TMA4268 V2026 Mock Exam 10 — Solution Proposal

Compiled for Anders Bekkevard

Companion to `mock-exam-10.tex` (same directory).

Mock for: May 18, 2026

This document is a worked-solution proposal in the style of the official Stefanie/Sara solutions to the 2024 and 2025 finals. Point values are quoted in the heading of each solved sub-part. Partial-credit hints are inline. Refer back to `mock-exam-10.tex` for the problem statements.

Problem 1 (10 %) — Fill-in-the-blank concepts

Solution *(1 P per blank.)*

(1) **parametric**

(2) **prediction**

(3) **inference**

(4) **irreducible**

(5) **double descent**

(6) **generative**

(7) **LDA**

(8) **bagging**

(9) **boosting**

(10) **nested cross-validation**

Grading: 1 P per correct blank, no partial credit for "close" alternatives. (4) "Bayes-optimal" is a hard zero — Bayes error and irreducible error are not the same object (Bayes error is for classification, irreducible error is the regression noise variance, although the two concepts are kin). (7) "QDA" is the classic trap; QDA is the version without *the shared covariance assumption. (10) Plain "cross-validation" is wrong because the question explicitly asks for a procedure that separates the selection step from the assessment step.*

Problem 2 (28 %) — Multiple choice, T/F, short numeric

a) Bias–variance, noise, and benign overfitting *(3 %)*

Solution *(0.75 P per statement.)*

(i) **True**. With large irreducible noise σ^2 a flexible model wastes capacity fitting noise that no estimator can recover; the bias–variance trade-off tilts toward simpler (higher-bias, lower-variance) models. This is the prof's standard "high-noise regime favours less flexibility" rule.

(ii) **False**. σ^2 is the variance of the noise ε in $y_0 = f(x_0) + \varepsilon$; it is a property of the data-generating process and does not depend on n. The estimator-variance term $\mathrm{Var}(\hat{f}(x_0))$ typically shrinks like $1/n$, but the *irreducible* term does not.

(iii) **False**. The expected squared test error has a third additive term, the irreducible noise σ^2, which is the same for both estimators. With equal bias and lower variance, $\hat{f}_A$ has lower expected squared error *at* x_0, but "strictly lower" is too strong when the difference $\mathrm{Var}(\hat{f}_B) - \mathrm{Var}(\hat{f}_A)$ could be infinitesimal; more importantly the statement reads as if it ignored σ^2. *Accept "True" with a clean argument that the noise term cancels and the two variance terms differ strictly.*

(iv) **True**. This is the prof's headline benign-overfitting story: the implicit regularization of SGD steers an over-parameterized network toward a minimum-norm interpolator among the manifold of zero-training-loss solutions, and that bias is one of the structural ingredients that lets test error keep falling past the classical bias–variance bound.

Grading: (iii) is the deliberately ambiguous one; full credit for any answer accompanied by the correct identification that the σ^2 term is shared and only the variances differ.

b) Cross-validation: bias, variance, failure modes, one-SE rule *(4 %)*

Solution

(i) *(1 %)* The ten fold MSEs sum to 22.00, so

$$\mathrm{CV}_{10} = \frac{22.00}{10} = \boxed{2.20}.$$

Sample standard deviation (divisor $k - 1 = 9$):

$$s = \sqrt{\frac{\sum (v_i - 2.20)^2}{9}} = \sqrt{\frac{0.31}{9}} \approx 0.186, \qquad \widehat{\mathrm{SE}}(\mathrm{CV}_{10}) = \frac{s}{\sqrt{10}} \approx \boxed{0.06}.$$

Grading: 0.5 P for CV-MSE, 0.5 P for SE. Two decimals as requested.

(ii) *(1 %)* The CV minimum is at $\lambda = 0.10$ with $\mathrm{CV}_{10} = 2.15$. The one-SE bound is

$$2.15 + 0.06 = \boxed{2.21}.$$

Values within the band: $\lambda = 0.10$ (CV = 2.15) and $\lambda = 0.20$ (CV = 2.18). The one-SE rule picks the *simplest* (largest-λ) model among those, hence

$$\boxed{\hat{\lambda}_{1\mathrm{SE}} = 0.20}.$$

Grading: 0.5 P for the bound, 0.5 P for the right λ. Picking $\lambda = 0.10$ (the CV-min) is a hard miss — the whole point of the rule is that we step away from the minimum toward a simpler model.

(iii) *(1 %)* (A) **True**. LOOCV trains on $n - 1$ points (nearly the full sample $\Rightarrow$ low bias as an estimate of the test error of a size-n trained model), but the n training sets overlap almost completely, so the n fold residuals are highly correlated and their average has higher variance than the 5- or 10-fold average. (B) **False**. With temporally autocorrelated data, random folds break the time order and let future information leak into the training of a model that is then evaluated on its own past — the standard remedy is time-series CV (rolling-origin / blocked folds), not a bigger k. *Grading: 0.5 P each.*

(iv) *(1 %)* The per-fold MSEs are *not* independent (the training sets used to produce them overlap heavily), so the formula $s/\sqrt{k}$, which assumes independent observations, underestimates the true uncertainty of CV_K. *Grading: 1 P. Accept "the folds share most of their training data, so the per-fold MSEs are positively correlated and $s/\sqrt{k}$ is a stand-in not a real SE."*

c) Bootstrap: pairs vs. residuals, percentile CI *(3 %)*

Solution

(i) *(1 %)* **Paired (case-resampling) bootstrap**. Reason: the residual bootstrap assumes the model's functional form is correct and that residuals are i.i.d. and exchangeable across x; with random X and a possibly mis-specified model the paired scheme makes the weaker assumption that the rows themselves are i.i.d., so it remains valid under model mis-specification and under heteroscedastic noise. *Grading: 0.5 P for the right scheme, 0.5 P for the reason.*

