Lab2 - Probability

Michael Y.

September 23rd, 2018

Hot Hands

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 (http://psych.cornell.edu/sites/default/files/Gilo.Vallone.Tversky.pdf). 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.

Saving your code

Click on File -> New -> R Script. This will open a blank document above the console. As you go along you can copy and paste your code here and save it. This is a good way to keep track of your code and be able to reuse it later. To run your code from this document you can either copy and paste it into the console, highlight the code and hit the Run button, or highlight the code and hit command+enter or a mac or control+enter on a PC.

You'll also want to save this script (code document). To do so click on the disk icon. The first time you hit save, RStudio will ask for a file name; you can name it anything you like. Once you hit save you'll see the file appear under the Files tab in the lower right panel. You can reopen this file anytime by simply clicking on it.

Getting Started

Our 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. Let's load some data from those games and look at the first several rows.

```r
load("more/kobe.RData")
head(kobe)
```

```
##    vs game quarter time
## 1 ORL    1       1 9:47
## 2 ORL    1       1 9:07
## 3 ORL    1       1 8:11
## 4 ORL    1       1 7:41
## 5 ORL    1       1 7:03
## 6 ORL    1       1 6:01
##                                      description basket
## 1                 Kobe Bryant makes 4-foot two point shot      H
## 2                             Kobe Bryant misses jumper      M
## 3                      Kobe Bryant misses 7-foot jumper      M
## 4 Kobe Bryant makes 16-foot jumper (Derek Fisher assists)      H
## 5                          Kobe Bryant makes driving layup      H
## 6                                Kobe Bryant misses jumper      M
```

In this data frame, every row records a shot taken by Kobe Bryant. If he hit the shot (made a basket), a hit, `H`, is recorded in the column named `basket`, otherwise a miss, `M`, is recorded.

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:

$$[\text{H M} \mid \text{M} \mid \text{H H M} \mid \text{M} \mid \text{M} \mid \text{M}]$$

To verify this use the following command:

```
kobe$basket[1:9]
```

```
## [1] "H" "M" "M" "H" "H" "M" "M" "M" "M"
```

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?

Response to Exercise 1

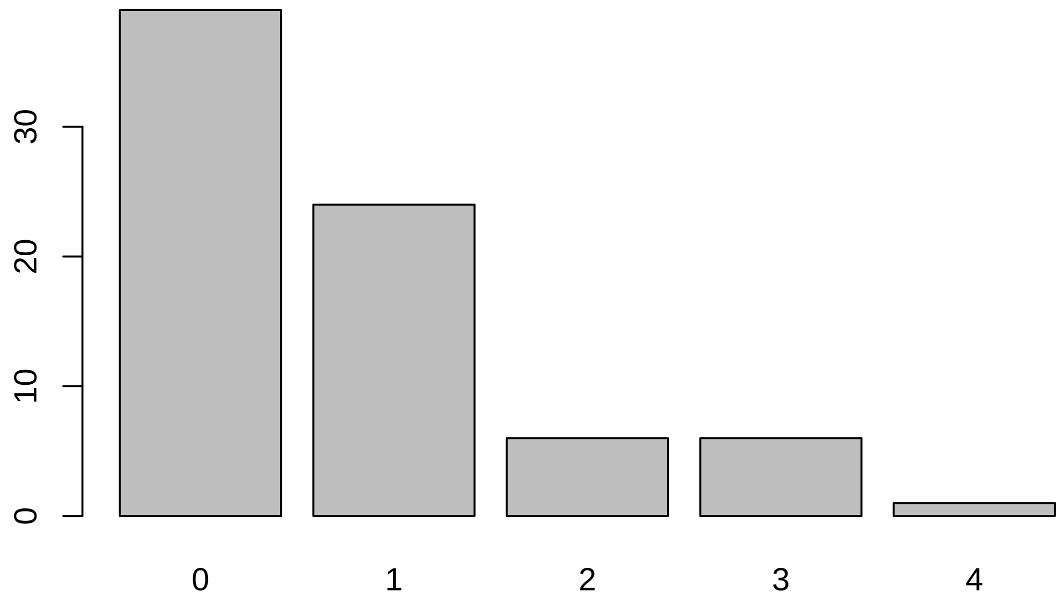
A streak of length 1 means one hit followed by one miss.
However, if the final shot is a *hit*, and it was preceded by a *miss*, then it too is a "streak of length one".

