Lab3_Probability

Heleine Fouda

2023-09-24

The Hot Hand

Basketball players who make several baskets in succession are described as having a *hot hand*. Fans and players have long believed in the hot hand phenomenon, which refutes the assumption that each shot is independent of the next. However, a 1985 paper by Gilovich, Vallone, and Tversky collected evidence that contradicted this belief and showed that successive shots are independent events. This paper started a great controversy that continues to this day, as you can see by Googling *hot hand basketball*.

We do not expect to resolve this controversy today. However, in this lab we'll apply one approach to answering questions like this. The goals for this lab are to (1) think about the effects of independent and dependent events, (2) learn how to simulate shooting streaks in R, and (3) to compare a simulation to actual data in order to determine if the hot hand phenomenon appears to be real.

Getting Started

Load packages

In this lab, we will explore and visualize the data using the `tidyverse` suite of packages. The data can be found in the companion package for OpenIntro labs, **openintro**.

Let's load the packages.

```
library(tidyverse)
library(openintro)
```

Data

Your investigation will focus on the performance of one player: Kobe Bryant of the Los Angeles Lakers. His performance against the Orlando Magic in the 2009 NBA Finals earned him the title *Most Valuable Player* and many spectators commented on how he appeared to show a hot hand. The data file we'll use is called `kobe_basket`.

```
glimpse(kobe_basket)
```

```
## Rows: 133
## Columns: 6
## $ vs          <fct> ORL, ORL, ORL, ORL, ORL, ORL, ORL, ORL, ORL, ORL, ORL, ORL~
## $ game        <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ quarter     <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3~
## $ time        <fct> 9:47, 9:07, 8:11, 7:41, 7:03, 6:01, 4:07, 0:52, 0:00, 6:35~
## $ description <fct> Kobe Bryant makes 4-foot two point shot, Kobe Bryant misse~
## $ shot        <chr> "H", "M", "M", "H", "H", "M", "M", "M", "M", "H", "H", "H"~
```

This data frame contains 133 observations and 6 variables, where every row records a shot taken by Kobe Bryant. The `shot` variable in this dataset indicates whether the shot was a hit (`H`) or a miss (`M`).

Just looking at the string of hits and misses, it can be difficult to gauge whether or not it seems like Kobe was shooting with a hot hand. One way we can approach this is by considering the belief that hot hand shooters tend to go on shooting streaks. For this lab, we define the length of a shooting streak to be the *number of consecutive baskets made until a miss occurs.*

For example, in Game 1 Kobe had the following sequence of hits and misses from his nine shot attempts in the first quarter:

H M | M | H H M | M | M | M

You can verify this by viewing the first 9 rows of the data in the data viewer.

Within the nine shot attempts, there are six streaks, which are separated by a "|" above. Their lengths are one, zero, two, zero, zero, zero (in order of occurrence).

1. What does a streak length of 1 mean, i.e. how many hits and misses are in a streak of 1? What about a streak length of 0?

Insert your answer here

Counting streak lengths manually for all 133 shots would get tedious, so we'll use the custom function `calc_streak` to calculate them, and store the results in a data frame called `kobe_streak` as the `length` variable.

```
kobe_streak <- calc_streak(kobe_basket$shot)
kobe_streak
```