(ii) *(1 %)* The percentile 95% CI is

$$\left[\hat{\theta}^*_{(0.025)},\ \hat{\theta}^*_{(0.975)}\right] \ = \ \left[\hat{\theta}^*_{(25)},\ \hat{\theta}^*_{(975)}\right],$$

i.e. the 2.5% and 97.5% empirical quantiles of the $B = 1000$ bootstrap replicates (the 25th and 975th order statistics). *Grading: 1 P. "Mean $\pm 1.96 \cdot \mathrm{SE}^*$" is a* different *CI (normal-approximation), worth* 0 *here.*

(iii) *(1 %)* (A) **True**. By definition $\widehat{\mathrm{SE}}_B(\hat{\theta}) = s(\hat{\theta}^*_1, \ldots, \hat{\theta}^*_B)$. (B) **True**. The course recommendation was $B \in [1000, 10000]$ for SE estimates, with more for quantile-based CIs. (C) **False**. The bootstrap estimates the *distribution* of $\hat{\theta} - \theta$ (or of $\hat{\theta}^* - \hat{\theta}$); it does not correct bias automatically. Bias-corrected bootstrap CIs exist (BCa) but the plain percentile CI does not adjust for it. *Grading: 0.33 P each.*

d) Backpropagation, mini-batch SGD, learning rate *(3 %)*

Solution *(0.75 P per statement.)*

(i) **True**. Backprop is the chain-rule recipe for computing $\partial L/\partial\theta$ layer by layer; the optimizer (SGD, Adam, ...) is a separate step that consumes that gradient.

(ii) **True**. $\widehat{\nabla L}_m = \frac{1}{m}\sum_{i\in\mathcal{B}} \nabla L_i$ is an unbiased estimator of the full-data gradient with variance $\propto 1/m$; reducing m cheapens each step at the cost of a noisier gradient (the noise itself is one source of the implicit regularization the prof flagged).

(iii) **False**. $\eta = 2$ is typically two orders of magnitude too large for tabular feed-forward training; standard ranges are 10^{-4} to 10^{-2}. Too-large η causes divergence, not faster convergence.

(iv) **True**. $\partial L/\partial \hat{f} = \hat{f} - y$ when $L = \frac{1}{2}(\hat{f} - y)^2$, and this residual is exactly the quantity that seeds the backward pass.

e) Collinearity, eigenvalues, dummy coding *(3 %)*

Solution

(i) *(1 %)* **True**. Perfect collinearity means a non-trivial null vector of $\mathbf{X}$ (here $\mathbf{X}_2 - 3\mathbf{X}_1 + 2\cdot\mathbf{1} = 0$), so $\mathbf{X}^\top\mathbf{X}$ is singular and the OLS normal equations have infinitely many solutions; only the predicted value $\hat{\beta}_1 X_1 + \hat{\beta}_2 X_2$ (and hence $\hat{\mathbf{y}}$) is uniquely identified.

(ii) *(1 %)* **False**. The collinearity diagnosis is correct, but the *standard* remedy is to drop one dummy (reference encoding) or equivalently to drop the intercept, restoring full column rank of $\mathbf{X}$. Adding a ridge penalty does numerically stabilise $\mathbf{X}^\top\mathbf{X}+\lambda I$, but the individual ridge-coefficients of the K dummies are not identified in the usual structural sense — they depend on λ and on the (arbitrary) choice of intercept column, so this is not the canonical remedy.

(iii) *(1 %)* **True**. The variance of the joint contribution $\mathrm{Var}(\hat\beta_1 X_1 + \hat\beta_2 X_2)$ includes the (large, *negative*) covariance term that cancels much of the per-coefficient variance inflation; this is why prediction can remain stable under near-collinearity even when individual coefficients are not, and why the joint test on $\{X_1, X_2\}$ can be highly significant while the marginal t-tests are not.

Grading: 1 P per statement. (iii) is the subtle one and is the deep reason "p-values of individually collinear predictors are misleading" — credit for any answer that names the covariance cancellation.

f) Boosting flavors: AdaBoost, gradient boosting, XGBoost *(3 %)*

Solution

(i) *(1 %)* **True**. Each tree contributes $\nu \cdot \hat T_m(x)$, so halving ν approximately doubles the M needed to reach the same cumulative ensemble; (M, ν) are jointly tuned because they trade off against each other.

(ii) *(1 %)* **False**. The prof's canonical range is $d \in \{1, \dots, 8\}$, with stumps ($d = 1$) as a routine default; deep trees ($d \in \{10, \dots, 50\}$) defeat the "many weak learners" principle by making each base learner already low-bias and high-variance, which boosting cannot easily unwind. Using stumps is the textbook recommendation, not a defeat of the bias–variance logic.

(iii) *(1 %)* **True**. All three claims are lecture-verbatim XGBoost differences: (a) leaf-weight penalty $\gamma|T| + \frac{1}{2}\lambda\sum_j w_j^2$ in the split objective, (b) Newton-style split scoring using both first and second derivatives of the loss, (c) row + column subsampling for variance reduction and speed.

g) Logistic regression with categorical × continuous interaction *(3 %)*

Solution

(i) *(1 %)* The odds factor for a unit increase in x depends on the group:

- Group A (reference): $\exp(0.20) \approx \boxed{1.22}$,
- Group B: $\exp(0.20 - 0.05) = \exp(0.15) \approx \boxed{1.16}$,
- Group C: $\exp(0.20 - 0.15) = \exp(0.05) \approx \boxed{1.05}$.

