میانی هم می‌توانند حافظه مصرف کنند.

مطالعهٔ تکمیلی: مستندات DBI

بخش هفتم • تحویل و بازتولید

خطایابی و بهبود سرعت

هنگام خطا، ابتدا اولین پیام معنی‌دار و خط مربوط به آن را پیدا کنید. حدس‌زدن چند تغییر هم‌زمان تشخیص علت را سخت می‌کند. ورودی همان خط را با str()، head() و تعداد ردیف‌ها بررسی کنید.

```
x <- c("10", "20", "bad")
y <- suppressWarnings(as.numeric(x))
failed <- is.na(y) & !is.na(x)
x[failed]

safe_mean <- function(x) {
  if (!is.numeric(x)) stop("x must be numeric")
  if (!any(!is.na(x))) return(NA_real_)
  mean(x, na.rm = TRUE)
}
safe_mean(c(10, NA, 20))
```

در این نمونه، هشدار تبدیل عمداً ساکت شده اما شکست‌ها صریح بررسی شده‌اند. حذف هشدار بدون بررسی خروجی، خطای داده را پنهان می‌کند. traceback() بلافاصله پس از خطای مدیریت‌نشده، مسیر فراخوانی را نشان می‌دهد؛ browser() می‌تواند اجرای تابع را برای بررسی متوقف کند.

```
system.time(sum(seq_len(1000000)))
object.size(iris)
stopifnot(nrow(iris) == 150)
all.equal(0.1 + 0.2, 0.3)
```

عدد اعشاری و برابری

اعداد ممیزشناور همیشه نمایش دقیق ده‌دهی ندارند. برای مقایسهٔ نتایج اعشاری از تحمل خطا یا all.equal استفاده کنید؛ برای مبالغ حساس، واحد کوچک‌تر و قواعد گردکردن مشخص داشته باشید.

بخش هفتم • تحویل و بازتولید

نسخه‌ها، renv و اجرای دوباره

تحلیل قابل‌بازتولید به کد تنها محدود نمی‌شود. دادهٔ ورودی، نسخهٔ R و بسته‌ها، ترتیب اجرای مراحل و پارامترهای تحلیل نیز مهم‌اند. در پروژهٔ همراه، فایل خام تغییر نمی‌کند و خروجی‌ها از اسکریپت ساخته می‌شوند.

```
sessionInfo()
writeLines(capture.output(sessionInfo()),
           "outputs/session-info.txt")

# Optional: run inside your own RStudio project.
renv::init()
renv::snapshot()
# On another setup of the same project:
# renv::restore()
```

renv::init() محیط وابستگی‌های پروژه را آماده می‌کند. snapshot() وضعیت بسته‌ها را در فایل قفل ثبت و restore() برای بازسازی آن وضعیت تلاش می‌کند. این فرایند جای ثبت نسخهٔ R، دادهٔ خام یا پیش‌نیازهای سیستم‌عامل را نمی‌گیرد و ممکن است به اینترنت نیاز داشته باشد.

چک نهایی پیش از تحویل

- R را Restart کنید و همهٔ مراحل را از ابتدا اجرا کنید.
- مسیر شخصی و وابستگی پنهان به Workspace را حذف کنید.
- تعداد ردیف‌ها، واحد پول و تعریف شاخص‌ها را ثبت کنید.
- seed را برای نمونه‌گیری یا الگوریتم تصادفی بنویسید.
- ورودی، اسکریپت، خروجی و راهنمای اجرا را همراه تحویل دهید.

> مطالعهٔ تکمیلی: راهنمای renv

بخش هشتم • پروژهٔ عملی

پروژهٔ کامل؛ تحلیل فروش ساختگی

قرار است برای یک فروشگاه آموزشی گزارشی بسازیم: پس از کنترل کیفیت داده، درآمد سفارش‌های تکمیل‌شده، میانگین سفارش، سهم شهرها و تغییر ماهانه را محاسبه کنیم. فایل sales_raw.csv شامل ۱۲۱ ردیف در چهار ماه است و عمداً چند مسئلهٔ کیفیت داده دارد.

ستون	تعریف
order_id	شناسهٔ یکتای یک سفارش تک‌دسته‌ای
date	تاریخ میلادی با قالب سال-ماه-روز
city / channel	شهر و کانال online یا store
category	دستهٔ book، course یا tool
quantity	تعداد واحد داخل همان سفارش
unit_price	قیمت واحد برحسب تومان
discount	نرخ تخفیف در بازهٔ صفر تا یک
status	completed، cancelled یا returned

در این تمرین، سفارش لغوشده و مرجوع‌شده وارد درآمد تکمیل‌شده نمی‌شوند. زمان و مبلغ واقعی بازپرداخت، هزینهٔ تأمین، مالیات و هزینهٔ جذب مشتری در فایل نیستند؛ بنابراین خروجی «سود» یا «جریان نقدی حسابداری» نیست.