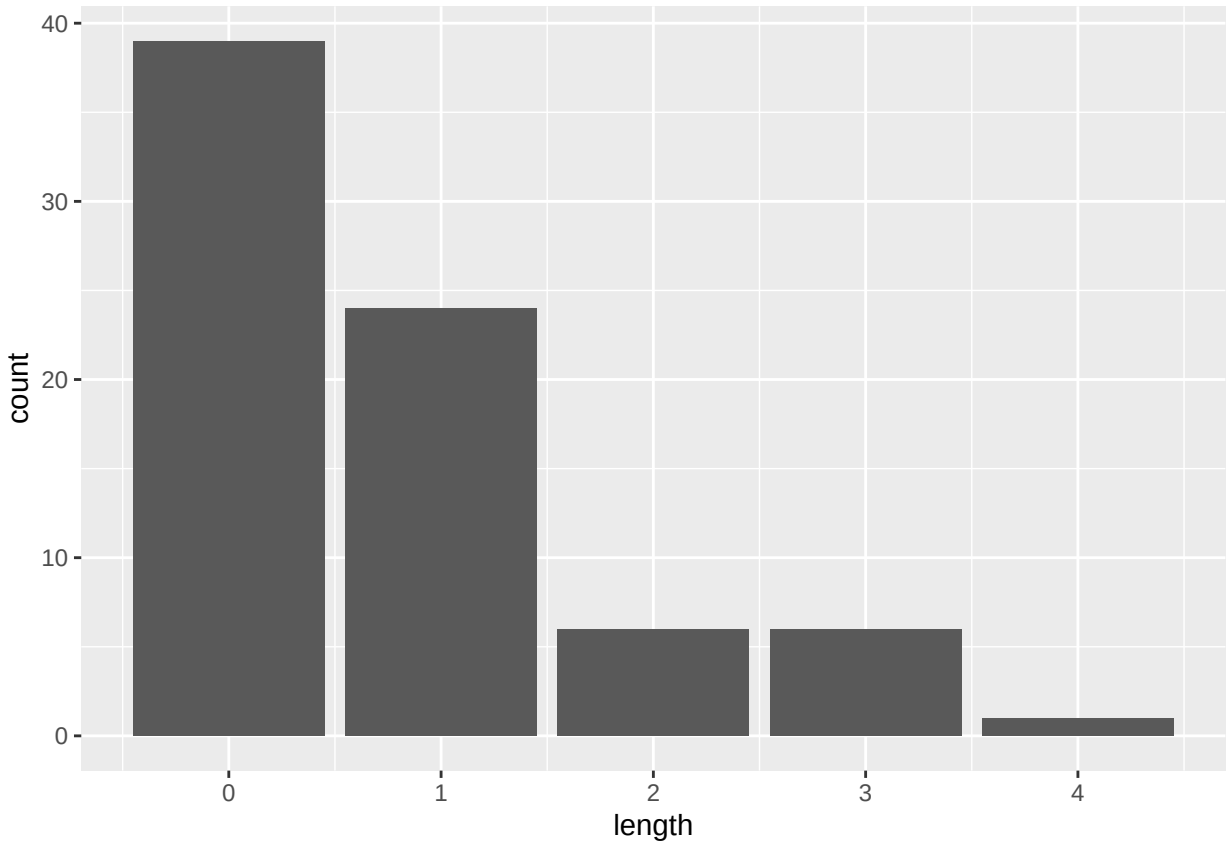
```
##    length
## 1       1
## 2       0
## 3       2
## 4       0
## 5       0
## 6       0
## 7       3
## 8       2
## 9       0
## 10      3
## 11      0
## 12      1
## 13      3
## 14      0
## 15      0
## 16      0
## 17      0
## 18      0
## 19      1
## 20      1
## 21      0
## 22      4
## 23      1
## 24      0
## 25      1
## 26      0
## 27      1
## 28      0
## 29      1
```

```
## 30      2
## 31      0
## 32      1
## 33      2
## 34      1
## 35      0
## 36      0
## 37      1
## 38      0
## 39      0
## 40      0
## 41      1
## 42      1
## 43      0
## 44      1
## 45      0
## 46      2
## 47      0
## 48      0
## 49      0
## 50      3
## 51      0
## 52      1
## 53      0
## 54      1
## 55      2
## 56      1
## 57      0
## 58      1
## 59      0
## 60      0
## 61      1
## 62      3
## 63      3
## 64      1
## 65      1
## 66      0
## 67      0
## 68      0
## 69      0
## 70      0
## 71      1
## 72      1
## 73      0
## 74      0
## 75      0
## 76      1
```

We can then take a look at the distribution of these streak lengths.

```
library(ggplot2)

ggplot(data = kobe_streak, aes(x = length)) +
  geom_bar()
```



2. Describe the distribution of Kobe's streak lengths from the 2009 NBA finals. What was his typical streak length? How long was his longest streak of baskets? Make sure to include the accompanying plot in your answer.

