Introduction to Machine Learning

D2

Fatih Kansoy

Oxford
fatih.ai/machinelearning

2026

Orientation

1. AI: outputs, history, and vocabulary
2. Business applications: decisions and value

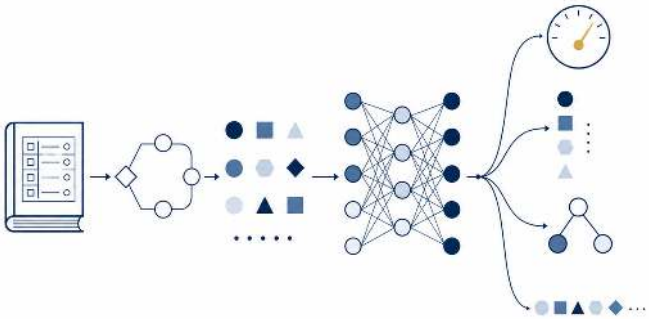
The machine-learning workflow

1. From an ambition to a learning problem
2. How training selects a rule
3. How a fitted rule is tested
4. What a performance result means
5. How model families differ
6. When predictions enter responsible use
7. Applying the complete workflow

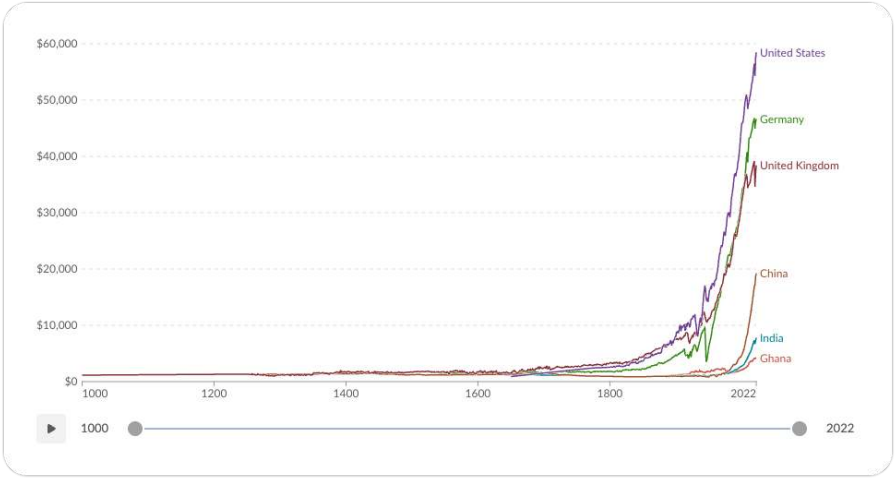
Orientation

Artificial intelligence: history and context

What an AI system produces, how its methods changed, and why this course begins with learning from data.



What happened and what is happening?

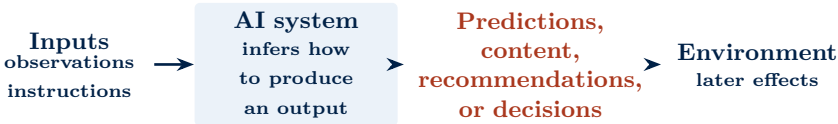


Source: Our World in Data.

AI is defined by what a system produces

For an explicit or implicit objective, an AI system **infers from inputs** how to generate outputs that can affect a physical or virtual environment.

Objective = criterion guiding an output; model = learned component; decision = use of an output.



AI is the broader system category. This course focuses on cases where a model inside the system is learned from data.

Source: OECD, updated definition of an AI system (2024).

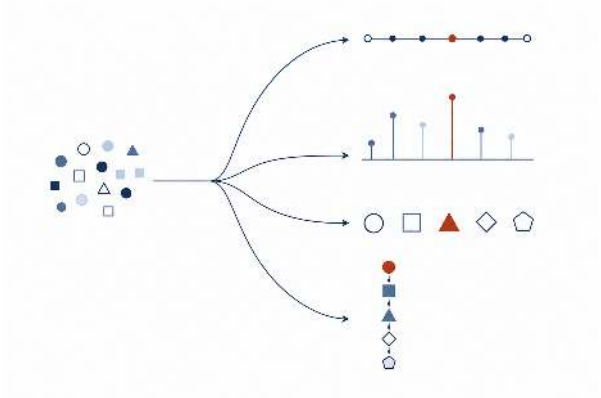
Definition and scope of artificial intelligence

> **Artificial intelligence**: building systems that perform tasks which normally require human intelligence, such as perceiving, reasoning, deciding, or producing language.

- **Narrow AI** performs one task envelope well: translation, chess, credit scoring. Every deployed system today is narrow.
- **General AI** would mean competence across arbitrary tasks. No such system exists, and it is not this week's subject.
- The definition says nothing about method. The methods that now deliver these capabilities are **learned from data**.

When a headline says "AI", ask first: which task, and how is the system judged on it?

Name the output before choosing the method



Number
ETA, demand, sale price

Probability or risk score
fraud, churn, default

Class after a rule
spam, identity, defect

Ranking
search, products, films

The required output determines the target, loss, and evaluation evidence.

Target = outcome or structure to learn; loss = how errors count; class = label after a decision rule.

Generative systems produce content from learned probability distributions; language models often generate a token sequence one choice at a time.

The terms answer different questions

SYSTEM CATEGORY

AI

Systems that infer outputs for objectives.

HOW A MODEL IS LEARNED

Machine learning

estimates models from data.

Deep learning

is ML using multilayer neural networks.

WHAT IS PRODUCED

Generative AI

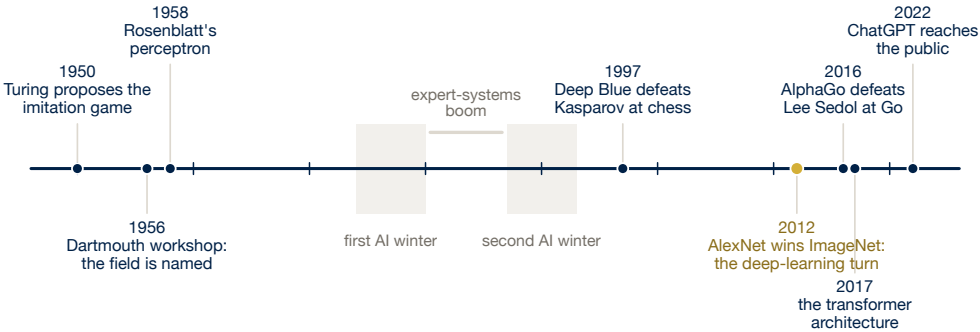
Systems intended to produce content: text, images, audio, code, and more.

Modern generative AI usually uses deep learning. But not every AI system learns from data, not every ML model is deep, and not every learned model generates content.

Source: OECD AI-system definition (2024); standard ML terminology.

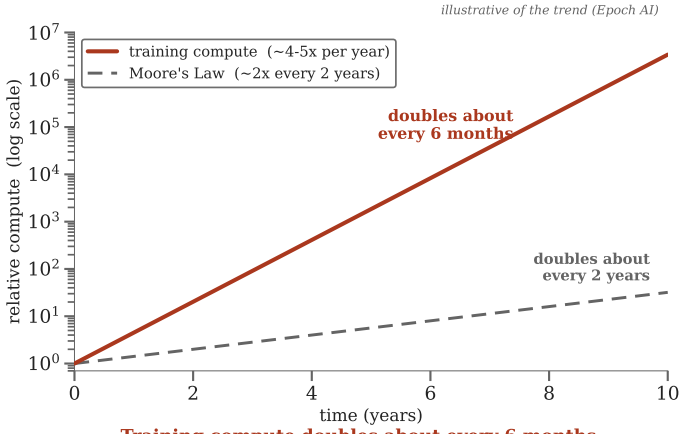
AI advanced in waves; methods accumulated

- In 1950 Alan Turing replaced “can machines think?” with a test of written answers; in 1956 a Dartmouth summer workshop named the field **artificial intelligence**.
- Twice, promises outran results and funding collapsed: the **AI winters**.
- The learning era after 2012 is the part this course examines.



Source: Turing (1950); Dartmouth proposal (1955/56); Computer History Museum (perceptron, expert systems, AI winters); IBM Research (Deep Blue); Krizhevsky et al. (2012); Silver et al., *Nature* (2016); Vaswani et al. (2017); OpenAI (2022).

Recent progress came from three reinforcing forces



Recorded examples
Larger datasets, including curated and labelled benchmarks.

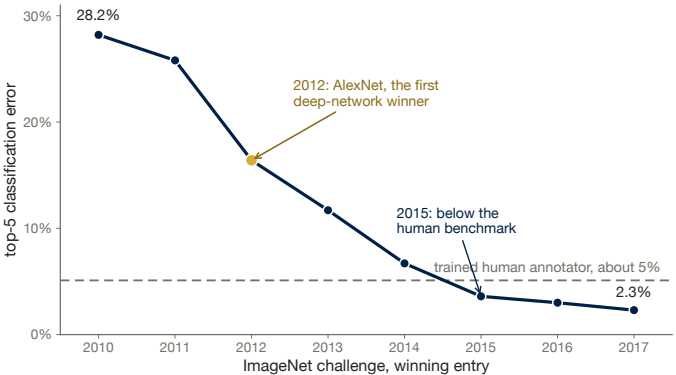
Compute
More experiments and larger training runs.

Learning methods
Architecture = permitted model structure; optimisation = procedure that adjusts it.

Epoch estimates that training compute for *selected notable models* rose about 4.5× per year from 2010. This is a historical sample trend, not a technological law; more compute does not guarantee usefulness. Scale still requires a defined test.

Source: Epoch AI, Trends in Artificial Intelligence (notable-model sample; accessed 2026).

