Practice Final — Solutions

MSE 125: Applied Statistics — Spring 2026 Practice

3 sections, 100 points, 90 minutes. Closed-book, no devices, no AI. The formula strip at the head of Section 1 is the only reference permitted. This practice exam mirrors the structure of the real exam; the data, scenarios, and specific numbers differ.

Section 1 — Tool literacy (25 pts, ∼22 min)

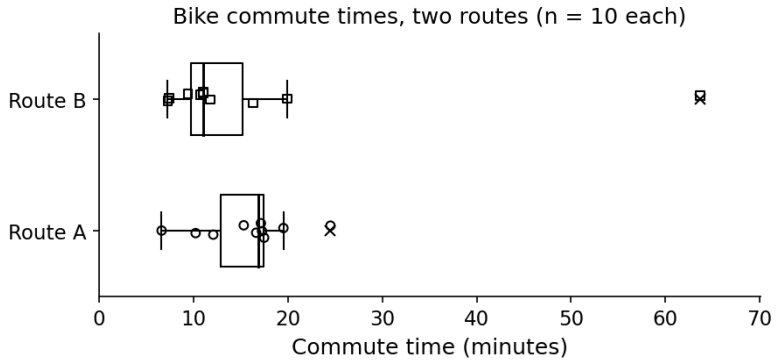
Formula strip (Section 1 only):

$$\text{Bonferroni cutoff} = \alpha/m \qquad \mathbb{E}[\text{false positives, all-null}] = m\alpha \qquad \text{Recall} = \frac{TP}{TP+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \qquad \text{Residual} = y - \hat{y} \qquad |t| = \frac{|\hat{\beta}|}{\text{SE}(\hat{\beta})} \qquad z_{0.975} \approx 1.96$$

1. (2 points) **Test selection.** You measure bike commute times (minutes) on two routes. The boxplots below show $n = 10$ observations per route; both distributions are right-skewed and Route B contains one clear outlier.



Which test is most appropriate for comparing the two routes' mean commute times?

A. Two-sample t-test

B. Permutation test on the difference in means

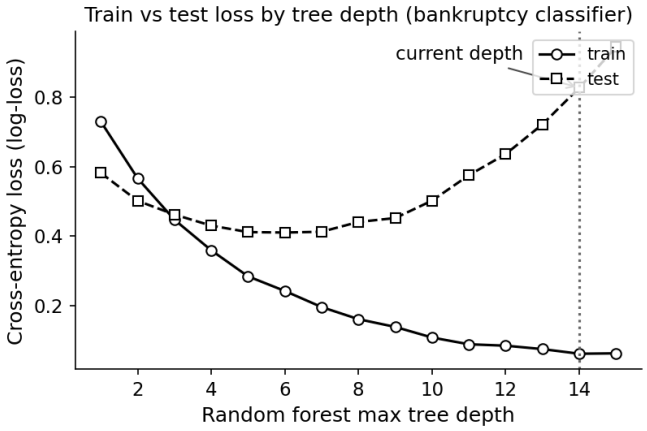
C. Sign test on each route's distribution

D. Bootstrap CI for each route's mean, separately

Solution.

Permutation test. At $n = 10$ with visible right-skew and an outlier, the t-test's normality assumption is unreliable (the CLT has not kicked in). A permutation test makes no distributional assumption beyond exchangeability under H_0, which a two-route comparison supports. Bootstrap CIs of each route separately do not compare the two routes. The sign test compares paired or median responses, not two independent groups' means.

2. (2 points) **Identify under/overfit.** The plot shows training and test cross-entropy loss for a random forest as a function of `max_depth`. The model is currently trained at the marked depth.



The model is currently:

A. Underfitting (high training and test loss)

B. Good fit (test loss at its minimum)

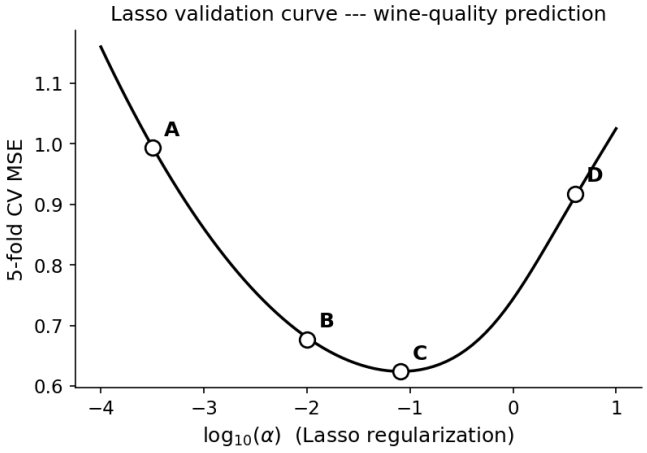
C. Overfitting (training loss very low, test loss rising)

D. Cannot tell from this plot

Solution.

