AI and Machine Learning

Day 3: From a spectacular score to a model that earns its place

Day 3 of 4

Fatih Kansoy
Oxford Summer School

Outline

1. Models that cut the feature space
2. Choosing flexibility fairly
3. Reading a score like a ranking
4. The audit, and the verdict
5. Close
6. Appendix

Opening audit: yesterday's five questions, answered

You wrote answers last night. Check them against these, briskly.

Q	Your answer should say
1	least squares minimises total squared error, and squared error points at the mean
2	the slope needs units, a comparison, and a scope: dollars per recorded mile, across these completed trips, as an association
3	19.27 multiplies the odds; it is not a 19-fold probability change
4	costs 4 and 80 give $\tau = 4/84 \approx 0.048$, and a fitted 0.30 clears it: call
5	an action meant to change behaviour needs uplift, not a response probability

Keep questions 4 and 5 to hand. Both work again today.

An AUC of 0.94, and an offer to deploy

0.94

The analytics team from Day 1 is back. Their new model, a random forest, scores an AUC of 0.94 on the bank data.

0.7838

Judged the same way, on the same records, yesterday's audited logistic.

Their offer: retire the logistic and deploy the forest.

One piece of equipment before you react. **AUC** is a ranking score: take one random subscriber and one random non-subscriber; AUC is the probability the model scores the subscriber higher, with a tie worth one half.

- a coin flip sits at 0.5
- perfect ranking at 1

> **Write one sentence on paper: deploy, or not, and your reason. Do not vote, and do not compare answers. We price your sentence at the close.**

Learning objectives for Day 3

Five outcomes, each one checkable. By the end of today you will be able to:

1. read a decision tree as regions, and say where its predictions come from
2. explain when averaging many trees helps, and what limits the help
3. run a cross-validation that never touches the validation or test records
4. read ROC and precision-recall curves, and pick the right one for a rare event
5. audit any impressive score with two questions

The fifth outcome closes the case you just wrote a verdict on.

The day in four parts

Part I **A model that cuts the feature space.** What a decision tree is, and what its leaves predict.

Part II **Choosing flexibility fairly.** The depth dial, cross-validation, random forests, and an introduction to boosting.

Part III **Reading a score like a ranking.** ROC, precision-recall, and the contact list.

Part IV **The audit, and the verdict.** Two questions, one reveal, and the price of 0.94.

On the week's chain, today lives in the model and evaluation links. The lab fits logistic regression, a tuned tree, and a random forest; boosting remains a lecture benchmark.

Where we are

1. Models that cut the feature space

2. Choosing flexibility fairly

3. Reading a score like a ranking

4. The audit, and the verdict

5. Close

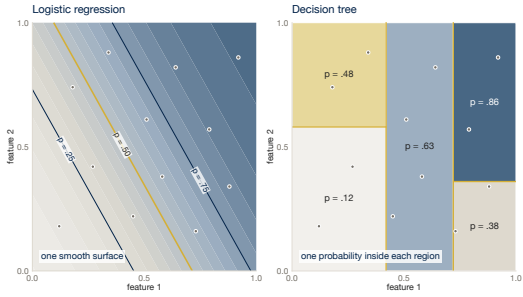
6. Appendix

One global rule versus regional rules

Day 2 closed with a question I promised to answer: what happens when the true rule has steps, corners, and interactions no line can hold?

The logistic index $\beta_0 + \mathbf{x}^\top \beta$ is one global recipe: each feature pushes every score the same way, by the same amount per unit.

Our bank data break that shape: response is higher among the youngest and oldest clients, and lowest in the middle. One age coefficient must choose one direction; a tree can learn both.



One smooth global surface against learned regional boundaries; illustrative.

A tree: yes/no questions ending in regions

Start with all 24,712 training records in one group.

1. ask one yes/no question about one feature, for example: is age above 60?
2. the answer sends each record left or right
3. ask a new question inside each branch, down to a fixed number of levels
4. records that run out of questions land in a **leaf**: one final group

The questions slice the feature space into non-overlapping regions, and every client, old or new, falls into exactly one. A 45-year-old meets at most three age questions in the tree you are about to see, and lands in the 30 to 58 group.

A tree is a rule you can read aloud: a few questions, then an answer.

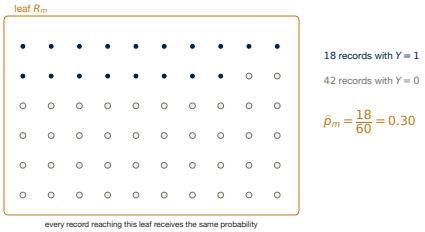
A leaf predicts its region's training mean

What number should each region announce? Day 2 answered this: under squared error, the best single guess for a group is its mean.

$$\widehat{f}(\mathbf{x}) = \bar{y}_{\text{leaf}(\mathbf{x})}$$

$\text{leaf}(\mathbf{x})$: the region client $\mathbf{x}$ falls into · $\bar{y}$: the average outcome of its training records

For a yes/no target, the average is the leaf's share of yes: a probability.



