Lab 4: Probability Practice

Felicia Liu

09/23/2020

- Due date: Friday, September 25 at 11:59pm PST.
- Late penalty: 50% late penalty if submitted within 24 hours of due date, no marks for assignments submitted thereafter.
- This assignment is graded on **correct completion**, all or nothing. You must pass all public tests and submit the assignment for credit.
- Submission process: Follow the submission instructions on the final page. Make sure you do not remove any `\newpage` tags or rename this file, as this will break the submission.

Helpful hints:

- Every function you need to use was taught during lecture! So you may need to revisit the lecture code to help you along by opening the relevant files on Datahub. Alternatively, you may wish to view the code in the condensed PDFs posted on the course website. Good luck!
- Knit your file early and often to minimize knitting errors! If you copy and paste code for the slides, you are bound to get an error that is hard to diagnose. Typing out the code is the way to smooth knitting! We recommend knitting your file each time after you write a few sentences/add a new code chunk, so you can detect the source of knitting errors more easily. This will save you and the GSIs from frustration! **You must knit correctly before submitting.**
- If your code runs off the page of the knitted PDF then you will LOSE POINTS! To avoid this, have a look at your knitted PDF and ensure all the code fits in the file (you can easily view it on Gradescope via the provided link after submitting). If it doesn't look right, go back to your .Rmd file and add spaces (new lines) using the return or enter key so that the code runs onto the next line.

Instructions

This lab focuses on calculating basic probability and understanding these concepts.

Section I: Group Discussion Questions

We will briefly review probability concepts and probability coding in R before jumping into applied problems from Baldi and Moore.

Discussion Question 1

Probability is a measure of how likely an event is to occur. Match each of the probabilities that follow with each statement of likelihood given. The probability is usually a more exact measure of likelihood than is the verbal statement.

- a) 0
- b) 0.001
- c) 0.3
- d) 0.6
- e) 0.99
- f) 1

Map the following statements to a probability value above:

- This event is unlikely.
- This event is impossible, it can never occur.
- This event will almost always occur.
- This event will occur more often than not.
- This event will always occur.
- This event will very rarely occur.

