Homework 4: Discrete Random Variables

In general, your goal in an explanation is to write enough that a student from class who has attended lecture, but not thought about the problem yet, could understand your approach, verify your reasoning, and believe your answer is correct. While we do not usually need to see arithmetic, you must include enough work that in principle one could rederive your answer with only a scientific calculator.

Unless a problem states otherwise, you should leave your answer in terms of factorials, combinations, etc., for instance 26^7 or $26!/7!$ or $26 \cdot \binom{26}{7}$ are all good forms for final answers.

Remember that you must "select pages" to identify which problem is on which page as part of your gradescope upload.

Submission: You must upload a **pdf** of your written solutions to Gradescope under "HW 4 [Written]". The use of LaTeX is *required*. Your *optional EC* code will be submitted under "HW 4 [Extra Credit]" as a file called `bloom.py`. The written part of the EC question should be submitted to the same assignment as the rest of your written assignment.

Due Date: This assignment is due at **11 PM Tuesday July 25**.
You will submit the written problems as a PDF to Gradescope. Please put each numbered problem on its own page of the PDF (this will make selecting pages easier when you submit), and ensure that your PDFs are oriented correctly (e.g. not upside-down or sideways). The coding problem will also be submitted to Gradescope.

Note that there will be **no leeway** with the submission time. Any submission to Gradescope after 11:00:00 PM on the deadline day will be considered late. We recommend submitting a few minutes before the deadline in order to avoid technical issues with submissions.

Collaboration: Please read the full collaboration policy. If you work with others, you must still write up your solution independently and name all of your collaborators somewhere on your assignment.

Full submission instructions are in the course syllabus.

1. Distribution Description [7 points]

Let X be a discrete random variable that takes integer values from 1 to 11 (both inclusive). X has the following cumulative distribution function (CDF):

$$F_X(n) = \begin{cases} 0 & \text{if } n < 1 \\ \frac{\lfloor (n) \rfloor \cdot \lfloor (n+2) \rfloor}{143} & \text{if } 1 \leq n \leq 11 \\ 1 & \text{if } n > 11 \end{cases}$$

Find the probability mass function (PMF) for X.

2. Turnabout is Fair Play [7 points]

You want to open a casino with a new card game. In the game, a dealer will shuffle a (full) deck of (standard) cards so they are fully-shuffled (i.e. every ordering is equally likely). The dealer then flips over the top three cards.

To play, a player pays $\$5$.

- If all three cards are spades, the player gets a payout of $\$20$ (not including the five dollars they already paid)
- If exactly two cards are spades, the player gets a payout of z
- If exactly one of the cards is a spades, the player gets a payout of $\$10$.
- If none of the cards are spades, the player gets no payout.

You want all the games in your casino to be **fair games** – the expected profit (payout minus cost) of your game should be $\$0$. What should the value of z be? Give your answer as a simplified fraction (don't worry if your number couldn't be paid out with standard U.S. currency)

3. Play It by Ear [14 points]

You have 12 pairs of earrings. 4 pairs are hoops, 5 pairs are studs, and 3 pairs are drops. Both earrings in a pair are identical (i.e., there is no difference between earrings for the left and right ear). However, each pair of earrings has a different design from the others.

A pair of earrings is "properly matched" if they are from the same set. A pair of earrings is "fashionably mismatched" if they are from different sets, but from the same type. For instance, 2 earrings from different stud designs are "fashionably mismatched". However, a stud and a hoop are not.

You (uniformly) randomly pair the earrings with each other. In each of these problems clearly define any random variables you need.

(a) What is the expected number of pairs of properly matched earrings? [7 points]

(b) What is the expected number of pairs of fashionably mismatched earrings? [7 points]

4. Better Drinks [11 points]

You just bought a set of n flavored syrups[1] to mix into coffee. Since you need the caffeine, but hate the taste of coffee, you mix in exactly four different syrup flavors into your drink.

Your coffee will taste good only if the four flavors you mix in are pairwise compatible – that is, if for every pair in the four flavors, those two are compatible. Each pair of flavors is compatible (independently) with probability p. What is the expected number of 4-flavor-combinations that will taste good?

(a) Carefully define random variables for this problem. [4 points]

(b) Calculate the expected number of 4-flavor-combinations that will taste good. [7 points]

5. Geometric Fish, Mon Dieu! [18 points]

Let X be Geometric with parameter p, let Y be Poisson with parameter λ and $Z = \max(X, Y)$. Assume that X and Y are independent. For each of the following problems, your final answers should not have summations. You may want to use the Taylor series expansion of e^x, that is, that for any x,

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

(a) (3 points) What is $\mathbb{P}(X > k)$? [Think about it from first principles]

(b) (5 points) Compute $\mathbb{P}(X > Y)$. Hint: Use the law of total probability to obtain that

$$\mathbb{P}(X > Y) = \sum_{k=0}^{\infty} \mathbb{P}(X > Y | Y = k) \cdot \mathbb{P}(Y = k).$$

(c) (5 points) Compute $\mathbb{P}(Z \geq X)$.