از دادهٔ خام به جدول معتبر؛ تصویر مفهومی مراحل آماده‌سازی.

دادهٔ ساختگی

اعداد این پروژه فقط برای یادگیری‌اند و وضعیت بازار، شهرها یا یک کسب‌وکار واقعی را نشان نمی‌دهند.

بخش هشتم • پروژهٔ عملی

پروژه؛ اجرای کنترل کیفیت

فایل R-course.Rproj را باز کنید و مرحلهٔ آماده‌سازی را اجرا کنید. کد کامل در فایل زیر است؛ کنترل ستون‌های لازم، تکرار دقیق، یکتایی شناسه، تاریخ، تعداد، قیمت، تخفیف و وضعیت را انجام می‌دهد.

```
source("scripts/02_clean_base.R", encoding = "UTF-8")
quality
rejected
head(orders)
stopifnot(!anyDuplicated(orders$order_id))
```

مرحله	تعداد ردیف
فایل خام	۱۲۱
تکرار کاملاً یکسان	۱
ردیف یکتا پس از حذف تکرار	۱۲۰
ردیف نامعتبر؛ تعداد خالی یا قیمت منفی	۲
ردیف معتبر در همهٔ وضعیت‌ها	۱۱۸
لغوشده یا مرجوع‌شدهٔ معتبر	۱۷
سفارش تکمیل‌شدهٔ قابل‌تحلیل	۱۰۱

از جدول روشن است که ۱۲۱ منهای ۱، منهای ۲، منهای ۱۷ برابر با ۱۰۱ است. کنارگذاشتن هر مرحله در خروجی ثبت شده است. سفارش با شناسهٔ یکسان ولی محتوای متناقض، تحلیل را متوقف می‌کند تا اختلاف پنهان نشود.

سه فایل قابل بررسی

quality.csv تعدادها را نگه می‌دارد؛ rejected_rows.csv شامل دو ردیف مسئله‌دار است؛ completed_orders.csv همان داده‌ای است که تحلیل روی آن انجام می‌شود. دادهٔ خام را برای رفع خطا بازنویسی نمی‌کنیم.

بخش هشتم • پروژهٔ عملی

پروژه؛ ساخت شاخص‌های مالی

درآمد خالصِ این تمرین، پس از تخفیف و فقط برای سفارش‌های تکمیل‌شده محاسبه می‌شود. ابتدا مقدار ناخالص هر سفارش را از تعداد ضربدر قیمت واحد بسازید و سپس تخفیف را اعمال کنید.

```
source("scripts/03_analysis_base.R", encoding = "UTF-8")
orders$gross <- orders$quantity * orders$unit_price
orders$net <- orders$gross * (1 - orders$discount)

sum(orders$net)
mean(orders$net)
1 - sum(orders$net) / sum(orders$gross)
```

شاخص	نتیجهٔ کنترل‌شده
تعداد سفارش تکمیل‌شده	۱۰۱
جمع درآمد پس از تخفیف	۷۷٬۴۱۸٬۵۰۰ تومان
میانگین مبلغ سفارش	حدود ۷۶۶٬۵۲۰ تومان
نرخ تخفیف وزنی بر مبلغ ناخالص	حدود ۵٫۹۳٪

برای سفارش سه‌واحدی با قیمت واحد ۲۰۰ هزار تومان و تخفیف ۱۰٪، مبلغ ناخالص ۶۰۰ هزار و مبلغ پس از تخفیف ۵۴۰ هزار تومان است. میانگین مبلغ سفارش با قیمت متوسط یک واحد کالا متفاوت است.

چرا تخفیف وزنی؟

میانگینِ سادهٔ درصد تخفیفِ یک سفارش ۱۰۰ هزار تومانی و یک سفارش ۱۰ میلیون تومانی، سهم مالی هرکدام را نادیده می‌گیرد. نسبت مجموع تخفیف به مجموع مبلغ ناخالص اثر مالی کل را نشان می‌دهد.

مقدار گردشده فقط برای نمایش است؛ در فایل خروجی، عدد کامل نگه داشته می‌شود. اگر خط مشی فروشگاه، گردکردن در سطح سفارش باشد، همان قاعده باید پیش از جمع اعمال و مستند شود.

