Chris Gregg
CS109

Spring 2026
June 8th

CS109 Final Exam

This is a closed calculator/computer/phone/smart-watch/smart-toothbrush exam. You are, however, allowed to use 6 pages of back/front, handwritten notes in the exam. You have 3 hours (180 minutes) to take the exam. The exam is 130 points, meant to roughly correspond to one point per minute of the exam, and is designed so that a well-prepared student can comfortably finish in about 130 minutes. You may want to use the point allocation for each problem as an indicator for pacing yourself on the exam.

In the event of an incorrect answer, any explanation you provide of how you obtained your answer can potentially allow us to give you partial credit for a problem. For example, describe the distributions and parameter values you used, where appropriate. It is fine for your answers to include summations, products, factorials, exponentials, and combinations. You can leave your answer in terms of Φ (the CDF of the standard normal) or Φ^{-1}. For example $\Phi(3/4)$ is an acceptable final answer. You can only leave your answer in **terms of code** when the problem explicitly says so.

Stanford Honor Code: The Honor Code is an undertaking of the Stanford academic community, individually and collectively. Its purpose is to uphold a culture of academic honesty. Students will support this culture of academic honesty by neither giving nor accepting unpermitted academic aid on this examination.

This course is participating in the proctoring pilot overseen by the Academic Integrity Working Group (AIWG), therefore proctors will be present in the exam room. The purpose of this pilot is to determine the efficacy of proctoring and develop effective practices for proctoring in-person exams at Stanford.

I acknowledge and accept the letter and spirit of the honor code. I pledge to write more neatly than I have in my entire life:

Signature: _____

First and Last Name (print): _____

Stanford Email (@stanford.edu): _____

Exam Break Sign-out

I pledge that during my exam break:

- I will not bring any paper, electronic devices (phone, smart watch, smart glasses, etc), or aid (permitted or unpermitted) *out of or into* the exam room, nor access any aid during the break.
- I will not communicate with anyone other than the course instructional staff about the content of the exam.

Signature Confirming Honor Code	**Exit Time**	**Return Time**	**Proctor Initial**	**Length (mins)**

If you are feeling unwell and are not able to complete the exam, please speak with the proctor.

1 Short Answer [17 Points]

You may not use code to answer any part of this problem.

(a) (5 points) Let $X \sim \text{Exp}(\lambda = 0.5)$. What is $P(X > 4)$?

Using the Exponential CDF $F(x) = P(X \leq x) = 1 - e^{-\lambda x}$ and $P(X > 4) = 1 - P(X <= 4)$:

$$P(X > 4) = 1 - (1 - e^{-0.5 \times 4}) = e^{-2}$$

(b) (6 points) A binary classification model always outputs 0.7 for the probability that $Y = 1$, regardless of the features given. In the test dataset, exactly 70% of examples have $Y = 1$.

(i) Would this model be considered *accurate*? Answer Yes or No and give one sentence of explanation.

(ii) Would this model be considered *calibrated*? Answer Yes or No and give one sentence of explanation.

(i) No. If we threshold at 0.5, the model predicts class 1 for *every* input, so it merely matches the majority base rate (70% accuracy) while completely ignoring the features — it has not learned anything discriminative.
(ii) Yes. Among all the examples to which the model assigns probability 0.7 (which is all of them), the label is actually 1 exactly 70% of the time, so its stated probability matches the observed empirical frequency.

(c) (6 points) The expected number of failed package deliveries a drone makes in a week depends on the weather, as shown in the table:

Weather	**Expected failed deliveries per week**
Clear	0.4
Rainy	1.5
Stormy	6.0

In a given region, 50% of weeks are clear, 35% are rainy, and the rest are stormy. For a randomly chosen week, what is the expected number of failed deliveries? You may leave your answer as a numerical expression.

By the law of total expectation, conditioning on the weather W:

$$E[\text{failures}] = \sum_w E[\text{failures} \mid W = w] P(W = w).$$

The stormy probability is $1 - 0.50 - 0.35 = 0.15$, so

$$E[\text{failures}] = (0.50)(0.4) + (0.35)(1.5) + (0.15)(6.0).$$

2 Fraud Detection [14 Points]

A bank monitors card transactions for fraud. From historical data:

- 2% of all transactions are fraudulent.
- If a transaction *is* fraudulent, detector 1 flags it with probability 0.95.
- If a transaction is *legitimate*, detector 1 flags it with probability 0.04.

You may not use code to answer any part of this problem. You do not need to simplify your answers to decimals.