Insert your answer here

```
summary(kobe_streak)
```

```
##      length
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.7632
##  3rd Qu.:1.0000
##  Max.   :4.0000
```

Compared to What?

We've shown that Kobe had some long shooting streaks, but are they long enough to support the belief that he had a hot hand? What can we compare them to?

To answer these questions, let's return to the idea of *independence*. Two processes are independent if the outcome of one process doesn't effect the outcome of the second. If each shot that a player takes is an independent process, having made or missed your first shot will not affect the probability that you will make or miss your second shot.

A shooter with a hot hand will have shots that are *not* independent of one another. Specifically, if the shooter makes his first shot, the hot hand model says he will have a *higher* probability of making his second shot.

Let's suppose for a moment that the hot hand model is valid for Kobe. During his career, the percentage of time Kobe makes a basket (i.e. his shooting percentage) is about 45%, or in probability notation,

$$P(\text{shot } 1 = \text{H}) = 0.45$$

If he makes the first shot and has a hot hand (*not* independent shots), then the probability that he makes his second shot would go up to, let's say, 60%,

$$P(\text{shot } 2 = \text{H} \,|\, \text{shot } 1 = \text{H}) = 0.60$$

As a result of these increased probabilites, you'd expect Kobe to have longer streaks. Compare this to the skeptical perspective where Kobe does *not* have a hot hand, where each shot is independent of the next. If he hit his first shot, the probability that he makes the second is still 0.45.

$$P(\text{shot } 2 = \text{H} \,|\, \text{shot } 1 = \text{H}) = 0.45$$

In other words, making the first shot did nothing to effect the probability that he'd make his second shot. If Kobe's shots are independent, then he'd have the same probability of hitting every shot regardless of his past shots: 45%.

Now that we've phrased the situation in terms of independent shots, let's return to the question: how do we tell if Kobe's shooting streaks are long enough to indicate that he has a hot hand? We can compare his streak lengths to someone without a hot hand: an independent shooter.

Simulations in R

While we don't have any data from a shooter we know to have independent shots, that sort of data is very easy to simulate in R. In a simulation, you set the ground rules of a random process and then the computer uses random numbers to generate an outcome that adheres to those rules. As a simple example, you can simulate flipping a fair coin with the following.

```
coin_outcomes <- c("heads", "tails")
sample(coin_outcomes, size = 1, replace = TRUE)
```

```
## [1] "heads"
```

The vector `coin_outcomes` can be thought of as a hat with two slips of paper in it: one slip says `heads` and the other says `tails`. The function `sample` draws one slip from the hat and tells us if it was a head or a tail.

Run the second command listed above several times. Just like when flipping a coin, sometimes you'll get a heads, sometimes you'll get a tails, but in the long run, you'd expect to get roughly equal numbers of each.