Overfitting. Training loss continues to fall while test loss rises — the model is memorizing training-specific patterns that do not generalize. The right depth is the one that minimizes *test* loss, which is at a *lower* `max_depth` than where the model sits now.

3. (2 points) **Best point on a validation curve.** The plot shows mean 5-fold CV mean squared error for Lasso as a function of $\log_{10}(\alpha)$ on a wine-quality prediction task. Four candidate operating points A, B, C, D are marked.



Which point should you choose for deployment?

A. A (smallest α)

B. B (small α)

C. C (CV-error minimum)

D. D (largest α)

Solution.

C. The deployment choice is the hyperparameter that minimizes cross-validation error. A and B under-regularize (high variance); D over-regularizes (high bias and the loss shoots up as the model shrinks toward zero).

4. (2 points) **Cross-validation protocol.** You have a panel of standardized test-score records for 200 schools across 6 academic years. You want to estimate how well your model will generalize to a *new school* the district has not yet onboarded. Which CV protocol gives an honest estimate?

A. Random 5-fold cross-validation across all (school, year) records

B. Walk-forward validation by year

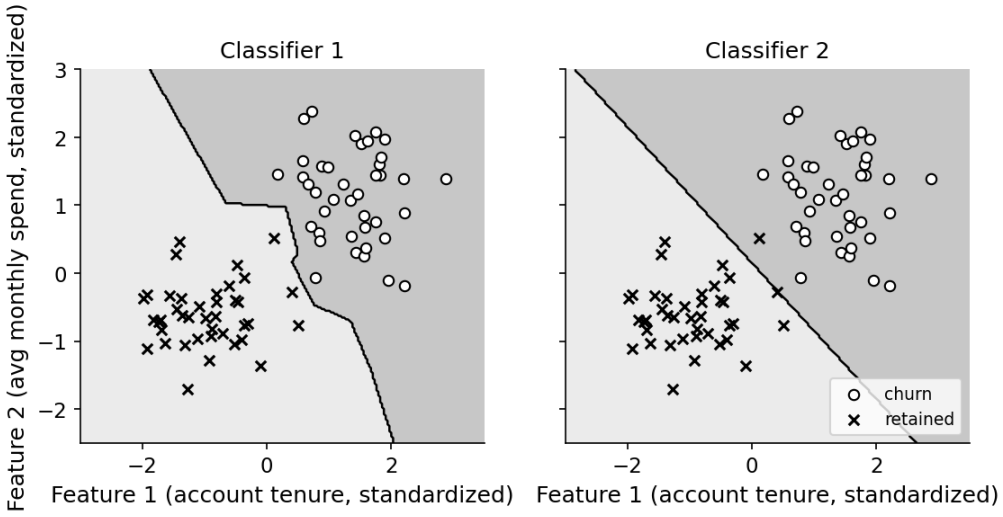
C. Grouped CV by school (hold out all records from a school in each fold)

D. Stratified k-fold, stratified by year

Solution.

Grouped CV by school. The deployment task asks about a school the model has never seen. Random folds leak the same school's data across train and test (the model can "memorize" that school's level rather than learning what generalizes). Walk-forward by year fixes the temporal leak but still trains and tests on the same schools. Stratification by year preserves seasonal balance but again does not isolate schools. Only grouped CV holds out the relevant unit of generalization.

5. (2 points) **Decision boundary identification.** The two panels below show classifier decision boundaries on the same 2D customer-churn scatter. Classifier 1 produces a jagged, locally-shaped boundary; Classifier 2 produces a single straight line.



Which panel was produced by which model?

A. Classifier 1: linear SVM; Classifier 2: k-nearest neighbours

B. Classifier 1: k-nearest neighbours; Classifier 2: linear SVM

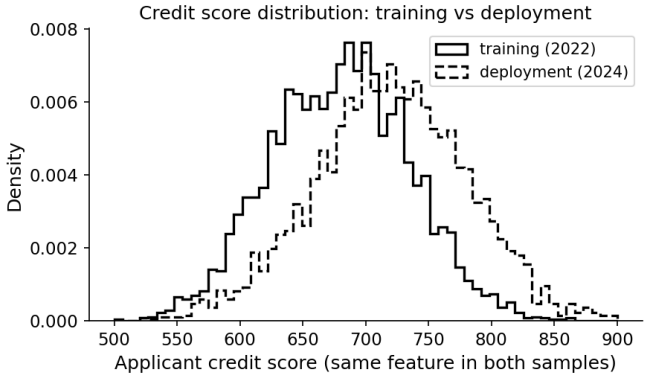
C. Both: k-nearest neighbours, with different k values

D. Both: linear SVMs, with different features

Solution.