A streak of length 0 means zero hits followed by one miss, i.e.,
it is a **miss** which was either:
* preceded by another miss (which was a part of a separate streak), or
* it was the first shot of the period under evaluation.

End of Response to Exercise 1

The custom function `calc_streak`, which was loaded in with the data, may be used to calculate the lengths of all shooting streaks and then look at the distribution.

```
kobe_streak <- calc_streak(kobe$basket)
barplot(table(kobe_streak))
```



Note that instead of making a histogram, we chose to make a bar plot from a table of the streak data. *A bar plot is preferable here since our variable is **discrete** – counts – instead of **continuous**.

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?

Response to Exercise 2

The distribution resembles a log-normal distribution. There are 39 "streaks" of zero (in other words, a miss, preceded by another miss), which just barely exceeds half of the total number of "streaks", which is 76.

Thus, the mode and the median are both ZERO, while the average (i.e., the mean) is 'r mean(kobe_streak).' The "typical" streak length is thus **zero** . The longest streak of baskets was 4 in a row; this occurred only once.

I note that the supplied function calc_streak contains a bug!

In respect of the above example, the function incorrectly reports a spurious "zero" after the correct data, i.e.,

```
kobe$basket[1:9]
```

```
## [1] "H" "M" "M" "H" "H" "M" "M" "M" "M"
```

```
calc_streak(kobe$basket[1:9])
```

```
## [1] 1 0 2 0 0 0 0
```

Note that there are four zeros at the end of the result, but according to this:

> 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]
>
> 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).

there should only be three zeros at the end of the sequence, *not* four.

The supplied `calc_streak` function is coded as follows:

```
calc_streak
```

```
## function(x){
##   y <- rep(0,length(x))
##   y[x == "H"] <- 1
##   y <- c(0, y, 0)
##   wz <- which(y == 0)
##   streak <- diff(wz) - 1
##   return(streak)
## }
## <bytecode: 0x00000000185e65b8>
```

The bug in the above function occurs on the line

```
y <- c(0,y,0)
```

The appending of a zero at the end of the sequence of "y" (which denotes **hit** baskets with 1 and **missed** shots with 0) *only works in the case where the* **final** *shot in the sequence under observation is a* ***hit***, i.e., ***"H"***. In the case where the final shot is a ***miss***, i.e., ***"M"***, the appending of a zero causes the result from `calc_streak` to be ***wrong***.

This bug can be corrected as follows:

```
my_calc_streak <- function(x){
  y <- rep(0,length(x))
  y[x == "H"] <- 1
  y <- c(0, y)                          # pre-pend a "zero" in front of the "y" sequence, in all cases
  if (x[length(x)]=="H")  y <- c(y, 0)   # append a "zero" at the end ONLY if the final shot was a hit.
  wz <- which(y == 0)
  streak <- diff(wz) - 1
  return(streak)
}
```

The first 9 shots: H, M, M, H, H, M, M, M, M

The incorrect results, using the supplied function `calc_streak` :

```
calc_streak(kobe$basket[1:9])
```

```
## [1] 1 0 2 0 0 0 0
```

The correct results, using my corrected function, `my_calc_streak` :

```r
my_calc_streak(kobe$basket[1:9])
```

```
## [1] 1 0 2 0 0 0
```

End of Response to Exercise 2

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 hot hands? 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 **[sic]** 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 hot hands? We can compare his streak lengths to someone without hot hands: 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.