Grading: 0.33 P each. Common slip: using $\exp(\hat\beta_x) = e^{0.20}$ *for all three groups (forgetting the interaction).*

(ii) *(1 %)* For group C, $x = 4$:

$$\hat\eta = -1.50 + 0.20 \cdot 4 + 1.20 + (-0.15) \cdot 4 = -1.50 + 0.80 + 1.20 - 0.60 = -0.10.$$

$$\hat p = \sigma(-0.10) = \frac{1}{1 + e^{0.10}} \approx \boxed{0.475}.$$

Grading: 0.5 P for $\hat\eta$*, 0.5 P for* $\hat p$*.*

(iii) *(1 %)* **False**. $\hat{\beta}_j$ in logistic regression is the change in *log-odds* per unit increase in x_j — equivalently, $\exp(\hat{\beta}_j)$ is the *odds-ratio*. The change in probability is nonlinear in x_j and depends on the current value of $\hat{p}$ (it is largest near $\hat{p} = 0.5$ and shrinks toward 0 at either extreme).

h) Neural-network regularization *(3 %)*

Solution

(i) *(1 %)* With $C = 5$ and $\varepsilon = 0.10$ the off-target mass per class is $\varepsilon/(C-1) = 0.10/4 = 0.025$; the on-target mass is $1 - \varepsilon = 0.90$. The original one-hot $(0, 0, 1, 0, 0)$ becomes

$$\boxed{(0.025,\ 0.025,\ 0.90,\ 0.025,\ 0.025)}.$$

Grading: 1 P for the full vector, 0.5 P if only the on-target entry is right.

(ii) *(1 %)* (A) **True**. Dropout masks units randomly at train time and is switched off at test time; weights are typically rescaled (test-time scaling) or activations are scaled at train time (inverted dropout) so that the expected unit input matches between regimes. (B) **False**. Early stopping monitors a held-out *validation* set, not the training loss. Stopping at the training-loss minimum essentially never stops (training loss usually decreases monotonically with more epochs). *Grading: 0.5 P each.*

(iii) *(1 %)* **False**. The course-recommended dropout range was 0.2–0.5; 0.5 on hidden layers (and even higher on the input layer) is a standard, well-tested setting and the prof did not flag it as too aggressive. *Grading: 1 P. Accept "False" with any reasonable counter-claim citing the course range.*

i) PCA: variance, scree, score *(3 %)*

Solution

(i) *(1 %)* For *standardized* variables each has unit variance, so the total variance equals $p = 5$ (and indeed $\sum_j \lambda_j = 2.10 + 1.30 + 0.80 + 0.50 + 0.30 = 5.00$, as expected).

$$\mathrm{PVE}_{1:3} = \frac{2.10 + 1.30 + 0.80}{5.00} = \frac{4.20}{5.00} = \boxed{0.84}.$$

Grading: 0.5 P for the total, 0.5 P for the proportion.

(ii) *(1 %)* The PC1 score is the inner product $\phi_1^\top x^*$:

$$z_1^* = 0.55(1.0) + (-0.40)(2.0) + 0.45(-1.0) + 0.50(0.5) + (-0.30)(-0.5)$$

$$= 0.55 - 0.80 - 0.45 + 0.25 + 0.15 = \boxed{-0.30}.$$

Grading: 1 P. Sign-flip error (giving +0.30) gets 0.5 P if the arithmetic is otherwise clean.

(iii) *(1 %)* (A) **True**. Check: $0.55^2 + 0.40^2 + 0.45^2 + 0.50^2 + 0.30^2 = 0.3025 + 0.16 + 0.2025 + 0.25 + 0.09 = 1.005 \approx 1$ (round-off). (B) **False**. PCA is *not* invariant under rescaling; without standardization the variables with the largest raw variance dominate the first few PCs. This is exactly why we standardize before PCA. (C) **True**. The jth eigenvalue of the sample covariance / correlation matrix is the empirical variance of the jth score $z_j = \phi_j^\top x$. *Grading: 0.33 P each.*

Problem 3 (16 %) — Theory, math, pseudocode

a) LDA decision boundary and the LDA → QDA bridge *(8 %)*

Solution

(i) *(3 %)* The denominator of Bayes' rule does not depend on k, so $\arg\max_k \Pr(Y = k \mid X = x) = \arg\max_k \pi_k f_k(x) = \arg\max_k \log[\pi_k f_k(x)]$. Take the logarithm:

$$\log[\pi_k f_k(x)] \;=\; \log \pi_k \;-\; \tfrac{p}{2}\log(2\pi) \;-\; \tfrac{1}{2}\log|\boldsymbol{\Sigma}| \;-\; \tfrac{1}{2}(x-\boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}^{-1}(x-\boldsymbol{\mu}_k).$$

Expand the quadratic:

$$-\tfrac{1}{2}(x-\boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}^{-1}(x-\boldsymbol{\mu}_k) = -\tfrac{1}{2}x^\top \boldsymbol{\Sigma}^{-1}x + x^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k - \tfrac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k.$$

The terms $-\frac{p}{2}\log(2\pi)$, $-\frac{1}{2}\log|\boldsymbol{\Sigma}|$, and $-\frac{1}{2}x^\top\boldsymbol{\Sigma}^{-1}x$ do *not* depend on the class k (the first two are constants; the third depends on x alone, which is fixed when we do the $\arg\max_k$). Dropping them preserves $\arg\max_k$, leaving exactly

$$\delta_k(x) \;=\; x^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \;-\; \tfrac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \;+\; \log \pi_k. \quad \square$$

Grading: 3 P total. 1 P for "log of $\pi_k f_k(x)$, denominator dropped," 1 P for the quadratic expansion, 1 P for identifying and dropping the k-independent terms.

(ii) *(2 %)* Setting $\delta_A(x) = \delta_B(x)$:

$$x^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_A - \tfrac{1}{2}\boldsymbol{\mu}_A^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_A + \log \pi_A = x^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_B - \tfrac{1}{2}\boldsymbol{\mu}_B^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_B + \log \pi_B.$$

Rearrange:

$$x^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_A - \boldsymbol{\mu}_B) \;=\; \tfrac{1}{2}\big(\boldsymbol{\mu}_A^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_A - \boldsymbol{\mu}_B^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_B\big) \;-\; \log\frac{\pi_A}{\pi_B}.$$

This is exactly the claimed equation (the left side is linear in x, the right side is a scalar constant; hence the boundary is an affine hyperplane). *The boundary is linear in x because the quadratic $x^\top\boldsymbol{\Sigma}^{-1}x$ cancelled when we subtracted $\delta_A - \delta_B$ — this cancellation is possible precisely because both classes share the same $\boldsymbol{\Sigma}$. Grading: 1 P for the algebra, 1 P for the one-sentence reason (shared $\boldsymbol{\Sigma} \Rightarrow$ quadratic term cancels).*