Classifier 1 is k-NN; Classifier 2 is a linear SVM. A linear SVM's boundary in 2D is a hyperplane (a straight line), as shown in Classifier 2. k-NN's boundary is determined locally by the nearest training points, producing jagged, irregularly-shaped regions whose detail reflects the training points' positions — the Classifier 1 signature.

6. (2 points) **Identify distribution shift.** The plot below shows kernel-density estimates of the same input feature (applicant `credit_score`) at two times: the training data (collected 2022) and the deployment data (collected 2024, after a policy change in upstream credit reporting). The two curves have visibly different shapes.



What does this plot evidence?

A. Covariate shift ($P(X)$ has changed)

B. Label shift ($P(Y)$ has changed)

C. No shift; differences are within sampling noise

D. Cannot tell — need to see $P(Y \mid X)$

Solution.

Covariate shift. The plot shows the density of a feature X shifting between training and deployment — the definition of covariate shift, $P(X)$ changes. Label shift ($P(Y)$ change) cannot be diagnosed without Y. The shape difference is too large to attribute to sampling noise. Whether $P(Y \mid X)$ *also* changed is a separate question this plot does not address.

7. (2 points) **Lasso vs ridge vs OLS.** A colleague says: "I have 40 candidate features. I am required to keep all of them in the model (a contract obligation). The OLS fit is unstable across data refreshes — coefficients flip sign from month to month. I need a model with stable, lower-variance coefficient estimates that still uses every feature." Which model best supports this goal?

A. Ordinary least squares (OLS) with all 40 features

B. Ridge regression

C. Lasso regression

D. Decision tree with all 40 features available

Solution.

Ridge. The L2 penalty shrinks coefficients toward zero, reducing variance and stabilizing them across data refreshes, but rarely sets any coefficient exactly to zero — so every feature stays in the model as the contract requires. OLS is the unstable baseline that motivated the change. Lasso would set most coefficients to zero (violates the contract). A decision tree is not a linear model and does not produce coefficients at all.

8. (2 points) **Plot type for the question.** You want to know whether the relationship between daily temperature (continuous) and total bike-share rentals (continuous) is different on weekdays vs weekends. Which plot type is best?

A. Histogram of rentals, one panel per weekday/weekend

B. Bar chart of mean rentals by temperature bin

C. Scatter plot of rentals against temperature, with weekday/weekend distinguished by marker or panel

D. Box plot of rentals, one box per weekday/weekend

Solution.

Scatter plot with weekday/weekend distinguished. Two continuous variables plus a categorical grouping (weekday vs weekend) is the canonical use case for a scatter plot with the third dimension on marker or facet. Histograms and box plots show marginal distributions but lose the temperature–rentals joint structure. A bar chart of binned means destroys within-bin variation and hides curvature.

9. (2 points) **Confusion matrix → precision.** A radiology AI classifier flags scans as "suspicious for cancer." On 1,000 screening scans (100 actual cancer cases, 900 cancer-free), the model produces:

	Predicted cancer	**Predicted not cancer**
Actual cancer	27	73
Actual not cancer	3	897

Compute the model's precision (round to 2 decimal places):

Precision = **3cm**

Solution.

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{27}{27 + 3} = \frac{27}{30} = \boxed{0.90}.$$

(For grader context: recall here is $27/100 = 0.27$*. The model is precise but has low recall — it only catches about a quarter of actual cancers. For a screening tool that asymmetry is the opposite of what you want, which is exactly the kind of trade-off Problem 3 exercises.)*

10. (2 points) **Bonferroni cutoff.** You run $m = 50$ independent hypothesis tests and want to control the family-wise error rate at $\alpha = 0.05$. The Bonferroni per-test cutoff is:

    per-test $\alpha =$ **3cm**

    Solution.

    $\alpha/m = 0.05/50 = \boxed{0.001}$.

11. (1 point) **Expected false positives.** You run $m = 400$ hypothesis tests at $\alpha = 0.01$, and *every null hypothesis is true* (no real effects). How many false positives do you expect, on average?

    $\mathbb{E}[\text{FP}] =$ **3cm**

    Solution.

    $m\alpha = 400 \times 0.01 = \boxed{4}$.

12. (2 points) **Bootstrap CI → reject or fail to reject.** An A/B test of a new checkout flow estimates the difference in conversion rate (new − old). A bootstrap of $B = 10{,}000$ resamples gives a 95% confidence interval of $[-0.4\%,\ +1.2\%]$.

    At $\alpha = 0.05$, do you reject the null hypothesis H_0: difference $= 0$? Circle one and give a one-sentence reason.

    Reject **Fail to reject**

    Solution.

    Fail to reject. The 95% CI contains 0 (since $-0.4\% < 0 < +1.2\%$), so 0 is among the plausible values for the true difference at this confidence level. By the CI–test duality, this is equivalent to a two-sided test failing to reject at $\alpha = 0.05$.

