R Notebook

Code

The Birthday Paradox

"The birthday paradox, also known as the birthday problem, states that in a random group of 23 people, there is about a 50 percent chance that two people have the same birthday." This statement surely can't be true, or can it? Ignoring leap years, everyone knows there are 365 days in a year, 365 possible birth dates. So how can it be possible that if you filled 100 rooms with only 23 people that roughly 50 of those rooms have a repeated birthday? In this notebook we will go over historical data of US births, the mathematical reasoning of why this statement is actually true, and a simulation of 1000 rooms where we add one person at a time until we have a repeated birthday in the room. Although the mathematical proof makes sense, I want to see how real world data will perform.

US Births from 2000 to 2014

Data manipulation

The dataset is used in this notebook is pretty simple. It contains the year, month, day, and the number of births on that date for every day between 2000 and 2014. There isn't much cleaning that needs to be done; however, I do want to add a column that represents the day in the year. For example, January 1st would equal 1 and December 31st would equal 365. I also want to remove dates only in leap years to keep things simple.

Hide

```
library("readxl")
births <- read.csv("C:\\Users\\ryanp\\Documents\\PortfolioCode\\Birthday Paradox\\US_births_2000-2014_SSA.csv")
```

Hide

```
births <- births[!(births$month == 2 & births$date_of_month > 28), ]
```

Hide

```
# adding a value for the specific day of the year
# Jan 1 = 1, ..., Dec 31 = 365
day_of_year <- rep(1:365, length.out = nrow(births))
births <- cbind(births, day_of_year)
```

Hide

```
# assigning nice labels for the months instead of just the numeric value
births$month[births$month==1] <- 'Jan'
births$month[births$month==2] <- 'Feb'
births$month[births$month==3] <- 'Mar'
births$month[births$month==4] <- 'Apr'
births$month[births$month==5] <- 'May'
births$month[births$month==6] <- 'Jun'
births$month[births$month==7] <- 'Jul'
births$month[births$month==8] <- 'Aug'
births$month[births$month==9] <- 'Sep'
births$month[births$month==10] <- 'Oct'
births$month[births$month==11] <- 'Nov'
births$month[births$month==12] <- 'Dec'
```

Hide

```
head(births)
```

	year <int>	month <chr>	date_of_month <int>	day_of_week <int>	births <int>	day_of_year <int>
1	2000	Jan	1	6	9083	1
2	2000	Jan	2	7	8006	2
3	2000	Jan	3	1	11363	3
4	2000	Jan	4	2	13032	4
5	2000	Jan	5	3	12558	5
6	2000	Jan	6	4	12466	6

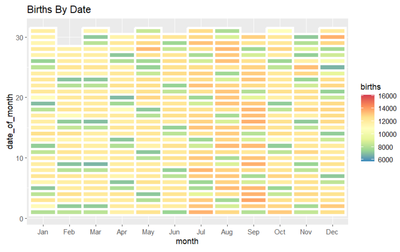
6 rows

Heatmap of US Births

Below is a heat map of the number of births per day in the US. The mathematical proof of the Birthday Paradox considers each day to have the same likelihood of being selected; however, in the heatmap you can tell that not all days are equal. The summer months of July, August, and September all have higher number of births than winter months of January and February. You can also see there are certain days that are blue/green which indicate that very relatively few births happen on these days. In the simulation we will be using the data in the heatmap for selecting a new person to add into a room, meaning that all days do not have an equal likelihood of being selected.

Hide

```
# creating our heatmap of births by month and day
library(ggplot2)
my_order = c('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec')
ggplot(births, aes(x = month, y = date_of_month, fill = births)) +
  geom_tile(color = "white",
            lwd = 1.5,
            linetype = 1) +
  scale_fill_distiller(palette = "Spectral") + ggtitle("Births By Date") +
  scale_x_discrete(limits = my_order)
```



Mathematical Reasoning

To calculate the probability of sharing a birthday with a group of people is actually quite simple. But its not as simple as:

I was born on August 8th. The probability of me being born on that day (if all days are equal) is $1/365$. The probability of my friend Lauren being born on that same day is, you guessed it, $1/365$. Pretty small odds, so let's say she's not born on the same day as me. Our next friend joining the room has two different days to not share, so $2/365$. And so on and so on, until we get to the 23rd person. So instinctively you would say 22 different birthdays in the room so the probability should be $22/365$, which is only 6%! But this is not correct.

