CSE 544, Spring 2020, Probability and Statistics for Data Science

Assignment 2: Random Variables Due: 2/24, in class

(7 questions, 70 points total)

I/We understand and agree to the following:

(a) Academic dishonesty will result in an 'F' grade and referral to the Academic Judiciary.

(b) Late submission, beyond the 'due' date/time, will result in a score of 0 on this assignment.

(write down the name of all collaborating students on the line below)

---

1. Transformation of Normal random variables **(Total 5 points)**

(a) If $X \sim Normal(\mu, \sigma^2)$ and $Y = aX + b$, with a > 0, prove that $Y \sim Normal(a\mu + b,\ (a\sigma)^2)$. (3 points)

(b) If $X$and Y are i.i.d. standard Normal RVs, show that $X + Y \sim Normal(0, 2)$. In general, the sum of any two independent Normal RVs is also a Normal. (2 points)

2. Introduction to Covariance (Total 5 points)

The covariance of two RVs X and Y is defined as: Cov(X,Y) = E[(X - E[X]) (Y - E[Y])] = E[XY] - E[X] E[Y].
Covariance of independent RVs is always zero.

(a) In an experiment, an unbiased/fair coin is flipped 3 times. Let X be the number of heads in the first two flips and Y be the number of heads in the last two flips. Calculate Cov(X,Y). (2 points)

(b) Let X be a fair 5-sided dice with face values {-5, -2, 0, 2, 5}. Let $Y = X^2$. Calculate Cov(X,Y). (2 points)

(c) Does a zero covariance imply that the RVs are independent? Justify your answer. (1 point)

3. Inequalities **(Total 10 points)**

Let X be a non-negative RV with mean μ and variance σ^2, and let t > 0 be some real number.

(a) Prove that $E[X] \geq \int_t^\infty xf(x)dx$. (3 points)

(b) Using part (a), prove that $\Pr(X > t) \leq \frac{E[X]}{t}$ (3 points)

(c) Using part (b), prove that $\Pr(|X - \mu| \geq t) \leq \frac{\sigma^2}{t^2}$ (4 points)

4. Functions of RVs **(Total 10 points)**

(a) Let $X_1, X_2, \dots, X_k$ be k independent exponential random variables with pdfs given by $f_{X_i}(x) = \lambda_i e^{-\lambda_i x},\ x \geq 0,\ \forall\, i \in \{1, 2, \dots, k\}$. Let $Z = \min\,(X_1, X_2, \dots, X_k)$.

i. Find the pdf of Z. (3 points)

ii. Find E[Z]. (1 point)

iii. Find Var(Z). (2 points)

(b) Let X and Y be two random variables with joint density function:

$f_{XY}(x, y) = \begin{cases} 2, & 0 \leq x \leq y \leq 1 \\ 0, & otherwise \end{cases}$. Find the pdf of $Z = XY$. (4 points)

5. Daenerys returns to King's Landing, almost. (Total 10 points)

In an alternate universe of Game of Thrones (or A Song of Ice and Fire, for fans of the books), Daenerys Targaryen is finally ready to leave Meereen and return to King's Landing. However, she does not know the way. From Meereen, if she goes East, she will wander around for 20 days in the Shadow Lands and return back to Meereen. If she goes West from Meereen, she will immediately arrive at the city of Mantarys. From Mantarys , she can go West by road or South via ship. If she goes South, her ship will get lost in the Smoking Sea and will be swept back to Meereen after 10 days. However, if she goes West from Mantarys, she will eventually reach King's Landing in 5 days. Let X denote the time spent by Daenerys before she reaches King's Landing. Assume that she is equally likely to take either of two paths whenever presented with a choice and has no memory of prior choices.

(a) What is E[X]? (3 points)

(b) What is Var[X]? (7 points)

(Hint: Be careful with Var[X]. You want to use conditioning.)

6. Stay Away From Stocks! (Total 15 points)

It has always been your dream to own a CMW car, which costs \$Y, but unfortunately you only have \$X, with X < Y both being positive integers. To overcome this shortage, you decide to bet on the stock market and buy shares of a stock for \$X. The stock value is known to change as a simple random walk, i.e., its value either increases or decreases by \$1 every day with probability 0.5. Assume that (i) you will liquidate your stock (convert stock into cash) once your stock reaches the target value of \$Y, and (ii) if your stock value decreases to \$0, then you can't recover (the two stopping criteria). Model the scenario as a discrete time Markov chain to answer the questions below.