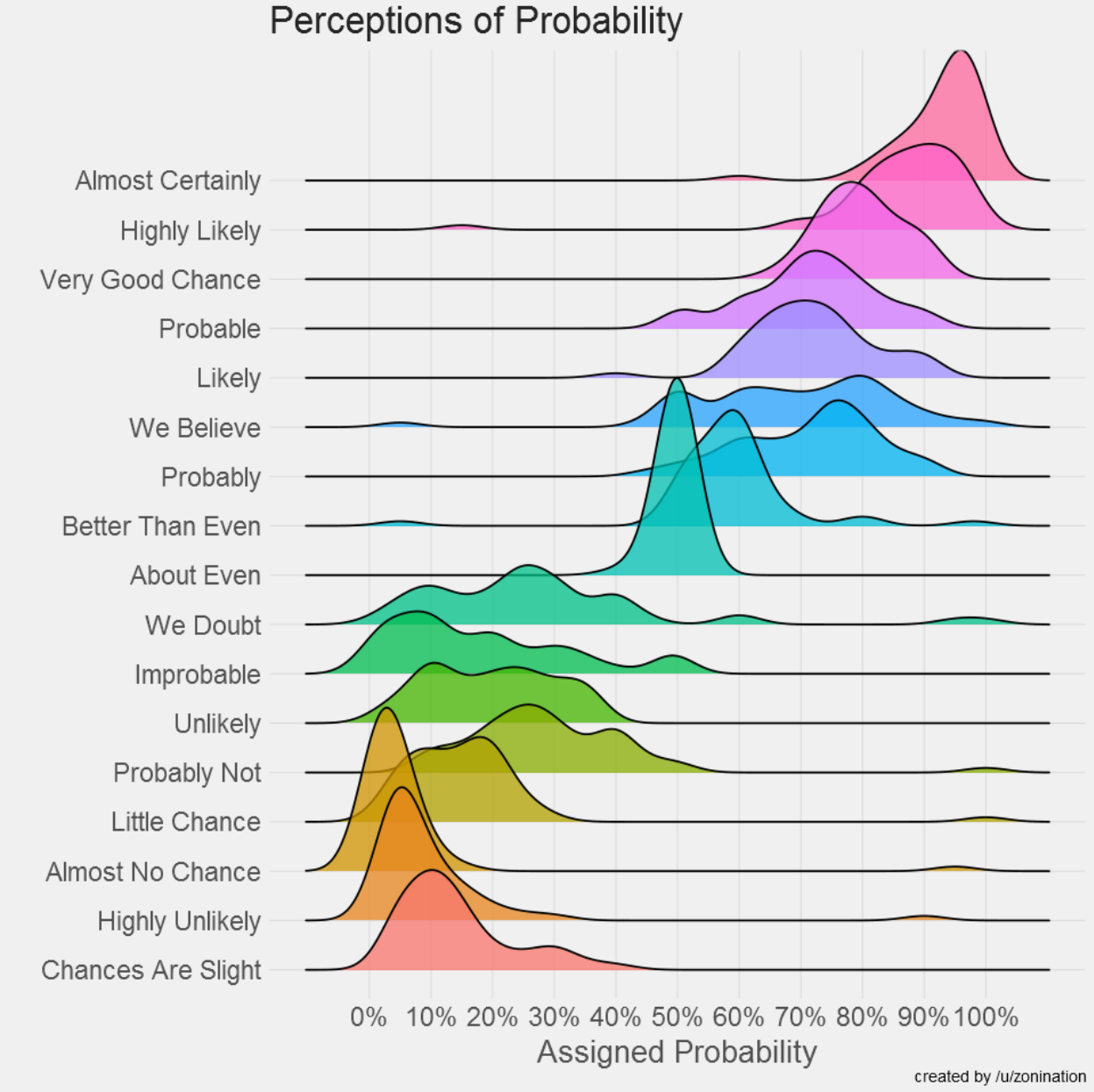
```
a <- "This event is impossible, it can never occur."
b <- "This event will very rarely occur."
c <- "This event is unlikely."
d <- "This event will occur more often than not."
e <- "This event will almost always occur."
f <- "This event will always occur."


check_problem1()
```

```
## [1] "Checkpoint 1 Passed: Correct, this event is impossiple"
## [1] "Checkpoint 2 Passed: Correct, this event will very rarely occur"
## [1] "Checkpoint 3 Passed: Correct, this event is unlikely"
## [1] "Checkpoint 4 Passed: Correct, this event will occur more often than not"
## [1] "Checkpoint 5 Passed: Correct, this event will almost always occur"
## [1] "Checkpoint 6 Passed: Correct, this event will always occur"
##
## Problem 1
## Checkpoints Passed: 6
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Discussion Question 2

Consider this image of perceptions of probability. This image is the smoothed distribution of how individuals perceive the probability associated with different statements. Those a wide distribution implies that different people associate the statement with different levels of probability, and a narrow probability implies that different individuals tend to associate the statement with probability levels that are more similar to one another.



Anything unexpected? Which terms have relatively narrow distributions? Which are relatively wider?

I noticed that there seem to be more wider distributions for the terms than there are narrow distributions. The terms that have relatively narrow distributions are: very good chance, probable, about even, almost no chance, highly unlikely, and chances are slight. The relatively wider distributions are: almost certainly, highly likely, likely, we believe, probably, better than even, we doubt, improbable, unlikely, probably not, and little chance.

Discussion Question 3-7

In this question we learn about the R code need to simulate events. One reason we simulate events is to calculate the probability of something happening over shorter and longer periods of time.

The easiest event to simulate is a coin flip. We call a coin "fair" if there is a 50% chance of landing on heads and an equal chance of landing on tails. We start with coin flips because it is a *binary* outcome. In public health, many events of interest are binary, like the occurrence of a specific disease or death.

Here is the code to clip a fair coin one time.

- Run the code over and over and see what happens.
- Do you get the same output as your neighbor?

```
# This is like flipping a fair coin one time. Because the coin is fair,
# there is a 50% chance of flipping heads
rbinom(n = 1, size = 1, prob = 0.5)
```

```
## [1] 0
```

Rather than running the above code over and over, we can change the `n` argument to a number >1 to simulate more than one flip. Give it a try:

```
# This is like flipping the fair coin 100 times.
one_hundred_flips <- rbinom(n = 100, size = 1, prob = 0.5)
one_hundred_flips
```

```
##   [1] 1 0 1 0 0 1 1 1 0 1 1 1 1 0 1 0 0 1 1 0 1 0 1 1 1 0 1 0 1 1 0 0 0 1 1 0 0
##  [38] 0 1 1 0 0 0 1 0 0 1 1 0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 0 1 1 0 0
##  [75] 0 1 1 0 1 0 1 1 1 0 0 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1
```