(a) (6 points) A transaction is flagged by detector 1. Write an expression for the probability that the transaction is actually fraudulent.

Let F be the event that the transaction is fraudulent and D_1 the event that detector 1 flags it. By Bayes' theorem,

$$P(F \mid D_1) = \frac{P(D_1 \mid F)P(F)}{P(D_1 \mid F)P(F) + P(D_1 \mid F^c)P(F^c)}$$
$$= \frac{(0.95)(0.02)}{(0.95)(0.02) + (0.04)(0.98)}$$

(b) (8 points) The bank also runs an *independent* second detector. If a transaction is fraudulent, detector 2 flags it with probability 0.90; if it is legitimate, detector 2 flags it with probability 0.10. Given the value of whether a transaction is fraudulent, the two detectors flag independently. A transaction is flagged by *both* detectors. Write an expression for the probability that the transaction is actually fraudulent.

Let D_1, D_2 be the events that detectors 1 and 2 flag the transaction. Conditioned on fraud status, the detectors are independent, so the likelihoods multiply:

$$P(D_1, D_2 \mid F) = (0.95)(0.90), \qquad P(D_1, D_2 \mid F^c) = (0.04)(0.10).$$

Applying Bayes' theorem,

$$P(F \mid D_1, D_2) = \frac{(0.02)(0.95)(0.90)}{(0.02)(0.95)(0.90) + (0.98)(0.04)(0.10)}$$

3 Support Chat Requests [15 Points]

A customer support team receives live-chat requests according to a Poisson process at an average rate of 8 requests per hour. You may not use code to answer any part of this problem. You do not need to simplify your answers to decimals.

(a) (5 points) Let T be the waiting time (in hours) from now until the next chat request arrives. State the distribution of T (including its parameter), and find the probability that no request arrives in the next 15 minutes.

For a Poisson process with rate $\lambda = 8$ per hour, the waiting time to the next arrival is

$$T \sim \text{Exp}(\lambda = 8).$$

Fifteen minutes is 0.25 hours, so

$$P(T > 0.25) = e^{-8\times 0.25} = e^{-2}.$$

(b) (5 points) Let N be the number of chat requests that arrive during a 30-minute window. State the distribution of N (including its parameter), and write an expression for the probability that *exactly* 3 requests arrive.

Over a 30-minute (0.5-hour) window, the expected number of requests is $8 \times 0.5 = 4$, so

$$N \sim \text{Poi}(\lambda = 4).$$

Using the Poisson PMF,

$$P(N = 3) = \frac{e^{-4} \cdot 4^3}{3!}.$$

(c) (5 points) Each chat request, independently, is "urgent" with probability 0.2. Let U be the number of urgent requests that arrive in a one-hour window. State the distribution of U (including its parameter) and give $E[U]$ and $\text{Var}(U)$.

By the thinning property of a Poisson process, independently marking each arrival with probability $p = 0.2$ yields a Poisson process with rate λp. Over one hour,

$$U \sim \text{Poi}(\lambda = 8 \times 0.2 = 1.6).$$

For a Poisson, the mean and variance both equal the parameter:

$$E[U] = 1.6, \qquad \text{Var}(U) = 1.6.$$

4 Warehouse Shipping [14 Points]

A warehouse ships packages whose weights are independent and identically distributed with mean $\mu = 2.5$ kg and variance $\sigma^2 = 1.44$ kg^2. A delivery truck is loaded with 100 packages. You may not use code to answer any part of this problem.

(a) (7 points) Let S be the total weight of the 100 packages. Write an expression for the approximate probability that the total weight exceeds 260 kg.

Let $S = \sum_{i=1}^{100} X_i$. Then

$$E[S] = 100 \times 2.5 = 250, \qquad \text{Var}(S) = 100 \times 1.44 = 144,$$

so the standard deviation of S is $\sqrt{144} = 12$. By the CLT, S is approximately $\mathcal{N}(250, 144)$. Standardizing,

$$P(S > 260) \approx P\left(Z > \frac{260 - 250}{12}\right) = P(Z > \frac{5}{6}) = 1 - \Phi\left(\tfrac{5}{6}\right).$$

(b) (7 points) The warehouse wants to advertise a weight capacity c for the truck such that the total weight of 100 packages stays *below* c with probability 0.95. Write an expression for c.