Illustrative: 18 of the 60 training records in this leaf are yes, so the leaf predicts 0.30.

A tree estimates Day 2's conditional mean $m(\mathbf{x})$, as a step function.

The age tree: a U-shape in eight leaves

A depth-3 tree of subscription on age alone, fitted on the training records only.

age range	clients	leaf share of yes
17–22	221	0.33
23–29	3,228	0.153
30–58	20,285	0.094
59–60	447	0.154
61–70	299	0.458
71–76	123	0.382
77–78	23	0.739
79–98	86	0.523

Response falls from youth into the working years, then rises steeply past 60: more than 20,000 clients sit in the 30–58 leaf at 0.094, while the 61–70 leaf sits at 0.458. A single coefficient on age had to pick one direction. The tree found both.

The 23-client leaf: memorisation in miniature

Look at the 77–78 leaf: 23 clients, share 0.739. About 17 of its 23 clients subscribed.

Should you announce 0.739 to the bank?

- an average of 23 outcomes is fragile: deal a slightly different sample and that share could move a long way
- the 30–58 leaf's 0.094 averages more than 20,000 outcomes and would barely twitch

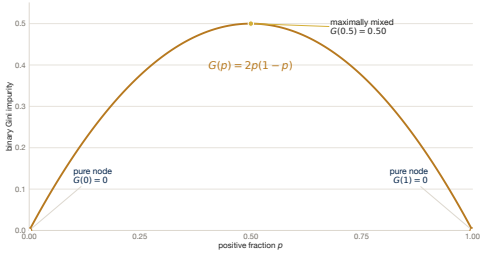
A tiny leaf is the tree memorising a handful of individuals and presenting the memory as a pattern.

> **Read a leaf's number together with its count. When trees grow deep in Part II, every leaf drifts towards the 23-client kind.**

How the tree chooses its questions

The tree tries every candidate: each feature, each cut point. It scores a candidate by how pure the two resulting groups would be; a group is pure when it is nearly all yes or nearly all no.

One named measure is **Gini impurity**: for a group with yes-share p, it equals $2p(1-p)$. Zero for a pure group, largest at fifty-fifty. The tree keeps the question that lowers the children's combined impurity most, then repeats inside each child.



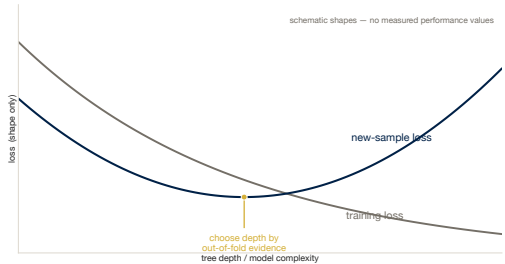
Gini impurity $2p(1-p)$: zero in pure groups, largest at an even split.

A greedy procedure: one best cut at a time, never revised. Appendix A holds the gain formula and the entropy alternative.

Depth: the dial that sets how far the tree bends

Depth counts the questions along the longest path: depth 1 gives two regions, depth 3 gave our eight age groups, and each extra level can double the leaves, up to 2^{depth}.

- turn the dial up: regions multiply and shrink, leaves drift towards the 23-client kind, then towards single clients
- turn the dial down: the tree cannot bend where the data do



Shape only, no measured values: training loss falls with depth while new-sample loss turns upward.

You must choose the depth, and the training score cannot be the chooser. Choosing it fairly is Part II's opening problem.

Part I checkpoint: you can read a tree

You can now:

- read a tree as yes/no questions that cut the feature space into regions
- say what a leaf predicts, and name the object the whole tree estimates: the conditional mean $m(\mathbf{x})$, in steps
- treat a small leaf's average with suspicion, and say in one line what Gini impurity measures

Next, the depth dial goes on trial.

Where we are

1. Models that cut the feature space

2. Choosing flexibility fairly

3. Reading a score like a ranking

4. The audit, and the verdict

5. Close

6. Appendix

Choosing the depth with yesterday's tools

You need nothing new to start. Day 1's split assigned the jobs: training fits, validation compares, the test set audits once.

So I fitted sixteen trees on the 24,712 training records, one per depth from 1 to 16, all on the valid pre-call features. Then I scored each tree twice:

- on its own training records
- on the 8,238 validation records it never saw

The judge is AUC, the ranking score defined at the opening.