بخش هشتم • پروژهٔ عملی

پروژه؛ مقایسهٔ شهرها و ماه‌ها

```
city_summary <- aggregate(net ~ city, data = orders, sum)
city_summary <- city_summary[order(-city_summary$net), ]
monthly <- aggregate(net ~ month, data = orders, sum)
monthly <- monthly[order(monthly$month), ]
monthly$growth_pct <- c(NA, 100 * diff(monthly$net) /
                        head(monthly$net, -1))
```

شهر	درآمد پس از تخفیف، تومان
Tehran	۲۳٬۲۲۱٬۵۰۰
Tabriz	۲۳٬۰۱۷٬۵۰۰
Isfahan	۱۵٬۷۲۶٬۰۰۰
Shiraz	۱۵٬۴۵۳٬۵۰۰

ماه میلادی	درآمد، تومان	تغییر نسبت به ماه قبل
2026-01	۲۲٬۹۹۷٬۵۰۰	قابل محاسبه نیست
2026-02	۱۶٬۶۶۲٬۰۰۰	۲۷٫۵۵٪ کاهش
2026-03	۱۹٬۹۹۳٬۰۰۰	۱۹٫۹۹٪ افزایش
2026-04	۱۷٬۷۶۶٬۰۰۰	۱۱٫۱۴٪ کاهش

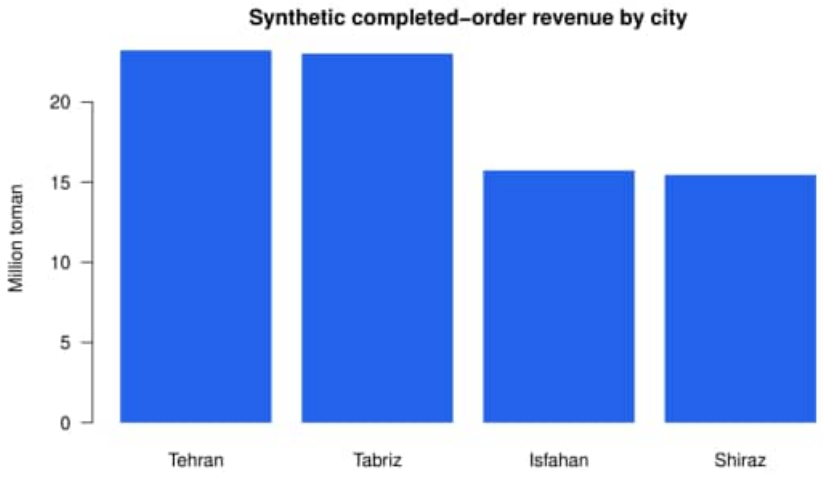
مقدار ماه اول NA است، چون ماه پیش از آن در فایل نیست. اگر درآمد ماه قبل صفر باشد، درصد تغییر با این فرمول تعریف مناسبی ندارد؛ در آن وضعیت تغییر مبلغ را گزارش کنید. مرتب‌سازی زمانی پیش از diff() ضروری است.

تفسیر درست رتبه

شهر دارای درآمد بیشتر لزوماً نرخ تبدیل یا سود بیشتری ندارد. تعداد مخاطبان، هزینه و ترکیب کالاها در مقایسه دخیل‌اند؛ دادهٔ فعلی برای همهٔ این پرسش‌ها کافی نیست.

بخش هشتم • پروژهٔ عملی

پروژه؛ نمودار و گزارش نهایی



درآمد سفارش‌های تکمیل‌شده برحسب میلیون تومان؛ دادهٔ ساختگی پروژه.

نمونهٔ متن گزارش قابل‌تحویل

پس از حذف یک ردیف کاملاً تکراری، کنارگذاشتن دو ردیف نامعتبر و خروج ۱۷ سفارش لغوشده یا مرجوع‌شدهٔ معتبر، ۱۰۱ سفارش تکمیل‌شده بررسی شد. مجموع درآمد پس از تخفیف ۷۷٫۴۲ میلیون تومان و متوسط مبلغ سفارش حدود ۷۶۶٫۵ هزار تومان بود. درآمد تهران و تبریز در این داده به هم نزدیک است.

درآمد فوریه نسبت به ژانویه کاهش و در مارس نسبت به فوریه افزایش داشته است. این تغییرها صرفاً توصیفی‌اند؛ برای توضیح علت باید تعداد سفارش، ترکیب کالا، تخفیف و شرایط فعالیت بررسی شود. این مجموعهٔ ساختگی برای پیش‌بینی قابل‌اتکای بازار کافی نیست.

فایل‌های تحویل

- kpi.csv، city_summary.csv و monthly_summary.csv برای جدول‌ها.
- project_charts.pdf برای نمودار شهر و روند ماهانه.
- اسکریپت‌ها، دادهٔ خام، گزارش کیفیت و session-info.txt برای بازتولید.

