Quantitative Research Methods

Chapter 1: Introduction

Prof. Dr. Christoph Weisser

HSBI

Summer Semester 2026

The course at a glance

	Lecture	Topic
▶	1	**Ch. 1 + 2 — Introduction; what is statistical learning**
	2	Ch. 2 — Accuracy, bias–variance, Bayes classifier, KNN
	3–4	Ch. 3 — Linear regression (estimation, inference, diagnostics)
	5–6	Ch. 4 — Classification (logistic, LDA/QDA, naive Bayes, ROC)
	7	Ch. 5 — Resampling (validation, CV, bootstrap)
	8	Ch. 6 — Model selection and regularisation (ridge, lasso)
	9	Ch. 7 — Moving beyond linearity (splines, GAMs)
	10	Ch. 8 — Tree-based methods (trees, forests, boosting)
	11	Ch. 10 — Deep learning (MLPs, CNNs, training)
	12	Ch. 13 — Multiple testing (FWER, FDR)

Twelve sessions of 180 minutes. ▶ marks today's chapter. Short exercises every ∼20 min; extended exercises every ∼45 min; each chapter has a companion lab notebook.

Contents

Optional and advanced material is collected in the **appendix** at the end of the deck.

Notation in this chapter

Symbol	Meaning
n	number of observations (sample size)
p	number of variables (predictors)
x_i	predictor value(s) of the i-th observation
y_i	response value of the i-th observation
$\mathbf{X}$	$n \times p$ matrix of all predictor values
$\mathbf{y}$	n-vector stacking the responses $y_1, \ldots, y_n$
x_{ij}	value of variable j for observation i
$Y = f(X) + \varepsilon$	general model: systematic part plus noise
f	unknown true function linking X to Y
ε	random error term with mean zero
$\hat{f}$	estimate of f learned from the data
$\hat{Y} = \hat{f}(X)$	prediction of Y at input X

Sources and references

Primary textbook

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). *An Introduction to Statistical Learning, with Applications in Python.* Springer.

About these slides

Each idea is introduced with motivation, intuition, and a worked example. Slide titles are descriptive; key formulas are read symbol-by-symbol.

Learning objectives for this chapter

By the end of this chapter you will be able to:

- **Define** statistical learning and explain why it matters.
- **Distinguish** supervised from unsupervised learning, and regression from classification.
- **Articulate** the difference between prediction and inference, and explain when each goal applies.
- **Read** the standard notation: n, p, design matrix X, response vector y.
- **Recognise** the three motivating data sets (Wage, Smarket, NCI60) and the type of question each illustrates.
- **Set up** the Python tooling we will use throughout.

Where this chapter is used in industry

Sector	Concrete application	Which decision it drives
Retail	Weekly demand forecast per article and store	Replenishment orders, safety stock, allocation
Banking	Credit scoring at application; expected credit loss (IFRS 9)	Accept / decline, credit limit, loan-loss provisions
Insurance	Claim frequency and severity modelling	Premium rates and technical reserves
Manufacturing, aviation	Predictive maintenance from sensor and usage data	When to service a part before it fails
Telco, SaaS	Churn score per subscriber	Who receives a retention offer
E-commerce, streaming	Recommendation and search ranking	What the customer is shown next

In industry — The common thread

Every row needs the same three things: a response worth predicting, predictors that are *available before* the decision, and a decision that changes when the number changes. Without the third, the model is a report, not a tool.

Industry case in depth: predictive maintenance at a manufacturer

In industry — The learning problem

Response Y: will this asset need an intervention within the next N operating days or cycles? — binary, supervised and heavily imbalanced. **Predictors**: telemetry aggregates (vibration, temperature, pressure), operating hours and load, maintenance and fault history, asset age, ambient conditions.

The fitted output — and who signs it off

A *risk score per asset per period*, not a failure date. The decision: pull the asset into the next planned maintenance window, or let it run. Maintenance planning and reliability engineering own the score; operations sign off, because an intervention costs production time. The score is read by the production planner on the affected line, who either releases the asset into the window or pushes back because the order book cannot absorb the downtime.

One honest caveat

Failures are rare, so labels are scarce — and every failure the model prevents is a *censored* observation nobody can confirm: its own success degrades its future training data. The value lies in the decision it changes, not in the score.

Two business problems we follow all semester

Not twelve unrelated anecdotes: two business problems return chapter after chapter.

A retail bank's credit scorecard	
Ch. 00b	the log-odds scale a scorecard is built on
Ch. 04	the scorecard itself: PD, cut-off, reason codes
Ch. 05	independent validation, out-of-time testing
Ch. 06	which bureau attributes make the final card
Ch. 07	why scorecards bin inputs into bands
Ch. 08	the boosted challenger model
Ch. 13	fairness testing across many subgroups

A retailer's pricing and demand question	
Ch. 00b	elasticity from a log–log model
Ch. 02	what is irreducible in a demand forecast
Ch. 03	marketing mix modelling and the budget split
Ch. 07	the price-response curve that saturates
Ch. 08	the boosted demand forecast
Ch. 10	one network across thousands of series

In industry — Banking and retail: one problem — many chapters

Neither list is a tour of techniques. The same decision — accept this applicant, set this price — keeps coming back, and each chapter adds the piece the previous tool could not supply: a validated card, a shorter list of attributes, a curve that saturates. Methods are chosen by what the decision needs, never because they are new.

Outline

Roadmap of this chapter

How this chapter is organised

1. Why statistical learning matters today.
2. Definitions: supervised vs. unsupervised, regression vs. classification.
3. Three motivating examples: Wage, Smarket, NCI60.
4. Notation: n, p, design matrix X.
5. The Python ecosystem we will use.