> **Before the table arrives, commit to a guess: as depth grows, what will each of the two scores do?**

The depth sweep: training against validation

Six of the sixteen depths; the pattern holds throughout.

depth	training AUC	validation AUC
1	0.6964	0.7017
4	0.7545	0.7682
6	0.7902	0.7863
7	0.8005	0.7888
10	0.8291	0.7276
16	0.9311	**0.5937**

- the training column never stops rising: 0.9311 by depth 16
- the validation column peaks near depth 6 to 7, then collapses to 0.5937, barely above a coin flip
- small differences either way, like depth 1's, are noise; the shape is what matters

Day 1's overfitting warning light, drawn with real numbers.

One validation set is one draw

So we pick depth 7, where the validation column peaks at 0.7888? Not yet.

The validation set is one particular parcel of 8,238 records that the split happened to deal:

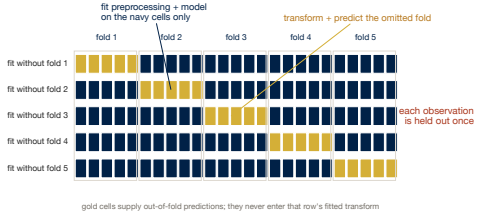
- compare sixteen candidates on the same parcel and crown the maximum, and part of what you crown is luck: the depth that happens to suit those records
- Day 1 warned that repeated looks turn the test set into a second validation set; the same logic bites the validation set itself, gently, every time it referees a many-way contest

We keep the referee but stop relying on a single one.

Five folds inside the training set

Cross-validation manufactures several referees from the training data alone. Cut the 24,712 training records into five folds, stratified so each keeps approximately the 11.3% yes-rate, then rotate:

1. hold out fold 1, fit on the other four, predict fold 1
2. repeat until each fold has had its turn
3. pool the 24,712 held-out predictions, compute one AUC



Each record is predicted exactly once, by a workflow fitted without it.

> **The original validation and test sets stay outside this CV loop; the price is five fits, the gain a judgement less dependent on one parcel.**

The sweep under the five-fold judge

The same sweep, now with the cross-validated column beside validation.

depth	validation AUC	CV AUC
1	0.7017	0.6894
4	0.7682	0.7456
5	0.7849	0.7679
6	0.7863	0.7687
7	0.7888	0.7582
16	0.5937	0.6063

Both judges agree on the story: rise, peak, collapse. They disagree slightly on the peak: validation points at depth 7, cross-validation at 5 to 6. That small disagreement is itself the lesson. From here, every modelling choice is refereed by CV inside the training set.

> **One split is a noisy judge; five folds average the noise away.**

Backtest overfitting: many trials, one history

We compared sixteen depths. Finance compares thousands of trading rules, and there the mathematics turns against you.

Bailey, Borwein, López de Prado, and Zhu analysed it (Notices of the American Mathematical Society, 2014): as more strategy configurations are tested against one price history, it becomes easier to select an impressive in-sample Sharpe ratio that reflects selection noise rather than durable skill.

The result concerns the act of selection itself, not any particular fund.

> **The defence is the same at every scale: count your trials, and hold out data the winner has never touched.**

A deep tree changes its mind when the data twitch

Why did the deep tree collapse? Its early questions decide everything below them. Refit the same depth-16 tree on a slightly different sample, and one early cut lands elsewhere; the whole staircase below it rearranges. Statisticians call this **high variance**: the fitted rule swings from sample to sample.

Here is the useful part. A high-variance rule is wrong in different directions on different draws. Average many such rules and much of the noise cancels, the way many shaky measurements average into one steady one.

So: grow many deep trees and average their predictions. One formula says exactly how much that helps, and it is the day's one piece of owned mathematics.

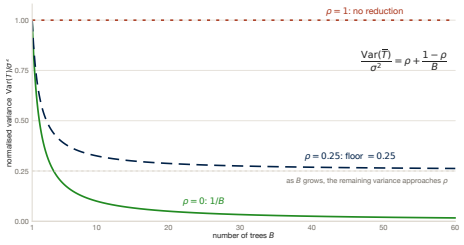
Averaging correlated trees: the variance floor

Trees grown from the same records make related mistakes:

$$\operatorname{Var}\left(\frac{1}{B}\sum_{b=1}^{B} T_b\right) = \rho\,\sigma^2 + \frac{(1-\rho)\,\sigma^2}{B}$$

T_b: tree b's prediction · σ^2: one tree's variance · ρ: average error correlation between two trees

The second term dies as B grows: more trees clear it away for free. The first term never moves (algebra in Appendix B).



Averaging removes idiosyncratic variance but not the shared part.

