Non-intuitive phenomena in probability

Mykola Shafranov

May 22, 2024

Introduction

In this project we will experimentally test several of the non-intuitive phenomena that appear in probability and statistics. Experiments will be made in R through simulations.

1. The birthday paradox

The first experiment is regarding the birthday paradox, also known as the birthday problem. It states that in a random group of 23 people, there is about a 50 percent chance that two people have the same birthday. If the number of people increases then the probability of two people having the same birthday starts to increase rapidly. We will check this statement for groups of 23, 40 and 50 random people.

```r
n <- 100000
group <- c(23, 40, 50)
probabilities <- c(0,0,0)

for (i in group) {
  shared_bdays <- 0

  for (j in 1:n) {
    birthdays <- sample(1:365, i, replace = TRUE)
    if (length(birthdays) != length(unique(birthdays))) {
      shared_bdays <- shared_bdays + 1
    }
  }

  probability <- shared_bdays / n
  probabilities[i] <- probability
}
```

```
Probability of at least two people sharing a birthday in a group
    of 23 people is 50.562 %

Probability of at least two people sharing a birthday in a group
    of 40 people is 89.113 %

Probability of at least two people sharing a birthday in a group
    of 50 people is 97.001 %
```

Intuitively we assume that the probabilities are lower that they actually are, but with this experiment we therefore confirmed the birthday paradox.

2. Urn with red and green balls

We are given an urn containing 100 balls; n of them are red and $100 - n$ are green, where n is chosen randomly in $[0, 100]$. We take a random ball out of the urn and see that it's red. We then discard it. The question is, if the next ball we pick out of the 99 remaining balls is more likely to be red, more likely to be green, or is equally likely to be red or green.

```r
n <- 100000
result <- rep(NA, n)

for (i in 1:n) {
  num_red <- sample(1:100, 1)
  num_green <- 100 - num_red
  urn <- c(rep("R", num_red), rep("G", num_green))
  urn <- sample(urn)
  if (urn[1] == "R") {
    result[i] <- urn[2]
  }
  else {
    result[i] <- NA
  }
}

result_counts <- table(result, useNA = "ifany")
probG <- result_counts["G"] / (result_counts["R"] + result_counts["G"])
probR <- result_counts["R"] / (result_counts["R"] + result_counts["G"])
```

The following table represents the quantity of each of the ball colors being picked second after 100000 simulations. *NA* represents the urns with the first ball being green in our simulation.

result	count
G	16948
R	33604
NA	49448

We see that the second ball picked, conditioned on the first ball being red, is more likely to be red.

```
Probability of the second ball we pick being red conditioned on the
    first picked ball being red is 66.47413 %
```

```
Probability of the second ball we pick being green conditioned on the
    first picked ball being red is 33.52587 %
```

3. Box with a white and a black balls

We have a box that contains one white ball and one black ball. We choose a ball at random and then return it to the box. If we chose a white ball then another white ball is added to the box, and if we chose a black ball then another black ball is added to the box. Played over time, in this instance we repeat this step 100 times (until we have 102 balls in our box), what is the probability that more than 80% of the chosen balls are white?

```r
n <- 100000  # of simulations
N <- 100  # of trials per simulation
target_hits <- 0

for (i in 1:n) {
  W <- 1
  B <- 1

  for (j in 1:N) {
    if (runif(1) < W / (W + B)) {
      W <- W + 1
    }
    else {
      B <- B + 1
    }
  }

  endoftrialW <- W / (W + B)
  if (endoftrialW > 0.8) {
    target_hits <- target_hits + 1
  }
}

Wgeq80 <- target_hits / n
```

```
Probability that more than 80% of the 100 chosen balls are white
    is 19.936 %
```

This is an example of the Polya urn process, for which it is known that all the ball configurations are equiprobable - all configurations have the same probability. In our case therefore the target is hit when the amount of white balls after 100 steps is in $[81, 101]$. That is total of 21 configurations with each having a probability of appearing of $\frac{1}{101}$, which means that the probability that one of the desired configurations appear is $\frac{21}{101} \approx 0.2$. With increasing the number of steps this number approaches 0.2.

4. Monty Hall problem

We come to a game show and see three doors in front of us. One of the doors hides a prize, let's say a car, and the other two doors have no prize. Monty Hall, who is a game show host, asks us to choose one of three doors. We say out loud which door we pick, but we don't open it right away. After this the host, who obviously knows where the prize is, opens one of the other two doors that we did not choose and does not have a car behind it. We are now left with two closed doors, one of which we initially picked. The car is behind one of those two closed doors, but we still don't know behind which one. The host then asks us if we would like to switch our door. What should be our choice?

```
n <- 100000
stayW <- 0
switchW <- 0

for (i in 1:n) {
  car <- sample(1:3, 1)
  choice <- sample(1:3, 1)

  reveal_options <- setdiff(1:3, c(car, choice))
  revealed <- sample(reveal_options, 1)
  remaining <- setdiff(1:3, c(choice, revealed))
  if (choice == car) {
    stayW <- stayW + 1
  }
  else {
    switchW <- switchW + 1
  }
}

stay_winrate <- stayW / n
switch_winrate <- switchW / n
```

Intuitively we assume that the chance to win is 50% since we are choosing between two doors. That is not the case. We see that by switching our doors we actually double our chances at winning.

```
Win rate if we open the initial door is 33.227 %
```

```
Win rate if we switch the door is 66.773 %
```

5. Dice experiments

5.a

We throw a dice until we get a six. What is the expected number of throws including the throw giving six conditioned on the event that all throws gave even numbers?

```
n <- 100000
good_throws <- 0

for (i in 1:n) {
  throws <- 0
  while (TRUE) {
    throws <- throws + 1
    throw <- sample(1:6, 1, replace = TRUE)
    if (throw %% 2 != 0) {
      throws <- 0
      next
    }
    if (throw == 6) {
      good_throws <- good_throws + throws
```

```r
      break
    }
  }
}

avg <- good_throws / n
```

We would assume that on average it would take us 3 throws since we eliminate all trials where we get any odd number, but in reality the expected number of throws is even less and converges to 1.5 as we increase the number of simulations.

```
Expected number of dice throws until rolling 6 including rolling 6
    conditioned on all throws being even is 1.50137
```

5.b

We start throwing the dice again. What is the expected number or throws until we throw two sixes in a row?

```r
n <- 100000
total2x6 <- 0

for (i in 1:n) {
  throws <- 0
  while (TRUE) {
    throws <- throws + 1
    throw <- sample(1:6, 1, replace = TRUE)
    if (throw == 6) {
      if (throws > 1 && previous == 6) {
        break
      }
    }
    previous <- throw
  }
  total2x6 <- total2x6 + throws
}

avg <- total2x6 / n
```

Intuitively we would say that our chances of getting a pair of sixes are $\frac{1}{6} \cdot \frac{1}{6} = \frac{1}{36}$, making us think that the expected number of throws is 36. That is not correct, since we most likely do not take into account the possibility of not getting another six after the first six. The correct interpretation is, since we are dealing with geometric distribution we would on average throw a dice 6 times before hitting our first six. Then we have either another six with a chance of $\frac{1}{6}$ or another number that is not six with a chance of $\frac{5}{6}$. If we denote X as a random variable that represents the number of throws until we hit two sixes in a row, we can express the expected number of throws:

$$E[X] = 6 + \frac{1}{6} \cdot 1 + \frac{5}{6} \cdot (1 + E[X]).$$

After some rearranging we get

$$E[X] = 42.$$

```
Average number of throws to get two sixes in a row is 41.94527
```