معیار پایان پروژه

پس از Restart کردن R، اجرای scripts/03_analysis_base.R باید دوباره همین تعداد سفارش و همین جمع درآمد را تولید کند.

بخش نهم • تمرین و ارزیابی

تمرین‌های پایه؛ قبل از دیدن پاسخ

برای هر تمرین یک بخش جدید در فایل practice.R بسازید. ابتدا حدس بزنید خروجی چه می‌شود، سپس اجرا کنید. پاسخ فقط عدد نهایی نیست؛ باید بتوانید توضیح دهید کدام داده وارد محاسبه شده است.

تمرین	درخواست
۱	بردار ۱۲، ۱۵، ۱۸، NA و ۲۵ را بسازید. تعداد کل، تعداد معتبر و میانگین معتبر را محاسبه کنید.
۲	از همان بردار، مقدارهای معتبرِ حداقل ۱۵ را نگه دارید.
۳	جدول سه دانشجو با نمره‌های ۱۸، ۱۴ و ۱۶ بسازید و ستون قبولی برای حداقل ۱۵ اضافه کنید.
۴	قیمت‌های ۱۰۰ و ۲۰۰ هزار تومان را با تابعی که ورودی تخفیف دارد، ۱۰٪ کاهش دهید.
۵	تاریخ‌های 05-01-2026 و 10-02-2026 را به Date تبدیل و فاصلهٔ روزها را حساب کنید.
۶	factor با برچسب‌های 10 و 20 بسازید و تفاوت تبدیل مستقیم و تبدیل از مسیر متن را نشان دهید.

سه سؤال برای توضیح شفاهی

- چرا mean(x) با وجود NA خروجی NA می‌دهد؟
- چرا در شمارش سفارش‌ها باید معنای هر ردیف را بدانیم؟
- چرا x == NA راه تشخیص مقدار گمشده نیست؟

> **اگر متوقف شدید**
>
> به نام تابع راهنما بگیرید، نوع شیء را با str ببینید و مسئله را به دو خط کوچک‌تر تقسیم کنید. هدف حفظ‌کردن کد نیست؛ پیدا کردن دلیل رفتار آن است.

بخش نهم • تمرین و ارزیابی

تمرین‌های داده و نمودار

تمرین	درخواست
۷	روی sales_raw.csv تعداد ردیف خام، تکراری دقیق و شناسه‌های تکراری را جداگانه بررسی کنید.
۸	با دادهٔ معتبر سفارش‌های تکمیل‌شده، تعداد سفارش و درآمد هر کانال را محاسبه کنید.
۹	برای جدول عریضِ دو شهر و دو ماه، نسخهٔ بلند بسازید و دوباره به عریض برگردانید.
۱۰	جدول شهر و منطقه بسازید و به سفارش‌ها وصل کنید. کنترل کنید تعداد سفارش عوض نشده باشد.
۱۱	برای دادهٔ iris نمودار پراکندگی بسازید؛ رنگ بر اساس Species باشد و PNG خروجی بگیرید.
۱۲	برای شهرها درآمد را بر تعداد سفارش تقسیم کنید و رتبه‌بندی میانگین سفارش را با رتبه‌بندی درآمد مقایسه کنید.

برای هر خروجی ثبت کنید

- نام داده و تعداد ردیف‌هایی که وارد محاسبه شدند.
- قاعدهٔ برخورد با مقدار گمشده یا تکراری.
- تعریف صورت و مخرج شاخص.
- عنوان، واحد و مسیر ذخیرهٔ شکل یا جدول.

دام تمرین ۱۰

عمداً یک شهر را در جدول منطقه دو بار تکرار کنید. ببینید کنترل یکتایی چگونه جلوی تکثیر مبلغ را می‌گیرد؛ بعد جدول مرجع را اصلاح کنید. از حذف بی‌دلیل سطر اضافه برای پنهان‌کردن خطا خودداری کنید.

بخش نهم • تمرین و ارزیابی

تمرین‌های مدل‌سازی و پروژه

تمرین	درخواست
۱۳	mtcars را با seed برابر ۴۲ به آموزش ۷۵٪ و آزمون ۲۵٪ تقسیم کنید؛ مدل mpg با wt و hp را بسازید و RMSE را با خط مبنا مقایسه کنید.
۱۴	توضیح دهید چرا استانداردسازی روی کل داده پیش از تقسیم می‌تواند نشت اطلاعات باشد.
۱۵	در مثال لجستیک، آستانهٔ ۰٫۳ و ۰٫۷ را روی همان مجموعهٔ آزمون برای مشاهدهٔ تغییر ماتریس خطا مقایسه کنید؛ از این آزمون برای انتخاب نهایی آستانه استفاده نکنید.
۱۶	برای پروژهٔ فروش، سهم درآمد هر شهر از کل را محاسبه کنید و بررسی کنید سهم‌ها به یک جمع شوند.
۱۷	یک پاراگراف گزارش بنویسید که درآمد، تعداد سفارش، کیفیت داده و یک محدودیت را پوشش دهد.
۱۸	R را Restart کنید و پروژه را بدون اجرای دستیِ خط‌های پراکنده، دوباره از ابتدا بسازید.