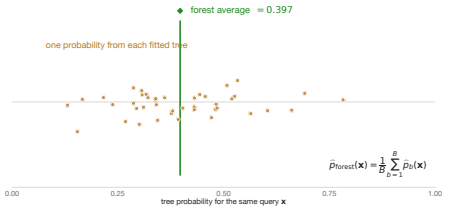
The floor is $\rho\sigma^2$: the lever is less correlated trees, not more trees.

The random forest: different rows, different features

To lower the floor, make the trees disagree. A random forest forces it twice:

- **the rows**: each tree trains on a bootstrap resample, 24,712 draws with replacement, a different mix every time
- **the features**: at every split, only a random subset may be considered, so trees lead with different questions

Both moves are designed to lower ρ. Our forest grows 300 trees, each deep and individually noisy.



A forest prediction is the average of its component-tree probabilities.

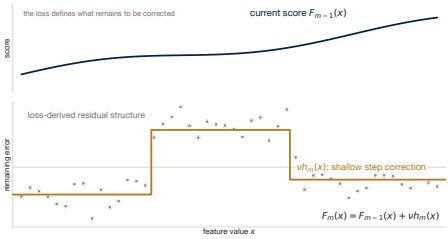
About 36.8% of the records miss any given resample; Appendix E does that arithmetic.

Boosting: small trees that follow the loss gradient

The forest's bet: grow big trees in parallel and average the noise away. Boosting makes the opposite bet: build small trees in sequence.

1. begin with a simple prediction
2. calculate each record's **negative loss gradient**: the direction that would reduce its current error
3. fit a small tree to those directions and add a cautious step
4. repeat to a chosen limit, or stop when held-out improvement ends

Each tree corrects the current ensemble rather than averaging (Appendix C).



A shallow tree adds a cautious correction where structured error remains.

On valid features, only boosting beats the logistic

Four candidates, same valid pre-call features, same judge: pooled five-fold CV inside the training set. Read it slowly; it resists a popular story.

model	CV AUC
single tree, CV-selected depth 6	0.7687
random forest, 300 trees	0.7709
logistic regression, Day 2's full model	0.7838
boosting	**0.7912**

- the forest, with its 300 trees, does not beat yesterday's single fitted line
- even at its best depth, a lone tree reaches 0.7687 and still trails
- boosting earns an edge of about 0.007; whether that is worth paying for waits for the verdict

> **Flexibility must earn its keep; here only boosting does.**

Part II checkpoint: you can choose flexibility fairly

You can now:

- choose a flexibility setting with a validation sweep, and explain why cross-validation steadies the choice
- state the variance of an average of correlated trees, and name the lever: decorrelation
- say in one sentence each how a forest and boosting are built, and quote today's comparison

The comparisons all leaned on AUC. Part III earns that number.

Where we are

1. Models that cut the feature space

2. Choosing flexibility fairly

3. Reading a score like a ranking

4. The audit, and the verdict

5. Close

6. Appendix

AUC is a pairwise ranking probability

Here is the opening definition, now with its mechanics. Take every possible pair of one actual subscriber and one actual non-subscriber. Ask, pair by pair: did the model score the subscriber higher? Count one for a correct ordering, one half for a tie, and zero for a reversal. AUC is the average.

Three consequences follow:

- a constant score ties every pair, so its half-credits average to 0.5; random scoring lands there in expectation
- AUC never mentions a threshold: it judges the whole ordering at once, which suits model comparison before anyone designs a decision
- boosting's 0.7912 reads plainly: draw one subscriber and one non-subscriber at random, and the probability of a correct ordering, plus half the probability of a tie, is about 0.79

Appendix D gives this statistic its formal name.

The ROC curve: the confusion matrix at every threshold

On Day 1 I called the 0.50 cut-off a convention and promised whole curves of thresholds. Here is the promise kept. Fix a threshold τ and the confusion matrix follows, with two shares:

$$\text{TPR} = \frac{TP}{TP + FN}, \qquad \text{FPR} = \frac{FP}{FP + TN}.$$

TPR: Day 1's recall · FPR: the share of non-subscribers flagged

Now sweep τ from 1 down to 0:

- at $\tau = 1$ nobody is flagged: the point $(0, 0)$; at $\tau = 0$ everyone is: $(1, 1)$
- every τ between is one point, and the trace is the ROC curve; AUC is exactly the area under it
- a no-skill scorer walks the diagonal

Precision-recall: the stricter judge at 11.3%

FPR divides by the non-subscribers, and non-subscribers are nearly nine records in ten here. Wrongly flag a few thousand of them and FPR barely moves: with rare events, ROC can look fine while your call list fills with waste. Day 1 met this shape once already: predicting no for everyone scored 88.7% accuracy.