Each section ends with a mini-check question to help you self-assess.

Why statistical learning is the most-asked-for skill

Every discipline now collects more data than it can analyse by hand.

The pressure

Storage is essentially free; sensors, transactions, and web interactions generate terabytes per minute. The bottleneck has shifted from *collecting* data to *understanding* it.

Hal Varian quote

"I keep saying that the sexy job in the next ten years will be statisticians. And I'm not kidding."

What this course is and is not

What it is

A course on the *statistical foundations* of learning. We focus on *ideas*, interpretability, and honest evaluation.

What it is not

A pure machine-learning engineering course. We will not deploy deep nets at production scale.

The course's promise

Two questions for every model: (1) *What does this model assume about the world?* (2) *What happens when those assumptions fail?*

Prerequisites

Mathematics

Linear algebra (matrices, eigenvectors), introductory calculus (derivatives, gradients), basic probability.

Statistics

Descriptive statistics; simple/multiple linear regression at the level of an introductory econometrics course.

Programming

Some Python experience is helpful but not required — we start from the basics in the lab sessions.

The convergence of statistics and computer science

Statistics emphasises

- models and assumptions
- inference and uncertainty
- interpretability
- rigorous quantification of error

Machine learning emphasises

- algorithms and computation
- predictive accuracy
- flexibility and scale
- large, complex data sets

Statistical learning

The intersection: methods that are computationally feasible *and* statistically principled.

Outline

What is statistical learning?

Working definition

A vast set of tools for *understanding data*. These tools are classified as either *supervised* (we predict an output) or *unsupervised* (we look for structure without an output).

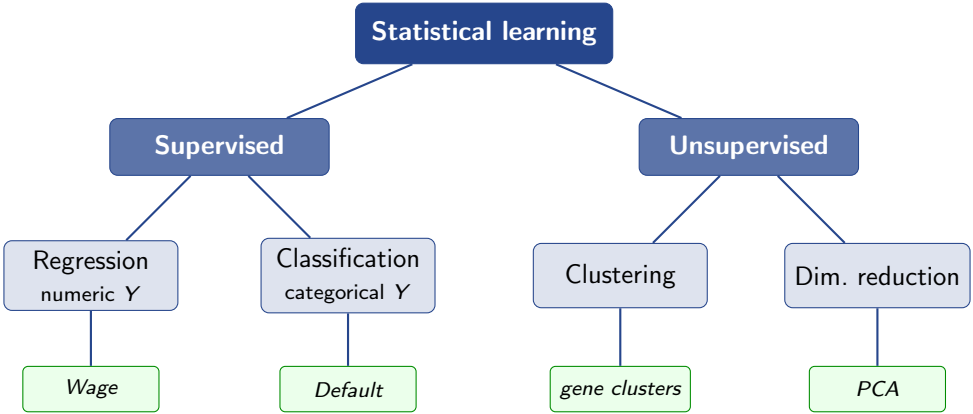
How to read this

- Supervised: regression (numerical Y) and classification (categorical Y).
- Unsupervised: clustering, dimension reduction, density estimation.
- Closely related fields: machine learning, data mining, pattern recognition, predictive analytics.

In industry — Analytics teams: which half a firm's questions fall into

Supervised, because the outcome was recorded: churn scores, credit decisions, demand forecasts. Unsupervised: segmentation, anomaly detection — no label, so no "accuracy".

The taxonomy of statistical learning



Takeaway

A response Y splits the field: supervised (predict Y) vs. unsupervised (discover structure).

The fundamental setup

We observe a sample

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

where each $x_i \in \mathbb{R}^p$ is a vector of p *predictors* and y_i is a *response*.

How to read this

Symbol	Meaning
n	Number of observations (sample size)
p	Number of predictors per observation
x_i	Vector of p predictors for observation i
y_i	Response for observation i (number or category)

In the unsupervised case there is no y_i — we only have $x_1, \ldots, x_n$.

Two distinct goals: prediction vs. inference

Prediction

Given a new x_0, what value of Y should we expect?

Inference

How does Y depend on X? Which features matter? In what direction? By how much?

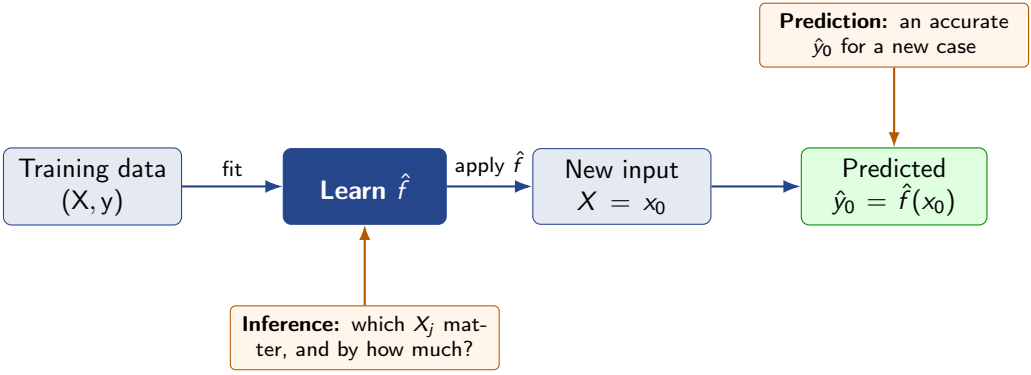
Two examples

Prediction: given a patient's measurements, will they survive 5 years?
Inference: which marketing channels actually drive sales?