We want c such that $P(S < c) = 0.95$ where $S \approx \mathcal{N}(250, 144)$. Standardizing,

$$P\left(Z < \frac{c - 250}{12}\right) = 0.95 \implies \frac{c - 250}{12} = \Phi^{-1}(0.95).$$

Solving for c,

$$c = 250 + 12\,\Phi^{-1}(0.95) \text{ kg}.$$

5 Hidden Chambers in the Pyramid of the Sun [18 Points]

Archaeologists want to detect hidden chambers inside the Pyramid of the Sun at Teotihuacán using muons and probability. A muon is a particle that originates in outer space and arrives at Earth according to a Poisson process. When muons pass through rock they travel in a straight line, but as they travel they are sometimes absorbed.

A muon detector is placed inside the pyramid and registers only muons traveling straight down. Our goal is to estimate how many meters of solid rock lie directly above the detector — a value that helps reveal hidden chambers.

If x is the number of meters of solid rock directly above the detector, then muons arrive at the detector at a rate of

$$\lambda(x) = 180 \cdot e^{-x/50} \quad \text{muons per month.}$$

You do not need to simplify your answers; you may leave them with integrals, sums, products, factorials, and exponentials.

(a) (6 points) Suppose the amount of rock above the detector is known to be exactly $x = 50$ meters. The arrival rate is then $\lambda(50) = 180e^{-50/50} = 180e^{-1}$ muons per month. Write an expression for the probability that in one month the detector registers *exactly* 20 muons.

Given $x = 50$, the monthly count is $N \sim \text{Poi}\big(\lambda(50)\big)$ with $\lambda(50) = 180e^{-1}$. By the Poisson PMF,

$$P(N = 20) = \frac{e^{-180e^{-1}}\big(180e^{-1}\big)^{20}}{20!}.$$

(b) (12 points) Let X be your belief, in meters, about the amount of rock above the detector. Your prior belief is that any value between 0 and 120 meters is equally likely: $X \sim \text{Uni}(0, 120)$. After one month, the detector has registered 20 muons. Write an expression for your updated (posterior) belief $f(X = x \mid N = 20)$. You may leave your answer as an expression containing an integral; you do not need to evaluate the integral.

This is a Bayesian update with a continuous prior and a discrete (Poisson) observation. We want

$$f(X = x \mid N = 20) = \frac{P(N = 20 \mid X = x)\, f(X = x)}{P(N = 20)}.$$

Likelihood. Given $X = x$, the count is Poisson with rate $\lambda(x) = 180\, e^{-x/50}$:

$$P(N = 20 \mid X = x) = \frac{e^{-\lambda(x)} \lambda(x)^{20}}{20!}, \qquad \lambda(x) = 180\, e^{-x/50}.$$

Prior. $f(X = x) = \dfrac{1}{120}$ for $0 \le x \le 120$.
Normalizing constant (by the law of total probability):

$$P(N = 20) = \int_0^{120} \frac{e^{-\lambda(u)} \lambda(u)^{20}}{20!} \cdot \frac{1}{120}\, du.$$

Posterior. For $0 \le x \le 120$ (the $\frac{1}{20!}$ and $\frac{1}{120}$ factors cancel):

$$f(X = x \mid N = 20) = \frac{e^{-\lambda(x)} \lambda(x)^{20}}{\displaystyle\int_0^{120} e^{-\lambda(u)} \lambda(u)^{20}\, du}, \qquad \lambda(x) = 180\, e^{-x/50}.$$

6 Logistic Regression [16 Points]

A logistic regression model predicts a binary label $Y \in \{0,1\}$ from a feature vector. For a datapoint with features x_1 and x_2, the model predicts

$$P(Y=1 \mid \mathbf{x}) = \sigma(\theta_0 + \theta_1 x_1 x_2), \qquad \sigma(z) = \frac{1}{1+e^{-z}}.$$

(a) (4 points) Suppose the trained weights are $\theta_0 = -1$ and $\theta_1 = 2$. For a new datapoint with $x_1 = 1$ and $x_2 = 1$, write an expression for $P(Y=1 \mid \mathbf{x})$ and simplify it as far as you can.

The linear term is

$$z = \theta_0 + \theta_1 x_1 x_2 = -1 + 2(1)(1) = 1.$$

Therefore

$$P(Y=1 \mid \mathbf{x}) = \sigma(1) = \frac{1}{1+e^{-1}}.$$

(b) (12 points) You are given a training set of n datapoints $\left(\mathbf{x}^{(i)}, y^{(i)}\right)$ with $y^{(i)} \in \{0,1\}$. Use maximum likelihood estimation to define the optimal estimates for θ_0 and θ_1. Write the log-likelihood of the training labels, derive the partial derivatives of the log-likelihood with respect to θ_0 and θ_1, and state the gradient-ascent update rules (with learning rate η) used to find the optimal values.