The precision-recall curve swaps FPR for **precision**, the share of your flagged list that really subscribed. Precision divides by your own list, so every false alarm costs it.

The baselines say it all:

- the ROC diagonal sits at 0.5, for any file
- the no-skill PR baseline is the yes-share itself, Day 1's training response rate: 0.1127; precision can fall below it at some thresholds

For a rare event, read the PR curve beside the ROC curve, and expect smaller numbers.

Average precision asks a different ranking question

Average precision, AP, weights precision at each increase in recall. It plays for precision-recall the one-number role AUC plays for ROC. A constant score's AP equals the yes-share.

model	CV AP
constant-score floor	0.1127
single tree, CV-selected depth 6	0.4130
random forest, 300 trees	0.4078
logistic regression	0.4395
boosting	0.4514

These numbers are not a "harsher" version of AUC. AUC asks how often a positive outranks a negative; AP summarises precision as recall grows, with prevalence as its baseline. Both rank boosting first; AP keeps rare-event call-list quality visible.

Two thresholds, two operating points

A ranking becomes a call list only when you cut it. If its scores are calibrated probabilities, Day 2's costs determine the cut: costs 4 and 80 give $\tau = 4/84 = 0.0476$, the 0.048 you derived. The day's selected model on the frozen course benchmark, cut twice:

τ	TP	FP	FN	TN	precision	recall
0.0476	813	4,361	115	2,949	0.157	0.876
0.50	233	103	695	7,207	0.693	0.251

- at the cost-derived cut: 87.6% of the 928 test subscribers found, 4,361 wasted calls accepted
- at the default: a short, clean list, and about three subscribers in four missed

Both points are legitimate if the probabilities are calibrated: one model, one curve, two cost tables. These rows use the frozen benchmark; Part IV explains its audit status.

The top-quarter call list: 2.81 times the random rate

The desk question, at last: the bank can afford to call about a quarter of the file. Whom does it call? Score every record, sort, call from the top.

$$\frac{0.316}{0.1126} \approx 2.81$$

0.316: subscription rate in the benchmark top quarter (2,059 records) · 0.1126: benchmark yes-share, what random dialling reaches

Same budget, nearly three times the subscribers found. This pays Day 1's last unpaid promise: the ranking output type now has its own measures.

One boundary survives from Day 2: ranking by response probability is still not ranking by uplift. Whom the call would change is a different question.

Part III checkpoint: you can read a ranking

You can now:

- read AUC as a pairwise ranking probability, and ROC as the confusion matrix at every threshold
- explain why precision-recall judges a rare event more strictly, with the two no-skill references: ROC AUC 0.5 and PR 0.1127
- turn a ranking into an operating point with costs, and into a contact list with a budget

Every tool the opening claim requires is now on the bench.

Where we are

1. Models that cut the feature space
2. Choosing flexibility fairly
3. Reading a score like a ranking
4. **The audit, and the verdict**
5. Close
6. Appendix

Two questions audit any impressive score

Before admiring any score, I ask two questions, in this order.

> **Question one: could every feature be known at the moment of prediction?** Day 1's information cutoff, asked column by column. It hunts timeline leakage: features written down after the outcome they predict.

> **Question two: does the evaluation imitate the deployment?** Does the split grade the model on the exam it will actually sit? A random split answers a stable-population question; a future-period question needs a split that respects time.

The rest of Part IV applies each question once. The first catches the 0.94. The second gets its own demonstration.

The reveal: the 0.94 model read the duration column

I asked the team for their feature list. It includes **duration**, the length of the final call. Day 1 ruled that column out at the information cutoff. Here is the full damage: same five folds, same training records, one column added.

model	valid features, CV AUC	with duration, CV AUC
single tree, CV-selected depth 6	0.7687	0.9184
random forest, 300 trees	0.7709	**0.9398**
logistic regression	0.7838	0.9328
boosting	0.7912	0.9475

The team's 0.94 is the forest's leaked row: 0.9398. Every model gains between 0.15 and 0.17 of AUC.

> **The improvement is Day 1's timeline leakage, priced in today's units.**

Every model exploits the leak differently

Look again at the leaked column.

- the ranking reorders: with duration inside, the forest overtakes the logistic, and boosting reaches 0.9475
- every model improves sharply; the forest gains the most here, but there is no rule that leakage gains must rise monotonically with model flexibility
- model capacity changes how efficiently the invalid signal is used, not whether the signal becomes valid

The forest also tells on itself. Ask the leaked forest how much each feature contributed to its splits: **duration** alone carries an importance share of 0.278, the largest in the model. Importance scores are rough guides as explanations; as leak detectors they point straight at the culprit.

