Chris Piech
CS109

Fall 2024

CS109 Final Exam

This is a closed calculator/computer exam. You are, however, allowed to use notes in 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, **integrals**, products, factorials, exponentials, and combinations.

You can leave your answer in terms of Φ (the CDF of the standard normal) or Φ^{-1} (the inverse CDF). For example $\Phi\left(\frac{3}{4}\right)$ is an acceptable final answer. Recall that the exam is going to be "curved" according to the difficulty of the questions and as such hard questions will not translate to lower grades.

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: ____

Family Name (print): ____

Given Name (print): ____

Email (preferably your gradescope email): ______________________________

1. Short Answer [17 points]

a. (5 points) Let $X \sim \text{Exp}(\lambda = 2.5)$. What is $P(X > 5)$?

b. (6 points) A binary classification machine learning model always outputs 0.6 for the probability that $Y = 1$, regardless of the features given. 60% of the test dataset has $Y = 1$.
i) Would this model be considered accurate? Answer Yes or No and give one sentence of explanation:

ii) Would it be considered calibrated? Answer Yes or No and give one sentence of explanation:

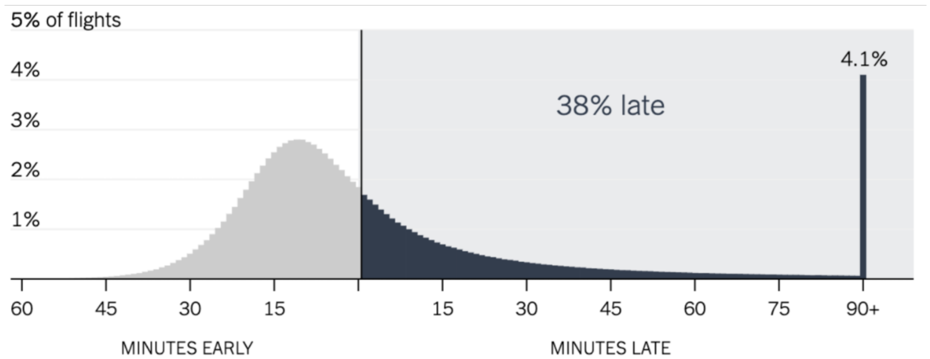
c. (6 points) As servers age, their probability of crashing increases. Here is a table of expected crashes per year for servers of different ages:

Server Age	Expected Crashes Per Year
1-2 years	0.5
3-4 years	3.2
5+ years	9.1

At a large computing center, 30% of servers are 1-2 years old, 50% are 3-4 years old, and the rest are at least 5 years old. A server is chosen at random to be assigned to a user. What are the expected number of crashes next year for this user's server?

2. Airline Flights [16 points]

38% of all flights in 2023 were late by one minute or more.
4.1% of all flights in 2023 were more than 90 minutes late.
Here is the full distribution of delays for flights in 2023 from the Bureau of Transportation Stats:



Assume all flights are independent: the number of minutes late of any one flight is independent of any other.

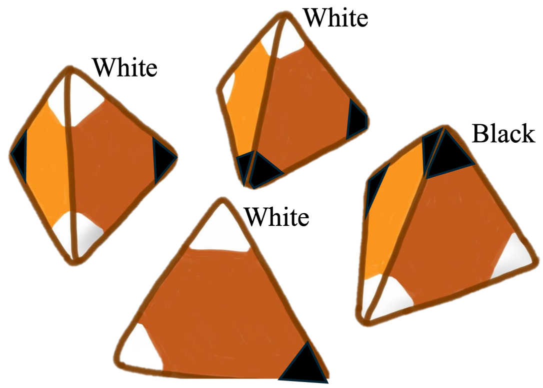
a. (5 points) If you take 3 flights in 2023, what is the probability that none are late?

b. (5 points) If you take 2 flights in 2023, what is the probability that either flight one or flight two is late?

c. (6 points) Given that a flight in 2023 is late, what is the probability that it is delayed by more than 90 minutes?

