Pol Sci 630: Problem Set 1 - Probability Theory and Distributions - Solutions

Prepared by: Jan Vogler (jan.vogler@duke.edu)

Grading Due Date: Friday, September 9th, 1.40 PM (Beginning of Lab)

R Programming

Problem 1

Insert your comments on the assignment that you are grading above the solution in bold and red text. For example write: "GRADER COMMENT: everything is correct!" Also briefly point out which, if any, problems were not solved correctly and what the mistake was. See below for more examples.

In order to make your text bold and red, you need to insert the following line at the beginning of the document:

```
\usepackage{color}
```

and the following lines above the solution of the specific task:

```
\textbf{\color{red} GRADER COMMENT: everything is correct!}
```

```
### a
factorial(10)

## [1] 3628800

factorial(8)
```

```
## [1] 40320
```

```
### b
factorial(15)/factorial(10)
```

```
## [1] 360360
```

```
factorial(10)/factorial(5)
```

```
## [1] 30240
```

```
### c
choose(12, 3)
```

```
## [1] 220
```

```
choose(9, 3)
```

```
## [1] 84
```

Problem 2

GRADER COMMENT: everything is correct!

```
### a
Multi = function(a, b, c) {
    if (-5 <= a & a <= 10 & -5 <= b & b <= 10 & -5 <= c & c <= 10) {
        print(a * b * c)
    } else {
        print("The values of the variables have to be between -5 and 10.")
    }
}

Multi(2, 3, 4)
```

```
## [1] 24
```

```r
Multi(-6, 3, 4)
```

```
## [1] "The values of the variables have to be between -5 and 10."
```

```r
### b
Permutation = function(n, k) {
    print(factorial(n)/factorial(n - k))
}

Permutation(n = 10, k = 8)
```

```
## [1] 1814400
```

```r
### c
DiceAverage = function(rolls) {
    die = c(1, 2, 3, 4, 5, 6)
    print(mean(sample(die, size = rolls, replace = T)))
}

DiceAverage(1000)
```

```
## [1] 3.507
```

```r
### Bonus
DiceAverage2 = function(rolls) {
    if (rolls%%1 == 0 & rolls >= 0) {
        die = c(1, 2, 3, 4, 5, 6)
        print(mean(sample(die, size = rolls, replace = T)))
    } else {
        print("The number of rolls must be a natural number")
    }
}

DiceAverage2(-10)
```

```
## [1] "The number of rolls must be a natural number"
```

```r
# Returns the error message written above.

DiceAverage2(1.5)

## [1] "The number of rolls must be a natural number"

# Returns the error message written above.

### Alternative solution for bonus question
DiceAverage3 = function(rolls) {
    if (rolls == round(rolls) & rolls >= 0) {
        die = c(1, 2, 3, 4, 5, 6)
        print(mean(sample(die, size = rolls, replace = T)))
    } else {
        print("The number of rolls must be a natural number.")
    }
}

DiceAverage3(-10)

## [1] "The number of rolls must be a natural number."

# Returns the error message written above.

DiceAverage3(1.5)

## [1] "The number of rolls must be a natural number."

# Returns the error message written above.
```

Probability Theory

Problem 3

GRADER COMMENT: everything is correct!

a) If a and b are independent events, are the following true or false?

1. True

2. False.

3. True.

b) In general, $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ and $P(A \cap B) = P(A|B)P(B)$ which, when combined, yield: $P(A \cup B) = P(A) + P(B) - P(A|B)P(B)$. If the two events are independent then $P(A|B) = P(A)$, giving $P(A \cup B) = P(A) + P(B) - P(A)P(B) = P(B)(1 - P(A)) + P(A)$.

We solve for $P(B)$ to get $P(B) = \frac{P(A \cup B) - P(A)}{1 - P(A)} = \frac{0.5 - 0.3}{1 - 0.3} = \frac{2}{7}$.

c) A committee contains fifteen legislators with ten men and five women. Find the number of ways that a delegation of six:

1. This is the number of ways 6 elements can be chosen from 15, or $\binom{15}{6}$.

2. Now we have the joint probability of two independent events: choosing 3 women from 5 and 3 men from 10. This is: $\binom{10}{3}\binom{5}{3}$.

3. Finally, we have the joint probability of two independent events: choosing 2 women from 5 and 4 men from 10, since there are twice as many men as women in the full group. This is: $\binom{10}{4}\binom{5}{2}$.