13. (2 points) **Regression coefficient distinguishable from zero?** A regression of a continuous outcome on a single predictor yields:

Predictor	Coefficient	Std error
`predictor_x`	1.2	0.9

    Is this coefficient statistically distinguishable from zero at $\alpha = 0.05$? Circle one and give a one-sentence reason.

    Yes **No**

Solution.

No. The t-statistic is $|t| = 1.2/0.9 \approx 1.33$, which is below the critical value ≈ 1.96 for a two-sided test at $\alpha = 0.05$. Equivalently, an approximate 95% CI is $1.2 \pm 1.96 \times 0.9 \approx [-0.56,\ 2.96]$, which contains 0.

Section 2 — Interpretation & EDA (35 pts, ∼33 min)

14. (12 points) **Regression coefficient table (bike-share demand).** A bike-share operator fits a multivariable linear regression of daily ridership (rides per day) on weather and calendar features from one year of operations. The table reports estimates, standard errors, and p-values; the reference category for season is `Winter`. A side table gives the standard deviations of two continuous predictors in the fitting sample.

Predictor	**Estimate**	**Std err.**	**p-value**
Intercept	1,840	110	< 0.001
`temperature` (per °F)	62.4	3.8	< 0.001
`season_Spring`	320	85	< 0.001
`season_Summer`	540	96	< 0.001
`season_Fall`	210	88	0.018
`humidity` (per %)	−7.8	1.6	< 0.001
`is_holiday`	−380	140	0.007

Predictor	**SD**
`temperature`	17 °F
`humidity`	14 %

(a) (3 points) Interpret the coefficient on `temperature` in one sentence, with units.

(b) (3 points) Interpret the coefficient on `season_Summer` relative to the omitted reference category, in one sentence.

(c) (3 points) Of `temperature` vs `humidity`, which predictor moves predicted ridership more per one-standard-deviation change in the predictor? Show your work.

(d) (3 points) A colleague proposes adding `previous_day_ridership` as a predictor. Will this help predict ridership **for tomorrow, in a forecast made the night before**? Why or why not?

Solution.

(a) Holding all other predictors fixed, each additional 1°F of average daily temperature is associated with about 62.4 more rides per day.

(b) Holding all other predictors fixed, a Summer day has about 540 more rides per day than an equivalent Winter day (the omitted reference category).

(c) 1-SD effects:

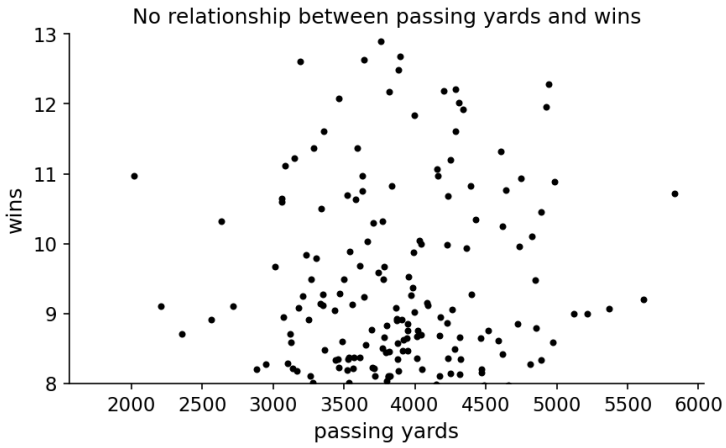
- `temperature`: $62.4 \times 17 \approx 1{,}060$ rides per SD.
- `humidity`: $|-7.8| \times 14 \approx 110$ rides per SD.

`temperature` moves predicted ridership about 10× as much per one-SD change.

(d) Yes. A forecast made the night before tomorrow has full access to yesterday's ridership count — that data is already in the books. `previous_day_ridership` is *available* at prediction time, and it likely carries real predictive signal (autocorrelation of demand).

(For grader: the analogous Airbnb question — "add `number_of_reviews` for a brand-new listing" — is "no" because the feature isn't computable at prediction time. Here the feature is *computable, so the answer flips. The lesson is identical: the question to ask is always "is this feature available when the model needs to predict?")*

15. (12 points) **EDA plot critique.** A junior analyst produces the scatter plot below from a team-season dataset (one point per team-season across many years of NFL play) and writes the caption: "*There is no relationship between passing yards and wins.*"



(a) (6 points) Identify **three specific problems** with the plot itself (not the conclusion). One short sentence per problem.

Problem 1:

Problem 2:

Problem 3:

(b) (3 points) What plot type, transformation, or rendering choice would better answer the underlying question? One-sentence justification.

(c) (3 points) Do you trust the analyst's conclusion that there is "no relationship" between passing yards and wins? Why or why not?

Solution.