3. Board Games [16 points]

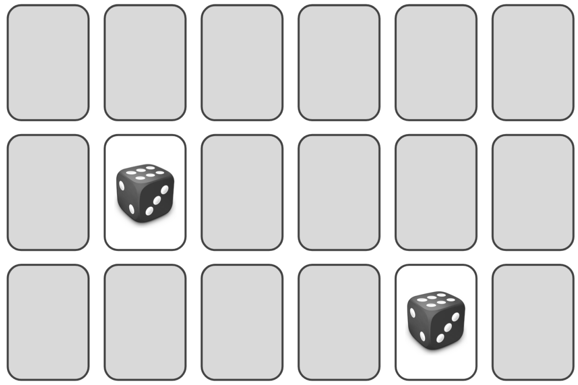
a. (8 points) The Royal Game of Ur is the oldest known board game. This game has four "ur dice." Here is a picture of the result of rolling the four ur dice where only one dice rolled a "black":



An "ur dice" is a four-sided pyramid with its four corners painted – two corners are white and two corners are black. A roll of an ur dice is "black" if the corner pointing upward is black. Each of the four corners is equally likely to be the corner pointing upward.

It is your turn and you will win the game if you either roll exactly four blacks or exactly two blacks. What is the probability that you win the game?

b. (8 points) In a game of Memory, 18 cards containing 9 distinct matching pairs are placed face down. If you randomly select two cards to turn over, what is the chance that the two cards match?



4. Song of the Quarter [25 points]

This quarter in CS109 there were 167 songs that were voted on. For each song, we have a list of votes where each vote is an integer in the set $\{1, 2, 3, 4, 5\}$. We assume all votes for a song are IID samples from the "true" distribution of CS109 opinion on the song.

For each song i we have m_i votes stored in a list `votes[i]` $= [x_1, x_2, \ldots, x_{m_i}]$. We have already calculated:

$$\mu_i = \frac{1}{m_i}\sum_{j=1}^{m_i} x_j \qquad \text{using } \texttt{np.mean(votes[i])}$$

$$\text{var}_i = \frac{1}{m_i}\sum_{j=1}^{m_i}(x_j - \mu_i)^2 \qquad \text{using } \texttt{np.var(votes[i])}$$

$$\text{svar}_i = \frac{1}{m_i - 1}\sum_{j=1}^{m_i}(x_j - \mu_i)^2 \qquad \text{using } \texttt{np.var(votes[i], ddof=1)}$$

a. (7 points) Song 1 has $m_1 = 45$ votes. We have calculated:

$$\mu_1 = 3.82 \qquad \text{var}_1 = 1.4 \qquad \text{svar}_1 = 1.5$$

Estimate the probability that the true average rating for song 1 is less than 3.

b. (8 points) Song 1 has $m_1 = 45$ votes. Song 2 has $m_2 = 36$ votes. We have calculated:

Song 1:	$\mu_1 = 3.82$	$\text{var}_1 = 1.4$	$\text{svar}_1 = 1.5$
Song 2:	$\mu_2 = 3.79$	$\text{var}_2 = 1.7$	$\text{svar}_2 = 1.8$

What is the probability that the true average of Song 1 is greater than the true average for Song 2?

c. (10 points) Write pseudo-code to calculate a p-value for the significance of the difference between the average of song 1 and song 2. That is, find the probability that the votes for both songs are samples from the same universal distribution and the observed difference in averages $|\mu_1 - \mu_2|$ is purely due to random chance.

Let **`votes[1]`** be the list of votes for song 1.
Let **`votes[2]`** be the list of votes for song 2.

```
# find the absolute difference of means between the two songs' votes
observed_diff = np.abs(np.mean(votes[1]) - np.mean(votes[2]))

m1 = len(votes[1])
m2 = len(votes[2])

# assume the null hypothesis: all votes came from same dist
pooled_samps = votes[1] + votes[2]

gt_obs_diff_counts = 0

for i in range(10000):
    # repeat the original experiment under the null hypothesis
    resample1 = np.random.sample(pooled_samps, size=m1, replace=True)
    resample2 = np.random.sample(pooled_samps, size=m2, replace=True)

    # count how often the difference in means is more extreme
    # than what we originally observed
    diff = np.abs(np.mean(resample1) - np.mean(resample2))

    if diff >= observed_diff:
        gt_obs_diff_counts += 1

# p-value = fraction of times we see something more extreme
# if we assume the null hypothesis
print(gt_obs_diff_counts / 10000)
```