If you wanted to simulate flipping a fair coin 100 times, you could either run the function 100 times or, more simply, adjust the `size` argument, which governs how many samples to draw (the `replace = TRUE` argument indicates we put the slip of paper back in the hat before drawing again). Save the resulting vector of heads and tails in a new object called `sim_fair_coin`.

```
sim_fair_coin <- sample(coin_outcomes, size = 100, replace = TRUE)
sim_fair_coin
```

```
##   [1] "tails" "tails" "tails" "heads" "tails" "heads" "heads" "heads" "tails"
##  [10] "heads" "tails" "heads" "heads" "tails" "heads" "tails" "tails" "heads"
##  [19] "tails" "heads" "heads" "tails" "heads" "tails" "heads" "heads" "heads"
##  [28] "heads" "heads" "tails" "heads" "tails" "tails" "tails" "tails" "heads"
##  [37] "heads" "tails" "heads" "tails" "heads" "tails" "tails" "heads" "heads"
##  [46] "heads" "tails" "tails" "heads" "heads" "heads" "heads" "tails" "tails"
```

```
##  [55] "tails" "heads" "tails" "heads" "tails" "tails" "tails" "heads" "heads"
##  [64] "heads" "heads" "heads" "heads" "tails" "tails" "tails" "heads" "tails"
##  [73] "tails" "heads" "heads" "tails" "tails" "heads" "tails" "tails" "tails"
##  [82] "heads" "tails" "tails" "tails" "heads" "heads" "heads" "heads" "tails"
##  [91] "heads" "heads" "heads" "heads" "heads" "tails" "heads" "heads" "tails"
## [100] "tails"
```

To view the results of this simulation, type the name of the object and then use `table` to count up the number of heads and tails.

```
sim_fair_coin
```

```
##   [1] "tails" "tails" "tails" "heads" "tails" "heads" "heads" "heads" "tails"
##  [10] "heads" "tails" "heads" "heads" "tails" "heads" "tails" "tails" "heads"
##  [19] "tails" "heads" "heads" "tails" "heads" "tails" "heads" "heads" "heads"
##  [28] "heads" "heads" "tails" "heads" "tails" "tails" "tails" "tails" "heads"
##  [37] "heads" "tails" "heads" "tails" "heads" "tails" "tails" "heads" "heads"
##  [46] "heads" "tails" "tails" "heads" "heads" "heads" "heads" "tails" "tails"
##  [55] "tails" "heads" "tails" "heads" "tails" "tails" "tails" "heads" "heads"
##  [64] "heads" "heads" "heads" "heads" "tails" "tails" "tails" "heads" "tails"
##  [73] "tails" "heads" "heads" "tails" "tails" "heads" "tails" "tails" "tails"
##  [82] "heads" "tails" "tails" "tails" "heads" "heads" "heads" "heads" "tails"
##  [91] "heads" "heads" "heads" "heads" "heads" "tails" "heads" "heads" "tails"
## [100] "tails"
```

```
table(sim_fair_coin)
```

```
## sim_fair_coin
## heads tails
##    53    47
```

Since there are only two elements in `coin_outcomes`, the probability that we "flip" a coin and it lands heads is 0.5. Say we're trying to simulate an unfair coin that we know only lands heads 20% of the time. We can adjust for this by adding an argument called `prob`, which provides a vector of two probability weights.

```
sim_unfr_coin <- sample(coin_outcomes, size = 100, replace = TRUE,
                        prob = c(0.2, 0.8))
```

`prob=c(0.2, 0.8)` indicates that for the two elements in the `outcomes` vector, we want to select the first one, `heads`, with probability 0.2 and the second one, `tails` with probability 0.8. Another way of thinking about this is to think of the outcome space as a bag of 10 chips, where 2 chips are labeled "head" and 8 chips "tail". Therefore at each draw, the probability of drawing a chip that says "head"" is 20%, and "tail" is 80%.

3. In your simulation of flipping the unfair coin 100 times, how many flips came up heads? Include the code for sampling the unfair coin in your response. Since the markdown file will run the code, and generate a new sample each time you *Knit* it, you should also "set a seed" **before** you sample. Read more about setting a seed below.