(a) Three problems (any three):

- **Truncated y-axis.** Wins range from 0 to 16 in a regular NFL season, but the axis starts at 8, hiding any team-seasons with fewer than 8 wins and exaggerating relative variation among the rest.
- **Missing units on the axes.** "Passing yards" without a window (per game? per season?) and "wins" without a window (per season? regular season vs total?) leaves the magnitudes uninterpretable.
- **Severe overplotting.** Points stack so densely in the middle of the cloud that any local trend is hidden by the visual mass.
- Also acceptable: no chart title beyond the editorial conclusion; the conclusion is presented as a title rather than as a separate annotation; tick labels are sparse and the implied scale is not anchored.

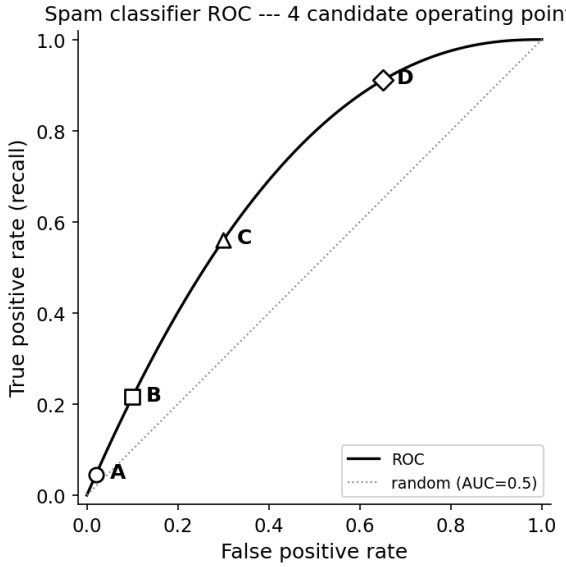
(b) A better plot: a scatter with **transparency** (α-blending) or **hex-bin density** to handle the overplot, a y-axis spanning the full 0–16 win range, units in both labels (e.g. "passing yards per game", "regular-season wins"), and a fitted smoother overlaid. With those fixes the eye can judge whether a trend exists through the cloud.

(c) No, the conclusion is not trustworthy from this plot. The plot's flaws (truncated axis, overplotting) actively hide whatever relationship is or isn't there. The analyst should reach a defensible plot before reaching a defensible conclusion. "No visible relationship in a broken plot" is not evidence of no relationship.

16. (11 points) **Spam classifier.** An email provider fits a classifier on 2022 messages to predict whether each new message is spam. At the default threshold $\hat{p} = 0.5$, the confusion matrix on a held-out 2022 test set of 1,000 messages (300 actual spam, 700 legitimate) is:

	Predicted spam	**Predicted not spam**
Actual spam	170	130
Actual not spam	210	490

The ROC curve below shows the model's full TPR-vs-FPR trade-off. Four candidate operating points A, B, C, D are labeled.



(a) (3 points) Which of A, B, C, D corresponds to the displayed confusion matrix? Circle one.

$A \quad B \quad C \quad D$

(b) (4 points) The email provider's product team says "customers complain loudly when legitimate emails land in their spam folder — we need to **minimize false positives**." Should they move the threshold **up** or **down**? What does that trade against?

(c) (4 points) The model was trained on 2022 spam patterns. By 2025, spammers have changed their tactics in response to public spam-detection write-ups. **Name the phenomenon**

and recommend **one concrete action** the provider should take before continuing to rely on this model.

Solution.

(a) C**.** From the confusion matrix:

- TPR = Recall = $170/(170 + 130) \approx 0.57$
- FPR = $210/(210 + 490) = 0.30$

The operating point is approximately $(0.30,\ 0.57)$ — this is point C on the curve. The other labeled points sit at A near $(0.02, 0.05)$, B near $(0.10, 0.22)$, and D near $(0.65, 0.92)$ along the same curve.

(b) Move the threshold up. A higher threshold predicts "spam" more rarely, so legitimate emails get flagged less often — false-positive rate drops. The trade-off is that the true-positive rate (recall) also drops: more actual spam slips into the inbox. The product call privileges customer trust (no good emails lost) over inbox cleanliness (some spam delivered).

(c) The phenomenon is **concept drift** (the relationship $P(Y \mid X)$ between observable features and "is spam" has changed because spammers actively adapt). *Distribution shift* more broadly, or *covariate shift* if the framing is about message-feature distributions changing, also earn full naming credit.

One concrete action (any one of):

- Retrain the model on recent labeled spam, including 2024–25 examples that reflect current tactics.
- Set up an adversarial-monitoring pipeline: track model performance on a recent, freshly-labeled sample; alert when accuracy degrades.
- Move to an online-learning setup where the classifier updates continuously as users flag spam in their inboxes.
- Recalibrate the decision threshold on recent labeled outcomes — the optimal cut for the new mix of spam vs not-spam may have moved.