در تمرین ۱۵ هدف آموزشی، فهم اثر آستانه است. در کاربرد واقعی، مقایسه و انتخاب آستانه باید روی اعتبارسنجی انجام شود و آزمون نهایی تا پایان دست‌نخورده بماند.

ارزیابی شخصی

برای هر تمرین از صفر تا دو امتیاز بدهید: صفر یعنی اجرا نمی‌شود؛ یک یعنی اجرا می‌شود ولی تفسیر یا کنترل ندارد؛ دو یعنی اجرا، کنترل و توضیح کامل است. اگر در کنترل داده و تعریف شاخص مشکل دارید، پیش از مدل‌های پیچیده به پروژهٔ فروش برگردید.

بخش نهم • تمرین و ارزیابی

پاسخ تمرین‌های پایه

```
# 1 and 2
x <- c(12, 15, NA, 18, 25)
c(total = length(x), valid = sum(!is.na(x)),
  average = mean(x, na.rm = TRUE))
x[!is.na(x) & x >= 15]
# total=5, valid=4, average=17.5; selected=15,18,25

# 3
students <- data.frame(name = c("Sara", "Ali", "Nima"),
                       score = c(18, 14, 16))
students$passed <- students$score >= 15

# 4: valid nonnegative prices, one rate between 0 and 1
net_price <- function(price, discount = 0) {
  stopifnot(all(is.finite(price)), all(price >= 0),
            length(discount) == 1, is.finite(discount),
            discount >= 0, discount <= 1)
  price * (1 - discount)
}
net_price(c(100000, 200000), 0.1)
# 90000, 180000

# 5 and 6
d <- as.Date(c("2026-01-05", "2026-02-10"))
as.integer(diff(d))
# 36
f <- factor(c("10", "20"))
as.numeric(f)
as.numeric(as.character(f))
```

در تمرین ۱ مخرج میانگین ۴ است، چون یک مقدار گمشده داریم. در تمرین ۳ فقط دانشجوی نمرهٔ ۱۴ مردود می‌شود. در تمرین ۶ تبدیل مستقیم کد سطح‌ها را می‌دهد، ولی تبدیل از متن به عدد مقادیر ۱۰ و ۲۰ را بازمی‌گرداند.

پاسخ‌های متفاوت می‌توانند درست باشند

اگر کد شما همان تعریف و همان کنترل‌ها را با R پایه یا بستهٔ دیگری پیاده می‌کند، لازم نیست دقیقاً شبیه این کد باشد.

بخش نهم • تمرین و ارزیابی

پاسخ تمرین‌های داده

```
library(dplyr)
source("scripts/02_clean_base.R")

# 7
c(raw_rows = nrow(raw),
  exact_duplicates = sum(duplicated(raw)))
# 121, 1

# 8
orders |>
  group_by(channel) |>
  summarise(n_orders = n(), revenue = sum(net), .groups = "drop")

# 10
lookup <- data.frame(
  city = c("Tehran", "Shiraz", "Tabriz", "Isfahan"),
  region = c("central", "south", "northwest", "central")
)
stopifnot(!anyDuplicated(lookup$city))
x <- left_join(orders, lookup, by = "city",
               relationship = "many-to-one")
stopifnot(nrow(x) == nrow(orders), !anyNA(x$region))

# 12
orders |>
  group_by(city) |>
  summarise(n = n(), revenue = sum(net),
            average = sum(net) / n(), .groups = "drop") |>
  arrange(desc(average))
```

پاسخ تمرین ۹ همان الگوی pivot_longer و pivot_wider در درس تبدیل شکل جدول است. برای تمرین ۱۱، نمونهٔ ggplot2 و ggsave را با مسیر دلخواه اجرا کنید؛ فایل را باز کنید و وجود راهنمای رنگ و واحد محورها را بررسی کنید.

کنترل تعداد ردیفِ اتصال لازم است اما کافی نیست؛ وجود منطقهٔ گمشده یا کلید نامعتبر را هم بررسی کنید. در مسئلهٔ واقعی، جدول منطقه باید از منبع معتبر کسب‌وکار تهیه شود.