A breakthrough is meaningful only inside a defined test



ImageNet made one claim comparable by fixing:

1. a labelled image task;
2. a benchmark: shared task, data, metric, and held-out cases;
3. a top-5 error metric: an error when the correct label is absent from the five highest-scored labels;
4. images withheld from fitting.

AlexNet's 2012 result was a large advance on this **defined classification problem**—not evidence of general intelligence.

Source: Russakovsky et al. (2015); official ILSVRC results archive (2016–17); AlexNet in Krizhevsky, Sutskever, and Hinton (2012). The plotted benchmark series and paper-specific test comparisons should not be interchanged.

We begin where outcomes are attached to historical cases

Setting	What is observed	What is learned	Course
Supervised	inputs x and outcomes y	an input-to-outcome rule	Days 1–3
Unsupervised	inputs x only	structure: groups or directions	Day 4
Reinforcement	states or observations, actions, and resulting rewards	a policy for acting	outside scope

x = recorded inputs; y = later outcome; policy = rule for acting; reward = consequence used to judge an action.

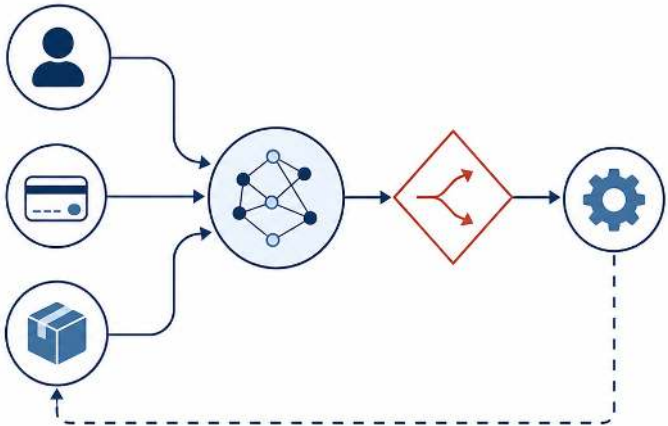
Self-supervision constructs training targets from the data themselves—for example, predicting a hidden or next token. It is a training strategy, not a separate business objective.

Next: how model outputs enter business decisions, and what evidence can establish value.

Orientation

Business applications of AI

A model becomes useful only when its output changes a decision and the resulting value—and harm—can be measured.



Machine learning applications by business function

Marketing
who will leave,
and who to target

Finance
fraud, and
credit risk

Operations
demand forecasts,
and maintenance

Customer
recommendations,
support triage

HR
screening,
handle with care

Risk
early warning
of trouble

This is not one department's tool. It is everyone's.

Netflix recommendation systems

Netflix's own engineers report that recommendations influence about **80%** of the hours watched, and estimate that personalisation saves **more than $1bn a year**, mostly through customers who stay.

Note who is doing the reporting. These are the company's own figures, and no outside party has audited them.

Source: Gomez-Uribe & Hunt, ACM TMIS 2016; Netflix's own ~2015 estimate.

~80%

of hours streamed are shaped by what the model recommends

A good recommender is not a feature. It is the business.

Real-time payment fraud detection

Global payment-card fraud losses reached about **$33.4bn** in 2024. Every card payment is scored for risk in **real time**, far faster than any human review could manage.

This is machine learning you never see, running on every card payment, everywhere, all day.

Source: The Nilson Report, card-fraud losses worldwide, 2024 (card fraud specifically).

$33.4bn

global card-fraud losses in a single year, 2024

At this speed and scale, the model is the only thing fast enough.

Email spam, phishing, and malware filtering

Google states that Gmail blocks **more than 99.9%** of spam, phishing and malware, filtering on the order of **15 billion** emails a day.

You never notice the ones it stops. That is the point.

Source: Google Safety Center (a vendor self-report, not an independent audit).

>99.9%

of spam, phishing and malware blocked before you see it

The best ML often shows up as nothing going wrong.

Generative AI use and coding productivity

The newest wave writes and codes. ChatGPT reports about **800 million** weekly users. In **one controlled study**, about 95 developers doing *a single* coding task finished **55% faster** with an AI assistant.

A striking result, but one narrow task run by the vendor. The same rules of evidence still apply.

Source: OpenAI (Oct 2025); GitHub Copilot study, 2022 (∼95 developers, one greenfield task, wide confidence interval).

55%

faster on *one* coding task, in a single vendor study

The newest wave, and the same discipline applies.

Estimated economic impact of generative AI

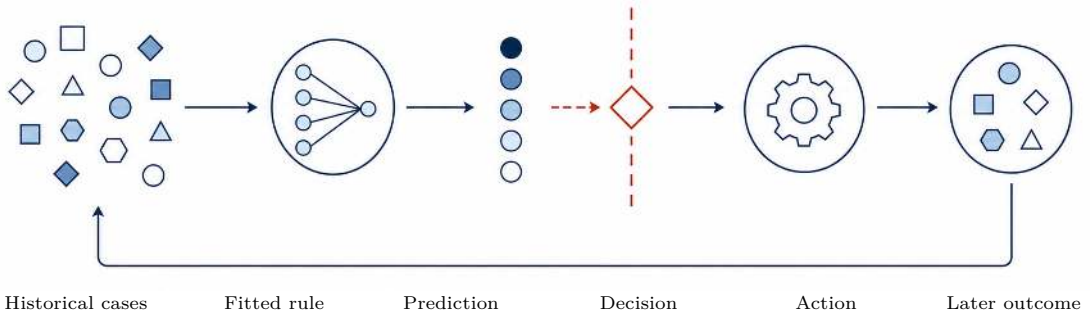
$2.6–4.4 trillion

a year that generative AI *could* add to the world economy, across dozens of use cases: a **projection**, not a measured figure

Source: McKinsey Global Institute, "The economic potential of generative AI," 2023. A modelled estimate, sensitive to adoption; other houses give different numbers.

A real opportunity, and an estimate, not a promise.

A model creates value only through a decision chain

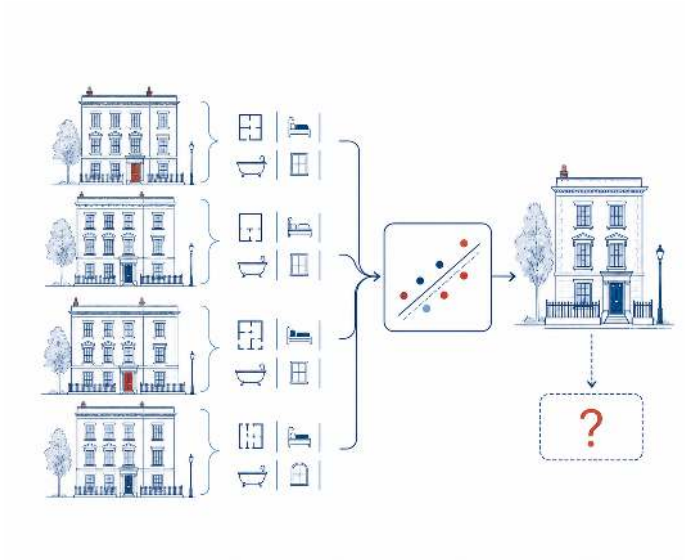


Part 1 now makes this chain testable: what is predicted, from which historical cases, at what moment, and for whose use?

Part 1

From an ambition to a learning problem

What must be learned, from which historical evidence, at what moment, and for whose use?



One request; three questions

1. estimate a benchmark for a realised completed sale price;
2. recommend an asking price;
3. estimate how renovation would change the sale price.

The first is a prediction. The second is a decision problem. The third is causal or interventional.

Causal = expected outcome if we change something.

Machine learning begins with a promise that can be checked

Case	one London flat in the stated scope
Prediction moment	initial valuation or listing intake
Permitted inputs	facts already recorded at that moment
Target and horizon	realised arm's-length sale price if completion occurs within 180 days
Learning evidence	earlier qualifying transactions
Intended use	a benchmark for an asking-price discussion

Use what was knowable at intake to estimate a later, precisely defined outcome for a relevant new flat.

A wrong case, clock, target, population, or use defines the wrong learning problem before any algorithm is fitted.

A training example connects what was knowable then to what happened later

Initial intake: t_0	$\Longrightarrow$	**Outcome:** $t_1 \le t_0 + 180$ **days**
floor area bedrooms location band building age available before the prediction		realised transaction price after an arm's-length completion not available at t_0

$$\big(\mathbf{x}_i(t_0),\, y_i(t_1)\big), \qquad t_1 > t_0.$$

i = case; $\mathbf{x}_i(t_0)$ = recorded inputs then; $y_i(t_1)$ = later outcome.

A target is complete only when both its **event** and its **time horizon** are specified.

A1 · Timed cases and training data →

A completed-sale model learns only from completed sales

What happens within 180 days?	Is a price target observed?	Role in this price model
Arm's-length sale completes	Yes: realised completed price	**Qualifying training case**
Withdrawn or unsold	No completed-price target	**Outside this target**
Sale completes after 180 days	Outside the defined horizon	**Outside this target**
Non-arm's-length transaction	Price is not comparable	**Outside this scope**

The benchmark is conditional on the completion definition.

Predicting whether completion occurs is a separate binary target. Excluded cases must be disclosed; they are not zero-price sales.

The learner sees a recorded property, not the property itself

Recorded

- floor area
- bedrooms
- location band
- building age

Missing or imperfect

- condition and light
- lease details
- urgency
- negotiation

Feature = recorded input supplied to the learner.

Features are an organisational representation of the property. More rows cannot recover information that was never recorded.