Insert your answer here Set.seed @ 1917 resulted in: 13 heads and 87 tails

```
set.seed(1917)
sim_unfair_coin <- sample(coin_outcomes, size = 100, replace = TRUE,
                        prob = c(0.2, 0.8))
sim_unfair_coin
```

```
##   [1] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "heads"
##  [10] "tails" "tails" "tails" "heads" "tails" "heads" "tails" "tails" "tails"
##  [19] "tails" "tails" "tails" "tails" "heads" "tails" "tails" "tails" "tails"
##  [28] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
```

```
##  [37] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
##  [46] "tails" "heads" "tails" "heads" "tails" "tails" "tails" "tails" "tails"
##  [55] "tails" "tails" "tails" "tails" "tails" "heads" "tails" "tails" "tails"
##  [64] "tails" "tails" "tails" "tails" "tails" "tails" "heads" "tails" "tails"
##  [73] "tails" "tails" "heads" "tails" "tails" "tails" "tails" "heads" "tails"
##  [82] "heads" "heads" "tails" "tails" "tails" "tails" "heads" "tails" "tails"
##  [91] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
## [100] "tails"
```

```
sim_unfair_coin
```

```
##   [1] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "heads"
##  [10] "tails" "tails" "tails" "heads" "tails" "heads" "tails" "tails" "tails"
##  [19] "tails" "tails" "tails" "tails" "heads" "tails" "tails" "tails" "tails"
##  [28] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
##  [37] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
##  [46] "tails" "heads" "tails" "heads" "tails" "tails" "tails" "tails" "tails"
##  [55] "tails" "tails" "tails" "tails" "tails" "heads" "tails" "tails" "tails"
##  [64] "tails" "tails" "tails" "tails" "tails" "tails" "heads" "tails" "tails"
##  [73] "tails" "tails" "heads" "tails" "tails" "tails" "tails" "heads" "tails"
##  [82] "heads" "heads" "tails" "tails" "tails" "tails" "heads" "tails" "tails"
##  [91] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
## [100] "tails"
```

```
table(sim_unfair_coin)
```

```
## sim_unfair_coin
## heads tails
##    13    87
```

A note on setting a seed: Setting a seed will cause R to select the same sample each time you knit your document. This will make sure your results don't change each time you knit, and it will also ensure reproducibility of your work (by setting the same seed it will be possible to reproduce your results). You can set a seed like this:

```
set.seed(35797)                 # make sure to change the seed
```

The number above is completely arbitraty. If you need inspiration, you can use your ID, birthday, or just a random string of numbers. The important thing is that you use each seed only once in a document. Remember to do this **before** you sample in the exercise above.

In a sense, we've shrunken the size of the slip of paper that says "heads", making it less likely to be drawn, and we've increased the size of the slip of paper saying "tails", making it more likely to be drawn. When you simulated the fair coin, both slips of paper were the same size. This happens by default if you don't provide a `prob` argument; all elements in the `outcomes` vector have an equal probability of being drawn.

If you want to learn more about `sample` or any other function, recall that you can always check out its help file.

```
?sample
```

Simulating the Independent Shooter

Simulating a basketball player who has independent shots uses the same mechanism that you used to simulate a coin flip. To simulate a single shot from an independent shooter with a shooting percentage of 50% you can type

```
shot_outcomes <- c("H", "M")
sim_basket <- sample(shot_outcomes, size = 1, replace = TRUE)
```

To make a valid comparison between Kobe and your simulated independent shooter, you need to align both their shooting percentage and the number of attempted shots.

4. What change needs to be made to the `sample` function so that it reflects a shooting percentage of 45%? Make this adjustment, then run a simulation to sample 133 shots. Assign the output of this simulation to a new object called `sim_basket`.