Let $s^{(i)} = x_1^{(i)} x_2^{(i)}$ and $\hat{p}^{(i)} = \sigma\left(\theta_0 + \theta_1 s^{(i)}\right)$. Each label is Bernoulli$(\hat{p}^{(i)})$, so the log-likelihood is

$$LL(\theta_0, \theta_1) = \sum_{i=1}^{n} \left[y^{(i)} \log \hat{p}^{(i)} + \left(1 - y^{(i)}\right) \log\left(1 - \hat{p}^{(i)}\right) \right].$$

Using the identity $\dfrac{\partial}{\partial z}\sigma(z) = \sigma(z)\left(1-\sigma(z)\right)$ and the chain rule, the partial derivatives simplify to

$$\frac{\partial LL}{\partial \theta_0} = \sum_{i=1}^{n} \left(y^{(i)} - \hat{p}^{(i)} \right), \qquad \frac{\partial LL}{\partial \theta_1} = \sum_{i=1}^{n} \left(y^{(i)} - \hat{p}^{(i)} \right) s^{(i)}.$$

Setting these to zero gives the MLE conditions, but they have no closed-form solution, so the optimal $\hat{\theta}_0, \hat{\theta}_1$ are found by gradient ascent, repeatedly stepping in the direction of the gradient:

$$\theta_0 \leftarrow \theta_0 + \eta \sum_{i=1}^{n} \left(y^{(i)} - \sigma\left(\theta_0 + \theta_1 s^{(i)}\right) \right),$$

$$\theta_1 \leftarrow \theta_1 + \eta \sum_{i=1}^{n} \left(y^{(i)} - \sigma\left(\theta_0 + \theta_1 s^{(i)}\right) \right) s^{(i)}.$$

7 Testing a New Medicine [12 Points]

You are developing a medicine that sometimes has a desired effect and sometimes does not. With FDA approval, you are allowed to test your medicine on 9 patients. You observe that 7 have the desired outcome. Before running any experiments, your belief about p, the probability that the medicine has the desired effect, was uniform over $[0,1]$.

(a) (6 points) Give a random-variable representation for p, the probability that the medicine has the desired effect, after observing the data. Name the distribution and give its parameters, and briefly justify your answer.

A uniform prior on $[0,1]$ is Beta$(1,1)$. With a Beta prior and Bernoulli/Binomial data, the posterior is Beta with the counts of successes and failures added to the prior parameters. Here we observed 7 successes and $9-7=2$ failures, so

$$p \mid \text{data} \sim \text{Beta}(\alpha = 1+7,\ \beta = 1+2) = \text{Beta}(8,3).$$

(b) (6 points) Write an expression for the probability that $p > 0.6$. You may use code for this answer, or you may leave your answer in terms of fractions, factorials, integrals, products, etc., without simplifying.

Letting $F_{\text{Beta}(8,3)}$ denote the CDF of the posterior,

$$P(p > 0.6) = 1 - F_{\text{Beta}(8,3)}(0.6).$$

Equivalently, $P(p > 0.6) = \displaystyle\int_{0.6}^{1} \frac{x^7(1-x)^2}{B(8,3)}\,dx$, where $B(8,3)$ is the Beta function.

8 Debugging a Bootstrap [12 Points]

To survey opinion on a sensitive topic, a pollster gives each of n respondents seven true/false statements: six harmless statements that each person answers "true" with probability 0.5 (independently), plus one *delicate* statement that a person answers "true" with the unknown probability p we care about. Each respondent i reports only X_i, the total number of the seven statements they marked "true." Because $E[X_i] = 6(0.5) + p = 3 + p$, we estimate p with $\hat{p} = \big(\text{average of the } X_i\big) - 3$.

We now want to use **bootstrapping** to estimate the probability that our estimate $\hat{p}$ is within 0.1 of the true p. Following the bootstrap recipe, we treat the original estimate as the "true" value, repeatedly resample, and measure how often the resampled estimate lands within 0.1 of it. A colleague wrote the code below to do this, but it contains **three bugs**.

```
1  import numpy as np
2
3  def estimate_p(samples):
4      # E[X_i] = 6 * 0.5 + p = 3 + p, so p = (average of X) - 3
5      return np.median(samples) - 3
6
7  def prob_within_0_1(samples, n_boot=10000):
8      n = len(samples)
9      # treat the original estimate as the  true  p
10     p_hat = estimate_p(samples)
11     n_close = 0
12     for b in range(n_boot):
13         resample = np.random.choice(samples, size=n, replace=False)
14         p_boot = estimate_p(resample)
15         if abs(p_boot - p_hat) < 0.1:
16             n_close += 1
17     return n_close / n
```

Identify all three bugs. For each one, **state the line number** of the offending line, explain in one or two sentences why it is incorrect, and give the corrected line. (4 points each.)