A prediction is evidence for a decision, not the decision itself

Recorded profile $\longrightarrow$ model benchmark: £540k

What the model used	recorded features and relationships learned from earlier qualifying sales
What the discussion adds	condition not captured in the record, urgency, market strategy, seller constraints, negotiation and consequences

$$\underbrace{£540\text{k}}_{\text{predictive evidence}} + \underbrace{\text{context, objectives, constraints}}_{\text{decision judgement}} \not\equiv \underbrace{£565\text{k}}_{\text{illustrative asking price}}$$

A model can inform the discussion. It does not own the action, and a prediction is not evidence of what an intervention would cause.

The feedback available determines the learning setting

Historical target information	Learning problem	Output for a new case
Realised completed sale price	**Supervised regression**	A numerical benchmark $\widehat{y}$
Completed within 180 days: yes/no	**Supervised classification**	An event probability $\widehat{p}$
No supplied target column	**Unsupervised learning**	Groups or a compact representation

$$\widehat{y} = f(\mathbf{x}), \qquad \widehat{p}(\mathbf{x}) \approx \Pr(Y = 1 \mid \mathbf{X} = \mathbf{x}).$$

$\widehat{\cdot}$ = model prediction; $\Pr(\cdot)$ = probability.

A probability is not yet an action; unsupervised structure is not automatically meaningful.

A2 · Output, class and action →

Text can create its own next prediction target

"A fitted rule must be judged on _____."

$$\underbrace{\text{context}}_{\text{model input}} \longrightarrow \underbrace{p(\text{next token} \mid \text{context})}_{\text{model output}}, \qquad \underbrace{\text{observed next token}}_{\text{training target}} \longrightarrow \text{loss}.$$

Token = unit of text the model reads or predicts.

- **Self-supervision** constructs prediction targets from the observation itself.
- Next-token prediction is a common pretraining objective for a large language model; it is not a direct objective for truth, safety, or organisational usefulness.

The text supplies abundant training targets without requiring a human label for every example.

Text can create its own next prediction target

"A fitted rule must be judged on new cases."

$$\underbrace{\text{context}}_{\text{model input}} \longrightarrow \underbrace{p(\text{next token} \mid \text{context})}_{\text{model output}}, \qquad \underbrace{\text{observed next token}}_{\text{training target}} \longrightarrow \text{loss}.$$

Token = unit of text the model reads or predicts.

- **Self-supervision** constructs prediction targets from the observation itself.
- Next-token prediction is a common pretraining objective for a large language model; it is not a direct objective for truth, safety, or organisational usefulness.

The text supplies abundant training targets without requiring a human label for every example.

In reinforcement learning, an action changes what is learned next

$$\underbrace{s_t}_{\text{observed queue}} \longrightarrow \underbrace{a_t}_{\text{signal action}} \longrightarrow \underbrace{(r_{t+1}, s_{t+1})}_{\text{reward and changed situation}} .$$

The action changes the next evidence; feedback may be delayed. Learning settings differ by the information and feedback available, not by brand.

Audit the learning brief before naming an algorithm

1. *Use AI to price homes better.*
2. *At intake, use the final negotiated price to predict the sale price.*
3. *Train on completed sales and claim the model covers every listing.*

Which field is missing or broken?

case **moment** **information** **event/horizon** **population** **use**

If the brief cannot be bounded, model comparison is premature.

We have defined what must be learned. Now: how can examples select one rule?

Part 2

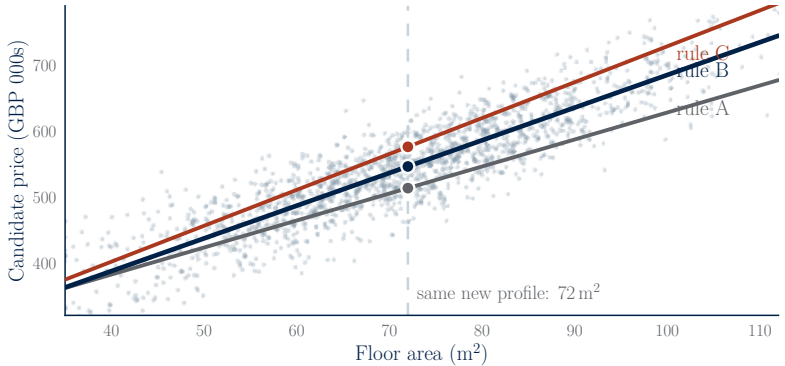
How training selects one rule

Candidate rules make different misses; loss and an objective determine how those misses guide fitting.

Before fitting, a model family permits many answers

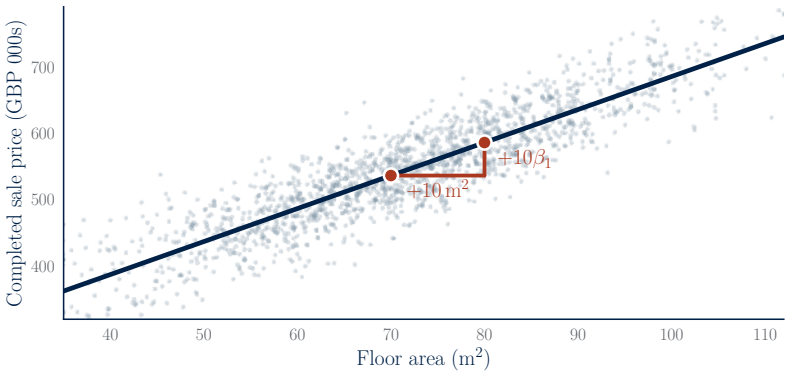
$$\mathcal{F} = \{\, f : \mathbf{x} \mapsto \text{candidate prediction} \,\}.$$

$\mathcal{F}$ = permitted family; f = candidate rule.



The area-only family makes the choice visible; a production valuation would require the fuller feature record defined in Section 1.

Parameters are learned inside settings that govern the fit



$$\widetilde{y}^{(\theta,\lambda)} = f_{\theta;\lambda}(\mathbf{x})$$

θ = learned parameters; λ = chosen fit settings.

Learned inside the fit

$$\theta = (\alpha, \beta_1)$$

α: benchmark at 70 m^2
β_1: change per extra m^2

Governing settings

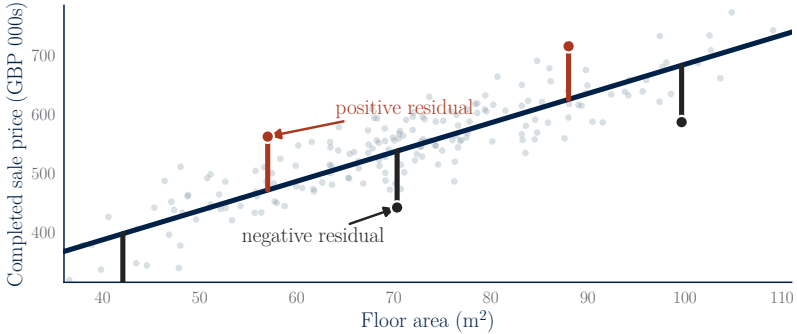
- penalty strength
- fitting settings
- allowed complexity

A3 · Model and fitting objects →

An observed outcome turns a candidate prediction into a residual

$$\widetilde{y}_i^{(\theta,\lambda)} = f_{\theta;\lambda}(\mathbf{x}_i), \qquad e_i(\theta,\lambda) = y_i - \widetilde{y}_i^{(\theta,\lambda)}.$$

$e_i > 0$: outcome above the prediction $\qquad$ $e_i < 0$: outcome below the prediction



A residual records direction and size; it does not yet say how severely the miss should count.

Two large misses can add to almost zero

$$+£46.8\text{k} \; + \; (-£46.8\text{k}) \; \approx \; 0$$

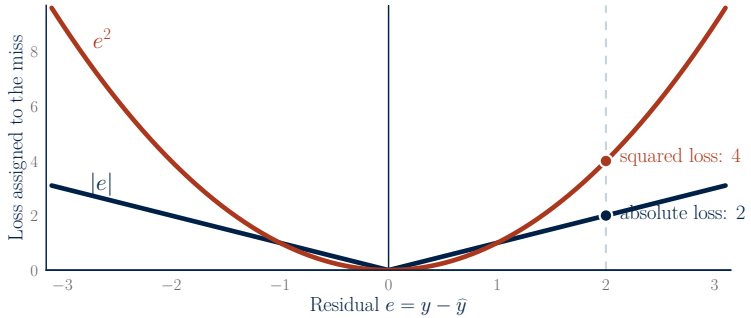
The signed total suggests almost no error.

$$|+£46.8\text{k}| + |-£46.8\text{k}| = £93.6\text{k}$$

The model missed by about £46,800 on *each* property.

Direction matters for diagnosis, but a signed sum cannot value the total size of the mistakes: positive and negative residuals cancel.

Loss decides how each miss should count



$L_1(y, \widehat{y}) = |y - \widehat{y}|$

Direct proportion: double the miss, double its charge.

$L_2(y, \widehat{y}) = (y - \widehat{y})^2$

Disproportionate influence: large misses grow much faster.

Mathematical loss encodes a fitting priority; it is not identical to financial, social, or legal harm.

A4 · Why loss changes the fitted centre →

Changing the loss can change which rule wins

Same observed outcomes:
$500, 500, 500, 600$ (GBP000).

	A	B
Absolute loss	100	160
Squared loss	10,000	7,600

Absolute loss selects A.

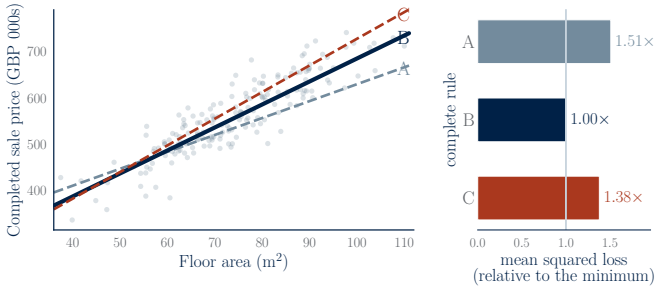
Squared loss selects B.