Insert your answer here With a seed set @ (1310), the simulation gives us: 70 H and 63 M

```
set.seed(1310)
sim_basket<- sample(shot_outcomes, size = 133, replace = TRUE)
                        prob = c(0.45, 0.55)

sim_basket
```

```
##   [1] "H" "H" "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "H" "H" "H" "H" "H" "M"
##  [19] "H" "M" "M" "M" "M" "H" "M" "H" "H" "H" "H" "H" "M" "M" "M" "M" "M" "H"
##  [37] "M" "H" "H" "M" "H" "H" "M" "H" "M" "M" "H" "H" "M" "H" "M" "H" "M" "H"
##  [55] "M" "M" "M" "H" "M" "M" "M" "H" "H" "M" "M" "M" "H" "H" "M" "M" "H" "M"
##  [73] "M" "M" "M" "M" "H" "M" "H" "H" "H" "H" "M" "M" "M" "H" "M" "H" "H" "H"
##  [91] "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "M" "H" "M" "H" "H" "H" "M" "M"
## [109] "H" "H" "H" "M" "H" "M" "M" "M" "M" "H" "H" "H" "H" "H" "H" "H" "H" "M"
## [127] "H" "M" "H" "M" "H" "M" "H"
```

```
sim_basket
```

```
##   [1] "H" "H" "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "H" "H" "H" "H" "H" "M"
##  [19] "H" "M" "M" "M" "M" "H" "M" "H" "H" "H" "H" "H" "M" "M" "M" "M" "M" "H"
##  [37] "M" "H" "H" "M" "H" "H" "M" "H" "M" "M" "H" "H" "M" "H" "M" "H" "M" "H"
##  [55] "M" "M" "M" "H" "M" "M" "M" "H" "H" "M" "M" "M" "H" "H" "M" "M" "H" "M"
##  [73] "M" "M" "M" "M" "H" "M" "H" "H" "H" "H" "M" "M" "M" "H" "M" "H" "H" "H"
##  [91] "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "M" "H" "M" "H" "H" "H" "M" "M"
## [109] "H" "H" "H" "M" "H" "M" "M" "M" "M" "H" "H" "H" "H" "H" "H" "H" "H" "M"
## [127] "H" "M" "H" "M" "H" "M" "H"
```

```
table(sim_basket)
```

```
## sim_basket
##  H  M
## 70 63
```

Note that we've named the new vector `sim_basket`, the same name that we gave to the previous vector reflecting a shooting percentage of 50%. In this situation, R overwrites the old object with the new one, so always make sure that you don't need the information in an old vector before reassigning its name.

With the results of the simulation saved as `sim_basket`, you have the data necessary to compare Kobe to our independent shooter.

Both data sets represent the results of 133 shot attempts, each with the same shooting percentage of 45%. We know that our simulated data is from a shooter that has independent shots. That is, we know the simulated shooter does not have a hot hand.

More Practice

Comparing Kobe Bryant to the Independent Shooter

5. Using `calc_streak`, compute the streak lengths of `sim_basket`, and save the results in a data frame called `sim_streak`.