بخش نهم • تمرین و ارزیابی

پاسخ مدل‌سازی و معیار قبولی پروژه

پاسخ اجرایی تمرین ۱۳ در scripts/04_statistics_models.R آمده است. تقسیم تصادفی ثابت است و مدل با میانگین آموزش به‌عنوان خط مبنا مقایسه می‌شود. هر دو معیار باید بر همان ردیف‌های آزمون محاسبه شوند.

```
# 15: continue the logistic example
for (threshold in c(0.3, 0.7)) {
  pred_class <- as.integer(prob >= threshold)
  print(table(actual = factor(test$bought, levels = 0:1),
              predicted = factor(pred_class, levels = 0:1)))
}

# 16: independent project calculation
source("scripts/02_clean_base.R")
by_city <- aggregate(net ~ city, data = orders, sum)
by_city$share <- by_city$net / sum(by_city$net)
stopifnot(isTRUE(all.equal(sum(by_city$share), 1)))
stopifnot(nrow(orders) == 101)
stopifnot(isTRUE(all.equal(sum(orders$net), 77418500)))
```

پاسخ تمرین ۱۴: پارامترهایی که با کل داده محاسبه شوند، اطلاعات توزیع آزمون را به آموزش وارد می‌کنند. ابتدا داده را جدا کنید؛ میانگین، انحراف معیار یا مدل جایگزینی را از آموزش بگیرید و عیناً روی آزمون اعمال کنید.

قبولی پروژهٔ نهایی

اجرای دوباره موفق باشد؛ شمارش کیفیت به ۱۰۱ سفارش برسد؛ درآمد ۷۷٬۴۱۸٬۵۰۰ تومان باشد؛ سهم شهرها به یک جمع شود؛ فایل‌های خروجی ساخته شوند؛ و گزارش روشن کند که داده ساختگی است و درآمد با سود فرق دارد.

اگر عدد شما متفاوت است، اول ترتیب پاک‌سازی، وضعیت سفارش، واحد پول و حذف تکرار را بررسی کنید. تغییر ظاهر جدول نباید عدد شاخص را عوض کند.

پیوست • مراجعهٔ سریع

برگهٔ تقلب؛ مبانی و داده

نیاز	کد یا تابع
ساخت بردار	c(10, 20, 30)
دنباله و تکرار	seq(1, 10, by = 2) و rep(0, 5)
نوع و ساختار	class(x)، typeof(x)، str(x)
اندازه	length(x)، nrow(df)، ncol(df)
خلاصه و نمایش	head(df)، summary(df)، names(df)
انتخاب ستون	df$price یا df[, "price"]
انتخاب ردیف	df[df$price > 0 & !is.na(df$price),]
گمشده	is.na(x) و sum(is.na(x))
تکرار	duplicated(df) و anyDuplicated(df$id)
متن	trimws(x)، gsub() و grepl()
تاریخ	as.Date(x, format = "%Y-%m-%d")
فایل	file.exists()، list.files()، file.path()
CSV	read.csv() و write.csv(..., row.names = FALSE)
شیء R با حفظ کلاس	saveRDS(x, "x.rds") و readRDS("x.rds")
انتخاب ستون و ردیف با dplyr	select() و filter()
تغییر و خلاصه	mutate()، group_by() و summarise()
تغییر شکل جدول	pivot_longer() و pivot_wider()
اتصال و بررسی تطبیق	left_join() و anti_join()

پیش از هر محاسبه

واحد داده، نوع ستون، تعداد ردیف، گمشده‌ها و معنای هر ردیف را بررسی کنید. این پنج کنترل از حفظ تعداد زیادی تابع مفیدترند.

پیوست • مراجعهٔ سریع

برگهٔ تقلب؛ آمار و خروجی

نیاز	کد یا تابع
مرکز	mean(x, na.rm = TRUE) و median(x)
پراکندگی	sd(x)، IQR(x) و quantile(x)
فراوانی و سهم	table(x) و prop.table(table(x))
پراکندگی دو متغیر	plot(x, y)
توزیع	hist(x) و boxplot(x)
نمودار لایه‌ای	ggplot(df, aes(x, y)) + geom_point()
ذخیرهٔ نمودار	ggsave("plot.png", plot = p, dpi = 300)
آزمون دو گروه	t.test(y ~ group, data = df)
آزمون زوجی	t.test(after, before, paired = TRUE)
جدول دسته‌ای	chisq.test(tab) یا fisher.test(tab)
همبستگی	cor.test(x, y, method = "pearson")
مدل خطی	lm(y ~ x1 + x2, data = df)
مدل لجستیک	glm(y ~ x, data = df, family = binomial())
خلاصهٔ مدل	summary(fit)، coef(fit) و confint(fit)
پیش‌بینی	predict(fit, newdata = test)
RMSE	sqrt(mean((actual - pred)^2))
محیط قابل‌بازتولید	set.seed(42)، sessionInfo() و renv::snapshot()
کنترل ساده	stopifnot() و all.equal()

توابع این جدول دستور انتخاب روش نیستند. انتخاب درست به نوع سؤال، استقلال مشاهده‌ها، طرح نمونه‌گیری، کیفیت داده و فرض‌های مدل وابسته است. برای گزارش، واحد و عدم‌قطعیت را کنار عدد بنویسید.