Which goal you have shapes the method

Most real problems involve both. Prediction can tolerate a black-box model; inference demands an interpretable $\hat{f}$ whose structure we can read off.

The supervised-learning workflow



Takeaway

We learn $\hat{f}$ from training data, then reuse it on new inputs — prediction targets $\hat{y}_0$; inference targets the structure of $\hat{f}$.

Exercise 1.1 — Prediction vs. Inference [Concept]

Exercise

For each scenario, decide whether the *primary* goal is **prediction** or **inference**, and justify in one line.

1. A bank wants to know *which* applicant characteristics raise default risk, and *by how much*.
2. A streaming service auto-fills the rating a user would give a film they have not seen.
3. An economist estimates the effect of a minimum-wage rise on employment.
4. A hospital flags high-risk patients from lab values; it need not explain the reasoning.
5. For a *pure inference* goal, would you prefer a linear model or a deep neural network? Why?

Solution — Exercise 1.1 (1/2)

Solution

How to decide. Ask what the *deliverable* is: an accurate value $\hat{Y}$ for new cases means *prediction*; a statement about *how* Y depends on the X_j (which ones, in which direction, by how much) means *inference*.

1. **Inference.** The deliverable is which predictors matter and the direction/magnitude of their effect, not a single forecast. The give-away words are "*which* characteristics" and "*by how much*" — both ask about the structure of f, so the bank needs a model whose coefficients it can read off and defend.
2. **Prediction.** Only the accuracy of the filled-in rating matters; *why* the user would rate it that way is irrelevant. The rating is consumed directly by the recommender — nobody inspects the mechanism — so a black-box $\hat{f}$ is perfectly acceptable.
3. **Inference.** The goal is a causal effect size (a coefficient with a confidence interval). A forecast of employment alone would not answer the policy question "what happens *if* the minimum wage rises?".

Solution — Exercise 1.1 (2/2)

Solution

4. **Prediction.** Accurate flagging is what counts; interpretability is secondary here. The hospital acts on the flag itself (direct attention to high-risk patients), so the model is judged purely on its held-out accuracy.
5. **Linear model.** Its coefficients have direct meaning and come with standard errors, confidence intervals and p-values, so we can quantify *how* Y depends on each X_j. A deep network predicts well but is a black box — its weights do not answer an inference question.

Rule of thumb: prediction cares only about $\hat{Y}$; inference cares about the *structure* of f.

Common mistake

Classifying by the *tool* instead of the *question* ("it uses regression, so it must be inference"). Any model can serve either goal — what decides is whether the answer you hand over is $\hat{Y}$ itself or the shape of f.

Outline

Three motivating examples set the tone

We will study three very different data sets, each illustrating one major flavour of problem.

Takeaway

- **Wage**: regression, mostly inference.
- **Smarket**: classification, weak signal.
- **NCI60**: unsupervised, structure discovery.

In industry — From HR to trading: the same three flavours

A *Wage*-type question is a pay-equity or compensation review — the coefficient is the deliverable. A *Smarket*-type question is a trading or forecasting signal with a very low signal-to-noise ratio. An *NCI60*-type question is segmenting customers or products where no label exists.

Example 1: Wage — a regression problem

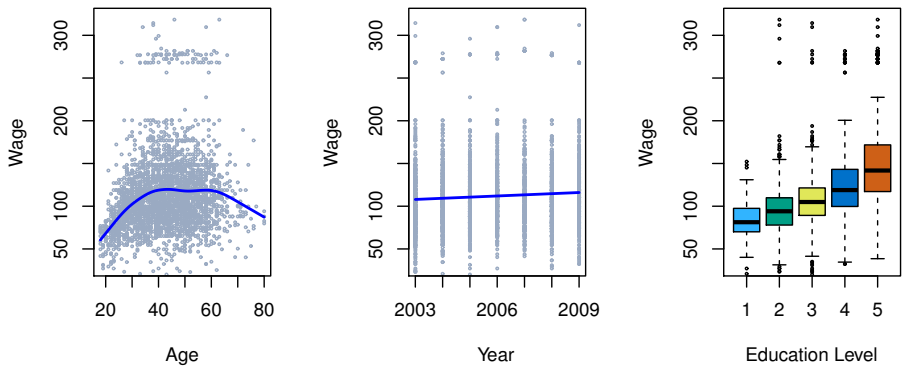
The data

Survey of 3,000 men working in the central Atlantic region of the U.S. Response `wage` is continuous, so this is a *regression* problem.

Variables

- Response: `wage` (annual income in thousands of dollars).
- Predictors: `age`, `year`, `education`, plus marital status, race, region.

Wage data: wage vs. age, year, and education



Source: Figure 1.1 — ISLP, James et al. (2023).