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

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

The vector `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(outcomes, size = 100, replace = TRUE)
number_of_heads <- sum(as.integer(sim_fair_coin=="heads"))
```

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" "heads" "heads" "heads" "heads" "heads" "heads"
##   [9] "tails" "heads" "tails" "tails" "tails" "tails" "tails" "tails"
##  [17] "heads" "tails" "tails" "tails" "heads" "heads" "tails" "tails"
##  [25] "tails" "heads" "tails" "heads" "heads" "heads" "heads" "tails"
##  [33] "tails" "heads" "tails" "heads" "heads" "heads" "tails" "tails"
##  [41] "heads" "heads" "heads" "tails" "heads" "tails" "tails" "heads"
##  [49] "heads" "tails" "tails" "heads" "heads" "tails" "heads" "heads"
##  [57] "heads" "heads" "tails" "heads" "heads" "tails" "heads" "heads"
##  [65] "tails" "heads" "heads" "tails" "tails" "heads" "tails" "tails"
##  [73] "tails" "tails" "heads" "heads" "tails" "heads" "heads" "heads"
##  [81] "tails" "heads" "heads" "heads" "tails" "heads" "tails" "heads"
##  [89] "heads" "tails" "heads" "heads" "tails" "tails" "tails" "tails"
##  [97] "tails" "tails" "tails" "heads"
```

```
table(sim_fair_coin)
```

```
## sim_fair_coin
## heads tails
##    52    48
```

Since there are only two elements in `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_unfair_coin <- sample(outcomes, size = 100, replace = TRUE, prob = c(0.2, 0.8))
sim_unfair_coin
```

```
##   [1] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
##   [9] "heads" "tails" "tails" "heads" "tails" "tails" "tails" "tails"
##  [17] "tails" "tails" "heads" "tails" "tails" "tails" "tails" "tails"
##  [25] "tails" "heads" "tails" "tails" "heads" "tails" "tails" "tails"
##  [33] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "tails"
##  [41] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "heads"
##  [49] "tails" "tails" "tails" "tails" "tails" "tails" "tails" "heads"
##  [57] "tails" "tails" "tails" "tails" "heads" "tails" "heads" "tails"
##  [65] "tails" "tails" "tails" "tails" "tails" "tails" "heads" "tails"
##  [73] "tails" "tails" "heads" "tails" "tails" "tails" "tails" "tails"
##  [81] "heads" "tails" "heads" "tails" "tails" "heads" "tails" "tails"
##  [89] "heads" "tails" "tails" "heads" "tails" "tails" "tails" "tails"
##  [97] "tails" "tails" "tails" "tails"
```

```
unfair_number_of_heads <- sum(as.integer(sim_unfair_coin=="heads"))
unfair_number_of_heads
```

```
## [1] 16
```

`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?

Response to Exercise 3

In the simulation of flipping the unfair coin 100 times, the number of "heads" was ***16*** , which is close to the expected result (i.e., **20**, based upon the probability of 0.2 .)

End of Response to Exercise 3

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 we 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.

```
##### display suppressed for knit
#?sample
```

Simulating the Independent Shooter

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

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

To make a valid comparison between Kobe and our simulated independent shooter, we 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`.

Response to Exercise 4

We need to add `prob = c(0.45,0.55)` to the *sample** function call:

```
sim_basket <- sample(outcomes, size = 133, replace = TRUE, prob = c(0.45,0.55))
sim_basket
```

```
##   [1] "M" "H" "H" "M" "H" "H" "H" "M" "M" "M" "H" "H" "M" "M" "M" "M" "M"
##  [18] "H" "H" "M" "H" "H" "M" "M" "M" "M" "H" "M" "H" "H" "M" "H" "H" "H"
##  [35] "M" "M" "H" "M" "H" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "M"
##  [52] "H" "H" "H" "M" "H" "M" "M" "M" "H" "M" "H" "M" "H" "M" "H" "H" "M"
##  [69] "M" "M" "H" "M" "M" "H" "M" "M" "M" "M" "M" "M" "M" "H" "H" "H" "M"
##  [86] "H" "M" "M" "H" "H" "M" "H" "M" "M" "M" "M" "H" "M" "H" "H" "H" "H"
## [103] "H" "M" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "H" "H" "M" "M"
## [120] "H" "M" "M" "M" "M" "M" "M" "H" "M" "M" "H" "M" "M" "M"
```

```
number_of_hit_baskets <- sum(as.integer(sim_basket=="H"))
number_of_hit_baskets
```

```
## [1] 58
```

```
proportion_of_hit_baskets <- number_of_hit_baskets/133
proportion_of_hit_baskets
```

```
## [1] 0.4360902
```

End of response to Exercise 4

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`, we have the data necessary to compare Kobe to our independent shooter. We can look at Kobe's data alongside our simulated data.