(d) (5 points) Compute $\mathbb{P}(Z \leq Y)$.

[1] vanilla, french vanilla, peppermint, caramel, hazelnut, pumpkin spice, etc.

6. Animal Crossing: 312 Horizons [16 points]

Answer each of the following questions. Make sure that each of your answers is **not** in the form of a summation for this problem. *In each case, include the expectation and variance of N as part of your answer.*

(a) Let $M = \frac{N^2 - N + 4}{3}$. What is the expected value of M if N equals some fixed positive integer c with probability 1? (Your answer will be a function of c.)

(b) Let $M = \frac{N^2 - N}{3}$. What is the expected value of M if N has a geometric distribution with parameter p? (Your answer will be a function of p.)

(c) What is the expected value of $M = \binom{N}{2}$ if $N = 7X + 9$, where X is a Bernoulli random variable with parameter p? (Your answer will be a function of p.)

(d) What is the expected value of $M = (N - \frac{1}{2})^2$ if $N = X_1 + X_2 + 2X_3$ where X_1, X_2, X_3 are independent Poission random variables with parameters $\lambda_1, \lambda_2, \lambda_3$? (Your answer should be a function of $\lambda_1, \lambda_2, \lambda_3$).

7. Bloom Filters [Extra Credit]

Google Chrome has a huge database of malicious URLs, but it takes a long time to do a database lookup (think of this as a typical `Set`). They want to have a quick check in the web browser itself, so a space-efficient data structure must be used. A **Bloom filter** is a **probabilistic data structure** which only supports the following two operations:

- `add(x)`: Add an element x to the structure.
- `contains(x)`: Check if an element x is in the structure. If either returns "definitely not in the set" or "could be in the set".

It does **not** support the following two operations:

- delete an element from the structure.
- return an element that is in the structure.

The idea is that we can check our Bloom filter to see if a URL is in the set. The Bloom filter is always correct in saying a URL definitely isn't in the set, but may have false positives – it may say that a URL is in the set when it isn't. Only in these rare cases does Chrome have to perform an expensive database lookup to know for sure.

Suppose that we have k **bit arrays** $t_1, \ldots, t_k$ each of length m (all entries are 0 or 1), so the total space required is only km bits or $km/8$ bytes (as a byte is 8 bits). Suppose that the universe of URL's is the set $\mathcal{U}$ (think of this as all strings with less than 100 characters), and we have k ***independent and uniform*** hash functions $h_1, \ldots, h_k : \mathcal{U} \to \{0, 1, \ldots, m-1\}$. That is, for an element x and hash function h_i, pretend $h_i(x)$ is a **discrete** Unif$[0, m-1]$ random variable. Suppose that we implement the `add` and `contains` function as follows:

Bloom Filter Operations

function initialize(k,m)
    for $i = 1, \ldots, k$: **do**
        $t_i =$ new bit array of m 0's
function add(x)
    for $i = 1, \ldots, k$: **do**
        $t_i[h_i(x)] = 1$
function contains(x)
      return $t_1[h_1(x)] == 1 \wedge t_2[h_2(x)] == 1 \wedge \cdots \wedge t_k[h_k(x)] == 1$

Refer to Section 9.4 of the textbook and the relevant lecture for more details on Bloom filters.

(a) Implement the functions `add` and `contains` in the `BloomFilter` class of `bloom.py`.

To solve this task, we have set up a corresponding edstem lesson here. Press the Mark button above the terminal to run the unit tests we have written for you. Passing these unit tests is not enough. We have written a number of different tests for the Gradescope autograder. Your score on Gradescope will be your actual score - you have unlimited attempts to submit.

(b) Let's compare this approach to using a typical `Set` data structure. Google wants to store 1 million URLs, with each URL taking (on average) 23 bytes.

- How much space (in MB, 1 MB = 1 million bytes) is required if we store all the elements in a set?
- How much space (in MB) is required if we store all the elements in a Bloom filter with $k = 10$ hash functions and $m = 800,000$ buckets? Recall that 1 byte = 8 bits.

(c) Let's analyze the time improvement as well. Let's say an average Chrome user attempts to visit 36,500 URLs in a year, only 1,000 of which are actually malicious. Suppose it takes half a second for Chrome to make a call to the database (the `Set`), and only 1 millisecond for Chrome to check containment in the Bloom filter. Suppose the false positive rate on the Bloom filter is 4%; that is, if a website is not malicious, the Bloom filter will incorrectly report it as malicious with probability 0.04. What is the time (in seconds) taken if we only use the database, and what is the *expected* time taken (in seconds) to check all 36,500 strings if we used the Bloom filter + database combination described earlier?