(iii) *(2 %)* With $\boldsymbol{\mu}_A = (1,0)^\top$, $\boldsymbol{\mu}_B = (0,3)^\top$, $\boldsymbol{\Sigma} = \mathbf{I}_2$, and $\pi_A = \pi_B$ (so $\log(\pi_A/\pi_B) = 0$):

$$\begin{aligned} \boldsymbol{\mu}_A - \boldsymbol{\mu}_B &= (1,-3)^\top, \\ \boldsymbol{\mu}_A^\top\boldsymbol{\mu}_A &= 1^2 + 0^2 = 1, \\ \boldsymbol{\mu}_B^\top\boldsymbol{\mu}_B &= 0^2 + 3^2 = 9. \end{aligned}$$

Plug in:

$$(1,-3)\,x = \tfrac{1}{2}(1-9) - 0 = -4 \quad\Longleftrightarrow\quad \boxed{x_1 - 3x_2 = -4}.$$

Grading: 1 P for the substitutions, 1 P for the final equation. Sign-flip in $\boldsymbol{\mu}_A - \boldsymbol{\mu}_B$ is a common slip; deduct 0.5 P.

(iv) *(1 %)* For QDA, $\boldsymbol{\Sigma}_k$ depends on k, so the quadratic term $-\frac{1}{2}x^\top\boldsymbol{\Sigma}_k^{-1}x$ in $\delta_k(x)$ no longer cancels between classes when one subtracts $\delta_A - \delta_B$. The resulting boundary $\delta_A(x) = \delta_B(x)$ therefore contains a non-zero $x^\top(\boldsymbol{\Sigma}_A^{-1} - \boldsymbol{\Sigma}_B^{-1})x$ term and is quadratic in x. *Grading: 1 P. Accept any answer that names the $x^\top\boldsymbol{\Sigma}^{-1}x$ term as the surviving culprit.*

b) Pseudocode: k-fold CV with the one-SE rule *(4 %)*

Solution

(i) *(3 %)*

```
Input: dataset (X, y) of size n; grid Lambda = {lambda_1, ..., lambda_T}
       ordered so larger lambda = simpler model; number of folds K = 10.

1. Randomly permute the indices {1, ..., n}; partition into K disjoint
   folds F_1, ..., F_K of (nearly) equal size.

2. For each lambda in Lambda:
     For each fold k = 1, ..., K:
        train_k = (X, y) with rows in F_k removed
        Fit model M_{lambda, k} on train_k
        MSE_{lambda, k} = mean squared error of M_{lambda, k} on F_k
     CV(lambda)  = (1/K) * sum_k MSE_{lambda, k}
     SE(lambda)  = sd(MSE_{lambda, 1}, ..., MSE_{lambda, K}) / sqrt(K)

3. lambda_min  = argmin_lambda CV(lambda)
   bound       = CV(lambda_min) + SE(lambda_min)
   lambda_1SE  = max { lambda in Lambda : CV(lambda) <= bound }
                    (largest = simplest model whose CV is within the SE band)

4. Refit model on the FULL dataset (X, y) at lambda = lambda_1SE.

Return: the refitted model and lambda_1SE.
```

Grading: 3 P. 0.5 P for the random partition, 1 P for the nested "for lambda / for fold" structure, 0.5 P for the SE formula, 0.5 P for the one-SE choice as $\max\{\lambda : \mathrm{CV}(\lambda) \leq \textit{bound}\}$ *(the* largest *qualifier is essential), 0.5 P for the final refit on the full data.*

(ii) *(1 %)* The CV curve is itself estimated with noise, so within an SE band of the minimum the differences are statistically indistinguishable; the one-SE rule breaks the tie in favour of the *simpler* model, which (i) generalizes more reliably out of sample and (ii) is more interpretable, at essentially no test-error cost. *Grading: 1 P. Accept "protect against noise in the CV estimate by picking the simplest model in the noise band."*

c) Backpropagation: sigmoid + binary cross-entropy *(4 %)*

Solution

(i) *(2 %)* Step 1: $\partial L/\partial \hat{p}$. Differentiate $L = -[y \log \hat{p} + (1-y)\log(1-\hat{p})]$:

$$\frac{\partial L}{\partial \hat{p}} = -\frac{y}{\hat{p}} + \frac{1-y}{1-\hat{p}} = \frac{-y(1-\hat{p}) + (1-y)\hat{p}}{\hat{p}(1-\hat{p})} = \frac{\hat{p}-y}{\hat{p}(1-\hat{p})}.$$

Step 2: $\partial \hat{p}/\partial z_2$. Since $\hat{p} = \sigma(z_2)$,

$$\frac{\partial \hat{p}}{\partial z_2} = \sigma(z_2)(1-\sigma(z_2)) = \hat{p}(1-\hat{p}).$$

Chain together:

$$\frac{\partial L}{\partial z_2} = \frac{\hat{p}-y}{\hat{p}(1-\hat{p})} \cdot \hat{p}(1-\hat{p}) = \boxed{\hat{p}-y}. \quad \square$$

The denominator from the cross-entropy derivative cancels exactly with the sigmoid derivative — this is why sigmoid + cross-entropy is the standard pairing (no vanishing-gradient problem at saturated outputs, unlike sigmoid + squared error). *Grading: 2 P. 0.5 P for $\partial L/\partial \hat{p}$, 0.5 P for $\partial \hat{p}/\partial z_2$, 1 P for showing the cancellation.*

(ii) *(1 %)* $z_2 = w_2 h + b_2$, so $\partial z_2/\partial w_2 = h$. Chain:

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z_2} \cdot \frac{\partial z_2}{\partial w_2} = \boxed{(\hat{p} - y) \cdot h}.$$

Grading: 1 P.