Bug 1 — wrong estimator. The line `return np.median(samples) - 3` should use the mean. The estimator was derived from $E[X_i] = 3 + p$, and an expectation is estimated by the sample *mean*, not the median; the median of X does not equal $3 + p$ in general, so this returns a biased estimate of p. Fix:

```
return np.mean(samples) - 3
```

Bug 2 — resampling without replacement. The line `np.random.choice(samples, size=n, replace=False)` uses `replace=False`, but a bootstrap must resample *with* replacement. With `replace=False`, each "resample" of size n is just a permutation of the original data, so its mean (and hence `p_boot`) always equals `p_hat`. The bootstrap then captures no sampling variability and the function reports that the estimate is within 0.1 essentially 100% of the time, regardless of the data. Fix:

```
resample = np.random.choice(samples, size=n, replace=True)
```

Bug 3 — wrong denominator. The line `return n_close / n` divides by the sample size. We are computing the fraction of the `n_boot` resamples whose estimate landed within 0.1, so we must divide by the number of resamples, `n_boot`, not `n`. As written, the result can exceed 1 and is not a valid probability. Fix:

```
return n_close / n_boot
```

9 Decision Tree Entropy [12 Points]

A decision tree has been trained on a dataset of 80 patients to predict whether a heart is normal or abnormal (a binary `Label`). A decision tree is trained to create decision nodes that maximize the reduction in expected entropy of the labels, reflecting how well a split separates the data.

The root node splits the data on the value of a binary feature "C.4". Datapoints with C.4 $= 0$ go to the left child; datapoints with C.4 $= 1$ go to the right child. From the trained model we know:

- There are 80 datapoints in total. 40 have `Label` $= 1$ and 40 have `Label` $= 0$.
- 52 datapoints have C.4 $= 0$ (left child). Of those, 17 have `Label` $= 1$ and 35 have `Label` $= 0$.
- 28 datapoints have C.4 $= 1$ (right child). Of those, 23 have `Label` $= 1$ and 5 have `Label` $= 0$.

Give all entropies in bits, using $\log_2$ when computing $H(X)$.

(a) (4 points) Compute the entropy of the labels in the original dataset (before the split).

The MLE is $\hat{p} = 40/80 = 0.5$, so

$$H_{\text{orig}} = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1 \text{ bit.}$$

(b) (5 points) Compute the entropy of the labels in the left child and in the right child, then compute the *expected* entropy of the labels in the child node that a randomly chosen training datapoint lands in You should leave fractions and logarithms in your final answer.

Left child: $\hat{p}_L = 17/52 \approx 0.327$,

$$H_L = -\tfrac{17}{52}\log_2 \tfrac{17}{52} - \tfrac{35}{52}\log_2 \tfrac{35}{52} \approx 0.912 \text{ bits.}$$

Right child: $\hat{p}_R = 23/28 \approx 0.821$,

$$H_R = -\tfrac{23}{28}\log_2 \tfrac{23}{28} - \tfrac{5}{28}\log_2 \tfrac{5}{28} \approx 0.677 \text{ bits.}$$

A random datapoint goes left with probability $52/80$ and right with probability $28/80$, so the expected entropy after the split is

$$H_{\text{exp}} = \frac{52}{80}(0.912) + \frac{28}{80}(0.677) \approx 0.830 \text{ bits.}$$

(c) (3 points) Without performing any new calculations, state which of the two child-node entropies from part (b) is smaller, and explain why in terms of the label distributions in the two children.

The **right child** has the smaller entropy. Its labels are much more *pure*: 23 of its 28 datapoints share the same label ($\hat{p}_R \approx 0.82$), so it sits far from the maximum-uncertainty point of $p = 0.5$ where Bernoulli entropy peaks. The left child is more evenly mixed ($\hat{p}_L \approx 0.33$, closer to 0.5), so there is more uncertainty about a datapoint's label there and its entropy is larger. Intuitively, the more lopsided the class split in a node, the less uncertain — and the lower the entropy.