Section 3 — Diagnose & supervise (40 pts, ∼35 min)

17. (15 points) **AI code review.** A bike-share operator asks an AI agent: "*Predict, for each day, whether ridership will exceed station capacity.*" The dataset (`bike-ridership.csv`) has one row per day, sorted by date, with weather and calendar features and a binary target `exceeds_capacity` (the operator says exceedance is uncommon, somewhere around 5% of days). The agent returns the following code:

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import KFold, cross_val_score

df = pd.read_csv("bike-ridership.csv").sort_values("date")

features = ["temp_f", "humidity", "is_weekend", "is_holiday"]
X = df[features]
y = df["exceeds_capacity"]

cv = KFold(n_splits=5, shuffle=True, random_state=0)
model = RandomForestClassifier(n_estimators=100, random_state=0)

scores = cross_val_score(model, X, y, cv=cv, scoring="accuracy")
print(f"CV accuracy: {scores.mean():.3f}")
print("Model accuracy is above 0.90 -- looks good, ship it.")
```

(a) (6 points) Identify the **two bugs** in this code. Name each (a short label is fine) and quote the offending line.

Bug 1 name:

Offending line:

Bug 2 name:

Offending line:

(b) (4 points) On a future week with similar weather to the training data, what fraction of the days when ridership actually exceeds capacity does the deployed model correctly flag? Circle one and give a one-sentence reason.

High (≈ 1) **Moderate (≈ 0.4–0.7)** **Near zero**

(c) (3 points) For each bug, write the **one-line fix** (pseudocode is fine).

Fix 1:

Fix 2:

(d) (2 points) Name **one sanity check** that would catch *either* bug before deployment.

Solution.

(a) Two bugs.

1. **Temporal leakage from random k-fold on time-ordered data.** The offending keyword is `shuffle=True` in the `KFold` constructor. The data is daily, sorted by date. Shuffling the folds places adjacent days (Friday and Saturday) in different folds, so each validation day is surrounded by training days from the same week. The CV score reflects same-week prediction, not future prediction.
2. **Wrong metric for an imbalanced target.** The offending argument is `scoring="accuracy"` on the `cross_val_score` call. With $\sim$5% positives, a constant "predict no exceedance" baseline scores about 0.95 accuracy. The reported 0.93–0.95 CV accuracy is indistinguishable from that baseline — accuracy can't tell whether the model has learned anything.

(b) Near zero. The operator's question is "what fraction of actual capacity-exceeding days does the model catch?" — that's recall on the positive class. Under accuracy with a $\sim$5% positive rate, the model is rewarded for predicting "no exceedance" everywhere; the random forest most likely learns to predict the majority class essentially always, so recall on the positive class is near zero. The headline "0.93 accuracy" hides that the model never (or almost never) flags an exceedance day.

(c) Fixes.

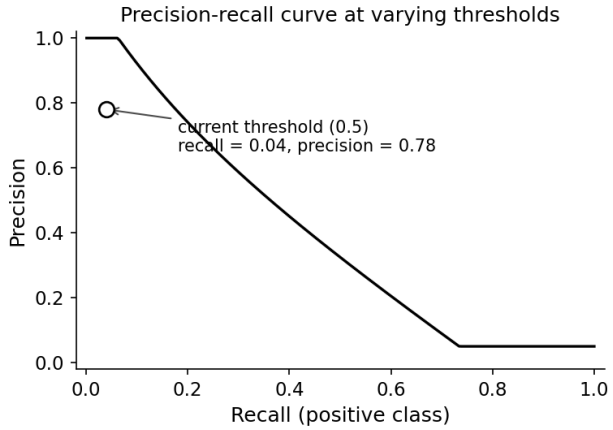
1. Replace random k-fold with a time-aware split: `cv = TimeSeriesSplit(n_splits=5)`.
2. Report a metric that reflects positive-class performance: `scoring="f1"` (or `"recall"`, `"average_precision"`), and print a confusion matrix rather than only the headline number.

(d) Sanity check (any one, full credit):

- Compare to a constant-majority baseline (`DummyClassifier(strategy="most_frequent")`): if the model's accuracy matches the baseline's, accuracy is telling you nothing.
- Inspect the class balance before picking a metric: `y.mean()` of 0.05 should disqualify "accuracy" from the metric list.
- Print the confusion matrix on a held-out window, not just a scalar score.
- Compare CV-reported score to a held-out final window in chronological order; a large gap signals leakage.

18. (12 points) **Diagnose the phenomenon.** For each scenario, name **two plausible causes** of the observed phenomenon. For each cause, write one sentence on **how you'd check it**.