(iii) *(1 %)* Continue the chain through the hidden layer:

$$\frac{\partial L}{\partial w_1} = \underbrace{(\hat{p} - y)}_{\partial L/\partial z_2} \cdot \underbrace{w_2}_{\partial z_2/\partial h} \cdot \underbrace{\mathbb{1}[z_1 > 0]}_{\partial h/\partial z_1} \cdot \underbrace{x}_{\partial z_1/\partial w_1} = \boxed{(\hat{p} - y)\, w_2\, \mathbb{1}[z_1 > 0]\, x}.$$

Grading: 1 P. Common slip: forgetting the ReLU indicator $\mathbb{1}[z_1 > 0]$ (deduct 0.5 P).

Problem 4 (20 %) — Data analysis: diabetes progression

a) OLS with quadratic, interaction, collinear pair *(7 %)*

Solution

(i) *(1 %)* The model has 10 estimated coefficients: intercept, `age`, `sex`, `bmi`, $I(\texttt{bmi}^2)$, `bp`, `s1`, `s2`, `s5`, and `bmi:sex`. Residual d.f. $= n_{\text{train}} - 10 = 350 - 10 = \boxed{340}$, consistent with the printed output. *Grading: 1 P (0.5 P for the count, 0.5 P for the d.f. check).*

(ii) *(2 %)* **Near-multicollinearity** between `s1` (total cholesterol) and `s2` (LDL cholesterol). The two pieces of evidence are: (a) the problem statement gives $\text{cor}(\texttt{s1}, \texttt{s2}) \approx 0.97$ on the training set, and (b) the SEs of $\hat{\beta}_{\texttt{s1}}$ and $\hat{\beta}_{\texttt{s2}}$ are 18.0 and 17.5, roughly 4–6× larger than those of the other (mutually weakly correlated) predictors. With $\mathbf{X}^\top\mathbf{X}$ near-singular along the (`s1, s2`) direction, $\text{Var}(\hat{\beta}_j) = \sigma^2[(\mathbf{X}^\top\mathbf{X})^{-1}]_{jj}$ explodes for those two coordinates while predictions and the joint contribution remain well-identified. *Grading: 2 P. 1 P for naming collinearity, 1 P for citing both the correlation and the inflated SEs.*

(iii) *(1 %)* Each p-value tests $H_0 : \beta_j = 0$ *conditional on all other predictors, including the other near-collinear partner*, so the test is asking "can `s1` be set to zero *given* that `s2` is already in the model?" (and vice versa) — both answer yes individually, even though dropping *both* typically worsens fit substantially. The correct procedure is an F-test on the joint contribution of $\{\texttt{s1}, \texttt{s2}\}$, or to drop just one (or combine, e.g. via PCA / a sum). *Grading: 1 P. Accept any answer that names the conditional nature of the t-tests under collinearity.*

(iv) *(2 %)* For each patient, all other (standardized) predictors are at zero, so only the intercept, `sex`, `bmi`, $I(\texttt{bmi}^2)$, and `bmi:sex` terms contribute.

Patient A (`sex` $= 0$, `bmi` $= 1$):

$$\widehat{\texttt{prog}}_A = 152.5 + (-10.2)(0) + 24.0(1) + 3.8(1)^2 + (-9.0)(1)(0) = 152.5 + 24.0 + 3.8 = \boxed{180.3}.$$

Patient B (`sex` $= 1$, `bmi` $= 1$):

$$\widehat{\texttt{prog}}_B = 152.5 + (-10.2)(1) + 24.0(1) + 3.8(1)^2 + (-9.0)(1)(1)$$

$$= 152.5 - 10.2 + 24.0 + 3.8 - 9.0 = \boxed{161.1}.$$

The interaction $\hat{\beta}_{\texttt{bmi:sex}} = -9.0$ implies the slope of `prog` in `bmi` is 9.0 units smaller for males than for females. *Grading: 1 P each. Forgetting the `bmi:sex` term for patient B is a hard miss (deduct 0.5 P).*

(v) *(1 %) Convention:* standardized `bmi` has mean 0 and SD 1, so the values +1 and +2 correspond to one and two SDs above the training mean of raw BMI. For a female (`sex` = 0), going from `bmi` = 1 to `bmi` = 2:

$$\begin{aligned}
\Delta_{\text{linear}} &= 24.0 \cdot (2 - 1) = 24.0, \\
\Delta_{\text{quadratic}} &= 3.8 \cdot (2^2 - 1^2) = 3.8 \cdot 3 = 11.4, \\
\Delta_{\text{interaction}} &= -9.0 \cdot 0 \cdot (2 - 1) = 0, \\
\Delta_{\text{total}} &= 24.0 + 11.4 = \boxed{35.4}.
\end{aligned}$$

Grading: 1 P. The trap is to forget the quadratic contribution and answer 24.0*; deduct 0.5 P.*

b) Lasso with 10-fold CV *(4 %)*

Solution

(i) *(1 %)* With $p = 9$ predictors and a standardized design (no intercept column in $\mathbf{X}$), the lasso objective is

$$\min_{\beta_0, \boldsymbol{\beta}} \; \tfrac{1}{2} \big\| \mathbf{y} - \beta_0 \mathbf{1}_n - \mathbf{X}\boldsymbol{\beta} \big\|_2^2 \; + \; \lambda \sum_{j=1}^{p} |\beta_j|,$$

where the intercept β_0 is *not* penalized (and is typically eliminated by centring the response) and the columns of $\mathbf{X}$ are standardized to make the L^1 penalty scale-invariant. *Grading: 1 P. Penalizing the intercept is the canonical wrong answer; deduct fully.*

(ii) *(1 %)* The dummy-coding scheme determines which level becomes the "zero" reference, and the lasso penalty is applied to the contrast coefficients, not to a symmetric group penalty (unless one uses group lasso); so different reference levels give different effective penalties on the same categorical predictor, and which dummy gets shrunk to zero is a function of the coding choice rather than of the data. The practical consequence: a dummy shrunk to zero *means the corresponding level cannot be distinguished from the reference*, not that the predictor is irrelevant. *Grading: 1 P.*