```
kobe$basket
```

```
##   [1] "H" "M" "M" "H" "H" "M" "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "M"
##  [18] "H" "H" "H" "M" "M" "H" "M" "H" "H" "H" "M" "M" "M" "M" "M" "M" "H"
##  [35] "M" "H" "M" "M" "H" "H" "H" "H" "M" "H" "M" "M" "H" "M" "M" "H" "M"
##  [52] "M" "H" "M" "H" "H" "M" "M" "H" "M" "H" "H" "M" "H" "M" "M" "M" "H"
##  [69] "M" "M" "M" "M" "H" "M" "H" "M" "M" "H" "M" "M" "H" "H" "M" "M" "M"
##  [86] "M" "H" "H" "H" "M" "M" "H" "M" "M" "H" "M" "H" "H" "M" "H" "M" "M"
## [103] "H" "M" "M" "M" "H" "M" "H" "H" "H" "M" "H" "H" "H" "M" "H" "M" "H"
## [120] "M" "M" "M" "M" "M" "M" "H" "M" "H" "M" "M" "M" "M" "H"
```

```
sim_basket
```

```
##   [1] "M" "H" "H" "M" "H" "H" "H" "M" "M" "M" "H" "H" "M" "M" "M" "M" "M"
##  [18] "H" "H" "M" "H" "H" "M" "M" "M" "M" "H" "M" "H" "H" "M" "H" "H" "H"
##  [35] "M" "M" "H" "M" "H" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "M"
##  [52] "H" "H" "H" "M" "H" "M" "M" "M" "H" "M" "H" "M" "H" "M" "H" "H" "M"
##  [69] "M" "M" "H" "M" "M" "H" "M" "M" "M" "M" "M" "M" "M" "H" "H" "H" "M"
##  [86] "H" "M" "M" "H" "H" "M" "H" "M" "M" "M" "M" "H" "M" "H" "H" "H" "H"
## [103] "H" "M" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "H" "H" "M" "M"
## [120] "H" "M" "M" "M" "M" "M" "M" "H" "M" "M" "H" "M" "M" "M"
```

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.

On your own

Comparing Kobe Bryant to the Independent Shooter

Using `calc_streak`, compute the streak lengths of `sim_basket`.

Note: as I have established above that `calc_streak` contains a bug, I am instead going to use `my_calc_streak` , in which I have fixed the bug.

Kobe Streak Info

```
kobe$basket
```

```
##   [1] "H" "M" "M" "H" "H" "M" "M" "M" "M" "H" "H" "H" "M" "H" "H" "M" "M"
##  [18] "H" "H" "H" "M" "M" "H" "M" "H" "H" "H" "M" "M" "M" "M" "M" "M" "H"
##  [35] "M" "H" "M" "M" "H" "H" "H" "H" "M" "H" "M" "M" "H" "M" "M" "H" "M"
##  [52] "M" "H" "M" "H" "H" "M" "M" "H" "M" "H" "H" "M" "H" "M" "M" "M" "H"
##  [69] "M" "M" "M" "M" "H" "M" "H" "M" "M" "H" "M" "M" "H" "H" "M" "M" "M"
##  [86] "M" "H" "H" "H" "M" "M" "H" "M" "M" "H" "M" "H" "H" "M" "H" "M" "M"
## [103] "H" "M" "M" "M" "H" "M" "H" "H" "H" "M" "H" "H" "H" "M" "H" "M" "H"
## [120] "M" "M" "M" "M" "M" "M" "H" "M" "H" "M" "M" "M" "M" "H"
```

