Lecture 07 - Friday Demo

Wesley Burr

September 28, 2018

Concept of the Day

```
library(tidyverse)
```

Today, instead of having a demo data set that we'll explore, we're going to compute simulations instead, because the topic right now is probability. Thus, instead of loading data, we're going to create it ourselves.

Example 1

Our first example is going to explore probability as a frequentist interpretation: repeated trials, over time, and the proportion which match. Imagine we have a bag with balls in it, and the balls are either red or black. There are 10 total balls, 6 red and 4 black. If you reach into the bag, pull out a ball, record its colour, then put it back in the bag, what is the probability of pulling a black ball out?

Since probability is

$$\frac{\text{outcomes we want}}{\text{all outcomes}}$$

we have

```
p <- 4 / 10
p
```

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

Now let's pull some balls out of this bag by using the **sample()** function. Remember the argument to **sample()** we talked about: **replace = TRUE**.

```
balls <- sample(c("red", "red", "red", "red", "red", "red",
                  "black", "black", "black", "black"), size = 100, replace = TRUE)
```

What did we just do? We repeatedly pulled a ball out of a bag, with six red balls and 4 black balls. What does it look like?

```
balls[1:20]
```

```
##  [1] "red"   "red"   "black" "red"   "black" "red"   "black" "red"
##  [9] "black" "red"   "red"   "black" "black" "black" "red"   "black"
## [17] "red"   "red"   "red"   "black"
```

So it worked! We pulled out **red** and **black** balls, sampling one at a time, 100 total draws (with replacement back in the bag). What are our frequencies?

```
balls == "black"
```

```
##   [1] FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE
##  [12]  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE
##  [23]  TRUE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE
##  [34] FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
##  [45]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE
##  [56]  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE
##  [67]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
##  [78]  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [89] FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## [100] FALSE
```

This comparison is a **logical**: we're asking the computer "Which of the components of **balls** is equal to 'black'?". Now, how do we figure out the number without counting ... 1, 2, 3, BORED.

```r
sum(balls == "black")
```

```
## [1] 46
```

Now why did this work? Because Booleans (logicals), variables which are TRUE/FALSE, are equivalent to 1s (TRUE) and 0s (FALSE). So adding them up means "how many 1s are there?", which is the same as "how many TRUEs are there?".

```r
sum(balls == "black") / 100
```

```
## [1] 0.46
```

```r
sum(balls == "red") / 100
```

```
## [1] 0.54
```

Are these the probabilities we'd expect, 0.4 and 0.6? Not quite! Let's try with 1000 samples.

```r
balls <- sample(c("red", "red", "red", "red", "red", "red",
                  "black", "black", "black", "black"), size = 1000, replace = TRUE)
sum(balls == "black") / 1000
```

```
## [1] 0.398
```

```r
sum(balls == "red") / 1000
```

```
## [1] 0.602
```

That's quite a bit closer. What about 10,000?

```r
balls <- sample(c("red", "red", "red", "red", "red", "red",
                  "black", "black", "black", "black"), size = 10000, replace = TRUE)
sum(balls == "black") / 10000
```

```
## [1] 0.3955
```

```r
sum(balls == "red") / 10000
```

```
## [1] 0.6045
```

And closer again. This is the **frequentist** interpretation: the long-run proportion of times that something happens is its **probability**.

Example 2: Conditioning

Let's setup a simulation to explore conditioning. Imagine a situation where we again have a bag of balls, this time with 5 red balls and 5 black balls. Two balls are drawn without replacement. If the first ball is black, find the probability that the second ball is also black.

We know from the last example that to get numbers which round to "close", we need lots of trials. So let's do 10,000 right from the start. But this time, we need to draw 2 balls at a time, so **sample()** doesn't really do what we want. We can **sample(. . . , size = 2, . . .)** - but how do we draw 2 balls per sample, 10,000 total times?

```r
first_ball <- vector(length = 10000)
second_ball <- vector(length = 10000)
for(j in 1:10000) {
  draw <- sample(c("red", "red", "red", "red", "red",
                   "black", "black", "black", "black", "black"), size = 2, replace = FALSE)
  first_ball[j] <- draw[1]
  second_ball[j] <- draw[2]
}
first_ball[1:10]
```

```
##  [1] "red"   "red"   "red"   "black" "black" "black" "red"   "red"
##  [9] "black" "black"
```

```r
second_ball[1:10]
```

```
##  [1] "black" "red"   "black" "red"   "red"   "black" "black" "black"
##  [9] "black" "black"
```

So now we have 20,000 records: 10,000 'first balls drawn' and 10,000 'second balls drawn'. What did we want again? The probability that the **second ball is black** given that the **first ball is black**.

```r
which_ones <- which(first_ball == "black")
which_ones[1:10]
```

```
##  [1]  4  5  6  9 10 11 14 18 19 23
```

The **which()** command tells us **exactly which ones** of the components of a vector are "whatever we ask". In this case, it's "which components of the vector are equal to 'black'?". Now that we have this, we can use it to find P(B), since B is "first ball drawn is black".

```r
PB <- length(which_ones) / 10000
PB
```

```
## [1] 0.4974
```

(so about half of the first balls drawn were black, which makes sense - think about the probability of drawing one ball from this bag with 5 & 5)

Now, we need **A and B** - that is, the first ball is black **and** the second ball is black.

```r
which_first <- which(first_ball == "black")
which_second <- which(second_ball == "black")
which_first[1:10]
```

```
##  [1]  4  5  6  9 10 11 14 18 19 23
```

```r
which_second[1:10]
```

```
##  [1]  1  3  6  7  8  9 10 12 13 16
```

So it looks like the there are some draws which are one of the **A and B** events: black for both the first ball and the second. So let's figure out what all of those are?

```r
P_AandB <- length(intersect(which_first, which_second)) / 10000
P_AandB
```

```
## [1] 0.2154
```

This is a new command we haven't seen before: **intersect**. It's the **AND** function for sets - it says "which numbers are in both **which_first** and **which_second**?".

So now we have a number for **P(A and B)** and a number for **P(B)**. Put these together:

```
P_AgivenB <- P_AandB / PB
P_AgivenB
```

```
## [1] 0.4330519
```

So according to our simulation, the probability of the second ball drawn being black, **given** that the first ball drawn was black, is around 0.44. What's the **actual** probability, computed theoretically?

Theory

The probability of drawing a black ball from the bag with 5 black, 5 red is clearly 0.5. So the probability of B, "the first ball drawn from a full bag of 10 balls is black" is 0.5. Then all we need is **P(A and B)**. These events are clearly dependent: when you draw the first ball out, it **changes** the probability of the second ball. We can still compute it:

$$\begin{aligned} P(A \text{ and } B) &= P(A|B) \cdot P(B) \\ &= \frac{4}{9} \cdot \frac{5}{10} = \frac{20}{90} = \frac{2}{9} = 0.222222. \end{aligned}$$

Then our conditional probability is

$$P(A|B) = \frac{\frac{2}{9}}{\frac{5}{10}} = \frac{4}{9} = 0.44444444.$$

So our simulation was pretty close: the probability of the second ball being black, given that the first ball was black is $\frac{4}{9}$ or 0.444. Not bad!

Challenge

Take the above simulation, read it carefully, then find the conditional probability of the second ball being **red** given that the first ball was black. Do it both via theory (the exact number), and as a simulation to see how it works when we do it "for real".