(iii) *(1 %)* Lasso reduces the *variance* of the OLS predictions at the cost of introducing some bias. The near-collinear pair (`s1, s2`) inflates the variance of the OLS predictions through their coefficient variances; lasso resolves the ambiguity by shrinking one of them (here: 7 nonzeros out of 9), which substantially lowers prediction variance, and that variance reduction outweighs the bias introduced by shrinkage. *Grading: 1 P. Accept "variance reduction at the cost of some bias, exploiting the collinear pair where OLS variance was inflated."*

(iv) *(1 %)* **One-SE rule:** from the CV curve at $\hat{\lambda}_{\min}$, compute $\widehat{\text{SE}}(\text{CV}_K)$ and pick the *simplest* model (largest λ, fewest nonzero coefficients) whose CV is within one SE of the minimum. The practitioner's preference is reasonable because (i) the differences within the SE band are statistically indistinguishable, and (ii) a model with 5 predictors instead of 7 is more stable across resamples and easier to interpret — both core arguments for the rule. *Grading: 1 P (0.5 for the rule, 0.5 for a reasonable justification).*

c) Gradient boosting — learning-rate / depth tradeoff *(5 %)*

Solution

(i) *(2 %)* Squared-error gradient boosting:

```
Initialize  f^{(0)}(x) = mean(y)                [the constant minimizing SSE]
For m = 1, ..., M:
    r_i = y_i - f^{(m-1)}(x_i)                 [negative gradient = residual]
    Fit a regression tree h_tilde_m to (X, r) using subroutine T(.)
    Update  f^{(m)}(x) = f^{(m-1)}(x) + nu * h_tilde_m(x)
Return f^{(M)}
```

Grading: 2 P. 0.5 P initialisation, 0.5 P fitting to residuals (= negative gradient under SSE), 0.5 P the ν-scaled update, 0.5 P the final $f^{(M)}$.

(ii) *(1 %)* Halving ν approximately *doubles* the required M^* (here: $\nu = 0.10 \to M^* = 700$, $\nu = 0.05 \to M^* = 1{,}400$, $\nu = 0.01 \to M^* = 6{,}500$; the last factor is ≈ 9 rather than 5 because smaller ν also reduces effective bias more gradually). Smaller ν slows the rate at which the ensemble fits, lowering variance per added tree at the cost of needing more trees to reach the same bias. *Grading: 1 P. Accept "$M^* \cdot \nu \approx$ constant."*

(iii) *(1 %)* Deep trees are themselves low-bias / high-variance learners; combined with *small* ν the ensemble takes many slow steps each adding lots of variance, and combined with *large* ν each big step over-fits the local residual — in either case boosting can't undo the variance contributed by a non-weak base learner, whereas shallow trees keep each step weak so that ν alone controls the bias–variance budget. *Grading: 1 P. Any answer naming the "deep tree = strong base learner = boosting can't compensate" argument.*

(iv) *(1 %)* **Unsound** for gradient boosting — M is a genuine tuning parameter and too-large M *does* overfit (test error eventually rises). Random forests behave differently: more trees only reduces variance, the ensemble does not overfit in B, so "set $B = 500$ to be safe" is fine for RF but not for boosting. *Grading: 1 P.*

d) GAM and random forest *(4 %)*

Solution

(i) *(1 %)* A cubic regression spline with $K = 2$ interior knots has $K+d+1 = 2+3+1 = 6$ basis functions; removing the global intercept leaves $\boxed{5}$ degrees of freedom for the bs(`bp`, …) term. *Grading: 1 P.*

(ii) *(1 %)* Sum up:

- intercept: 1;
- $s($`age`$, \mathrm{df} = 4)$: 4;
- `sex` (single dummy): 1;
- $s($`bmi`$, \mathrm{df} = 5)$: 5;
- bs(`bp`, …) (from (i)): 5;
- `s1`, `s2`, `s5` (linear): $1 + 1 + 1 = 3$.

Total: $1 + 4 + 1 + 5 + 5 + 3 = \boxed{19}$ degrees of freedom. *Grading: 1 P. Off-by-one slips on the spline conventions are common; accept any answer between* 18 *and* 20 *with correct accounting per term.*

(iii) *(1 %)* For regression random forests, the standard default is `mtry` $= p/3 = 9/3 = \boxed{3}$. One picks `mtry` $< p$ to *decorrelate* the bagged trees: bagging only reduces variance when the trees are weakly correlated, and forcing each split to consider only a random subset of features prevents the same one or two strong predictors (here: `bmi`, `s5`, `bp`) from dominating every tree. *Grading: 1 P.*

(iv) *(1 %)* The random-forest variable-importance plot reports a *magnitude* of effect (e.g. mean decrease in OOB MSE) but *not a sign / direction*: it tells you which predictors matter but not whether higher `bmi` is associated with higher or lower disease progression. An OLS coefficient does both (and supplies an SE for inference). *Grading: 1 P. Accept "no sign" or "no functional form" or "no confidence interval / inference."*

Problem 5 (26 %) — Data analysis: South African heart disease

a) Logistic regression with a continuous-by-binary interaction *(8 %)*

Solution

(i) *(2 %) Encoding:* `famhist` $= 1$ means family history, 0 means no family history (as given). Each additional mmol/L of `ldl` changes the log-odds by $\hat{\beta}_{\texttt{ldl}} + \hat{\beta}_{\texttt{ldl:famhist}} \cdot \texttt{famhist}$:

- `famhist` $= 0$: factor $= \exp(0.18) \approx \boxed{1.197}$.
- `famhist` $= 1$: factor $= \exp(0.18 + 0.15) = \exp(0.33) \approx \boxed{1.391}$.