```
sum(one_hundred_flips) #this takes the summation of all the values (i.e., it counts to 1's)
```

```
## [1] 48
```

```
# note: We can't use `summarize()` to calculate the sum() because `one_hundred_flips` is not a
# data frame and dplyr functions can only run computations on data frames.
```

3. How many 1's do you expect to get when you flip this coin 100 times? You expect to get 50 1's because there is a 50% chance of getting a 1 and 50% chance of getting a 0 when the coin is flipped. Out of 100, you should get 50 of each.

4. How many 1's did you get?

I got 54 1's.

This is equal to `sum(one_hundred_flips)` from above.

5. Flip the coin 100 times again and assign it to a new variable.

```
one_hundred_flips_2 <- rbinom(n = 100, size = 1, prob = 0.5)

one_hundred_flips_2
```

```
##   [1] 0 1 1 0 1 1 1 1 1 1 0 0 1 0 0 0 1 0 1 0 0 0 0 0 1 1 0 1 0 1 1 1 1 0 1 1 1
##  [38] 0 1 0 0 1 0 1 1 1 1 0 1 0 0 0 1 0 1 0 0 1 1 0 0 1 1 1 1 0 1 1 1 1 1 0 1 1
##  [75] 1 0 0 1 1 0 1 0 0 0 1 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0
```

```
sum(one_hundred_flips_2)
```

```
## [1] 54
```

```
check_problem5()
```

```
## [1] "Checkpoint 1 Passed: Correct! They are integers"
## [1] "Checkpoint 2 Passed: Correct! Sample size is 100"
## [1] "Checkpoint 3 Passed: Correct! Outputs are values of zero and one, it's binary"
##
## Problem 5
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

6. How many 1's do you get? Is it the same as your previous 100 flips? I got 51 1's and this is not the same as my previous 100 flips.

In public health, we are often interested in binary events that are uncommon or rare, like the risk of a disease or death. We can still use this code to simulate these events by changing the `prob` argument to the risk that the outcome of interest occurs. For example, setting `prob = 0.05` is like setting the risk of the event to 5%.

Rather than coin flips, `n` could represent people in a population, or participants in a research study.