Insert your answer here

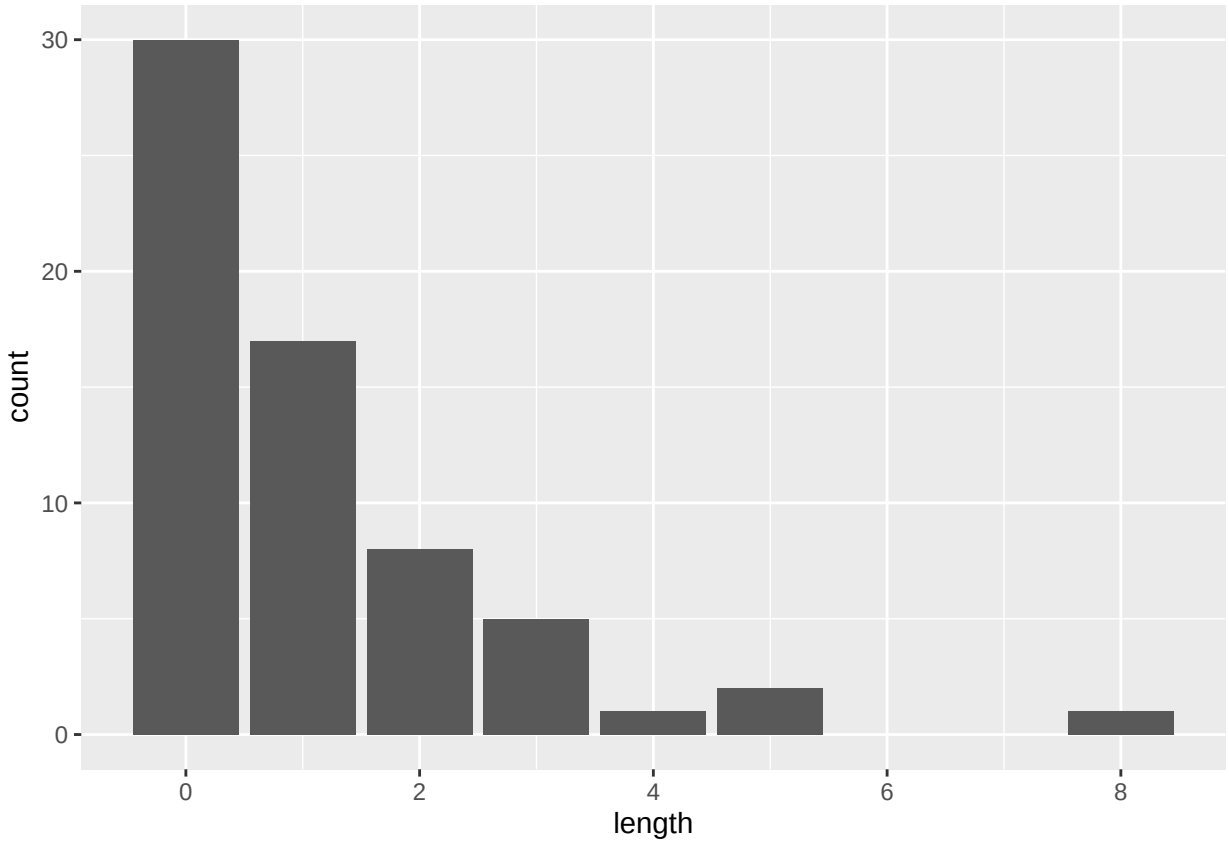
```
sim_streak <- calc_streak(sim_basket)
summary(sim_streak)
```

```
##      length
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :1.000
##  Mean   :1.094
##  3rd Qu.:2.000
##  Max.   :8.000
```

6. Describe the distribution of streak lengths. What is the typical streak length for this simulated independent shooter with a 45% shooting percentage? How long is the player's longest streak of baskets in 133 shots? Make sure to include a plot in your answer.