Problem 4

GRADER COMMENT: everything is correct!

a) Each time a fundraiser is observed or not observed, the legislator should again update his beliefs, in each case using posterior beliefs at the end of the previous month as his priors in the present month. Further, as $Pr(f|I_r)$ increases, the degree to which observing a fundraiser (or not observing one) is informative increases as well. (Note that is need not be perfectly informative as $Pr(f|I_r)$ approaches 1, however, as $Pr(f| \sim I_r)$ is logically independent of $Pr(f|I_r)$ and need not change as $Pr(f|I_r)$ changes.) We assume that the legislator's posterior belief is 0.6 at the end of December, as in the problem in Section 9.2.3. After observing no fundraiser in January, his posterior belief is $\Pr(I_r| \sim f) = \frac{Pr(\sim f|I_r)Pr(I_r)}{Pr(\sim f|I_r)Pr(I_r)+Pr(\sim f|\sim I_r)Pr(\sim I_r)} = \frac{(0.3)(0.6)}{(0.3)(0.6)+(0.6)(0.4)} = 0.43$. If he now sees one in February, this changes to $\Pr(I_r|f) = \frac{Pr(f|I_r)Pr(I_r)}{Pr(f|I_r)Pr(I_r)+Pr(f|\sim I_r)Pr(\sim I_r)} = \frac{(0.8)(0.43)}{(0.8)(0.43)+(0.4)(0.57)} = 0.6$. Finally, if he does not see one in March, his posterior is $\Pr(I_r| \sim f) = \frac{Pr(\sim f|I_r)Pr(I_r)}{Pr(\sim f|I_r)Pr(I_r)+Pr(\sim f|\sim I_r)Pr(\sim I_r)} = \frac{(0.1)(0.6)}{(0.1)(0.6)+(0.6)(0.4)} = 0.2$. So two positive signals and two negative ones drop the final posterior below the prior because later signals are more informative.

b) There are 36 possible outcomes for the dice rolls of player A and player B. All these outcomes are equally likely. There are several ways to compute the probability of player A having a strictly greater number than player B. One possible way is the following:

The probability of player B to roll any number from 1 to 6 is $\frac{1}{6}$, i.e. each possible number occurs with probability $\frac{1}{6}$.

If player B rolls a 1, player A can beat him with five different outcomes, i.e. the numbers from 2 to 6. This implies that the likelihood of beating him is $\frac{5}{6}$.

If player B rolls a 2, player A can beat him with four different outcomes, i.e. the numbers from 3 to 6. This implies that the likelihood of beating him is $\frac{4}{6}$.

Following this logic and applying it to all outcomes, player A will beat player B with the following probability:

$\sum_{i=1}^{6} Pr(B \text{ rolling } i) * Pr(A \text{ beating } B|B \text{ rolling } i)$

$= \frac{1}{6} * \frac{5}{6} + \frac{1}{6} * \frac{4}{6} + \frac{1}{6} * \frac{3}{6} + \frac{1}{6} * \frac{2}{6} + \frac{1}{6} * \frac{1}{6} + \frac{1}{6} * \frac{0}{6} = \frac{15}{36} = \frac{5}{12}$

This is the probability for player A to win a single game against B.

We have to model the probability that player A will win four games against player B with a binomial distribution.

$\binom{5}{4}(\frac{5}{12})^4(\frac{7}{12})^1 = 0.088$

Additionally, player A could also win all five games against player B, so you have to add a second probability.

$\binom{5}{5}(\frac{5}{12})^5(\frac{7}{12})^0 = 0.012$

The total probability of both outcomes is approximately 0.1 or 10 percent.

Problem 5 (Bonus Problem)

GRADER COMMENT: everything is correct!

Let's write a function to represent the Monty Hall problem. What is the proportion of successful trials, for any number of trials n, if you always switch to the other door when Monty reveals an empty one? The theoretical expectation derived from the calculation in the lecture was 2/3. The following function returns an empirical proportion of successes for any number of trials.

```
Switching = function(trials) {
    successes = rep(0, trials)
    # Create a null vector with the length of the number of trials
    for (i in 1:trials) {
        # For every trial, indexed by 'i', do the following
        prizeoptions = c(1, 2, 3)
        # The prize can be located behind door 1, 2, or 3
        prize = sample(prizeoptions, size = 1)
        # Draw a random location of the prize
        doorchosen = sample(prizeoptions, size = 1)
        # Choose a door at random
        if (doorchosen == prize) {
            # If the door you chose is the same door as the door of the prize
            doorshown = sample(prizeoptions[-prize], size = 1)
            # Monty will sample between the other two doors and show you one at random
        } else {
            # If the door you chose is not the door with the prize
```

```
            doorshown = prizeoptions[-c(prize, doorchosen)]
            # Monty has to show you the third door that is empty
        }
        switchtodoor = prizeoptions[-c(doorshown, doorchosen)]
        # You will always switch to the door that is NOT the door you chose
        # originally and NOT the door that Monty has revealed to be empty
        if (switchtodoor == prize) {
            # If you switched to the correct door
            successes[i] = 1
            # The entry at position 'i' of the successes vector will be recoded to 1
        }
    }
    print(sum(successes)/trials)
    # Eventually we will calculate the sum of successes and divide it by the
    # number of trials
}

Switching(1000)

## [1] 0.653

# Returns approximately 0.68.
```