    (a) (6 points) *"I fit a logistic regression classifier. Its accuracy on a held-out set is 0.96, but its recall on the positive class is 0.04."* The precision-recall curve at varying classification thresholds is shown below; the operating point at the default threshold $\hat{p} = 0.5$ is marked.



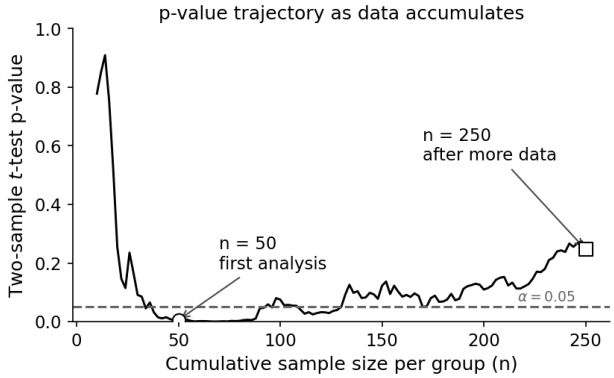
Cause 1:

Check:

Cause 2:

Check:

(b) (6 points) *"We ran a two-sample t-test on a difference in means and got $p = 0.03$ on $n = 50$ observations per group. We collected 200 more observations per group; the p-value is now around 0.25."* The trajectory of the p-value as the sample grows is shown below.



Cause 1:

Check:

Cause 2:

Check:

Solution.

(a) Accuracy 0.96, recall on positives 0.04. Any two of:

- **Severe class imbalance.** Positives are rare; a model that predicts "negative" for everyone scores high accuracy and zero recall. **Check:** compute `y.mean()`; if it's ~ 0.04, accuracy is bounded above by predicting all-negative.
- **Default threshold too high for this problem.** The PR curve shows the model can achieve higher recall by lowering the cutoff at the cost of precision. **Check:** move along the PR curve — recompute precision and recall at threshold 0.3 or 0.2 and see how the trade evolves.
- **Positive class too rare to learn from at this sample size.** Even a sensible model can't find a decision rule when it has, say, 40 positives to learn from. **Check:** count the absolute number of positives in training; if it's small (dozens or fewer), consider class-weighting, oversampling, or collecting more positive examples.

- **Wrong loss / no class weighting.** Cross-entropy with default class weights treats the rare class as equally important per-example, which is the opposite of what the operator needs. **Check:** refit with `class_weight="balanced"` and compare recall.

(b) p-value moves from 0.03 (n=50) to 0.25 (n=250). Any two of:

- **The original $p = 0.03$ was a Type I error.** Under a true null hypothesis, p-values are uniform on $[0, 1]$, so values below 0.05 happen 5% of the time by construction. The first analysis got lucky in the wrong direction; the larger sample is closer to the truth. **Check:** run a permutation test on the full $n = 250$ sample as a non-parametric sanity check; compare its p-value to the t-test result.
- **Winner's curse / regression to the mean.** Effects estimated from significance-filtered samples are biased upward; the small-sample effect was real but its size was overstated. As more data arrives, the estimate regresses toward its true (smaller) value, and p rises. **Check:** report a CI on the effect size and watch it shrink and move toward zero as n grows; compare effect-size estimates between the first and combined samples.
- **Sampling mechanism differs between the two batches.** The new 200 observations came from a different time window, a different recruitment channel, or a different operator. **Check:** compare baseline covariate distributions across the first 50 and the additional 200; look for visible shifts that could dilute the effect.
- **Heterogeneous subgroups.** The original sample over-represented a subgroup where the effect is real; the new sample dilutes it. **Check:** stratify by the most likely subgroup variable (e.g., site, cohort, season) and test within strata.

Grader: any two defensible causes per scenario with sensible checks earn full credit. The same cause restated counts as one. Award partial credit for sensible diagnoses outside this list with sensible reasoning.

19. (12 points) **Unsupervised interpretation & decision (EPL player clustering).** An analyst clusters English Premier League player-season records with k-means at $k = 4$ on five standardized per-90-minute features: goals (G), assists (A), tackles (TKL), passes (PASS), and saves (SV). The cluster centroids (in standardized units, where 0 is league average and +1 is one SD above) and cluster sizes are below.