The lower total wins. Loss is part of the learning design, not a neutral score added afterwards.

The training objective scores a complete rule

$$\widehat{f} \in \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) \,.$$

$\mathcal{F}$: permitted rules L: how one case counts $n^{-1}\sum_i$: one score for the rule **arg min**: lowest-scoring rule(s).



The objective ranks complete rules on training evidence; it does not establish future error.

A5 · Least-squares derivation →

Training updates the candidate, scores again, then stops

$$\boldsymbol{\theta}^{(k)} \longrightarrow \widehat{\mathbf{y}}^{(k)} \longrightarrow \widehat{\mathcal{R}}_{\text{train}}\left(\boldsymbol{\theta}^{(k)}\right) \longrightarrow \boldsymbol{\theta}^{(k+1)} \longrightarrow \cdots$$

current state → predictions → score → updated state; repeat until the stopping condition is met.

k = update step; $\widehat{\mathcal{R}}_{\text{train}}$ = training score.

Fixed during this fit

training cases, model family, loss, governing settings

Allowed to change

the current parameters or learned state

The **model family** says which rules are permitted; the **learning algorithm** says how the fitting procedure searches them.

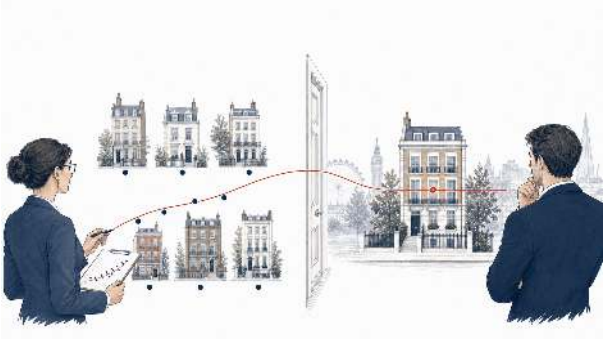
Some procedures solve directly; others update, grow, or search. Iteration is common, not the definition of learning.

A6 · Gradient update; A7–A8 related terms →

Fitting and prediction happen in different phases

$$\widehat{f} \leftarrow \mathrm{Fit}(\mathcal{D}_{\mathrm{train}}; \lambda), \qquad \widehat{y}_{\mathrm{new}} = \widehat{f}(\mathbf{x}_{\mathrm{new}}).$$

$\mathcal{D}_{\mathrm{train}}$ contains historical feature–outcome pairs; $\mathbf{x}_{\mathrm{new}}$ arrives without its outcome.

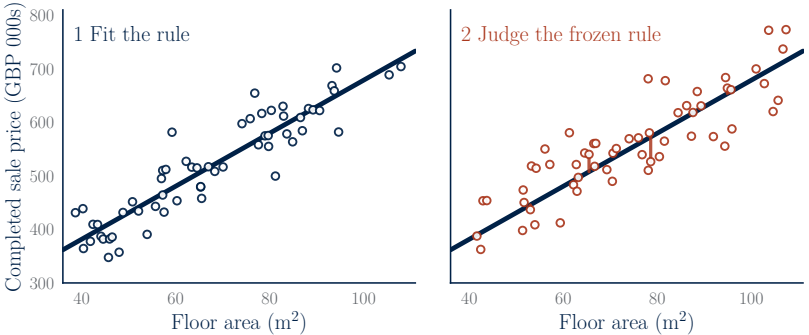


Preprocessing learned during fitting belongs to the fitted procedure and must be reused unchanged on the new case.

A fitted model has completed training—not earned trust

$$\widehat{\mathcal{R}}_{\text{train}}(\widehat{f}) \quad \not\approx \quad \mathcal{R}_{\text{deploy}}(\widehat{f}).$$

$\mathcal{R}_{\text{deploy}}$ = expected loss for the stated intended use.



Training fit alone cannot establish performance on relevant new cases.

The generated earlier and later cohorts explain an evidential boundary, not a documented London chronology. The next question is which independent judge could make the future-performance claim credible.

Part 3

How a fitted rule is tested

Training produces a candidate rule. Independent, deployment-relevant evidence must determine whether that rule travels.

Training data cannot independently judge the rule they selected

$$\mathcal{D}_{\text{train}} \xrightarrow[\text{selects}]{\text{Fit}} \widehat{f}$$

The training cases helped this rule win; their score is not an independent judgement of the winner.

Training score shows how the selected rule fitted the cases that participated in selection.

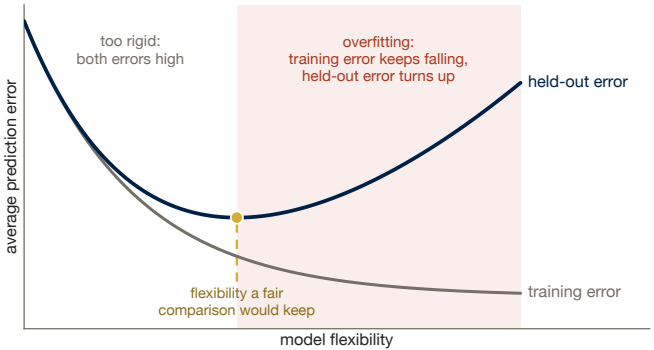
Independent score uses relevant cases that had no influence on fitting, preprocessing, selection, threshold choice, or revision.

Training evidence chooses the candidate. A protected information boundary lets new evidence judge whether it transfers.

A9 · Empirical and intended-use risk →

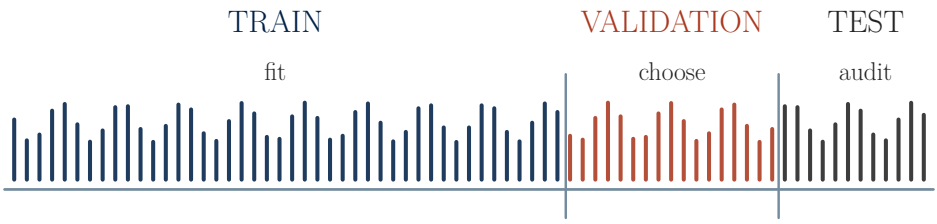
Generalisation and overfitting

Training observations shaped the rule. A low training loss is necessary, but it is not evidence that the learned pattern will travel.



The job is not to remember the past; it is to perform on the next case.

Training, validation, and test are roles—not percentages



The names describe permitted influence, not universal percentages.

The required data volume and dependence determine how these roles are implemented; there is no universal 60/20/20 rule.

Evidence used to revise a workflow is no longer its final audit

test result $\longrightarrow$ report the bounded claim

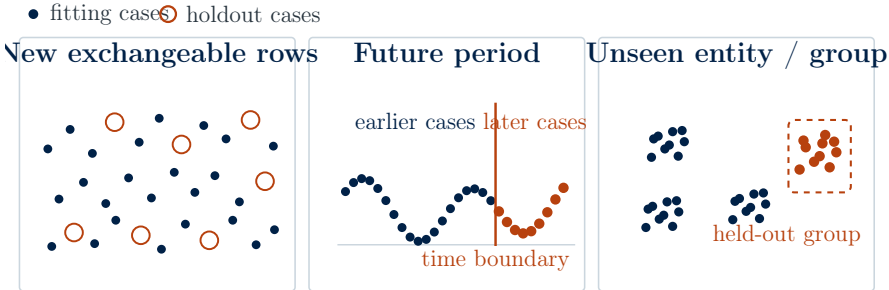
test result $\longrightarrow$ change the workflow $\longrightarrow$ the test has entered selection.

A valid final audit therefore requires four commitments:

1. freeze preprocessing and model;
2. freeze the operating rule;
3. open the test evidence;
4. report the result and its scope.

Repeatedly reopening the same test evidence converts it into another validation source.

The holdout must represent what will actually be new

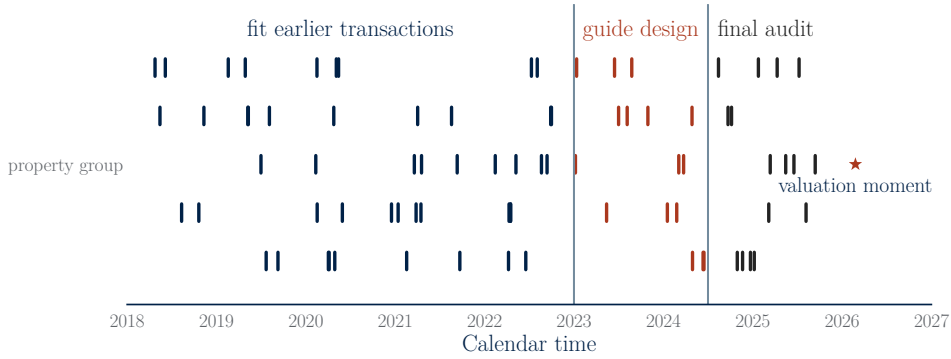


New exchangeable rows	A later period	An unseen entity or group
random split	chronological split	group-aware split

Exchangeable = cases that can plausibly arrive under the same conditions.

The evaluation split must reproduce what will be new when the model is used.

For the London claim, later transactions should judge earlier fitting



Earlier transactions fit the procedure; a later period guides choices; the latest eligible transactions provide the final audit. Related properties must not be allowed to leak across the boundary.

This chronology illustrates the evaluation design. The transactions are generated teaching cases, not a documented London market history.