What the Wage plots tell us

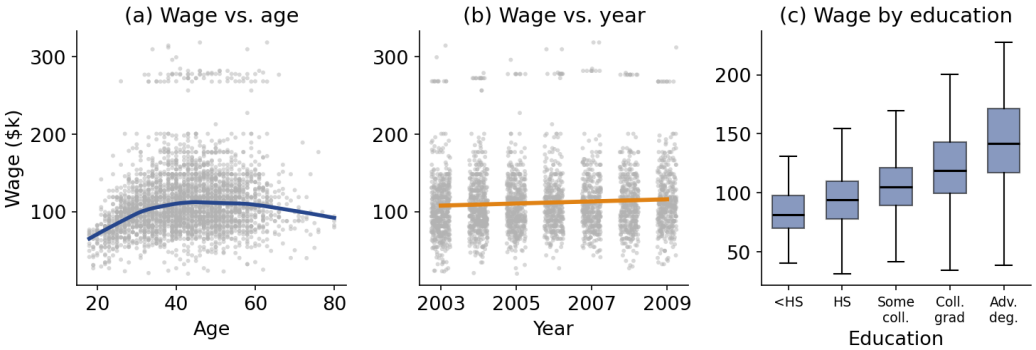
Reading the three panels left to right:

- **Wage vs. age**: increasing until around 60, then a slight decline — a *nonlinear* relationship.
- **Wage vs. year**: a slow upward trend — wages grow modestly over time.
- **Wage vs. education**: a strong, monotone effect — higher education levels mean higher wages.

Modelling implication

A purely linear regression in age would miss the curvature. We will need polynomial terms or splines (Ch. 7).

Wage data up close: age, year, and education



Takeaway

Wage rises then dips with age (nonlinear), drifts gently upward over the years (linear), and climbs steeply with education.

Example 2: Smarket — a classification problem

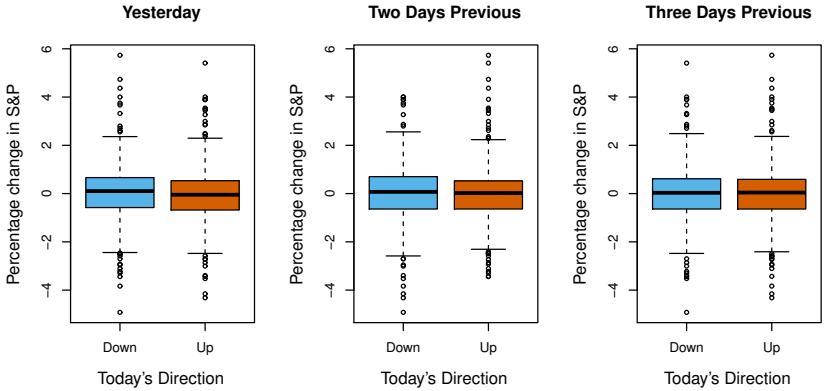
The data

Daily percentage returns of the S&P 500 from 2001–2005. Response `Direction` (`Up`/`Down`) is binary, so this is a *classification* problem.

Variables

- Response: `Direction` (`Up` / `Down`).
- Predictors: `Lag1`, . . . , `Lag5` (previous five days' returns) and `Volume`.

Smarket: previous-day returns by direction



Source: Figure 1.2 — ISLP, James et al. (2023).

Reading the boxplots

Each panel compares returns from one, two, and three days ago for days where the market went `Up` vs. `Down`.

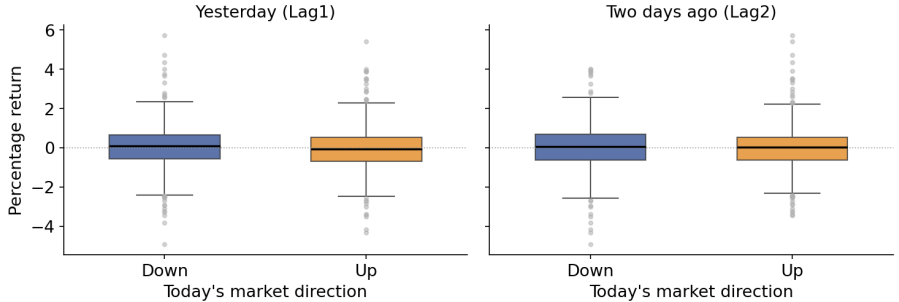
The bad news

The boxplots are *nearly identical*. Past returns barely predict the next day's direction — consistent with the efficient-markets hypothesis.

A cautionary tale

Even when a problem "looks like" a classification problem, it does not follow that we can solve it well. Prediction is only possible if predictors actually carry information about the response.

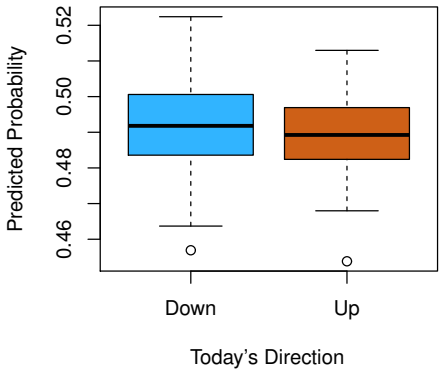
Recreating Figure 1.2 from the raw Smarket data



Takeaway

Median `Lag1` from `Smarket.csv`: $+0.10\,\%$ before Down days vs. $-0.05\,\%$ before Up days — a 0.15-point gap against boxes spanning $\pm 0.6\,\%$. Yesterday's return barely separates the classes: market prediction is hard.

Smarket: a tiny but real signal



Quadratic discriminant analysis on the same predictors yields about 60 % accuracy on held-out 2005 data — a small but potentially profitable edge.

Source: Figure 1.3 — ISLP, James et al. (2023).