```
kobe_streak <- my_calc_streak(kobe$basket)
kobe_streak
```

```
##  [1] 1 0 2 0 0 0 3 2 0 3 0 1 3 0 0 0 0 0 1 1 0 4 1 0 1 0 1 0 1 2 0 1 2 1 0
## [36] 0 1 0 0 0 1 1 0 1 0 2 0 0 0 3 0 1 0 1 2 1 0 1 0 0 1 3 3 1 1 0 0 0 0 0
## [71] 1 1 0 0 0 1
```

```
sum(kobe_streak)
```

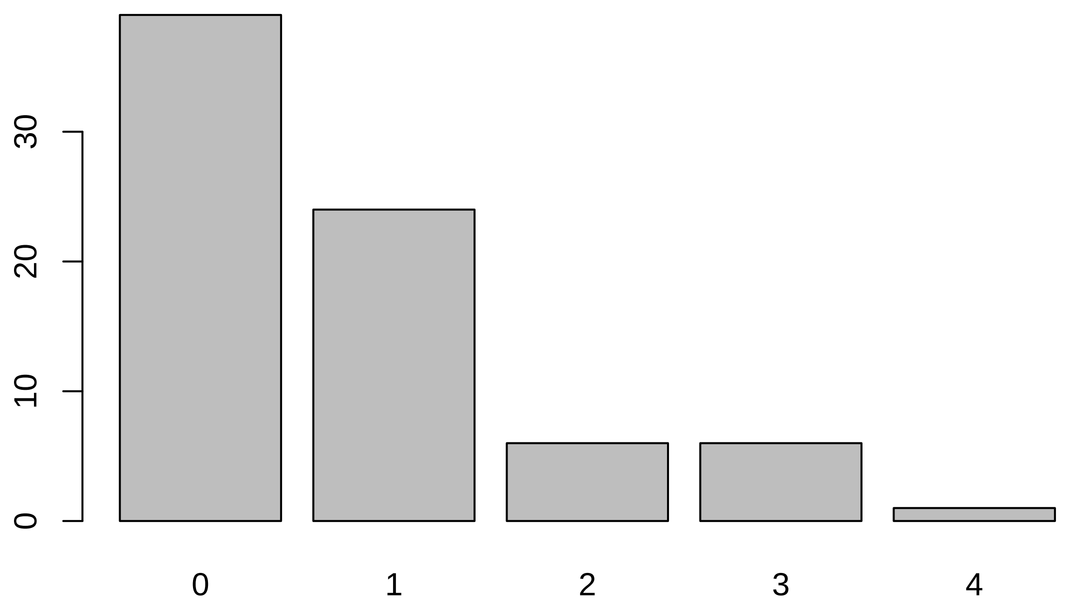
```
## [1] 58
```

```
table(kobe_streak)
```

```
## kobe_streak
##  0  1  2  3  4
## 39 24  6  6  1
```

Kobe streak barplot

```
barplot(table(kobe_streak))
```



Sim Streak info

```
sim_basket
```

```
##   [1] "M" "H" "H" "M" "H" "H" "H" "M" "M" "M" "H" "H" "M" "M" "M" "M" "M"
##  [18] "H" "H" "M" "H" "H" "M" "M" "M" "M" "H" "M" "H" "H" "M" "H" "H" "H"
##  [35] "M" "M" "H" "M" "H" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "M"
##  [52] "H" "H" "H" "M" "H" "M" "M" "M" "H" "M" "H" "M" "H" "M" "H" "H" "M"
##  [69] "M" "M" "H" "M" "M" "H" "M" "M" "M" "M" "M" "M" "M" "H" "H" "H" "M"
##  [86] "H" "M" "M" "H" "H" "M" "H" "M" "M" "M" "M" "H" "M" "H" "H" "H" "H"
## [103] "H" "M" "M" "M" "M" "M" "H" "H" "M" "M" "H" "H" "H" "H" "H" "M" "M"
## [120] "H" "M" "M" "M" "M" "M" "M" "H" "M" "M" "H" "M" "M" "M"
```