پیوست • مراجعهٔ سریع

خطاهای رایج؛ از پیام به راه‌حل

خطا یا نشانه	علت محتمل و اقدام نخست
object ... not found	خط سازندهٔ شیء اجرا نشده یا نام با حروف متفاوت نوشته شده است.
could not find function	بسته بارگذاری نشده، تابع هم‌نام اشتباه است یا نام تابع غلط است.
non-numeric argument	ورودی محاسبه متن یا نوع نامناسب دارد؛ str را بررسی کنید.
undefined columns selected	ستون در داده وجود ندارد؛ names و جداکنندهٔ فایل را ببینید.
subscript out of bounds	شماره یا نام انتخابی از محدودهٔ داده بیرون است.
cannot open the connection	مسیر، وجود فایل، مجوز خواندن یا پوشهٔ خروجی را بررسی کنید.
NAs introduced by coercion	حداقل یک مقدار به نوع هدف تبدیل نشده است؛ ورودی‌های شکست‌خورده را جدا کنید.
missing value where TRUE/FALSE needed	شرط if مقدار NA داده است؛ گمشده را پیش از شرط مدیریت کنید.
figure margins too large	فضای نمودار نسبت به حاشیه‌ها کوچک است؛ پنجره را بزرگ یا حاشیه را تنظیم کنید.
علامت + در Console	دستور ناقص است؛ پرانتز و کوتیشن را ببندید یا با Esc لغو کنید.
جمع فروش بعد از join زیاد شد	کلید سمت راست یکتا نیست یا رابطهٔ اتصال درست تعریف نشده است.
عدد فارسی یا متن ناخوانا	کدگذاری، قالب عدد، فونت و مسیر ورود/خروج داده را جداگانه بررسی کنید.

وقتی کمک می‌خواهید

متن کامل خطای اول، چند ردیف دادهٔ بی‌نام، str ورودی، کد حداقلی تولیدکنندهٔ خطا و sessionInfo را آماده کنید. تصویر خطا به‌تنهایی معمولاً اطلاعات کافی ندارد.

پیوست • مسیر یادگیری

برنامهٔ مطالعه و قدم بعد

زمان پیشنهادی زیر فقط برای برنامه‌ریزی است؛ هر روز حدود ۴۵ تا ۹۰ دقیقه اجرای فعال در نظر بگیرید. اگر یک تمرین اجرا شد اما نمی‌توانید خروجی را توضیح دهید، آن بخش هنوز کامل نشده است.

بازه	موضوع و خروجی
روزهای ۱ و ۲	نصب، محیط، پروژه و اولین اسکریپت
روزهای ۳ تا ۵	بردار، نوع داده، جدول، تابع و گمشده
روزهای ۶ تا ۸	ورود CSV/Excel، پاک‌سازی، گروه‌بندی و اتصال
روزهای ۹ و ۱۰	آمار توصیفی، نمودار پایه و ggplot2
روزهای ۱۱ تا ۱۳	آزمون آماری، همبستگی و رگرسیون
روزهای ۱۴ و ۱۵	ارزیابی پیش‌بینی، لجستیک و سری زمانی
روزهای ۱۶ تا ۱۸	پروژهٔ فروش، گزارش، اجرای دوباره و پاسخ تمرین

بعد از این جزوه مسیر را با مسئله انتخاب کنید

- تحلیل کسب‌وکار: SQL، تعریف شاخص و طراحی گزارش قابل‌بازتولید.
- پژوهش: طراحی مطالعه، استنباط، مدل‌های متناسب با طرح و تحلیل حساسیت.
- پیش‌بینی: اعتبارسنجی، مهندسی ویژگی و ارزیابی روی دادهٔ دیده‌نشده.
- ساخت ابزار: Shiny، ساختار برنامه، کنترل ورودی و مدیریت دسترسی.

چه زمانی آمادهٔ پروژهٔ بعدی هستید؟

وقتی بتوانید از یک فایل خام تازه، با قواعد مکتوب، جدول معتبر بسازید؛ یک نمودار مرتبط تولید کنید؛ عدد اصلی را کنترل کنید؛ و نتیجه را با محدودیت‌های آن توضیح دهید. حفظ همهٔ توابع شرط شروع پروژهٔ بعدی نیست.