Example 3: NCI60 — an unsupervised problem

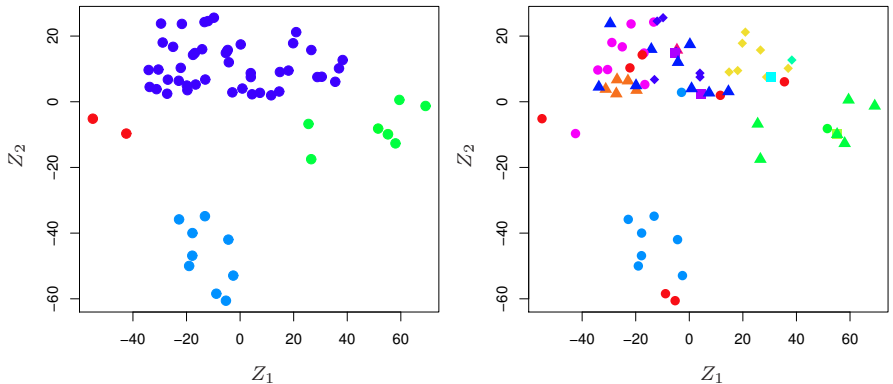
The data

NCI60: 64 cancer cell lines, 6,830 gene expression measurements each. *No* response variable — we don't know what to predict.

Takeaway

Goal: discover *groups* of similar cell lines (clustering) and *visualise* the high-dimensional data in 2D (dimension reduction). This is unsupervised learning.

NCI60: the first two principal components



Cell lines of the same cancer type cluster together. PCA recovered biologically meaningful structure from raw gene expression alone — without ever seeing the labels.

Source: Figure 1.4 — ISLP, James et al. (2023).

What the three examples illustrate

The course's three flavours

- **Wage**: a quantitative response, mostly an *inference* question $\rightarrow$ regression.
- **Smarket**: a categorical response, mostly a *prediction* question $\rightarrow$ classification.
- **NCI60**: no response, structural questions $\rightarrow$ unsupervised learning.

These three flavours structure the rest of the course.

Mini-check: types of problems

Quick self-assessment

For each of the following, decide: regression, classification, or unsupervised?

1. Predicting next year's GDP growth from macro indicators.
2. Detecting which customers will cancel their subscription.
3. Grouping news articles into topics without prior labels.
4. Forecasting house prices from features.
5. Identifying the digit shown in a 28×28 pixel image.

Answers

1. regression. 2. classification. 3. unsupervised. 4. regression. 5. classification.

Notation: n, p, and the design matrix

Sample size and dimension

- n: number of observations
- p: number of predictors (features)

Generic notation

Object	Notation
Scalars	lower-case italic, a, b, x, y
Vectors of length n	bold lower-case, $\mathbf{a}$
Vectors of length $\neq n$	lower-case italic, x_i
Matrices	bold upper-case, $\mathbf{A}, \mathbf{X}$

Exercise 1.2 — Regression, Classification, or Unsupervised? [Concept]

Exercise

Classify each task. If it is *supervised*, also say whether it is **regression** or **classification** and name the response Y.

1. Predict a used car's selling price from mileage, age, brand.
2. Segment 64 cell lines described by 6,830 gene-expression values into similar groups.
3. Predict whether an incoming e-mail is spam.
4. Forecast tomorrow's electricity demand (MWh) from weather and calendar features.
5. Reduce 20 survey questions to a few underlying "factors".
6. Recognise which of the digits 0–9 a handwritten image shows.

Solution — Exercise 1.2 (1/2)

Solution

Method. First look for a designated response Y: none $\Rightarrow$ *unsupervised*. If there is one, its *type* decides: quantitative $\Rightarrow$ regression, qualitative $\Rightarrow$ classification.

1. **Supervised regression**; Y = price (quantitative). Past sales provide labelled pairs (features, price), and the target is a number on a continuous scale — exactly the setting squared-error loss is made for.
2. **Unsupervised** (clustering); there is no response Y. All 6,830 gene-expression values are inputs — none of them is a label — and the goal is to *discover* groups, not to predict anything.
3. **Supervised classification**; Y = spam / not-spam (binary, qualitative). The training e-mails are already labelled, and the output is a category, not a quantity.

Solution — Exercise 1.2 (2/2)

Solution

4. **Supervised regression**; Y = demand in MWh (quantitative). Weather and calendar variables are the predictors; historical demand records supply the labels.
5. **Unsupervised** (dimension reduction); no response Y. We seek a few latent "factors" that summarise 20 correlated answers — PCA / factor-analysis territory (Ch. 12).
6. **Supervised classification**; Y = digit $0, \ldots, 9$ (10 qualitative classes). The pixel intensities are *numeric predictors*, but numeric inputs do not make it a regression — the output is one of ten categories, and a "7" is not numerically closer to an "8" than to a "1" in any sense that squared error could use.

Decision rule: no $Y \Rightarrow$ unsupervised. With a Y: quantitative $\Rightarrow$ regression, qualitative $\Rightarrow$ classification. The type of the *response* decides — not the predictors.

Common mistake

Letting the predictors vote — e.g. calling digit recognition "regression" because pixels are numbers, or task 1 "classification" because `brand` is categorical. Only the type of Y matters.

Outline

Why Python?

Python is the most widely used language in industry data science and machine learning.

The scientific stack

NumPy, pandas, matplotlib, scikit-learn, statsmodels, ISLP. All free, mature, and well-documented.

How to read this

The *first* edition of the textbook used R. The methods are identical — only the syntax changes. We will use Python throughout.

In industry — Banking and retail: the production stack

pandas, scikit-learn and statsmodels are the default analytics stack in banks, insurers and retailers: `statsmodels` where a coefficient has to be defended, `scikit-learn` where a score has to be produced every night.