Cluster	G	A	TKL	PASS	SV	Size
1	+1.8	+0.3	−0.6	+0.2	−0.2	35
2	+0.1	+1.4	−0.2	+1.3	−0.2	80
3	−0.6	−0.5	+1.3	+0.5	−0.2	90
4	−0.4	−0.4	−0.4	−0.6	+2.5	18

(a) (4 points) Give each cluster a one-phrase plain-English name based on its centroid. A few words each is fine.

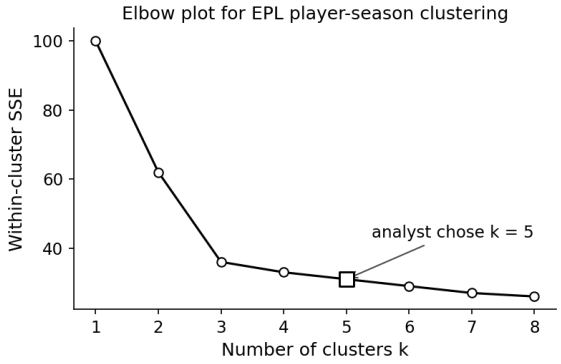
Cluster 1: **8cm**

Cluster 2: **8cm**

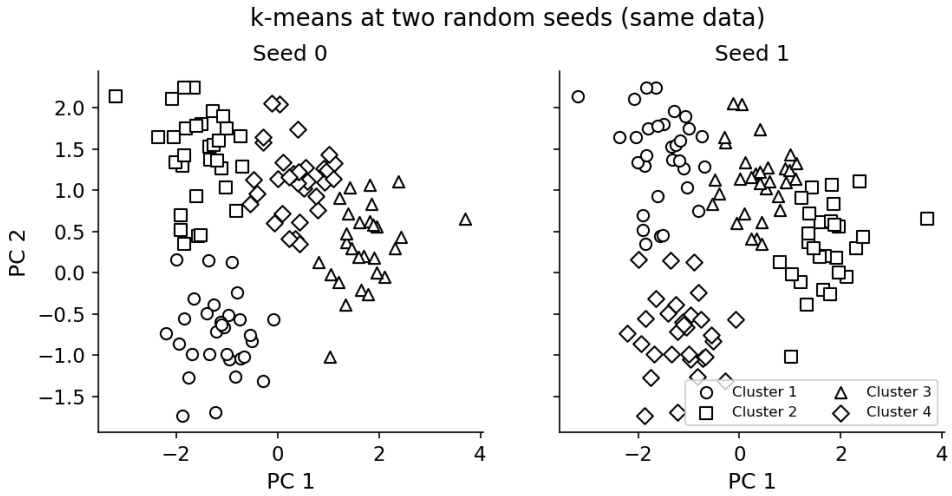
Cluster 3: **8cm**

Cluster 4: **8cm**

(b) (4 points) The analyst chose $k = 5$. The elbow plot below shows within-cluster sum of squares (WSS) as a function of k. Does the plot justify $k = 5$? If not, what k would you choose, and why?



(c) (4 points) Two runs of k-means at different random seeds, side by side, are shown below: many players land in different clusters across the two runs. The team's recruiting department wants to publish an internal shortlist of players in "Cluster 2" as midfielder targets. What does this comparison reveal, and what should the analyst do before publishing?



Solution.

(a) Cluster names (representative; any sensible synthesis of the centroid signal earns credit):

- **Cluster 1** ($n = 35$): strikers / goal-scorers. Far above average on goals, near-zero on other roles, small population. The small n is consistent with the relatively few specialist scorers per season.
- **Cluster 2** ($n = 80$): playmaking midfielders. High assists and passes, modest everything else — the orchestrator profile.
- **Cluster 3** ($n = 90$): defenders. High tackles, elevated passes, low goals and assists, low saves — the defensive contributor profile.
- **Cluster 4** ($n = 18$): goalkeepers. Very high saves; everything else well below average; small population. Saves at +2.5 SD with $n = 18$ is the unambiguous signature.

(b) Does the plot justify $k = 5$? The elbow is at $k = 3$. The marginal reduction in WSS from $k = 3$ to $k = 5$ is small compared to the drop from $k = 2$ to $k = 3$. **The plot does not justify $k = 5$ on its own;** a defensible reading is $k = 3$.

That said, the four roles in part (a) (strikers, midfielders, defenders, goalkeepers) suggest $k = 4$ is also a defensible choice on *interpretability* grounds even if the elbow doesn't insist on it. Full credit for picking $k = 3$ from the elbow alone, or for arguing for $k = 4$ explicitly on non-elbow

grounds (position structure, downstream-use requirements). Picking $k = 5$ requires evidence the elbow doesn't show.

(c) What the seed comparison reveals. k-means is non-deterministic: different random initializations converge to different local optima of the WSS objective. Cluster labels ("Cluster 1," "Cluster 2") are not stable across runs — the bucket called "Cluster 2" in one run may be a different set of players from "Cluster 2" in another run. Publishing a shortlist that singles out "Cluster 2" is unreliable: a re-run would change which players are on it.

What to do before publishing (any one, full credit):

- Re-run k-means with many random seeds (say, 50) and act only on players who land in the same cluster across most runs (stable subset / consensus clustering).
- Fix the random seed *and* report stability diagnostics (e.g., average pairwise Rand index across seeds) so the recruiting team knows the shortlist is one valid partition of many.
- Use a stability-explicit method: hierarchical clustering with bootstrap, or consensus clustering.
- Validate the cluster assignment against an external label (registered position, club role, salary tier) before publishing.

Grader: full credit for any answer that addresses the stability concern with a concrete mitigation. Answers that only describe the non-determinism without recommending an action: 2/4. "Use a different algorithm" without naming or justifying: 1/4.