(a) What is the probability that your stock value reaches the high of $\$Y$? (4 points)

(b) What is the probability that your stock value reaches low of $\$0$? (3 points)

(c) What is the expected value of your stock at the end? (2 points)

(d) Solve parts (a), (b), and (c) above via simulation (in python). Simulate the stock value as a random walk. To simulate a random walk, start with an initial value as the current state. Generate a uniform random variable $u = U[0,1]$, if $u < p$ (p is the probability of increase in stock value), then increase the current state by 1, else decrease it by one. Keep repeating this process until you meet either of the stopping criteria. To calculate the above-mentioned probability of events and the expected values, we will go with the frequentist interpretation of probability based on large number of repeated trials. $P(A) = \frac{N_A}{N}$ and $E[X] = \frac{\sum X_i}{N}$ where N_A is the number of favorable events and N is the number of trials and X_i is the outcome of i^{th} trial. So essentially you will be repeating the random walk N times ($N \gg 1$) and calculate the quantities asked for in 6(a) -- 6(c).

Submit your code via the google form link https://forms.gle/uFF4U4Th7YYAhxRn6. For this question, your code submission should include a python file, a2_6.py. The script should have a function **`[a, b, c]← rand_walk(init_val, final_high, final_low, prob, N)`** where the returned values **`a, b, c,`** are the answers for 6(a) -- 6(c), respectively, and the function arguments **`init_val`**, **`final_high`**, **`final_low`**, **`prob`** and **`N`** are initial stock value, final stock high, low values (stopping criteria), probability of upward movement, and the number of trials, respectively. For 6(d), also mention the final answers in your hardcopy submission for the following test cases in the specified format. (6 points)

<u>Test cases</u>

Case #	Init_val	Final_high	Final_low	Prob	N
1	100	150	0	0.50	10000
2	100	200	0	0.52	10000
3	200	250	0	0.54	10000

<u>Output format</u>

TEST CASE 1 >> (answer for 6(a)) (answer for 6(b)) (answer for 6(c))

TEST CASE 2 >> (answer for 6(a)) (answer for 6(b)) (answer for 6(c))

TEST CASE 3 >> (answer for 6(a)) (answer for 6(b)) (answer for 6(c))

7. Dependence on past 2 states (Total 15 points)

Consider the Clear-Snowy problem from class. However, this time, assume that the weather tomorrow depends on the weather today AND the weather yesterday. While this does not seem to follow the Markovian property, you can modify the state space to work around this issue. Use the following notation and transition probability values:

Pr[Weather tomorrow is X_{i+1}, given that weather today is X_i and weather yesterday was X_{i-1}]
= Pr[X_{i+1} | X_i X_{i-1}] (note that each X is either *c* or *s*).
Pr[c | c c] = 0.9; Pr[c | c s] = 0.8; Pr[c | s c] = 0.5; Pr[c | s s] = 0.1.

(a) Find the eventual (steady-state) Pr[c c], Pr[c s], Pr[s c], and Pr[s s]. Show your Markov chain and the transition probabilities. (7 points)

(b) In steady-state, what is the probability that it will be snowy 3 days from today. (3 points)

(c) Solve the problem of finding the steady state probability via simulation (in python). You need to find the steady state by raising the transition matrix to a high power ($\boldsymbol{\pi} = P^k; k \gg 1$) and then take any row of the exponentiated matrix ($\boldsymbol{\pi}[i, :]$) as the steady state. For taking power of matrix in python, you can use **np.linalg.matrix_power(matrix, power)**. After you obtain the steady state distribution, solve part (b) numerically. (5 points)

Submit your code via the google form link https://forms.gle/uFF4U4Th7YYAhxRn6. For this question, your code submission should include a python file, a2_7.py. The script should have a function **a ←** **`steady_state_power`** **(transition matrix)**, where **`steady_state_power`** **()** should have the implementation of Power method and the return value **a** is the final steady state. Also, in the hardcopy submission, you should mention the final steady state you obtained in the following format:

`Steady_State:` `Power iteration >> [xx, xx, xx, xx]`