Average precision inflates too: boosting's 0.4514 becomes 0.6612. A strong validation score cannot rescue a feature that arrives too late.

Every learned preprocessing step lives inside its fold

Question one has a quieter cousin, and cross-validation multiplies it. Day 1 named it evaluation leakage: information from held-out records leaking into how the model is prepared. The instance you know: standardise the columns using all records, then split.

With five folds, each preparation step has five chances to make that mistake. The rule, which modern toolkits automate:

- treat preprocessing as part of the model
- scaling, filling in missing values, encoding categories: every step that learns numbers from data is refit inside each fold, on that fold's training portion only

You refit one pipeline five times, once per fold. The concept is what matters; today's lab shows the code.

A recession task tests the second question

Our stratified bank split matches the bank's stable-population claim. Now a task where it cannot work: predict whether the US economy will be in recession exactly 12 months later.

The data, from FRED, the Federal Reserve's public data service:

- 521 monthly rows, 1982 to 2025
- three features: two Treasury yield-curve spreads (T10Y3M, T10Y2Y) and the unemployment rate (UNRATE)
- target: recession status exactly 12 months later, from the NBER-based indicator (USREC); base rate 0.069

The model: a random forest, 300 trees, the same recipe as ours.

> **Before the result, decide: what must the split respect here?**

Two evaluation designs answer different questions

evaluation design	AUC
random five-fold CV over all 521 months	0.934
train to 2009, hold out 2010 onwards	0.727

- random CV trains on later observations and near-neighbours of many months it judges; 0.934 is not a forecast into an unseen future regime
- the chronological holdout asks the harder question: can earlier history rank recession risk in a later period?

The later cohort contains only 2 positive months among 185, so its AUC is highly unstable and illustrative; it is not evidence that 0.727 is the model's durable forecasting ability.

> **A score is inseparable from the population, period, and information pattern on which it was measured.**

The 0.207 gap is a warning, not an effect estimate

The forest and three features are the same, but the two rows do not sit the same exam:

- their training information differs: random CV can train on later regimes; the chronological model cannot
- their scoring populations differ: random CV scores the whole history; the holdout scores only 2010 onwards
- the target is observed 12 months later, so a live backtest would also leave a 12-month label gap before every test period

Therefore $0.934 - 0.727 = 0.207$ cannot be interpreted as a causal estimate of "the cost of shuffling." It shows how an apparently small design choice can change the question being measured.

State the population, period, and information pattern before comparing two scores.

Case study: the COVID imaging review

> Roberts and colleagues (Nature Machine Intelligence, 2021) reviewed the models proposed for detecting COVID-19 from chest X-rays and CT scans. From more than two thousand candidate studies, 62 met the bar for detailed examination. The reviewers judged none fit for clinical use. Among the recurring causes: data leakage and biased evaluation designs.

The finding is not evidence that medical AI is useless. Hundreds of teams with real data and urgent motives produced models whose evaluations could not support deployment.

> **The failures sat where today's two questions look.**

Layer 1 of the verdict: reject the 0.94

Apply question one and the opening claim comes apart.

- the forest's 0.9398 reads a post-call column: it answers an after-the-call question, and the bank needs a before-the-call model
- strip the leak and the same forest scores 0.7709, below the logistic model the team proposed to retire, 0.7838

So the first layer of the verdict: reject the offer as made. The spectacular score was not evidence of a better model. It was Day 1's timeline leakage, exploited more efficiently by a more flexible model, which is what flexible models do to leaks.

Layer 2 of the verdict: select boosting, then lock it

Leak removed, today's menu reads: boosting 0.7912, logistic 0.7838, forest 0.7709, tree 0.7687, all by cross-validation inside the training set. That referee selects boosting.

The frozen course benchmark reports:

> benchmark AUC 0.806, benchmark AP 0.486

against a yes-share of 0.1126 · Part III's operating points and contact list use the same frozen cohort

Day 2 has already displayed this cohort, so it is no longer a pristine lockbox for a revised model. In a live project, lock today's winner and evaluate it on a fresh later cohort. These values are a common course benchmark, not a second independent audit.

A modest edge, and a judgement

What would the bank actually gain?

- in desk terms: the top-quarter call list reaches subscribers at 2.81 times the random rate, and the cost-derived threshold finds 87.6% of them; those numbers show operational value, subject to calibration and a fresh audit
- against yesterday's logistic, boosting's cross-validated edge is about 0.007 of AUC

Whether 0.007 justifies a system that is harder to build, explain, and maintain is a judgement about costs, and arithmetic alone cannot settle it. You met the precedent yesterday: Netflix never fully put its prize-winning ensemble into production.