The core scientific stack

Library	Role
`numpy`	numerical arrays, linear algebra
`pandas`	data frames, CSV I/O, group-by
`matplotlib`	plotting (low-level)
`seaborn`	plotting (statistical)
`scipy`	scientific routines (optimisation, stats)
`statsmodels`	inference-oriented regression / GLM
`scikit-learn`	general-purpose ML
`ISLP`	book companion package

Standard imports

```
import numpy as np # arrays and linear algebra
import pandas as pd # data frames (tables)
import matplotlib.pyplot as plt # plotting

# models, data splitting / CV, and error metrics:
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import mean_squared_error, accuracy_score

from ISLP import load_data # the book's datasets
```

Adopt these as the default first cell of every notebook.

30-second tour: NumPy

```
import numpy as np
x = np.array([1, 2, 3, 4]) # 1-D array (a vector)
A = np.array([[1, 2], [3, 4]]) # 2-D array (a matrix)

x.shape # (4,) -- one axis of length 4
A.shape # (2, 2) -- 2 rows, 2 columns
A @ A # matrix product
np.linalg.inv(A) # matrix inverse
np.mean(x), np.std(x) # mean and standard deviation
```

30-second tour: pandas

```python
import pandas as pd
df = pd.read_csv("Wage.csv") # load CSV into a DataFrame

df.shape # (3000, 11) -- 3000 rows, 11 columns
df.head() # show the first 5 rows
df["wage"].mean() # mean of a single column
df.groupby("education")["wage"].mean() # mean wage per group
df.describe() # summary statistics for every column
```

30-second tour: scikit-learn

```
from sklearn.linear_model import LinearRegression

X = df[["age", "year"]].values # predictor matrix (n x 2)
y = df["wage"].values # response vector (n,)
model = LinearRegression() # choose the model...
model.fit(X, y) # ...and estimate it from data

print(model.coef_, model.intercept_) # fitted slopes, intercept
yhat = model.predict(X) # predicted wage for each row
```

Every scikit-learn model follows the same `fit` / `predict` API.

Outline

Datasets used throughout the book

ISLP comes with a curated collection of datasets that illustrate every major method.

Takeaway

All accessible via the `ISLP` Python package, also as CSV files. Most are tabular and modest in size — chosen to run on a laptop in seconds.

In industry — Data engineering: why company data never arrives tidy

The equivalent table does not exist in a firm: it has to be built by joining warehouse tables *as of the decision date*, and most of a project's time goes into that join. Pulling in a field that was only recorded *after* the decision is the classic leakage bug — the model looks excellent in testing and fails in production.

Exercise 1.3 — Notation: n, p, and X [Math]

Exercise

For five patients we record age (yrs), `BMI`, and resting heart rate `HR`; the response is systolic blood pressure `SBP`.

Patient	age	BMI	HR	SBP
1	45	24.0	70	128
2	60	29.5	82	145
3	52	26.1	75	138
4	38	22.3	68	120
5	71	31.0	90	158

1. State n and p.
2. Give the dimensions of the design matrix X and the response vector y.
3. Write the predictor vector x_3 and give the value $x_{2,3}$.
4. If an intercept column of ones is added, what are the dimensions of the model matrix?
5. Is this a regression or a classification problem?

Solution — Exercise 1.3 (1/2)

Solution

1. Count the rows and the predictor columns of the table: $n = 5$ observations (one per patient); $p = 3$ predictors (age, `BMI`, `HR`). `SBP` is the response, *not* a predictor — the variable we model is never counted in p.
2. Rows = patients, columns = predictors, so $\mathrm{X} \in \mathbb{R}^{5\times 3}$; stacking the five responses gives $\mathrm{y} = (128,\ 145,\ 138,\ 120,\ 158)^\top \in \mathbb{R}^5$ (a 5×1 column).
3. Read off *row* 3 of the table: $x_3 = (52,\ 26.1,\ 75)^\top$ — all predictors of patient 3. For $x_{2,3}$ the *first* index is the row (observation), the *second* the column (predictor): predictor 3 (`HR`) of patient 2, i.e. $x_{2,3} = 82$.

Solution — Exercise 1.3 (2/2)

Solution

4. Adding a column of ones gives $\tilde{\mathbf{X}} \in \mathbb{R}^{5\times 4}$: the row count stays at $n = 5$, while the column count grows from $p = 3$ to $p + 1 = 4$ — the extra column of ones multiplies the intercept β_0 in a linear model.
5. **Regression**: `SBP` is quantitative (blood pressure in mmHg, on a continuous scale), so regression methods and squared-error loss apply.

Note on indexing: x_{ij} is row i (observation), column j (predictor). Row i is one patient across all predictors; column j is one predictor across all patients.

Common mistake

Reading $x_{2,3}$ column-first as "column 2, row 3", which would give the `BMI` of patient 3 (26.1) instead of the correct $x_{2,3} = 82$; and counting the response `SBP` into p, which would wrongly give $p = 4$.

Outline

The book in one slide

Ch. 2	Statistical Learning (foundations)
Ch. 3	Linear Regression
Ch. 4	Classification (logistic, LDA, QDA, naive Bayes)
Ch. 5	Resampling Methods (CV, bootstrap)
Ch. 6	Linear Model Selection and Regularisation
Ch. 7	Moving Beyond Linearity (splines, GAMs)
Ch. 8	Tree-Based Methods (CART, RF, boosting, BART)
Ch. 9	Support Vector Machines
Ch. 10	Deep Learning
Ch. 11	Survival Analysis and Censored Data
Ch. 12	Unsupervised Learning (PCA, clustering)
Ch. 13	Multiple Testing

The narrative arc

- Chapters 2–4 establish the *vocabulary* and *baseline* methods (regression and classification).
- Chapter 5 teaches you how to *honestly evaluate* models.
- Chapter 6 takes the linear model beyond OLS (subset selection, ridge, lasso).
- Chapters 7–10 add increasing flexibility: nonlinear, trees, SVMs, deep nets.
- Chapter 11 specialises to time-to-event data.
- Chapter 12 returns to unsupervised problems.
- Chapter 13 reasons about many simultaneous tests.

