Project 2: Checking Probability using Simulation (Monty Hall Problem/Birthday Paradox)

Jose Lazarte

September 30, 2019

Monty Hall Problem

Pencil/Pen Approach

In the show Let's Make a Deal, the contestant wanted to win a brand new car. However, the car was hidden behind one of three doors. The following values represents different statistics of the show Let's Make a Deal.

With 3 doors in play, there is a $\frac{1}{3}$ chance of choosing the door with the car. And there is a $\frac{2}{3}$ chance of choosing a door with a goat. Therefore, the probabilities of the 3 doors are as follows:

Let a be the event that the car is behind door number 1, the probability is $\frac{1}{3}$. Let b be the event that the car is behind door number 2, the probability is $\frac{1}{3}$. Let c be the event that the car is behind door number 3, the probability is $\frac{1}{3}$.

Therefore we can say that the probability of:

$P(C|a)$

$= \frac{P(C \cap a)}{P(a)}$

$= \frac{1/6}{1/3} = \frac{1}{2}$

And:

$P(C|b)$

$= \frac{P(C \cap b)}{P(b)}$

$= \frac{1/3}{1/3} = 1$

Finally:

$P(C|c)$

$= \frac{P(C \cap c)}{P(c)}$

$= \frac{0}{1/3} = 0$

Therefore the probability of $P(C)$ is $\frac{1}{2}$

Using Baye's Theorem we can calculate: $P(a|C) = \frac{P(C|a)P(a)}{P(C|a)P(a)+P(C|a')P(a')} = \frac{1/6}{1/6+2/6} = \frac{1}{3}$

and:

$P(b|C) = \frac{P(C|b)P(b)}{P(C|b)P(b)+P(C|b')P(b')} = \frac{1/3}{1/3+0} = 1$

Monte Carlo Simulation

The following code will be broken down line by line to explain what is occuring and how the simulation is running.

```
doors <- c("1","2","3")
xdata <- c()
for(i in 1:1000)
{
  prize <- sample(doors,1)
  pick <- sample(doors,1)
  opened_door <- sample(doors[which(doors != pick & doors != prize)],1)
  switchyes <- doors[which(doors != pick & doors != opened_door)]
  if(pick==prize){xdata<-c(xdata,"noswitchwin")}
  if(switchyes==prize){xdata<-c(xdata,"switchwin")}
}

sum(xdata=="switchwin")/length(xdata)
sum(xdata=="noswitchwin")/length(xdata)
```

The first two lines:

```
doors <- c("1","2","3")
xdata <- c()
```

Create variables named doors and xdata that contain characters 1, 2, 3 and the result of whether the contestant switched or did not switch, respectively.

The next line:

```
for(i in 1:1000)
```

The for loop is stated to run the code within it 1000 times, thus, creating a simulation.

The next line:

```
  prize <- sample(doors,1)
  pick <- sample(doors,1)
```

The 2 lines are also creating variables named prize and pick, within them is function sample(). The purpose of sample(doors,1) is to randomly pick an element from the pool, in other words, sample(doors,1) is returning 1 randomly picked element from doors. Then it is assigning that value to prize and pick.

The next line:

```
opened_door <- sample(doors[which(doors != pick & doors != prize)],1)
```

This following line creates a variable named opened_door that stores the door that will be opened. The boolean statement doors != pick & doors != prize with which return the index position of the values that do not match the statement. Therefore, it allows for sample to return 1 value that passes the boolean statement from the vector doors and is given to the variable opened_door.

The next line:

```
switchyes <- doors[which(doors != pick & doors != opened_door)]
```

Once again another variable is created named switchyes. This variable is given the value of one of the elements that is neither picked or picked to be opened, therefore, there will always be 1 element left. Unlike the previous line, this is always one value so it does not need to be sampled. Using the which() function it returns the position of the number that satisfies the boolean statement and the [] allows to return the value from the corresponding position and vector.

The next line:

```
if(pick==prize){xdata<-c(xdata,"noswitchwin")}
```

This if loop is only satisfied when the pick variable is the same as the prize variable. If it is TRUE, then the vector xdata will have the string “noswitchwin” added. If it is FALSE, then the program will continue to the next line.

The next line:

```
if(switchyes==prize){xdata<-c(xdata,"switchwin")}
```

As aforementioned, the if loop will only execute when the statement is satified. In this line of code that statement is if switchyes is the same as prize then "switchwin" will be added to xdata.

The next line:

```
sum(xdata=="switchwin")/length(xdata)
sum(xdata=="noswitchwin")/length(xdata)
```

This final 2 lines are outside of the for loop therefore it will only run once. The function sum() returns the numerical value of the summed values in a dataset. Therefore it will add all the values that match either "switchwin" or "noswitchwin". After that it will divide by the length of xdata, meaning length() returns the total number of elements in a set of data. In this case it returns the number of elements in xdata. By putting the two together we arrive to the numerical values of the simulation.

Birthday Paradox

Pencil/Paper Approach

Utilizing a 365 day year and 35 student class size, we can approach the following questions.

-The probability of the first person entering the class having there own unique birthday is: $365/365 = 1$

-Assuming that the first person had a unique birthday, the probability that the second person entering the class has a unique birthday is: $364/365$

-Assuming the first two people had unique birthdays, the probability that the third person entering the class has a unique birthday is: $365/36 * 364/365 * 364/365 \approx 0.9918$

-Assuming the first three people had unique birthdays, the probability that the fourth person entering the class has a unique birthday is: $365/365 * 364/365 * 364/365 * 363/365 \approx 0.9836$

-Using a similar assumption for the first n-1 people, the probability that the nth person entering the class has a unique birthday is: $\frac{365!}{(365-n)!*365^n}$

-As of the writing of this project, we have 35 people in the class, the probability that all 35 people have a unique birthday is: $\frac{365!}{((365-35)!*365^{35})} \approx 0.1856$

-In contrast, the probability that at least two people share a birthday is: $1 - 0.1856 = 0.8144$

Simulation

```
calender <- c(1:365)

numStud <- 35

simNum <- 1000

totMean <- c()



for(i in 1:simNum)

{

  xdata <- c(rep(0,numStud))


  students <- sample(calender,numStud,replace=TRUE)

  for(p in 1:numStud)

  {

    for(q in 1:numStud)

    {

      if(p!=q & students[p]==students[q])

        xdata[p] = 1

    }

  }


  totMean <- c(totMean, sum(xdata==1)/numStud)

}

sum(totMean)/simNum


1-(sum(totMean)/simNum)
```

Utilizing the following algorithm, we receive an average of the averages of the number of students whose birthday repeats. This result is very close to our projected number in the aforementioned Pencil/Pen approach.

Modifying the algorithm results in a class size of around 350-300 to achieve a 50/50 ratio. This may be due to the algorithm as it differs greatly from the calculated 23 class size that supposedly reaches the 50/50 ratio.