پیوست • منابع قابل کلیک

منابع رسمی؛ مبانی و کار با داده

متن و مثال‌های این جزوه برای مسیر آموزشی حاضر تنظیم شده‌اند. برای جزئیات آرگومان‌ها، سازگاری نسخه‌ها و امکانات تازه، منابع رسمی زیر را بررسی کنید. نام منابع و نشانی‌ها در PDF قابل کلیک‌اند. تاریخ بررسی منابع: ۶ سپتامبر ۲۰۲۶.

مطالعهٔ تکمیلی: R Project؛ معرفی و دسترسی به منابع

مطالعهٔ تکمیلی: CRAN؛ دریافت R و بسته‌ها

مطالعهٔ تکمیلی: An Introduction to R؛ راهنمای رسمی مبانی

مطالعهٔ تکمیلی: R Data Import/Export؛ ورود و خروج داده

مطالعهٔ تکمیلی: Posit؛ دانلود RStudio

مطالعهٔ تکمیلی: RStudio IDE User Guide؛ محیط و تنظیمات

مطالعهٔ تکمیلی: dplyr؛ فعل‌های کار با داده

مطالعهٔ تکمیلی: dplyr؛ قواعد اتصال جدول‌ها

مطالعهٔ تکمیلی: tidyr؛ تبدیل جدول به حالت بلند

مطالعهٔ تکمیلی: readr؛ نوشتن فایل و کدگذاری خروجی

مطالعهٔ تکمیلی: readxl؛ ورود فایل‌های Excel

مطالعهٔ تکمیلی: DBI؛ ارتباط استاندارد R با پایگاه داده

دانلود و نسخهٔ نرم‌افزار

به‌جای لینک مستقیمِ نصب‌کنندهٔ یک نسخهٔ خاص، به صفحهٔ رسمی دانلود مراجعه کنید تا فایل مناسب سیستم‌عامل و نسخهٔ پشتیبانی‌شده را انتخاب کنید.

پیوست • منابع قابل کلیک

منابع رسمی؛ نمودار، مدل و گزارش

مطالعهٔ تکمیلی: ggplot2؛ مرجع نمودارهای لایه‌ای

مطالعهٔ تکمیلی: ggsave؛ ذخیرهٔ شکل و تنظیم ابعاد

مطالعهٔ تکمیلی: geom_smooth؛ خط برازش و نوار عدم‌قطعیت

مطالعهٔ تکمیلی: t.test؛ آزمون میانگین و فاصلهٔ اطمینان

مطالعهٔ تکمیلی: lm؛ مدل خطی و فرمول‌ها

مطالعهٔ تکمیلی: glm؛ مدل‌های خطی تعمیم‌یافته

مطالعهٔ تکمیلی: prcomp؛ تحلیل مؤلفه‌های اصلی

مطالعهٔ تکمیلی: Quarto؛ اجرای R در گزارش

مطالعهٔ تکمیلی: renv؛ مدیریت وابستگی‌های پروژه

مطالعهٔ تکمیلی: Shiny؛ ساخت اولین برنامهٔ تعاملی

یادداشت فنی این نسخه

اسکریپت‌های پایه، پاک‌سازی و تحلیل فروش، نمونه‌های اصلی آمار و مدل‌سازی و نمودارهای پایه در محیط R نسخهٔ ۴٫۶٫۰ اجرا و نتایج اصلی کنترل شده‌اند. فایل‌های تمرین همراه، راهنمای اجرای مستقل دارند. نمونه‌های وابسته به بسته، Excel، گزارش و اپ با مستندات بازبینی شده‌اند، اما اجرای کامل آن‌ها در این نسخه تأیید نشده است؛ پیش‌نیازهای هر بخش در متن آمده‌اند.

ظاهر منوها، پیام نصب یا جزئیات خروجی چاپ‌شده ممکن است با نسخه و سیستم‌عامل فرق کند. برای مقایسهٔ تمرین، تعریف شاخص، تعداد ردیف و مقدار عددیِ کنترل‌شده را ملاک بگیرید. در محیط‌های متفاوت، نتایج تصادفی و محاسبات ممیزشناور ممکن است تفاوت جزئی داشته باشند.

> **از خواندن به اجرا**
>
> پروژه را از پوشهٔ خام تا گزارش نهایی یک‌بار خودتان انجام دهید. بهترین نشانهٔ یادگیری این است که بتوانید با داده‌ای تازه، همان فرایند را آگاهانه تکرار و نتیجه را توضیح دهید.