Takeaway

The arc: build *vocabulary* and baseline methods, learn to *evaluate honestly*, then add *flexibility* step by step before specialising.

Extended Exercise 1.1 — Anatomy of a loan-default study [Integrative]

Extended exercise (15 min)

A regional bank has records on **20,000** past personal-loan customers. For each it stored: annual `income`, current `debt`, credit-`utilisation` ratio, `loan_amount` requested, `employment_length` (yrs), number of prior `late_payments`, and `home_ownership` (own/rent/mortgage). For every past customer it also knows whether they eventually `defaulted` (Yes/No). The bank wants to **(a)** flag which *new* applicants are likely to default, and **(b)** understand *which* factors drive default and *by how much*, to justify decisions to a regulator.

1. Identify the response Y and the predictors; state n and p.
2. Regression or classification? Supervised or unsupervised? Justify each.
3. Which of goals (a) and (b) is *prediction* and which is *inference*?
4. Write the model $Y = f(X) + \varepsilon$. What does the irreducible error ε represent *here*? Give two concrete sources.
5. Sketch a sensible end-to-end workflow, and say which model family suits goal (b) given the regulator.

Solution — Extended Exercise 1.1 (1/2)

Solution

1. Response, predictors, n, p. The response is Y = `defaulted` (Yes/No). The predictors are the seven recorded features (`income`, `debt`, `utilisation`, `loan_amount`, `employment_length`, `late_payments`, `home_ownership`), so $p = 7$ and $n = 20{,}000$. Nuance: `home_ownership` has 3 levels — as *one* predictor $p = 7$, but dummy-encoding turns it into 2 columns, so the model matrix (with an intercept) has $6+2+1 = 9$ columns.

2. Problem type / supervision. Y is qualitative (binary) $\Rightarrow$ a **classification** problem — the type of Y decides, not the predictors. Because every training customer carries a *label* `defaulted`, this is **supervised** learning.

3. Prediction vs. inference. Goal (a), flagging new applicants, is **prediction**: we only need an accurate $\hat{Y}$. Goal (b), which factors matter and by how much for a regulator, is **inference**: we need the structure of f — signed, quantified effects.

Common mistake

Counting the response `defaulted` among the predictors (giving $p = 8$) — the response is never a column of X.

Solution — Extended Exercise 1.1 (2/2)

Solution

4. Model and irreducible error. We posit $Y = f(X) + \varepsilon$ with $E[\varepsilon] = 0$: f is the systematic part of default risk explained by the seven features; ε is everything they *cannot* explain. Here ε absorbs (i) *post-application life events* — a later job loss, illness, or divorce that no application-time feature can foresee; and (ii) *unmeasured or random factors* — a local recession, fraud, or plain chance. Even the *true* f cannot beat this noise floor $\mathrm{Var}(\varepsilon)$; for a Yes/No response its analogue is the Bayes error rate (Ch. 2).

5. A sensible workflow.

1. Assemble clean labelled data; *avoid leakage* (no post-outcome fields).
2. Split into train / test (or cross-validate); lock the test set away.
3. Preprocess: dummy-encode `home_ownership`, impute missing values, *standardise* numeric predictors.
4. For goal (b) use an **interpretable** model — *logistic regression* gives signed coefficients / odds ratios with confidence intervals a regulator can read; optionally add a flexible model (random forest) to gauge attainable accuracy for goal (a).
5. Evaluate on held-out data with a metric fit for *imbalanced* classes (ROC-AUC, precision/recall, or a cost-weighted error), not raw accuracy, and pick by cross-validated performance.

Outline

Pitfalls to remember from day one

The classic mistakes

1. Reporting training error and calling it "performance".
2. Ignoring data quality (selection bias, leakage, label noise).
3. Reading univariate associations as causal effects.
4. Forgetting to standardise features for distance-based methods.
5. Treating p-values as truth instead of evidence (Ch. 13).
6. Re-using the test set during model selection.

In industry — Project delivery: what kills a model project

Target leakage from fields written after the decision; training on a population different from the one being scored; nobody owning the model once it is live; and a score nobody can explain to a regulator or to a sales director.

Self-check questions

1. In your own words, what is the difference between *prediction* and *inference*? Give one example of each. (p. 20)
2. Which of the three motivating data sets (Wage, Smarket, NCI60) would benefit most from a flexible method, and why? (p. 38)
3. For the Boston housing data ($n = 506$, $p = 12$), what regime are we in: low-dimensional or high-dimensional? Explain. (pp. 36 and 40)
4. Write the design matrix X for $n = 4$ observations of $p = 2$ predictors plus an intercept. (pp. 54–55)

Get hands-on: the companion lab notebook

Companion notebook: `chapter_01_lab.ipynb`

These datasets are explored in the companion Jupyter notebook. Open it and run it cell by cell:

- §1 *The Wage data* — reproducing Fig. 1.1;
- §2 *The Stock Market data* — *Smarket* (Fig. 1.2);
- §3 *The NCI60 cancer cell lines*.

Data loads via the `ISLP` package or the bundled CSV files; the notebook ends with hands-on exercises.

Outline

Chapter 1 in one slide

What we accomplished

Established the *vocabulary*, *problem types*, and *tools* of statistical learning.