```
sim_streak <- my_calc_streak(sim_basket)
sum(sim_streak)
```

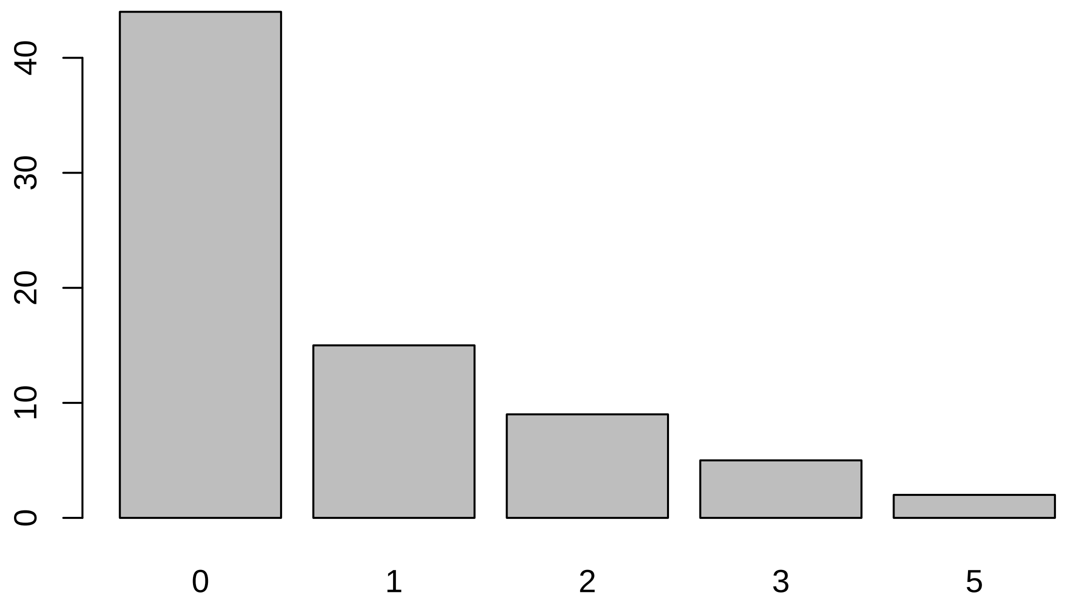
```
## [1] 58
```

```
table(sim_streak)
```

```
## sim_streak
##  0  1  2  3  5
## 44 15  9  5  2
```

Sim streak barplot

```
barplot(table(sim_streak))
```



More sim streak info

```
streaktable = table(sim_streak)
streaktable
```

```
## sim_streak
##  0  1  2  3  5
## 44 15  9  5  2
```

```
streakmedian = median(sim_streak)
streakmodenamed = streaktable[streaktable==max(streaktable)]
streakmode=names(streakmodenamed)
streakmode
```

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

```
streakmodequantity=as.numeric(streakmodenamed)
streakmodequantity
```

```
## [1] 44
```

```
streakmax = streaktable[length(streaktable)]
streakmaxlength = names(streakmax)
streakmaxlength
```

```
## [1] "5"
```

```
streakmaxcount = as.integer(streakmax)
streakmaxcount
```

```
## [1] 2
```

```
numshots = sum(sim_streak+1)
numshots
```

```
## [1] 133
```

```
numgood  = sum(sim_streak)
numgood
```

```
## [1] 58
```

```
pctgood  = numgood/numshots
pctgood
```

```
## [1] 0.4360902
```

```
nummissed = sum(as.integer(sim_basket=="M"))
nummissed
```

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

```
numstreaks = length(sim_streak)
numstreaks
```

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