> **Either decision can be sound: advance boosting to a fresh audit, or keep the simpler logistic. What is not sound is deciding on 0.94.**

Part IV checkpoint: you can audit a score

You can now:

- audit an impressive score with the two questions: feature timing first, then split realism
- describe what a leak does to a model menu: every score inflates, by 0.15 to 0.17 of AUC here, although the gains are not monotonic in flexibility
- state the two-layer verdict with its numbers: reject 0.9398, select boosting at 0.7912, then require a fresh locked audit

The verdict is ready to meet your opening sentence.

Where we are

1. Models that cut the feature space
2. Choosing flexibility fairly
3. Reading a score like a ranking
4. The audit, and the verdict
5. Close
6. Appendix

Your opening sentence meets the price of 0.94

Take out the sentence you wrote at the start of today. Hold it against the pricing.

Layer 1. The 0.94 was timeline leakage amplified by flexibility. On valid features the same forest falls to 0.7709 and loses to yesterday's logistic at 0.7838. Reject the offer.

Layer 2. Rebuilt on valid features and refereed by cross-validation, boosting earns a modest edge: 0.7912. The shared course benchmark is 0.806, with a contact list worth 2.81 times random dialling; a live revision still needs a fresh locked cohort.

> **The right verdict needed both layers, and at the start of today nobody in the room had either.**

What you fitted, compared, and inspected today

Object	What you learned to do
decision tree	read recursive regions and their leaf probabilities; control depth
random forest	explain parallel averaging, randomisation, and the correlation floor
gradient boosting	interpret sequential correction and a frozen benchmark comparison
judging tools	use CV, ROC, PR, AP, lift, and an operating budget for declared jobs
validity checks	test feature timing and whether the split imitates deployment

The application fits logistic regression, a tuned tree, and a random forest. Boosting remains an interpreted extension rather than a model you independently train.

The comparison discipline fits on one slide

Keep this slide. It is the day, and most of the week, in seven lines.

1. start from the decision and its costs: they choose the metric and the operating point
2. match the split to the deployment: random for a stable population, chronological for a future
3. fix the baseline before admiring anything: a constant score has AUC 0.5 and AP equal to the yes-share
4. compare candidates by cross-validation inside the training set
5. audit the final choice on the test set, once
6. check every feature against the information cutoff
7. write down what you chose and why, so the next person can audit you

Each line marks a place where, this week, you watched a score mislead someone.

Review questions for tomorrow

Write short answers tonight; Day 4 opens by auditing them.

1. A tree's leaf prediction is an average. Why, and how should you read a 23-client leaf?
2. Averaging correlated trees stops helping after a point. Which formula says why, and what is the lever?
3. Why do five folds judge a modelling choice better than one validation set?
4. When does a precision-recall curve tell you more than a ROC curve, and why?
5. A colleague reports a spectacular score. What two audit questions do you ask first?

Every answer is on today's slides; none needs a computer.

The bridge to Day 4

Every dataset this week arrived with answers attached: a recorded fare, a recorded yes or no. Every method you now own leaned on that column.

Tomorrow the labels disappear. What structure can be found when nothing marks the answer: which clients resemble each other, and which few forces move dozens of prices at once? That is clustering and principal components, and they lead to the representations behind generative AI.

One thread travels with us: today's audit is a snapshot, and a deployed model must also be watched. Monitoring belongs to tomorrow, and the week then closes with the question Day 1 left open: what a deployed system owes the people it scores.

> **Bring your five written answers. We start with them.**

Where we are

1. Models that cut the feature space
2. Choosing flexibility fairly
3. Reading a score like a ranking
4. The audit, and the verdict
5. Close
6. Appendix

Appendix A: impurity measures and the split gain

For a group with yes-share p, the two standard impurity measures are Gini and entropy; both are zero when the group is pure and largest at $p = 0.5$, and in practice they rarely disagree about the chosen split. A candidate question splits a parent of n records into children L and R; its gain Δ is the impurity removed:

$$G(p) = 2p(1-p), \qquad H(p) = -p\log_2 p - (1-p)\log_2(1-p),$$

$$\Delta = I_{\text{parent}} - \frac{n_L}{n} I_L - \frac{n_R}{n} I_R.$$

I: a group's impurity · n_L/n, n_R/n: the children's shares of the parent

The tree evaluates every feature and cut point, keeps the largest Δ, and repeats inside each child until a stopping rule bites (Appendix G).