Try changing the probability and seeing how the results vary.

```
rbinom(n = 100, size = 1, prob = 1)
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
rbinom(n = 100, size = 1, prob = 0)
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
rbinom(n = 100, size = 1, prob = 0.6)
```

```
##   [1] 1 1 1 1 0 1 0 1 1 1 0 0 1 0 1 1 0 0 1 0 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 0 1
##  [38] 1 0 0 1 1 1 1 0 1 0 1 0 1 0 1 0 1 1 1 1 1 0 0 0 1 0 0 1 1 1 1 1 1 1 1 0 1
##  [75] 1 1 0 0 1 0 0 1 0 1 0 1 0 1 1 1 1 0 0 1 0 0 0 1 0 0
```

```
rbinom(n = 100, size = 1, prob = 0.03)
```

```
##   [1] 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [75] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

7. Think of an event in public health that is relatively common (occurs roughly half of the time). Think of an event that is relatively rare (occurs $< 5\%$ of the time) Event that is relatively common: Air pollution, event that is relatively rare: Huntington disease

We will use the `rbinom` functions and related functions from the **binomial** family once we cover chapter 12.

Section II:

Example 9.5 "Blood Types" from page 220 of Edition 4 of Baldi & Moore

A person's blood type determines the kind of blood transfusion or organ transplant they can safely get. There are 8 different blood types based on the presence or absence of certain molecules on the surface of red blood cells. A person's blood type is given as a combination of a blood group (one of: O, A, B, or AB) and a Rhesus factor (either + or -).

Together, the blood group and Rhesus factor define the sample space S for the variable blood type.

8. Write out the sample space for blood type

S = {O+, O-, A+, A-, B+, B-, AB+, AB-}

Blood type possibilities

Once we have a sample space, we need to determine the probability associated with each event in the sample space. For blood types, the proportion of the population with each blood type varies by race and ethnicity. Within a given race/ethnic group, we can use the blood types' frequencies in that group to assign their respective probabilities. The American Red Cross reports that among Asian Americans there are:

39% blood type O+
1% blood type O-
27% blood type A+
0.5% blood type A-
25% blood type B+
0.4% blood type B-
7% blood type AB+
0.1% blood type AB-

Because 39% of all Asian Americans have blood type O+, the probability that a randomly chosen Asian American has blood type O+ is 39%, or 0.39.

Write out the probability model for blood type for Asian Americans. You can write this out by hand on a piece of paper. In R markdown, you can make a table using the following template.

Group	O+	O-	A+	A-	B+	B-	AB+	AB-
Probability	39%	1%	27%	0.5%	25%	0.4%	7%	0.1%

Reminders:

- A **probability model** is a mathematical description of a random phenomenon consisting of two parts: a sample space S and a way of assigning probabilities to events.
- An **event** is an outcome or a set of outcomes of a random phenomenon. That is, an event is a subset of the sample space.

9. What is the probability that blood type is equal to A+?
Using notation, calculate P(Blood type = A+)

P(Blood type = A+) = 27%

10. What is the probability that blood type is not equal to A+?

Using notation, calculate P(Blood type $\neq A+$)? Note: you can hover with your mouse over the text inside the dollar signs to see the equation. When you knit this file, the code inside the dollar signs will compile to show a not equals to sign.

P(Blood type $\neq A+$) = 100% -P(Blood type = A+) = 100% - 27% = 73%

11. What is P(Blood type = O+ or blood type = O-)?

Here we can apply the **addition rule for disjoint events.**

This states that two events A and B are **disjoint** (mutually exclusive) if they have no outcomes in common and so can never occur together.

If A and B are disjoint, **P(A or B) = P(A) + P(B)**.

P(Blood type = O+ or blood type = O-) = P(Blood type = O+) + P(blood type = O-) = 39% + 1% = 40%

Rhesus factor

Using the probability model for Asian American blood types, write out a new probability model for an Asian American's Rhesus Factor.

12. First, what is the sample space for Rhesus factor?

S = {+, -}

13. Write out the probability model for a Rhesus factor

Group	+	-
Probability	98%	2%

P(+) = P(O+) + P(A+) + P(B+) + P(AB+) = 39% + 27% + 25% + 7% = 98%

P(-) = P(O-) + P(A-) + P(B-) + P(AB-) = 1% + 0.5% + 0.4% + 0.1% = 2%

Recall the following rules of probability:

1. Any probability is a number between 0 and 1, inclusively.
2. All possible outcomes together must have probability 1.
3. If two events have no outcomes in common, the probability that one or the other occurs is the sum of their individual probabilities.
4. The probability that an event does not occur is 1 minus the probability that the event does occur.

Continuous Probability Model

In the previous question about blood types and Rhesus factors, we created probability models and applied probability rules to calculate the chance of events or combinations of events. This was an example of a **discrete** probability space, because the variables "blood type" and "Rhesus factor" were categorical events.

Another type of probability model is a **continuous probability model**. For continuous models, we most often use data visualization to plot the model and as a tool for calculating the probability of specific events.

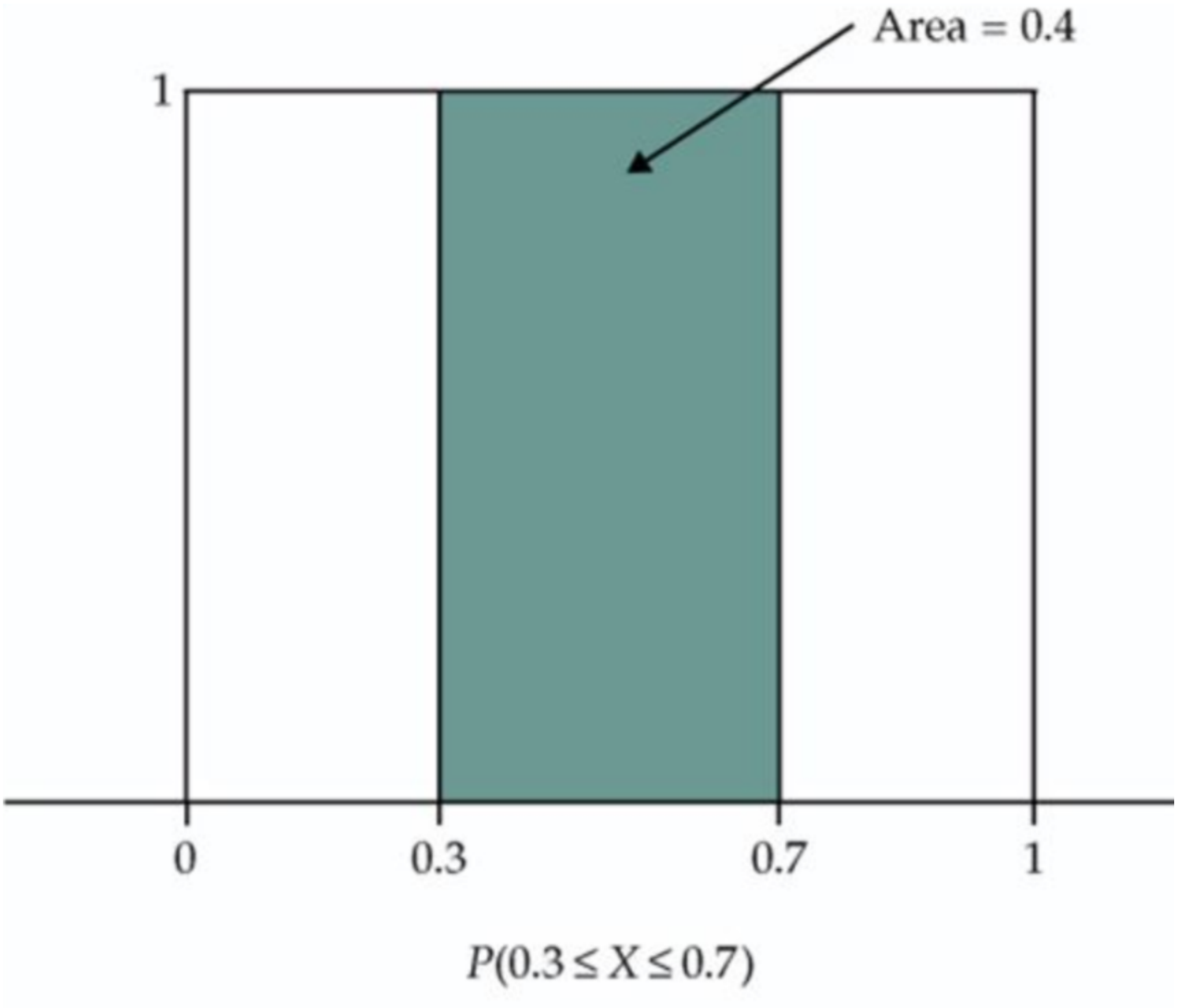
The simplest continuous probability model is the uniform distribution. A uniform distribution between the numbers 0 and 1 randomly chooses a number between 0 and 1 with equal probability. To sample a random number for this distribution we use the following code:

```
runif(n = 1, min = 0, max = 1)
```

```
## [1] 0.9565678
```

The following plot shows the uniform distribution between 0 and 1. Specifically, this image shows how to calculate the probability that a randomly chosen number from this distribution is between 0.7 and 0.3.

```
knitr::include_graphics("D04_uniform_density.png")
```



Using the uniform distribution, find the following probabilities. You can draw a sketch and shade the area you are calculating if that helps you:

14. Probability that a random variable X is less than or equal to 0.4, or P (X $\leq$ 0.4). Write your answer as a number between 0 and 1.

```
p14 <- 0.4

p14
```

```
## [1] 0.4
```

```
check_problem14()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p14= 0.4"
##
## Problem 14
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

15. Probability that a random variable X is less than 0.4, P (X $<$ 0.4). Write your answer as a number between 0 and 1.

```
p15 <- 0.4

p15
```

```
## [1] 0.4
```

```
check_problem15()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p15= 0.4"
##
## Problem 15
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

16. Probability that X is greater than or equal to 0.3 and less than or equal to 0.5, or P (0.3 $\leq$ X $\leq$ 0.5). Write your answer as a number between 0 and 1.

```
p16 <- 0.5-0.3


p16
```

```
## [1] 0.2
```

```
check_problem16()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
```

```
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p16= 0.2"
##
## Problem 16
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

17. Probability that X is less than 0.3 or greater than 0.5, or P(X $<$ 0.3 or X $>$ 0.5). Write your answer as a number between 0 and 1.

```
p17 <- 1-(0.5-0.3)


p17
```

```
## [1] 0.8
```

```
check_problem17()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p17= 0.8"
##
## Problem 17
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

How does your answer for 17 compare to 18.

Fill in the blank: The probability of any **individual** value under a continuous distribution is always *p18*.

```
p18 <- 0

p18
```

```
## [1] 0
```

```
check_problem18()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p18= 0"
##
## Problem 18
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Note: To calculate a probability under a continuous distribution, you need to calculate the area under the density curve, either below or above a specific point or between two given values.