```
sim_streak
```

```
##  [1] 0 2 3 0 0 2 0 0 0 0 2 2 0 0 0 1 2 3 0 1 1 0 0 0 2 0 3 3 1 0 0 1 1 1 2
## [36] 0 0 1 0 1 0 0 0 0 0 0 3 1 0 2 1 0 0 0 1 5 0 0 0 0 2 0 5 0 1 0 0 0 0 0
## [71] 1 0 1 0 0
```

(1) Describe the distribution of streak lengths.

In the above case, the number of successful shots was 58 out of a total of 133 attempts, for a shooting percentage of 0.4360902 . The total number of "streaks" was 75, which reflects 75 misses. The shape of the distribution is right-skewed with most results equal to zero or 1, and few long streaks.

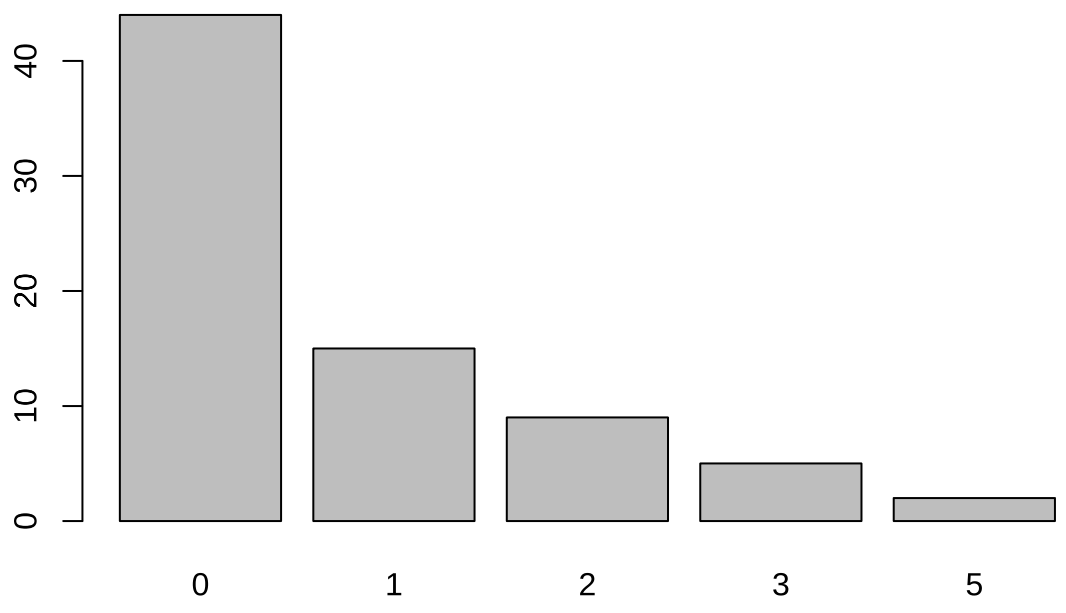
For the above simulation, the table of streak lengths is:

```
streaktable
```

```
## sim_streak
##  0  1  2  3  5
## 44 15  9  5  2
```

The barplot associated with this streaktable is:

```
barplot(streaktable)
```



What is the typical streak length for this simulated independent shooter with a 45% shooting percentage?

The "typical" streak length was 0, which occured 44 times.
The median was 0 , which is expected to be zero, as the shooting percentage used in the simulation is less than 50%.

How long is the player's longest streak of baskets in 133 shots?

Because of randomness, the result varies each time I re-knit the r-markdown file. On this run, the longest streak was 5 , which occured 2 times.

(2) 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?

No, because of randomness, it will not be the same. To be exactly the same, first, the total number of baskets would have to match. At a 45% probability of success, for 133 shots there should be 60 successes and 73 missed shots. Running the simulation repeatedly, the total number of successes changes slightly.

Somewhat similar?

Yes, the distribution of streaks should be somewhat similar to the previous results.

Totally different?

No, it would not be totally different.

Explain your reasoning.

The process which generates the hit or miss results relies on a 45% probability of each shot being good. With randomness, there should be "about" 60 successful shots and 73 misses, but the random number generator causes these values to fluctuate each time the simulation is run. Furthermore, the randomness associated with the sequence of successful baskets running in "streaks" is generally similar, but fluctuates. Thus, the distribution should be "somewhat similar" to the previous results, but not exactly the same.

(3) How does Kobe Bryant's distribution of streak lengths compare to the distribution of streak lengths for the simulated shooter?

Kobe Bryant's distribution of streak lengths does **not** look very different from 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?

No, we do **not** have such evidence.

Explain.

If Kobe Bryant's distribution reflected a "hot hand" then his results should reflect significantly **more *long*** streaks than the number we observe from the simulated shooter, whose streak lengths are random. Kobe's longest streak was **4** successful shots, and that occurred only **once** . Most random simulations also reflect such streaks, and some even reflect a larger number.