Leakage route 1: held-out cases influence what is learned

Wrong order

held-out values helped determine the transformation

Correct order

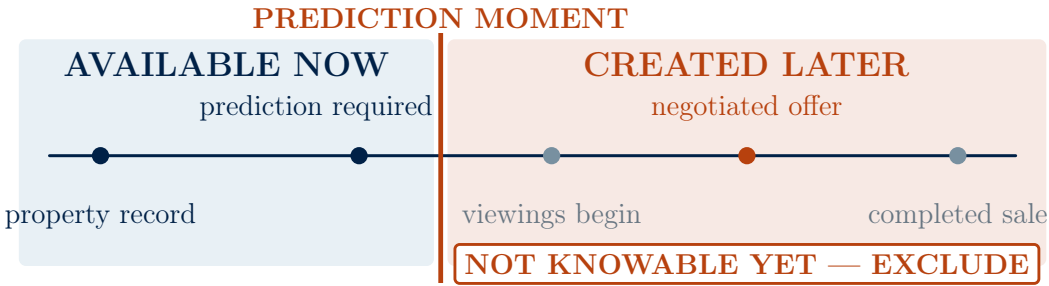
$$T_{\text{train}} = \text{FitTransformation}(\mathcal{D}_{\text{train}}), \qquad T_{\text{train}}(X_{\text{val/test}}).$$

Imputation, scaling, feature selection, and dimensionality reduction belong inside the training pipeline.

Imputation = replacing a missing input using a value learned from training data.

A11 · Leakage-safe fitted pipeline →

A feature must exist at the prediction moment



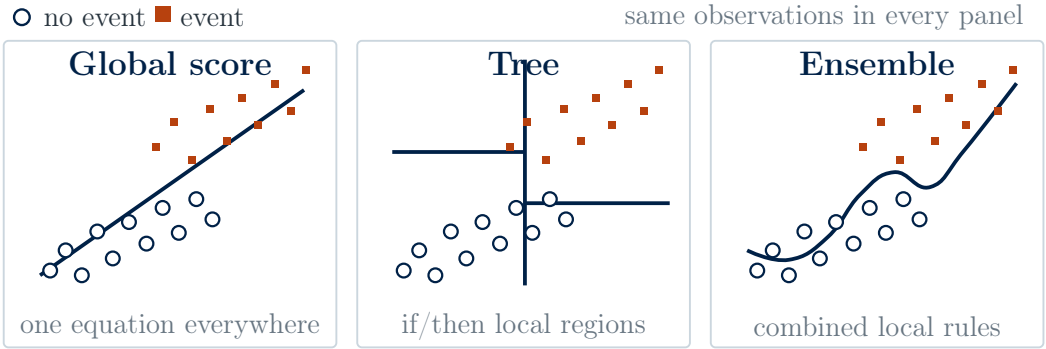
A field can exist in the final database and still be invalid for the earlier prediction; timing is part of the feature definition.

Part 5

How model families differ

The learning question and evaluation contract stay fixed; the permitted shape of the rule changes.

Model families permit different functional forms



The cases, target, objective, validation evidence, baseline and use stay fixed; only the language available for expressing the rule changes.

Functional form = set of shapes the model family allows for its prediction rule.

A global score uses the same additive coefficient in every case

Linear and logistic models use the same *functional form*:

$$s(\mathbf{x}) = \beta_0 + \mathbf{x}^\top \boldsymbol{\beta}.$$

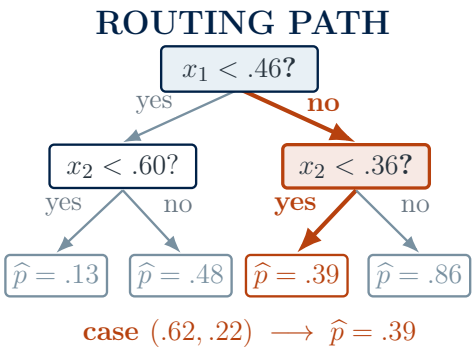
β_0 = intercept; $\boldsymbol{\beta}$ = vector of feature coefficients.

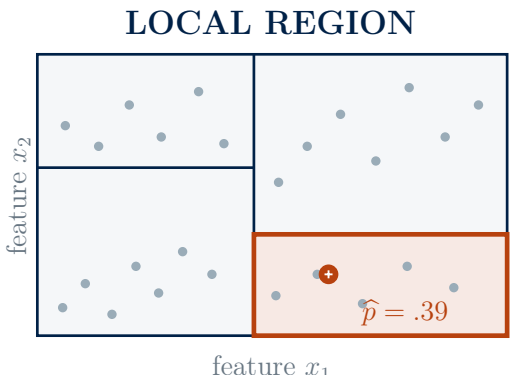
Target	Output	What stays global
Numerical	$\widehat{y} = s(\mathbf{x})$	one additive slope for each supplied feature
Binary	$\widehat{p} = \dfrac{1}{1 + e^{-s(\mathbf{x})}}$	the same additive score form, mapped to a probability

One coefficient adds the same amount to the score across cases unless variation is represented explicitly. For logistic regression, the resulting change in probability still depends on the starting score.

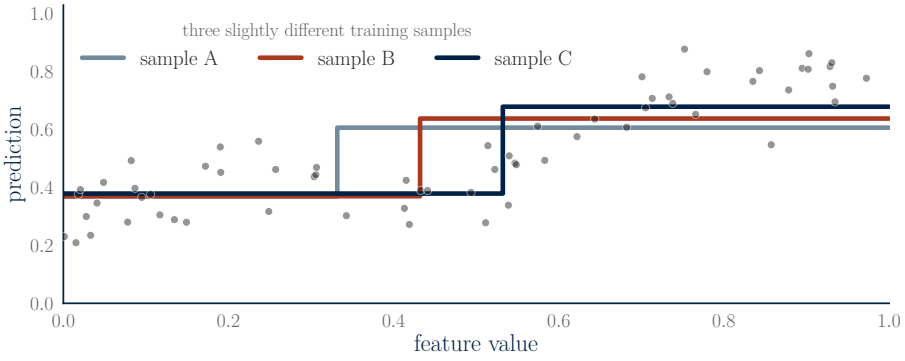
Global-score models are transparent, fast and valuable baselines, but their global additive structure can be too rigid for local interactions.

A decision tree routes each case to a local prediction



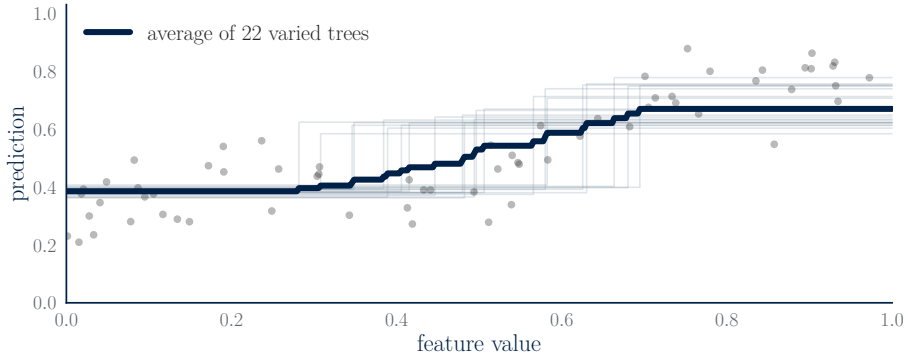


A single tree can change sharply when the training data change



Nearby samples can move a split and change many predictions at once.

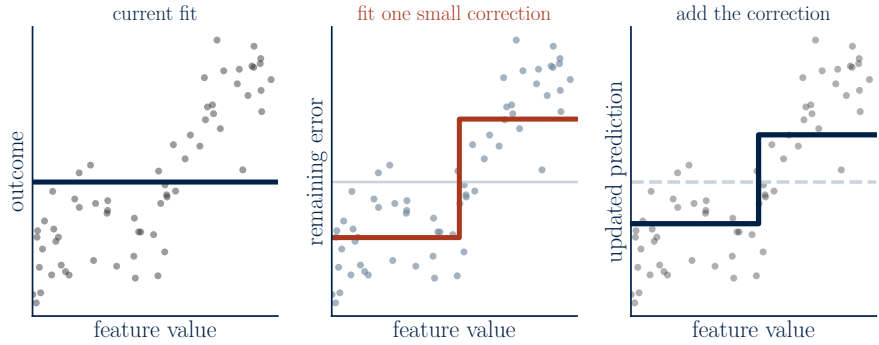
A random forest averages many varied tree predictions



Bootstrap samples and feature subsampling create variation; averaging reduces the instability of one tree.

Bootstrap = resample development cases with replacement; feature subsampling = use random subsets of features at splits.

Gradient boosting adds a small correction to remaining error

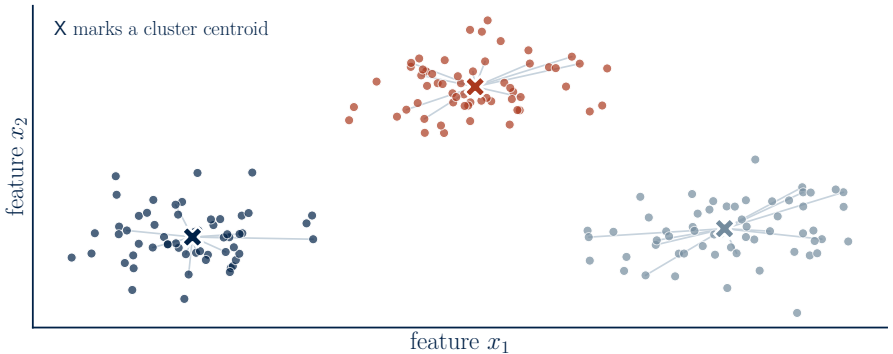


Each stage learns part of the current residual pattern; learning rate, number of stages and stopping are validation choices.