Calculating probabilities in R for the uniform distribution

We can calculate the previous probabilites using `punif()` in R. The `p` stands for probability and `unif` for uniform.

Note how the `runif()` function generates a **random** value from the **uniform** distribution.

`punif()` is a different type of function: it calculates the probability **below** the specified point (from the uniform density curve).

P(X less than or equal to 0.4), or $P(X \leq 0.4)$:

```
# Finds the probability below the point 0.4 from the uniform density curve
punif(0.4, min = 0, max = 1)
```

P(X greater than or equal to 0.4), or $P(X \geq 0.4)$:

```
# Finds the probability at or above the point 0.4 from the uniform density curve
punif(0.4, min = 0, max = 1, lower.tail = F)
```

Question 19: What does $P(X \leq 0.4)$ equal?

$P(X \leq 0.4) = 0.4$

Question 20: What does $P(X \geq 0.4)$ equal? Make a sketch on paper of the area being calculated. You do not need to upload this image.

$P(X \geq 0.4) = 0.6$

Make a sketch on paper shading the area under the uniform density curve that each of the following equations represent. You do not need to turn in your sketches. Then, use R to calculate the probabilities:

21. $P(0.3 \leq X \leq 0.5)$

```
#write your R code here.
p21 <- punif(0.5, min = 0, max = 1) - punif(0.3, min = 0, max = 1)

p21
```

```
## [1] 0.2
```

```
check_problem21()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p21= 0.2"
##
## Problem 21
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

22. $P(X < 0.3 | X > 0.5)$

```
p22 <- punif(0.3, min = 0, max = 1) + punif(0.5, min = 0, max = 1, lower.tail = F)

p22
```

```
## [1] 0.8
```

```
check_problem22()
```

```
## [1] "Checkpoint 1 Passed: Correct! It's numeric"
## [1] "Checkpoint 2 Passed: Correct! It takes value between 0 and 1"
## [1] "Checkpoint 3 Passed: Correct! p22= 0.8"
##
## Problem 22
## Checkpoints Passed: 3
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

Check your score

Click on the middle icon on the top right of this code chunk (with the downwards gray arrow and green bar) to run all your code in order. Then, run this chunk to check your score.

```
# Just run this chunk.
total_score()
```

```
##                      Test Points_Possible          Type
## Problem 1          PASSED               1    autograded
## Problem 2  NOT YET GRADED               1 free-response
## Problem 3  NOT YET GRADED               1 free-response
## Problem 4  NOT YET GRADED               1 free-response
## Problem 5          PASSED               1    autograded
## Problem 6  NOT YET GRADED               1 free-response
## Problem 7  NOT YET GRADED               1 free-response
## Problem 8  NOT YET GRADED               1 free-response
## Problem 9  NOT YET GRADED               1 free-response
## Problem 10 NOT YET GRADED               1 free-response
## Problem 11 NOT YET GRADED               1 free-response
## Problem 12 NOT YET GRADED               1 free-response
## Problem 13 NOT YET GRADED               1 free-response
## Problem 14         PASSED               1    autograded
## Problem 15         PASSED               1    autograded
## Problem 16         PASSED               1    autograded
## Problem 17         PASSED               1    autograded
## Problem 18         PASSED               1    autograded
## Problem 19 NOT YET GRADED               1 free-response
## Problem 20 NOT YET GRADED               1 free-response
## Problem 21         PASSED               1    autograded
## Problem 22         PASSED               1    autograded
```

Submission

For assignments in this class, you'll be submitting using the **Terminal** tab in the pane below. In order for the submission to work properly, make sure that:

1. Any image files you add that are needed to knit the file are in the `src` folder and file paths are specified accordingly.
2. You **have not changed the file name** of the assignment.
3. The file is saved (the file name in the tab should be **black**, not red with an asterisk).
4. The file knits properly.

Once you have checked these items, you can proceed to submit your assignment.

1. Click on the **Terminal** tab in the pane below.
2. Copy-paste the following line of code into the terminal and press enter.

cd; cd ph142-fa20/lab/lab04; python3 turn_in.py

3. Follow the prompts to enter your Gradescope username and password. When entering your password, you won't see anything come up on the screen–don't worry! This is just for security purposes–just keep typing and hit enter.
4. If the submission is successful, you should see "Submission successful!" appear as output.
5. If the submission fails, try to diagnose the issue using the error messages–if you have problems, post on Piazza.

The late policy will be strictly enforced, **no matter the reason**, including submission issues, so be sure to submit early enough to have time to diagnose issues if problems arise.