- Defined supervised vs. unsupervised, regression vs. classification.
- Walked through three motivating examples.
- Set up the standard notation and Python tooling.
- Surveyed the data sets we will see throughout the course.

Key formulas and notation at a glance

Data	n observations, p predictors; design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, response $\mathbf{y} \in \mathbb{R}^n$.
Supervised model	$Y = f(X) + \varepsilon$, with f unknown and $E[\varepsilon] = 0$.
Two goals	*Prediction*: $\hat{Y} = \hat{f}(X)$. *Inference*: understand the shape of f.
Response type	*Regression* (Y quantitative) vs. *classification* (Y qualitative).
Error	reducible (improve $\hat{f}$) vs. irreducible ($\mathrm{Var}(\varepsilon) = \sigma^2$).
Supervision	*supervised* (labels y available) vs. *unsupervised* (no y).

Vocabulary checklist

Term	Meaning
Statistical learning	Tools for understanding data
Supervised / unsupervised	With / without a response Y
Regression / classification	Y quantitative / qualitative
Prediction / inference	Forecast Y / explain how Y depends on X
Predictor / response	X_j / Y
Training / test set	Data used to fit / evaluate

Ten things to remember from Chapter 1

1. Statistical learning = tools for understanding data.
2. Two families: supervised and unsupervised.
3. Two flavours of supervised: regression and classification.
4. Two questions: prediction and inference.
5. The Wage example: regression, mostly inference.
6. The Smarket example: classification, very weak signal.
7. The NCI60 example: unsupervised structure discovery.
8. Notation: n obs., p predictors, X, y.
9. Tools: NumPy, pandas, scikit-learn, ISLP.
10. Always validate on data you did not train on.

What's next

Chapter 2: *Statistical Learning — the foundations*.

- Define the unknown function f.
- Distinguish parametric from nonparametric methods.
- Decompose prediction error into bias and variance.
- Meet our first concrete classifier: K-nearest neighbours.

Appendix — what is in here, and why

Nothing in the appendix is needed to follow the chapter: each slide is a formal derivation, a longer worked exercise, or a side topic. Read it when you want the full story, or when a later chapter sends you back here.

Contents of the appendix

- **The design matrix in full** — the anatomy of X and y drawn out entry by entry, and the notation for estimates. The one-slide version in the main deck ("Notation: n, p, and the design matrix") is all the chapter needs.
- **The dataset catalogue** — the two reference tables of every ISLP data set. Look them up when a lab asks for a data set you have not met.

The design matrix

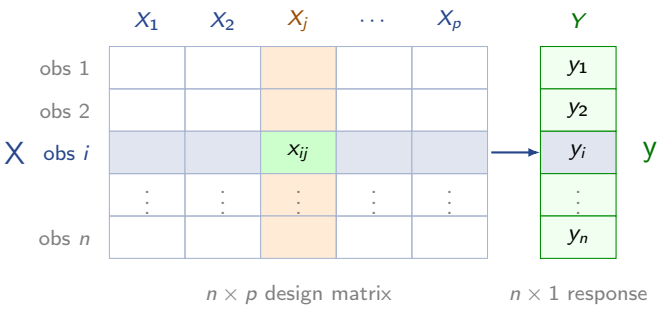
Stack the predictor vectors as rows:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix} \in \mathbb{R}^{n \times p}.$$

How to read this

- Row i: vector $x_i = (x_{i1}, \ldots, x_{ip})^\top$ — one observation, all predictors.
- Column j: vector x_j — all observations, one predictor.
- x_{ij}: value of the j-th predictor for the i-th observation.

Anatomy of the data: the matrix X and the vector y



Takeaway

A **row** is one observation x_i (all predictors); a **column** is one predictor across all observations; x_{ij} sits at row i, column j. Supervised learning pairs each row of X with its response y_i.

The response and estimates

Response vector

$$\mathrm{y} = (y_1, y_2, \ldots, y_n)^\top \in \mathbb{R}^n.$$

The hat convention

Estimated quantities wear a hat: $\hat{\beta}, \hat{f}, \hat{y}$. The hat reminds us that these are computed from a sample and therefore have sampling variability.

Common usage

Predicted (fitted) values: $\hat{y}_i$.
Residuals: $e_i = y_i - \hat{y}_i$.
Train vs. test: subscripts tr and te.

Datasets at a glance (1/2)

Name	n	p	**Response**	**Used in**
Wage	3000	10	wage (num.)	Ch. 1, 7
Smarket	1250	8	Direction (binary)	Ch. 1, 4
NCI60	64	6830	— (unsup.)	Ch. 1, 12
Auto	392	8	mpg (num.)	Ch. 3
Boston	506	12	medv (num.)	Ch. 3, 8
Carseats	400	10	Sales (num.)	Ch. 3, 8
Credit	400	10	Balance (num.)	Ch. 3, 6
Default	10000	3	default (binary)	Ch. 4
Heart	303	13	AHD (binary)	Ch. 4, 8

Datasets at a glance (2/2)

Name	n	p	**Response**	**Used in**
Caravan	5822	85	Purchase (binary)	Ch. 4
OJ	1070	17	Purchase (binary)	Ch. 8, 9
Hitters	322	19	Salary (num.)	Ch. 6, 8
College	777	17	Apps (num.)	Ch. 6
BrainCancer	88	8	survival	Ch. 11
USArrests	50	4	— (unsup.)	Ch. 12
Bikeshare	8645	14	bikers (num.)	Ch. 4