Grading: 1 P each. Forgetting the interaction in the second case (answering 1.197 *twice) is a hard miss.*

(ii) *(1 %)* The interaction $\hat{\beta}_{\texttt{ldl:famhist}} = 0.15$ is non-zero, so the slope of the log-odds in `ldl` *depends on* `famhist` and the "effect of LDL" is different in the two groups (specifically: LDL is a stronger risk factor among patients with family history). $\hat{\beta}_{\texttt{ldl}} = 0.18$ alone reports only the slope in the reference group (`famhist` $= 0$) and is therefore not a useful summary of "the effect of LDL" in this fitted model. *Grading: 1 P.*

(iii) *(3 %)* Compute $\hat{\eta}$ term by term:

$$\begin{aligned}
\hat{\eta} = & -5.50 \quad \text{(intercept)} \\
& + 0.045 \cdot 55 = +2.475 \quad (\texttt{age}) \\
& + 0.18 \cdot 5 = +0.900 \quad (\texttt{ldl}) \\
& + 0.85 \cdot 1 = +0.850 \quad (\texttt{famhist}) \\
& + 0.080 \cdot 4 = +0.320 \quad (\texttt{tobacco}) \\
& + 0.035 \cdot 55 = +1.925 \quad (\texttt{typea}) \\
& + 0.020 \cdot 25 = +0.500 \quad (\texttt{adiposity}) \\
& - 0.050 \cdot 27 = -1.350 \quad (\texttt{obesity}) \\
& + 0.15 \cdot 5 \cdot 1 = +0.750 \quad (\texttt{ldl:famhist})
\end{aligned}$$

Cumulative sum: $-5.50 + 2.475 - 3.025$; $+0.900 \to -2.125$; $+0.850 \to -1.275$; $+0.320 \to -0.955$; $+1.925 \to +0.970$; $+0.500 \to +1.470$; $-1.350 \to +0.120$; $+0.750 \to$

$$\boxed{\hat{\eta} = 0.870}.$$

$$\hat{p} \;=\; \sigma(0.870) \;=\; \frac{1}{1 + e^{-0.870}} \;\approx\; \boxed{0.705}.$$

Grading: 3 P. 1.5 P for $\hat{\eta}$ (deduct 0.5 P per arithmetic slip up to a floor of 1 P if the structure is right), 1 P for the sigmoid evaluation, 0.5 P for correctly including the interaction term.

(iv) *(1 %)* $\hat{p} = 0.705 > 0.5$, so the patient *would* be flagged as CHD at the default threshold. However 0.5 is rarely the right operating point for medical screening: the base rate is $\approx 35\%$, false negatives are clinically costly, and the threshold should be lowered (perhaps to 0.3) to raise sensitivity at the cost of some false positives. *Grading: 1 P. Accept "yes, flagged, but* 0.5 *is too high for screening with asymmetric costs."*

(v) *(1 %)* The model is an observational regression and reports *conditional associations, not causal effects*; the negative sign on `obesity` could be a product of confounding (e.g. obesity is correlated with `adiposity`, `bmi`, and `bp`, and "controlling for" those other predictors makes the residual contribution of obesity to CHD risk hard to interpret). The prof's verbatim line: "these are fancy correlations, not causal claims." *Grading: 1 P. Accept any answer naming confounding / observational design / "correlation is not causation."*

b) AdaBoost: deriving the classifier weight α_m *(6 %)*

Solution

(i) *(3 %)* Split the sum by correct vs. incorrect prediction of G:

$$\begin{aligned} Q(\alpha, G) &= \sum_i w_i^{(m)} \exp(-\alpha\, y_i G(x_i)) \\ &= \underbrace{\sum_{i:\, y_i = G(x_i)} w_i^{(m)} e^{-\alpha}}_{W_C\, e^{-\alpha}} + \underbrace{\sum_{i:\, y_i \neq G(x_i)} w_i^{(m)} e^{+\alpha}}_{W_W\, e^{+\alpha}}, \end{aligned}$$

where $W_C = \sum_{i:\,\text{correct}} w_i^{(m)}$ and $W_W = \sum_{i:\,\text{wrong}} w_i^{(m)}$. Differentiate with respect to α and set to 0:

$$\frac{\partial Q}{\partial \alpha} = -W_C e^{-\alpha} + W_W e^{+\alpha} = 0 \quad \Longrightarrow \quad e^{2\alpha} = \frac{W_C}{W_W} \quad \Longrightarrow \quad \alpha^* = \tfrac{1}{2} \log \frac{W_C}{W_W}.$$

Define the weighted error $\text{err}_m = W_W/(W_C + W_W)$, so $W_C/W_W = (1 - \text{err}_m)/\text{err}_m$, giving

$$\boxed{\alpha^* = \tfrac{1}{2} \log \frac{1 - \text{err}_m}{\text{err}_m}}. \quad \square$$

AdaBoost.M1 absorbs the $\frac{1}{2}$ into the weight-update rule, so the algorithm uses $\alpha_m = \log[(1 - \text{err}_m)/\text{err}_m]$ rather than half of it. *Grading: 3 P. 1 P for the correct/wrong split, 1 P for the derivative and $e^{2\alpha} = W_C/W_W$, 1 P for substituting* err_m.

(ii) *(1 %)* For $\text{err}_m = 0.30$:

$$\alpha_m = \log \frac{1 - 0.30}{0.30} = \log \frac{0.70}{0.30} = \log(2.333\ldots) \approx \boxed{0.85}.$$

$$\exp(\alpha_m) = \frac{0.70}{0.30} \approx \boxed{2.33}.$$

Grading: 0.5 P each.

(iii) *(1 %)* If $\text{err}_m > 0.5$, then $\alpha_m = \log[(1 - \text{err}_m)/\text{err}_m] < 0$; the base learner is worse than random and its sign is flipped in the ensemble vote (equivalently, the algorithm trusts $-G_m$). In practice AdaBoost is usually stopped or the learner re-selected when this happens. *Grading: 1 P. Accept "α_m becomes negative; G_m contributes with inverted sign."*

(iv) *(1 %)* A deep tree is already a low-bias / high-variance learner on its own. Boosting's strategy is to chain many *weak* learners so that each iteration reduces bias by a small, controlled amount; if each base learner is itself high-variance, the cumulative variance of the ensemble explodes and the "many weak learners" logic breaks — which is why AdaBoost (and most boosting) defaults to stumps or very shallow trees. *Grading: 1 P.*

c) Neural-network classifier with regularization *(6 %)*

Solution

(i) *(1 %)* Parameter count layer by layer:

- Hidden layer 1 ($7 \to 20$): $7 \cdot 20 = 140$ weights $+20$ biases $= 160$.
- Hidden layer 2 ($20 \to 10$): $20 \cdot 10 = 200$ weights $+10$ biases $= 210$.
- Output layer ($10 \to 1$): $10 \cdot 1 = 10$ weights $+1$ bias $= 11$.