Insert your answer here

```
ggplot(data = sim_streak, aes(x = length)) +
  geom_bar()
```




```
  theme_minimal()
```

```
## List of 97
##  $ line                       :List of 6
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                       :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                       :List of 11
##   ..$ family       : chr ""
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 11
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                      : NULL
##  $ aspect.ratio               : NULL
##  $ axis.title                 : NULL
##  $ axis.title.x               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top           :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
```

```
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom       : NULL
##  $ axis.title.y              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left         : NULL
##  $ axis.title.y.right        :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                 :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey30"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
```

```
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top           :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom        : NULL
##  $ axis.text.y               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left          : NULL
##  $ axis.text.y.right         :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
```

```
##  $ axis.ticks               : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.ticks.x             : NULL
##  $ axis.ticks.x.top         : NULL
##  $ axis.ticks.x.bottom      : NULL
##  $ axis.ticks.y             : NULL
##  $ axis.ticks.y.left        : NULL
##  $ axis.ticks.y.right       : NULL
##  $ axis.ticks.length        : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ axis.ticks.length.x      : NULL
##  $ axis.ticks.length.x.top  : NULL
##  $ axis.ticks.length.x.bottom: NULL
##  $ axis.ticks.length.y      : NULL
##  $ axis.ticks.length.y.left  : NULL
##  $ axis.ticks.length.y.right : NULL
##  $ axis.line                : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.line.x              : NULL
##  $ axis.line.x.top          : NULL
##  $ axis.line.x.bottom       : NULL
##  $ axis.line.y              : NULL
##  $ axis.line.y.left         : NULL
##  $ axis.line.y.right        : NULL
##  $ legend.background        : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.margin            : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing           : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x         : NULL
##  $ legend.spacing.y         : NULL
##  $ legend.key               : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size          : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height        : NULL
##  $ legend.key.width         : NULL
##  $ legend.text              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.align        : NULL
##  $ legend.title             :List of 11
##   ..$ family       : NULL
```

```
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.align        : NULL
##  $ legend.position           : chr "right"
##  $ legend.direction          : NULL
##  $ legend.justification      : chr "center"
##  $ legend.box                : NULL
##  $ legend.box.just           : NULL
##  $ legend.box.margin         : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing        : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ panel.background          : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.border              : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.spacing             : 'simpleUnit' num 5.5points
##   ..- attr(*, "unit")= int 8
##  $ panel.spacing.x           : NULL
##  $ panel.spacing.y           : NULL
##  $ panel.grid                :List of 6
##   ..$ colour       : chr "grey92"
##   ..$ linewidth    : NULL
##   ..$ linetype     : NULL
##   ..$ lineend      : NULL
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ panel.grid.major          : NULL
##  $ panel.grid.minor          :List of 6
##   ..$ colour       : NULL
##   ..$ linewidth    : 'rel' num 0.5
##   ..$ linetype     : NULL
##   ..$ lineend      : NULL
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ panel.grid.major.x        : NULL
##  $ panel.grid.major.y        : NULL
##  $ panel.grid.minor.x        : NULL
##  $ panel.grid.minor.y        : NULL
##  $ panel.ontop               : logi FALSE
##  $ plot.background           : list()
```

```
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ plot.title               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 1.2
##   ..$ hjust        : num 0
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 5.5points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.title.position      : chr "panel"
##  $ plot.subtitle            :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 5.5points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.caption             :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : num 1
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 5.5points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.caption.position    : chr "panel"
##  $ plot.tag                 :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 1.2
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
```

```
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.tag.position         : chr "topleft"
##  $ plot.margin               : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ strip.background          : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ strip.background.x        : NULL
##  $ strip.background.y        : NULL
##  $ strip.clip                : chr "inherit"
##  $ strip.placement           : chr "inside"
##  $ strip.text                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey10"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 4.4points 4.4points 4.4points 4.4points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ strip.text.x              : NULL
##  $ strip.text.x.bottom       : NULL
##  $ strip.text.x.top          : NULL
##  $ strip.text.y              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ strip.text.y.left         :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
```

```
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ strip.text.y.right        : NULL
##  $ strip.switch.pad.grid     : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ strip.switch.pad.wrap     : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

```
ggtitle("Simulation of Streak Distribution")
```

```
## $title
## [1] "Simulation of Streak Distribution"
##
## attr(,"class")
## [1] "labels"
```

7. If you were to run the simulation of the independent shooter a second time, how would you expect its streak distribution to compare to the distribution from the question above? Exactly the same? Somewhat similar? Totally different? Explain your reasoning.

Insert your answer here Assuming a seed was set for all the simulations (first and second), I would expect to see similarities in the resulting streak distributions.

8. How does Kobe Bryant's distribution of streak lengths compare to the distribution of streak lengths for the simulated shooter? Using this comparison, do you have evidence that the hot hand model fits Kobe's shooting patterns? Explain.

Insert your answer here Looking at the bar plots, one notices that both distributions are normally distributed. This means that both distributions are reflective of the central limit theorem which accounts for independent random variables. The central limit theorem makes it hard to side with any "Hot Hand Theory" or to explain Kobe's shooting patterns using a "Hot Hand Model."