Learning rate = fraction of each fitted correction added at one boosting stage.

A19 · Tree, forest and boosting expressions →

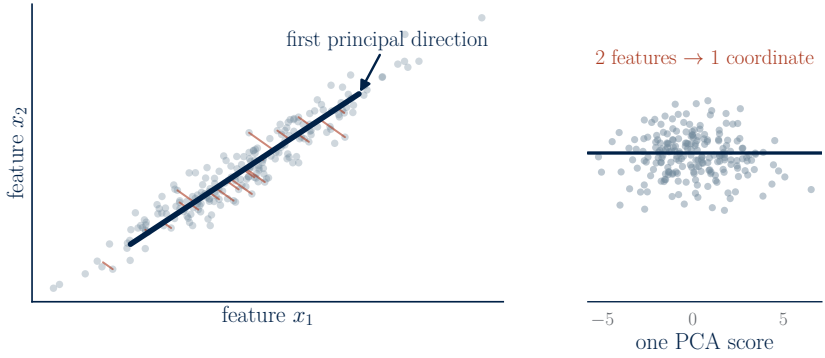
K-means assigns each observation to its nearest centroid



The objective reduces within-cluster squared distance. Scaling and the chosen representation therefore change the groups.

K = number of clusters; centroid = coordinate average of the points assigned to a cluster.

PCA preserves variation along a smaller set of directions



The first component is a direction; each observation is represented by its coordinate along that direction. No target outcome is supplied.

PCA = principal component analysis; a component = a direction in feature space chosen to capture variance.

A20 · Objectives, representation and scaling →

Technical appendix

Notation, derivations, and distinctions linked from Sections 1–7.

Technical appendix map

Foundations	A1 timed cases · A2 outputs and actions · A3 model objects · A4 loss centres · A5 least squares · A6 gradient update · A7 regularisation · A8 batches and epochs
Evaluation	A9 risk · A10 cross-validation · A11 leakage-safe pipelines · A12 numerical metrics · A13 baselines · A14 calibration · A15 classification metrics · A16 thresholds · A17 uncertainty
Model families	A18 global scores · A19 trees and ensembles · A20 unsupervised objectives · A21 neural layers and token loss
Responsible use	A22 prediction and intervention · A23 group metrics · A24 policy-shaped evidence · A25 provenance and sources

Each main-slide link opens the relevant detail; each appendix page returns to the exact teaching slide.

Appendix A1: from a timed case to training data

For case i, reconstruct the feature record at the intended prediction moment:

$$\mathbf{x}_i(t_0) = \Big[\text{area}_i \quad \text{bedrooms}_i \quad \text{location band}_i \quad \text{building age}_i\Big]^\top .$$

If the defined completion event occurs by $t_1 \le t_0 + 180$ days, record

$$y_i(t_1) = \text{realised completed sale price}_i.$$

The resulting supervised training data are

$$\mathcal{D}_{\text{train}} = \Big\{ \big(\mathbf{x}_i(t_{0i}), y_i(t_{1i})\big) : i \text{ meets the event, horizon, and scope rules} \Big\} .$$

Outside the completion definition, this price target is undefined—not zero. If those cases matter, model completion separately or redesign the task.

← **Main** · **Timed training example**

Appendix A2: output, class, and action are different

Object	Example
Regression prediction	$\widehat{y} = 546$ means a £546k benchmark
Classification score/probability	$\widehat{p} = 0.73$ estimates the probability of completing within 180 days
Class decision	$\mathbb{1}\{\widehat{p} \geq c\}$ after a threshold c is chosen
Organisational action	inspection, pricing discussion, outreach, or no action under a policy

$$\widehat{p} \quad \longrightarrow \quad \text{threshold and capacity rule} \quad \longrightarrow \quad \text{class/action.}$$

Accuracy requires a threshold; probability scoring rules assess scores directly. Action value also depends on consequences and capacity.

← **Main** · **Learning settings**

Appendix A3: the objects around one fitted model

Object	Precise role
Learning problem	specifies case, moment, information, target, population, and use
Model family $\mathcal{F}_\lambda$	the permitted candidates under governing settings λ
Parameters θ	values or structure learned within one fitting run
Learning algorithm A	the procedure that searches, solves, grows, or updates
Fitted model $\widehat{f}$	the frozen preprocessing and learned rule returned by fitting
Prediction $\widehat{y}$	the fitted rule's output for one supplied record

A hyperparameter may be fixed rather than tuned. Tree splits are learned structure; neural weights are learned numeric parameters.

$$\widehat{f} = A(\mathcal{D}_{\text{train}}; \mathcal{F}_\lambda, L, \lambda, \text{stopping rule})\,.$$

← **Main** · **Parameters and settings**

Appendix A4: why the two losses target different centres

For a constant prediction a, squared loss satisfies

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - a)^2 = \underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2}_{\text{does not depend on } a} + (a - \bar{y})^2.$$

Hence the unique minimiser is

$$\arg\min_{a} \frac{1}{n}\sum_{i}(y_i - a)^2 = \{\bar{y}\}.$$

For absolute loss, the subgradient changes sign when at least half the observations lie on each side of a, so

$$\arg\min_{a} \sum_{i} |y_i - a| = \left[y_{(\lceil n/2 \rceil)},\, y_{(\lfloor n/2 \rfloor + 1)}\right],$$

where $y_{(1)} \le \cdots \le y_{(n)}$ are the ordered outcomes.

Squaring changes more than cancellation: it gives extreme residuals greater influence and selects a mean-type centre. Absolute loss selects a median.

← **Main** · **Loss** **A4b** · **Population targets** →

Appendix A4b: loss determines the population target

At a particular feature profile $X = x$, imagine choosing a constant prediction a before the outcome is observed. Under squared loss,

$$\arg\min_a \mathbb{E}[(Y - a)^2 \mid X = x] = \mathbb{E}[Y \mid X = x].$$

$\mathbb{E}[\cdot]$ = expected value (probability-weighted average); conditioning on $X = x$ restricts that average to the stated feature profile.

Under absolute loss,

$$\arg\min_a \mathbb{E}[|Y - a| \mid X = x] \in \text{Median}(Y \mid X = x).$$

The loss tells the fitted function which conditional centre to approximate.

Changing the loss can therefore change predictions even when the data and model family are unchanged.

← **Main** · **Loss**

Appendix A5a: ERM becomes least squares

Take a linear family with intercept,

$$\widetilde{\mathbf{x}} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}, \qquad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \boldsymbol{\beta}_{1:p} \end{bmatrix}, \qquad f_{\boldsymbol{\beta}}(\mathbf{x}) = \widetilde{\mathbf{x}}^{\top} \boldsymbol{\beta}.$$

$\widetilde{\mathbf{x}}$: $\mathbf{x}$ augmented by 1; $\widetilde{\mathbf{X}}$: stacked augmented rows; $\|\cdot\|_2$: Euclidean length.

Stacking the augmented row vectors gives $\widetilde{\mathbf{X}} = [\,\mathbf{1}\ \mathbf{X}\,]$. Under squared loss, empirical-risk minimisation becomes

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{n} \left\| \mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\beta} \right\|_2^2 .$$

The derivative makes the fitting direction explicit:

$$\nabla_{\boldsymbol{\beta}} \frac{1}{n} \left\| \mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\beta} \right\|_2^2 = -\frac{2}{n} \widetilde{\mathbf{X}}^{\top} \left(\mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\beta} \right).$$

Empirical-risk minimisation supplies the grammar; the chosen family and loss determine the particular objective and derivative.

← **Main** · **Training objective**

Appendix A5b: the normal equations

At an interior minimum, setting the least-squares gradient to zero gives

$$\widetilde{\mathbf{X}}^{\top}\left(\mathbf{y}-\widetilde{\mathbf{X}}\widehat{\boldsymbol{\beta}}\right)=0,$$

or equivalently

$$\widetilde{\mathbf{X}}^{\top}\widetilde{\mathbf{X}}\widehat{\boldsymbol{\beta}}=\widetilde{\mathbf{X}}^{\top}\mathbf{y}.$$

Because the first column of $\widetilde{\mathbf{X}}$ is the intercept column $\mathbf{1}$, one component implies

$$\sum_{i=1}^{n}\widehat{e}_i=0.$$

This explains why fitted least-squares residuals with an intercept sum to zero; it does not mean the individual misses are small. If $\widetilde{\mathbf{X}}$ has full column rank, the unique solution can be written

$$\widehat{\boldsymbol{\beta}}=(\widetilde{\mathbf{X}}^{\top}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\top}\mathbf{y}.$$

With rank deficiency, use a pseudoinverse or another identified solution. Numerical software normally uses stable matrix factorisations rather than explicitly forming this inverse.

← **Main** · **Training objective**

Appendix A6: one gradient-based update

For a differentiable training objective $\widehat{\mathcal{R}}_{\text{train}}(\theta;\lambda)$, one common update is

$$\theta^{(k+1)} = \theta^{(k)} - \eta\nabla_\theta\widehat{\mathcal{R}}_{\text{train}}(\theta^{(k)};\lambda), \qquad \eta > 0.$$

$\nabla_\theta\widehat{\mathcal{R}}_{\text{train}}$ = gradient, the vector of objective slopes with respect to θ. A small η can be slow; an excessively large η can overshoot or diverge.

For the quadratic $q(\theta) = (\theta - 3)^2$, start at $\theta^{(0)} = 0$:

$$\nabla q(0) = 2(0 - 3) = -6.$$

With learning rate $\eta = 0.25$,

$$\theta^{(1)} = 0 - 0.25(-6) = 1.5, \qquad q(0) = 9, \quad q(1.5) = 2.25.$$

The update moved in a direction that lowered the objective.