5. What Name Doesn't Give Away Age? [20 points]

In the Name to Age problem in class, we came up with the following probability distribution that someone was born in year b given that their name is n:

$$P(B = b|N = n) = \frac{\text{count}(b, n)}{\sum_{y \in \text{years}} \text{count}(y, n)}$$

Write pseudo-code to choose the name that leaks the least information about age: specifically, the name where the distribution of the year they were born has the highest entropy. You can use the following variables and functions:

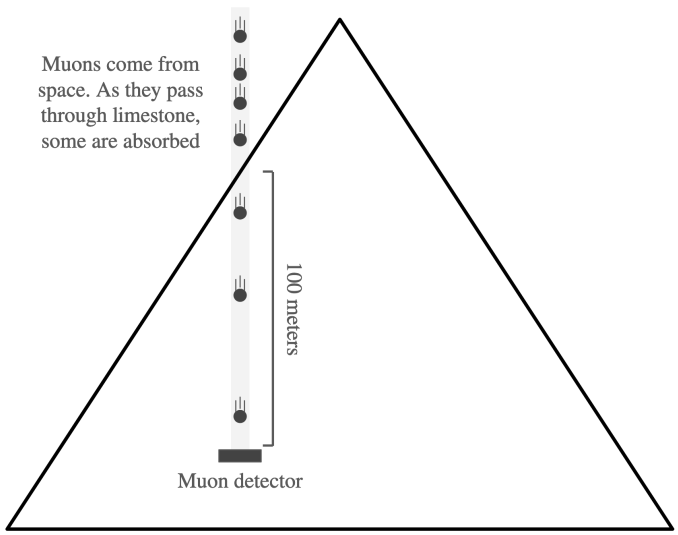
`all_names`, a list of all possible names to consider.
`all_years`, a list of all possible years to consider.
`count(year, name)`, returns the number of people born in the specified year with the specified name.

```
# helper functions
def calc_entropy(pmf_dict):
    entropy = 0
    for prob in pmf_dict.values():
        entropy -= prob * math.log2(prob)  # note the minus equals
    return entropy

def make_pmf_for_name(name):
    # implement the equation given for P(B = b | N = name), for all b

    counts_per_year = {year : count(year, name) for year in all_years}
    total = sum(counts_per_year.values())

    pmf = {year : counts_per_year(year) / total for year in all_years}
    return pmf


def find_best_name():
    # loop through all names, keeping track of the name
    # with the highest entropy seen so far

    best_name_so_far = ""
    best_entropy_so_far = 0

    for name in all_names:
        pmf = make_pmf_for_name(name)

        entropy = calc_entropy(pmf)

        if entropy > best_entropy_so_far:
            best_entropy_so_far = entropy
            best_name_so_far = name

    return best_name_so_far
```

6. Hidden Chambers in the Great Pyramid [20 points]

We are going to build a tool to predict the existence of hidden chambers in the Great Pyramid of Giza using muons and probability. A muon is a special type of particle that originates in outer-space and arrives at earth as a Poisson process. When the muons hit limestone, they do not change direction, but as they travel through the limestone, they sometimes get absorbed.

A muon detector is positioned inside a chamber in the Great Pyramid. It is 100 meters below the edge of the pyramid and will only detect muons traveling straight down. Our goal is to estimate: how many meters of limestone are above our detector? This number will help us detect any hidden chambers!



If you knew how many meters of limestone each muon was passing through, you could calculate the rate of muons arriving per month. If x is the meters of limestone, then the rate is $100 \cdot e^{-x/40}$ muons per month.

a. (6 points) Imagine the entire 100 meter path is limestone. In that case, the rate of muons arriving per month on the detection plate is $100 \cdot e^{-100/40} = 8.2$. What is the probability that in one month you would observe 12 muons?

b. (14 points) Let X be your belief in the meters of limestone above the detection plate. Your prior belief is that any number of meters from 0 to 100 is equally likely: $X \sim \text{Uni}(0, 100)$. After one month, your detection plate has been hit by 12 muons. What is your updated belief in X?
Recall: You may leave your answer with integrals or sums. You don't need to simplify for full credit.