Appendix B: the variance of an average of correlated trees

Let each tree's prediction T_b have variance σ^2, and let every pair of trees have correlation ρ, so each covariance is $\rho\sigma^2$. The variance of the average expands into B variance terms and $B(B-1)$ covariance terms:

$$\operatorname{Var}\left(\frac{1}{B}\sum_{b=1}^{B} T_b\right) = \frac{1}{B^2}\left[B\,\sigma^2 + B(B-1)\,\rho\,\sigma^2\right] = \rho\,\sigma^2 + \frac{(1-\rho)\,\sigma^2}{B}.$$

As $B \to \infty$ the second term vanishes and the variance settles at $\rho\sigma^2$.

Equal variances and a common pairwise correlation are simplifications; they keep the mechanism visible without changing its direction.

Appendix C: boosting as stagewise additive fitting

Write the ensemble after m rounds as F_m. Start from a constant, F_0. At each round, fit a small tree h_m to the current shortfall and add a damped step:

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

h_m: the round-m tree, fitted to the current shortfall · ν: the learning rate

- for squared error the shortfall is the residual $y_i - F_{m-1}(\mathbf{x}_i)$; for other losses it is the negative gradient of the loss, which names gradient boosting
- the number of rounds and ν are flexibility dials, chosen exactly as Part II chose depth: by CV inside the training set

Appendix D: AUC as the Mann-Whitney statistic

With n_+ actual positives, n_- actual negatives, and scores s, count the correctly ordered pairs, with ties worth one half:

$$\mathrm{AUC} = \frac{1}{n_+ n_-} \sum_{i:\, y_i = 1} \sum_{j:\, y_j = 0} \Big[\mathbf{1}(s_i > s_j) + \tfrac{1}{2}\, \mathbf{1}(s_i = s_j) \Big].$$

$\mathbf{1}(\cdot)$: 1 if the condition holds, else 0 · the double sum runs over every positive-negative pair

Two consequences the main deck used: a constant score ties every pair and lands exactly on 0.5, and AUC depends only on the ordering of the scores, never their absolute values.

Appendix E: the out-of-bag 36.8%

A bootstrap resample draws n rows from n with replacement. The chance that one given row is missed by a single draw is $1 - 1/n$; the chance that all n draws miss it is

$$\left(1 - \frac{1}{n}\right)^n \longrightarrow e^{-1} \approx 0.368.$$

So about 36.8% of the training rows sit outside any one tree's resample. Scoring each tree on its own left-out rows gives the forest a free internal error estimate, the out-of-bag estimate.

Useful for quick checks, and still internal to training: it never replaces the deployment-matched split or the one test audit.

Appendix F: cross-validation variants

Cross-validation inherits the deployment-matching duty of Day 1's Appendix C, fold by fold. Each deployment question from Day 1's split table has a matching fold design:

deployment question	matching CV variant
stable population	stratified K-fold: today's choice
a future period	temporal CV: train on past folds, judge the next period
a new person or firm	grouped K-fold
a new location	leave-one-site-out

The principle does not change at fold scale: the fold you judge on must imitate the cases the model will face. The FRED demonstration is what happens when it does not.

Appendix G: the controls that tame a single tree

control	what it does	more flexible when
maximum depth	caps the questions along any path	deeper
minimum leaf size	forbids leaves below a set count	smaller
minimum split size	stops splitting groups below a set count	smaller
cost-complexity pruning	grows deep, then removes cuts that do not pay on held-out data	less pruning

Every row is the same dial in different clothes, and you choose every setting the same way: by cross-validation inside the training set.

Forests take the other route: they keep individual trees deep and noisy, and buy stability through averaging instead.

References: data and cases

- James, G., Witten, D., Hastie, T., Tibshirani, R., and Taylor, J. (2023). *An Introduction to Statistical Learning*. Springer, ch. 8; Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*, 2nd ed. Springer.
- Bailey, D. H., Borwein, J. M., López de Prado, M., and Zhu, Q. J. (2014). "Pseudo-Mathematics and Financial Charlatanism." *Notices of the American Mathematical Society*, 61(5), 458–471.
- Roberts, M., Driggs, D., Thorpe, M., et al. (2021). "Common Pitfalls and Recommendations for Using Machine Learning to Detect and Prognosticate for COVID-19 Using Chest Radiographs and CT Scans." *Nature Machine Intelligence*, 3, 199–217.
- Chen, T., and Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System." *Proc. 22nd ACM SIGKDD*, 785–794.
- Moro, S., Rita, P., and Cortez, P. (2014). Bank Marketing [Dataset]. UCI Machine Learning Repository.
- Federal Reserve Bank of St. Louis, FRED: series T10Y3M, T10Y2Y, UNRATE, USREC.

Same frozen bank file and 60/20/20 split as Days 1 and 2. Day 3's computation opens the frozen benchmark only for the CV-selected model; Day 2 has already reported the same cohort, so the main deck does not call it a fresh independent audit.