Gradient descent is one fitting route, not the definition of machine learning and not the procedure used by every model.

← **Main · Training loop** **A7 · Regularisation →** **A8 · Batch and epoch →**

Appendix A7: regularisation adds another design choice

$$\widehat{\theta}_{\gamma} \in \arg\min_{\theta} \left\{ \widehat{\mathcal{R}}_{\text{train}}(\theta) + \gamma\, \Omega(\theta) \right\}, \qquad \gamma \geq 0.$$

γ is one governing setting. Comparing several values requires validation evidence rather than the final test set.

- $\widehat{\mathcal{R}}_{\text{train}}(\theta)$ rewards fit to the training cases;
- $\Omega(\theta)$ penalises a chosen form of complexity;
- γ controls the strength of that penalty.

For ridge-type regularisation,

$$\Omega(\boldsymbol{\beta}) = \|\boldsymbol{\beta}_{1:p}\|_2^2,$$

usually excluding the intercept. The meaning of the penalty depends on feature scale, so any scaling must be learned inside the training pipeline.

Regularisation changes the fitting objective; it does not provide independent evidence that the chosen γ generalises.

← **A6 · Gradient update**

Appendix A8: batch, iteration, and epoch

Suppose the training set has $n = 1{,}000$ cases and the fitting procedure uses mini-batches of 100.

Term	Meaning in this example
Mini-batch	the 100 cases used to compute one approximate update
Iteration	one update of the current learned state
Epoch	ten mini-batch iterations: one complete pass through all 1,000 cases
Five epochs	50 updates and five passes through the training cases

$$\text{iterations per epoch} = \frac{1{,}000}{100} = 10.$$

The relation changes with batch size, shuffling, incomplete final batches, and distributed training. Epoch is not a universal unit for trees or direct linear solvers.

← **A6 · Gradient update**

Appendix A9: empirical risk and intended-use risk

For any fixed rule f and a sample S,

$$\widehat{\mathcal{R}}_S(f) = \frac{1}{|S|} \sum_{i \in S} L(y_i, f(\mathbf{x}_i)) \,.$$

The intended-use quantity is a population expectation,

$$\mathcal{R}_{\text{use}}(f) = \mathbb{E}_{(X,Y)\sim P_{\text{use}}} \left[L(Y, f(X)) \right] .$$

If $\widehat{f}$ was selected using the training observations, then $\widehat{\mathcal{R}}_{\text{train}}(\widehat{f})$ evaluates the winner on the evidence that helped make it win. Its optimism is induced by selection; the realised ordering of two finite-sample estimates is not a mathematical certainty in every dataset.

Independent evidence estimates performance of the selected procedure; deployment relevance determines which population that estimate concerns.

← **Main** · **Independent evidence**

Appendix A10: selection, cross-validation, and final audit

For a fixed validation design, each candidate workflow is fitted without the validation cases and then compared on them:

$$\widehat{f}_\lambda = A(\mathcal{D}_{\text{train}}; \lambda), \qquad \widehat{\lambda} \in \arg\min_{\lambda \in \Lambda} \widehat{\mathcal{R}}_{\text{val}}(\widehat{f}_\lambda).$$

In K-fold cross-validation, aggregate the temporary validation roles:

$$\widehat{\mathcal{R}}_{\text{CV}}(\lambda) = \frac{1}{K} \sum_{k=1}^{K} \widehat{\mathcal{R}}_k(A(\mathcal{D}_{-k}; \lambda)) \,.$$

K = number of folds; $\mathcal{D}_{-k}$ = development data excluding fold k.

After selecting $\widehat{\lambda}$, refit the complete pipeline on the pre-authorised development evidence, freeze it, and audit once:

$$\widehat{f}_{\text{final}} = A(\mathcal{D}_{\text{development}}; \widehat{\lambda}), \qquad \widehat{\mathcal{R}}_{\text{test}}(\widehat{f}_{\text{final}}).$$

If cross-validation itself supplies the final performance estimate, an outer or nested resampling design is needed. Repeated final-test consultation still wears out the audit.

← **Main** · **Workflow selection**

Appendix A11: a leakage-safe fitted pipeline

Let T denote a transformation whose state must be learned, such as an imputer, scaler, feature selector, or PCA mapping. The valid order is

$$T_{\text{train}} = \text{FitTransformation}(\mathcal{D}_{\text{train}}), \qquad \widetilde{X}_{\text{train}} = T_{\text{train}}(X_{\text{train}}),$$

$$\widetilde{X}_{\text{val}} = T_{\text{train}}(X_{\text{val}}), \qquad \widetilde{X}_{\text{test}} = T_{\text{train}}(X_{\text{test}}).$$

The validation or test rows may be transformed, but they may not determine the transformation's learned state. The same information-flow rule applies inside every cross-validation fold. Unsupervised transformations may use X_{train} alone; supervised feature selection also uses y_{train}.

Evaluation leakage held-out rows influence preprocessing, selection, or tuning.

Target/timing leakage a feature contains the outcome, a proxy created after it, or information unavailable at the prediction moment.

← **Main** · **Leakage route 1**

Appendix A12: MAE, MSE, and RMSE

For residuals $e_i = y_i - \widehat{y}_i$ on m held-out cases,

$$\text{MAE} = \frac{1}{m}\sum_i |e_i|, \qquad \text{MSE} = \frac{1}{m}\sum_i e_i^2, \qquad \text{RMSE} = \sqrt{\text{MSE}}.$$

If the outcome is measured in pounds, MAE and RMSE are measured in pounds; MSE is measured in squared pounds. By the quadratic-mean inequality,

$$\text{RMSE} \geq \text{MAE},$$

with equality only when all absolute residuals have the same magnitude.

- MAE increases in direct proportion to an individual absolute miss.
- RMSE is more responsive to the upper tail because errors are squared before aggregation.
- Neither average reveals error direction, tail location, time variation, or subgroup concentration; retain residual diagnostics.

← **Main** · **Numerical performance**

Appendix A13: a baseline must match the loss

A no-feature baseline is fitted using training outcomes, frozen, and evaluated on the same held-out cases and metric as the model.

Evaluation objective	Training-fitted constant rule
Squared error	training mean $\bar{y}_{\text{train}}$
Absolute error	training median
Classification accuracy	training majority class
Squared probability loss or log loss	training event rate

For squared error, the decomposition

$$\frac{1}{n}\sum_i (y_i - a)^2 = \frac{1}{n}\sum_i (y_i - \bar{y})^2 + (a - \bar{y})^2$$

shows why the training mean is the optimal constant. A baseline fitted using held-out outcomes would itself leak audit information. ← **Main** · **Baseline comparison**

Appendix A14: calibration and probability losses

A probability model is calibrated when, approximately,

$$\Pr(Y = 1 \mid \widehat{p} \approx p) \approx p.$$

Thus a score may rank positive cases well while its numerical probabilities remain systematically too high or too low. Two common probability losses are

$$\text{Brier} = \frac{1}{n} \sum_i (y_i - \widehat{p}_i)^2,$$

$$\text{LogLoss} = -\frac{1}{n} \sum_i \left[y_i \log \widehat{p}_i + (1 - y_i) \log(1 - \widehat{p}_i) \right].$$

Both assess the probability before a classification threshold is imposed and penalise confident incorrect probabilities. Calibration plots, Brier score, log loss, and ranking/discrimination answer related but non-identical questions. ← **Main** · **Score and probability**

Appendix A15: confusion-matrix metric ledger

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \qquad \text{Precision} = \frac{TP}{TP + FP},$$

$$\text{Recall} = \frac{TP}{TP + FN}, \qquad \text{Specificity} = \frac{TN}{TN + FP}.$$

TP = true positive; TN = true negative; FP = false positive; FN = false negative.

For the validation ledger $(TN, FP, FN, TP) = (7{,}211, 99, 712, 216)$,

$$\text{Accuracy} = 90.16\%, \quad \text{Precision} = 68.57\%, \quad \text{Recall} = 23.28\%, \quad \text{Specificity} = 98.65\%.$$

Accuracy	conditions on every evaluated case
Precision	conditions on cases selected by the model
Recall	conditions on cases where the event actually occurred
Specificity	conditions on cases where the event did not occur

The numerator can contain the same count while the denominator changes the question. No metric chooses the operational consequence automatically.