Total: $160 + 210 + 11 = \boxed{381}$ parameters. *Grading: 1 P. Forgetting biases (giving* 350*) is a common miss; deduct 0.5 P.*

(ii) *(1 %)* Pre-activation:

$$\begin{aligned} z = w^\top x + b &= 0.10(0.5) + 0.30(1.0) + (-0.20)(-1.0) + 0.50(2.0) \\ &\quad + 0.05(0.5) + (-0.40)(0.5) + 0.20(-0.5) + (-0.10) \\ &= 0.05 + 0.30 + 0.20 + 1.00 + 0.025 - 0.20 - 0.10 - 0.10 \\ &= 1.175. \end{aligned}$$

Since $z > 0$, $h = \mathrm{ReLU}(z) = z \approx \boxed{1.18}$ (and $z \approx \boxed{1.18}$). *Grading: 1 P (0.5 P for z, 0.5 P for h).*

(iii) *(1 %)* Train-test gap of 98% $\to$ 70% is classic overfitting: with 381 parameters and only $\sim$ 350 training points, the network has enough capacity to memorize the training set, driving training error to near-zero (low *empirical* bias) while estimator variance is large and test error is much worse. Regularization forces the optimizer toward solutions with smaller capacity / smoother decision surfaces, trading a small increase in training error for a substantial decrease in test error. *Grading: 1 P.*

(iv) *(1 %)* (a) L^2 **weight decay** adds $\lambda\|\boldsymbol{\theta}\|_2^2$ to the training loss, which shrinks weights toward zero each step and biases the optimizer toward smaller-norm (smoother) solutions. (b) **Dropout** randomly masks a fraction p of hidden units at each training forward pass, preventing the network from depending on any single feature ("ensemble-of-subnetworks" interpretation); at test time *all* units are kept and the weights (or activations) are rescaled by the keep-probability so that the expected unit input matches the training-time distribution. (c) **Early stopping** monitors held-out validation loss and halts training at the epoch where validation loss first stops improving, returning those weights — which effectively limits the number of optimizer steps and so the effective capacity. *Grading: 1 P total (roughly 0.33 P each).*

(v) *(1 %) Treating the binary output as the two-class one-hot* $(0, 1)$ *for* $y = 1$ *and applying* $\varepsilon/(C-1)$ *with* $C = 2$*:* off-target mass $0.10/1 = 0.10$, on-target mass $1 - 0.10 = 0.90$, so the modified target is

$$\boxed{(0.10,\ 0.90)}.$$

Label smoothing prevents the network from being forced to drive its softmax / sigmoid outputs to exactly 0 or 1 (which would require unbounded logits) and acts as a regularizer

that is also robust to mislabelled training examples — the gradient contribution from a wrong label is bounded rather than infinite. *Grading: 1 P. Accept either symmetric convention* $(0.10, 0.90)$ *or* $(0.05, 0.95)$ *with an explicit statement of which convention is used; the exam's P2(h)(i) used the symmetric* $\varepsilon/(C-1)$ *form which here yields* $(0.10, 0.90)$.

(vi) *(1 %)* The classical bias–variance bound that "$p \gg n$ must overfit" is false in the over-parameterized regime: SGD's implicit bias toward minimum-norm interpolators, combined with regularization (weight decay / dropout / early stopping), can produce models that interpolate the training set and still generalize well — the "benign overfitting" / double-descent phenomenon. The friend's claim assumes the classical regime where it does not apply. *Grading: 1 P.*

d) Comparison of classifiers and class imbalance *(4 %)*

Solution

(i) *(2 %)* Logistic regression:

- Sensitivity $= 18/(18+20) = 18/38 \approx \boxed{0.47}$.
- Specificity $= 64/(10+64) = 64/74 \approx \boxed{0.86}$.
- Error rate $= (20+10)/112 = 30/112 \approx \boxed{0.27}$.

AdaBoost:

- Sensitivity $= 25/(25+13) = 25/38 \approx \boxed{0.66}$.
- Specificity $= 60/(14+60) = 60/74 \approx \boxed{0.81}$.
- Error rate $= (13+14)/112 = 27/112 \approx \boxed{0.24}$.

Grading: 0.33 P per number.

(ii) *(1 %)* The naive "no CHD" classifier misclassifies the 38 true positives, giving error $= 38/112 \approx \boxed{0.34}$. Accuracy alone is a poor discriminator here: the naive classifier already has 66% accuracy, AdaBoost has 76% and logistic regression 73%, all in the same range, and accuracy hides the fact that the naive classifier has 0% sensitivity — it never flags a true CHD patient. *Grading: 1 P.*

(iii) *(1 %)* For a screening task with costly false negatives, the relevant metric is **sensitivity** (recall on the positive class). AdaBoost has sensitivity 0.66 vs. logistic regression's 0.47, so **deploy AdaBoost**; the cost is a modest reduction in specificity (0.81 vs. 0.86), which under asymmetric costs is the right trade. *Grading: 1 P (accept any answer that names the right classifier and the right metric; deduct 0.5 P for a correct metric with the wrong classifier).*

e) KNN with mixed-unit predictors *(2 %)*

Solution

(i) *(1 %)* For a test point x_0, compute distances to all training points, pick the K closest, and classify x_0 by majority vote among those K neighbours' labels (with ties broken at random or by a tiebreak rule). *Grading: 1 P.*

(ii) *(1 %)* Euclidean distance on raw predictors lets the variable with the largest numerical range (here `age` in years, with values up to ~ 60) dominate the distance, swamping the contribution of all other features — so KNN effectively becomes a 1-D classifier on `age`. The standard remedy is to *standardize* (center and scale to unit SD) each continuous predictor before fitting KNN. *Grading: 1 P.*

End of solution. Total: $10 + 28 + 16 + 20 + 26 = 100$ points.