7. Sorted Random Values [23 points]

We want to reason about the values produced by the following python code, which creates 10 random uniform values in the range [0, 1] and then sorts them from low to high:

```
# generate 10 random uniform values
values = []
for i in range(10):
   value_i = random_uniform(0,1)  # sample from standard uniform
   values.append(value_i)

# sort all of the values ascending from low to high
sorted_values = sorted(values)
print(sorted_values)
```

Here is what the list could look like when printed. For clarity each value is rounded to two decimal places:

```
sorted_values = [0.03, 0.13, 0.45, 0.51, 0.52, 0.63, 0.69, 0.82, 0.88, 0.91]
        index :    0     1     2     3     4     5     6     7     8     9
```

a. (10 points) The first value produced by **`random_uniform`** is 0.4. What is the probability that it will end up at index 4 in the sorted list?
In other words: What is the probability that exactly 5 out of the 9 other values are greater than 0.4?

b. (10 points) Let X_4 be the value at index 4 in the sorted list. What is the probability density of $X_4 = x$? *In other words: What is the probability density that $X_4 = x$ given that you know exactly 5 out of 9 numbers are greater than x? Recall: You may leave your answer with integrals or sums.*

c. (3 points) What is the variance of X_4? Hint: can you recognize X_4 as a random variable we have studied in class?

8. MLE of Negative Binomial [23 points]

You are trying to estimate the p parameter of a Negative Binomial. You already know that $r = 5$. Recall that if $X \sim \text{NegBin}(r, p)$:

$$P(X = k) = \binom{k-1}{r-1} p^r (1-p)^{k-r}$$

You have n samples of X: [7, 12, 9, 12, 8, 12, . . .]. Let k_i be the i^{th} value in this dataset.
Assume that the samples are IID from the same Negative Binomial and that $r = 5$.

a. (5 points) If $p = 0.7$ and $r = 5$ what is the likelihood of seeing the first sample, $X = 7$?

b. (18 points) Explain how you would choose parameter p and provide any necessary derivatives.

9. Recalibrating an Uncalibrated Model [20 points]

You have an uncalibrated binary classification model that outputs values $\hat{p} \in [0, 1]$. These outputs are meant to be the probability that $Y = 1$. However, the outputs from this model are **not** well-calibrated. For instance, among all examples where $\hat{p} \approx 0.9$, it was the case that Y was 1 only 70% of the time. To recalibrate the model's outputs you decide to use Platt Recalibration, where the corrected probability that $Y = 1$ is:

$$P(Y = 1 \mid \hat{p}) = \sigma(a \cdot \hat{p} - 0.5)$$

$\sigma(z) = 1/(1 + e^{-z})$ is the sigmoid function and a is the parameter of the recalibration model. Here is the partial derivative of the Platt Recalibration model with respect to a:

$$\frac{\partial}{\partial a}\sigma(a \cdot \hat{p} - 0.5) = \sigma(a \cdot \hat{p} - 0.5) \cdot [1 - \sigma(a \cdot \hat{p} - 0.5)] \cdot \hat{p}$$

a. (4 points) For a new datapoint the uncalibrated model outputs $\hat{p}$ of 0.9. If you use Platt Recalibration with $a = 2$ what is the recalibrated probability that $Y = 1$?

b. (16 points) You are given a training dataset with n outputs from the uncalibrated model $(\hat{p}^{(i)}, y^{(i)})$ where $\hat{p}^{(i)}$ is the uncalibrated output and $y^{(i)} \in \{0, 1\}$ is the true binary outcome. Explain how you could estimate the value of a that makes the $y^{(i)}$ values as likely as possible. Solve for any and all partial derivatives required by your answer.

That's all folks! Thank you for the lovely quarter. You were a wonderful class. Airline data is real for US domestic flights. The Royal Game of Ur originated in ancient Mesopotamia around 4,600 years ago. Archeologists have found old boards, dice, and cuneiform tablets with rules. In March 2023 ScanPyramids team discovered a new void in the Great Pyramid using muon tomography and claim they are 99.9999% confident. The inventor of this technology won a Nobel Prize. In the real-life Song of the Quarter I used my confidence in a song averages to select which songs needed more votes. Platt Recalibration (which typically also has a learnable intercept term) is often the best method to fix calibration issues.