← **Main** · **Metric denominators**

Appendix A16: a cost-based threshold

Suppose $\widehat{p}$ is a calibrated event probability. Predicting positive incurs cost c_{FP} only when $Y = 0$; predicting negative incurs cost c_{FN} only when $Y = 1$.

$$\mathbb{E}[C(\text{positive}) \mid \widehat{p}] = c_{\mathrm{FP}}(1 - \widehat{p}), \qquad \mathbb{E}[C(\text{negative}) \mid \widehat{p}] = c_{\mathrm{FN}}\widehat{p}.$$

Choose positive when the first is no larger than the second:

$$c_{\mathrm{FP}}(1 - \widehat{p}) \le c_{\mathrm{FN}}\widehat{p} \quad \Longleftrightarrow \quad \widehat{p} \ge \frac{c_{\mathrm{FP}}}{c_{\mathrm{FP}} + c_{\mathrm{FN}}}.$$

This threshold changes when benefits, multiple actions, capacity constraints, uncalibrated scores, delayed outcomes, or heterogeneous consequences enter the decision. Those additions require an explicit decision model rather than a software default. ← **Main** · **Operating point**

Appendix A17: Wilson uncertainty for an accuracy estimate

What would vary if another comparable set of cases were drawn under the same sampling assumptions? For x correct classifications among n independent Bernoulli cases, $\widehat{p} = x/n$. The Wilson interval with standard-normal critical value z has centre and half-width

$$c = \frac{\widehat{p} + z^2/(2n)}{1 + z^2/n}, \qquad h = \frac{z}{1 + z^2/n}\sqrt{\frac{\widehat{p}(1-\widehat{p})}{n} + \frac{z^2}{4n^2}}.$$

Using $x = 7{,}427$, $n = 8{,}238$, and $z = 1.96$ gives

$$\widehat{p} = 0.90155, \qquad [c - h, c + h] \approx [0.89493, 0.90780].$$

The formula assumes independent Bernoulli cases and a fixed evaluated procedure. Here it is only a mechanical illustration around a validation proportion: this validation set participated in model selection, so the interval does not account for selection optimism. Dependence, time change, subgroup analysis and deployment shift require a design-aware calculation. ← **Main · Validation uncertainty**

Appendix A18: one global score, two output scales

For a numerical target, a linear model returns

$$\widehat{y} = \alpha_0 + \mathbf{x}^\top \boldsymbol{\alpha}.$$

For a binary target, logistic regression maps the same functional form of score into a probability:

$$\widehat{p} = \sigma(s) = \frac{1}{1+e^{-s}}, \qquad s = \beta_0 + \mathbf{x}^\top \boldsymbol{\beta} = \log\left(\frac{\widehat{p}}{1-\widehat{p}}\right).$$

Its coefficients are commonly fitted by minimising binary log loss,

$$\widehat{\boldsymbol{\beta}} \in \arg\min_{\boldsymbol{\beta}} -\sum_i \left[y_i \log p_i + (1-y_i)\log(1-p_i)\right], \qquad p_i = \sigma(\beta_0 + \mathbf{x}_i^\top \boldsymbol{\beta}).$$

The two coefficient vectors are fitted separately: the models share a global additive functional form, not numerical coefficient values. A one-unit increase in x_j, holding other inputs fixed, adds α_j to the linear prediction or β_j to the logistic log-odds; logistic odds are multiplied by e^{β_j}. These interpretations depend on the chosen units, transformations, interactions, included variables, and population. They describe the fitted rule's association; a causal interpretation requires a separate identification argument. ← **Main** · **Global-score models**

Appendix A19: tree and ensemble expressions

A fitted regression tree partitions feature space into terminal regions $R_1, \dots, R_M$ and returns one value in each region:

$$\widehat{f}_{\text{tree}}(\mathbf{x}) = \sum_{m=1}^{M} \widehat{c}_m \, \mathbf{1}\{\mathbf{x} \in R_m\}.$$

A random forest varies trees through resampled training cases and random feature subsets, then averages regression outputs or class probabilities:

$$\widehat{f}_{\text{RF}}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \widehat{f}_b(\mathbf{x}).$$

Gradient boosting constructs an additive sequence of small corrections:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu h_m(\mathbf{x}), \qquad 0 < \nu \le 1.$$

$\widehat{c}_m$ = fitted value in terminal region m; B = number of trees; h_m = fitted correction at stage m; ν = learning rate.

With squared loss, h_m fits the residual pattern left by F_{m-1}; with other differentiable losses, it follows the negative gradient. Validation must still choose depth, number of trees or stages, learning rate, and stopping. ← **Main** · **Tree families**

Appendix A20: unsupervised objectives depend on representation

Under Euclidean K-means, rescaling feature j by a_j changes its contribution to squared distance from

$$(x_{ij} - \mu_{c_i,j})^2 \quad \text{to} \quad a_j^2 (x_{ij} - \mu_{c_i,j})^2.$$

Standardising to $z_{ij} = (x_{ij} - \bar{x}_j)/s_j$ gives retained variables equal marginal variance, but it does not make their substantive influence equal. The representation and scaling remain modelling choices.

x_{ij} = feature j of case i; c_i = assigned cluster; $\mu_{c_i,j}$ = corresponding centroid coordinate; z_{ij} = standardised feature value.

For centred X with covariance $S = X^\top X/(n-1)$, the first PCA direction solves

$$\mathbf{v}_1 = \arg\max_{\|\mathbf{v}\|_2 = 1} \mathbf{v}^\top S \mathbf{v},$$

X = centred data matrix; n = number of cases; S = sample covariance matrix; $\mathbf{v}_1$ = first PCA direction; $\|\mathbf{v}\|_2$ = Euclidean length.

and is an eigenvector associated with the largest eigenvalue of S. K-means therefore privileges compact Euclidean groups; PCA privileges linear directions of high variance. Neither objective knows which structure is useful for the organisation's later decision. ← **Main** · **Unsupervised learning**

Appendix A21: neural layers and next-token loss

For layers $\ell = 1, \dots, L$,

$$\mathbf{h}^{(\ell)} = \phi_\ell\Big(\mathbf{W}_\ell \mathbf{h}^{(\ell-1)} + \mathbf{b}_\ell\Big), \qquad \mathbf{h}^{(0)} = \mathbf{x}.$$

$\mathbf{h}^{(\ell)}$ = layer-ℓ representation; $\mathbf{W}_\ell$, $\mathbf{b}_\ell$ = learned weights and biases; ϕ_ℓ = layer activation.

The parameters are the entries of the weight matrices and bias vectors. Backpropagation applies the chain rule to the training loss so an optimiser can update those parameters.

For a token sequence $w_1, \dots, w_T$, a common self-supervised language-model objective is

$$\mathcal{L}_{\text{token}} = -\sum_{t=1}^{T} \log p_\theta(w_t \mid w_1, \dots, w_{t-1}).$$

This trains conditional token probabilities. Whether a deployed assistant is helpful, grounded, safe, private, or appropriate requires system-level evidence beyond this pretraining objective.

← **Main** · **Deep models and LLMs**

Appendix A22: prediction and intervention are different estimands

Let $Y(1)$ denote the outcome if a customer is called and $Y(0)$ the outcome if the same customer is not called. The conditional treatment effect is

$$\tau(\mathbf{x}) = \mathbb{E}[Y(1) - Y(0) \mid X = \mathbf{x}].$$

$\mathbb{E}[\cdot]$ = expected value; A = observed call indicator (1 = called).

The outcome mean under contact is $\mu_1(\mathbf{x}) = \mathbb{E}[Y(1) \mid X = \mathbf{x}]$, whereas contacted historical records describe $m_{\text{obs}}(\mathbf{x}) = \Pr(Y = 1 \mid X = \mathbf{x}, A = 1)$. For one customer, only one of $Y(1)$ and $Y(0)$ is observed. Historical contact records do not automatically identify either potential-outcome mean or the effect of a new contact policy.

Estimating $\tau(\mathbf{x})$ requires an intervention design or defensible assumptions: consistency; conditional exchangeability given measured confounders; positivity of contact choices; no relevant interference between customers; and transportability to the intended policy. A randomised pilot can make this comparison credible when it is feasible and ethical. ← **Main** · **Response versus uplift**

Appendix A23: group metrics answer different denominator questions

For group g,

$$\text{Recall}_g = \Pr(\widehat{Y} = 1 \mid Y = 1, G = g), \qquad \text{FPR}_g = \Pr(\widehat{Y} = 1 \mid Y = 0, G = g),$$

$$\text{Precision}_g = \Pr(Y = 1 \mid \widehat{Y} = 1, G = g), \qquad \text{SelectionRate}_g = \Pr(\widehat{Y} = 1 \mid G = g).$$

G = group membership; Y = observed outcome; $\widehat{Y}$ = model's class decision.

Recall conditions on actual positive cases; false-positive rate conditions on actual negative cases; precision conditions on selected cases; selection rate conditions on everyone in the group. They therefore describe different experiences and can move differently when prevalence or threshold changes.

There is no context-free instruction that all group metrics must be equal. Labels can be flawed, groups can be heterogeneous, estimates can be uncertain, and several parity criteria can conflict.

Begin with the decision, the benefit or harm, and the people who bear each error; then choose and report the metrics that make those consequences visible. ← **Main** · **Unequal error burdens**

Appendix A24: policy-shaped data leave a counterfactual missing

Let $A = 1$ mean that a customer is called. Under a threshold policy,

$$A = \mathbf{1}\{\widehat{p}(X) \geq t\}.$$

$\widehat{p}(X)$ = predicted event probability from features X; t = call threshold; $Y(1), Y(0)$ = outcomes with and without a call.

For each customer, the policy reveals at most one factual outcome:

$$Y = A\,Y(1) + (1 - A)\,Y(0).$$

The counterfactual under the unreceived action remains unobserved. The next dataset is therefore generated partly by the current score, threshold, capacity, assignment and override behaviour.

If the ordinary outcome is recorded only for contacted customers, an additional label-observation problem also arises; that is a stricter *selective labels* case. Remedies can include explicit assignment and override logging, randomised exploration where appropriate, causal evaluation, and monitoring coverage as well as predictive error. ← **Main** · **Policy-shaped evidence**

Appendix A25: data, result, and case-study provenance

London case	synthetic fixed cohorts used only to demonstrate prediction, residuals, loss, chronology and evaluation discipline; no claim about current London prices
Bank data	UCI *Bank Marketing*, `bank-additional-full.csv`; 41,188 client–campaign records; target `y`; post-contact `duration` excluded
Bank split	stratified teaching split: 24,712 train, 8,238 validation and 8,238 untouched test; logistic pipeline selected on validation log loss; threshold 0.50 shown descriptively
Claim boundary	random row split, no stable client identifier and no causal no-call comparison: useful for teaching, not deployment-matched evidence
Published cases	Obermeyer et al., *Science* 366 (2019); Buolamwini and Gebru, *PMLR* 81 (2018); NIST AI RMF 1.0 (2023)

UCI Bank DOI · Obermeyer et al. · Gender Shades · NIST AI RMF

← **Main** · **Bounded evidence statement**