How we should be calculating the Birthday Paradox is by calculating the probability of *NOT* sharing a birthday with someone else in the room. So instead the math would look like this:

Person One was definitely born on a day so their probability was $365/365$ or 1. Person Two has a probability of $364/365$ of not sharing a birthday with Person One. So

$$\frac{365}{365} * \frac{364}{365}$$

is the probability that that don't share a birthday and

$$1 - \frac{365}{365} * \frac{364}{365}$$

is the probability they do share a birthday. Continuing this for 23 people we have:

$$\frac{364!}{342! * 365^{22}}$$

which equals 0.492703 likelihood that they do not share a birthday. And that means, with 23 people in a room there is a 50.729T percent chance that they DO share a birthday.

Paradox Simulation

The simulation

Before we run the simulation I need to create our population dataset. To do this I will simply just add each day of the year *n* times to an array. This will act as a giant waiting room of people to select from.

Hide

```
# creating a list of every single birthday from 2000 - 2014 to represent our population
# this list will be used like drawing a card from a deck without replacement
fullList <- c()
for (x in 1:nrow(births)) {
  currentDate <- rep(births$day_of_year[x],births$births[x])
  fullList <- c(fullList, currentDate)
}
```

The simulation itself is fairly simple. We will first start with an empty room and add one person at a time until the room contains two people with the same birthday (just day and month, not year). Once there are two people with the same birthday, the number of people in the room will be recorded, the room will be emptied, and the simulation will start again. The simulation will run 1,000 times to try to get a good distribution of results.

Hide

```
# creating a loop for the simulation
# results is an array of the number of people needed until a birthday is repeated
results <- c()
i <- 1
# run simulation 1000 times
while (i < 1001){
  # initialize listPop to be our entire population to pull from
  # initialize usedDates to represent the birthdays currently in the room
  # force stop the loop at 364 because we are guaranteed to have a match in the next loop
  listPop <- fullList
  usedDates <- c()
  peopleInRoom <- 1
  while (peopleInRoom < 365){
    # pick a random birthday from our total population
    # remove that birthday from the population list
    # add birthday to the usedDates list
    # if the birthday is already in the room stop adding people to the room
    dateIndex <- match(sample(listPop, 1), listPop)
    datePicked = listPop[dateIndex]
    listPop <- listPop[-c(dateIndex)]
    if (datePicked %in% usedDates){
      usedDates <- c(usedDates,datePicked)
      break
    } else {
      usedDates <- c(usedDates,datePicked)
    }
    peopleInRoom <- peopleInRoom + 1
  }
  # add the number of people needed in the room to the results array
  results <- c(results, length(usedDates))
  i <- i + 1
}
```

The results

Hide

```
summary(results)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   2.00   15.00   23.00   24.54   33.00   77.00 
```

Hide

```
paste("Number of iterations with less than 23 people in the room:", sum(results <= 23))
```

```
[1] "Number of iterations with less than 23 people in the room: 513"
```

Hide

```
paste("Percent of iterations with less than 23 people in the room:", sum(results <= 23) / length(results))
```

```
[1] "Percent of iterations with less than 23 people in the room: 0.513"
```

Hide

```
paste("Average number of people in the room before birthday is repeated:",mean(results))
```

```
[1] "Average number of people in the room before birthday is repeated: 24.54"
```

Hide

```
paste("Lowest number of people before birthday is repeated:",min(results))
```

```
[1] "Lowest number of people before birthday is repeated: 2"
```

Hide

```
paste("Most number of people before birthday is repeated:",max(results))
```

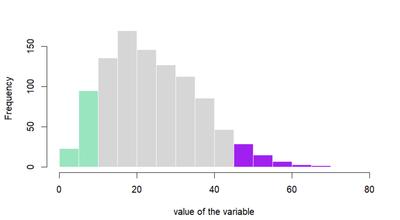
```
[1] "Most number of people before birthday is repeated: 77"
```

Hide

```
my_hist=hist(results , breaks=20  , plot=F)

my_color= ifelse(my_hist$breaks < 7, rgb(0.2,0.8,0.5,0.5) , ifelse (my_hist$breaks >=45, "purple", rgb(0.2,0.2,0.2,0.2) ))

plot(my_hist, col=my_color , border=F , main="" , xlab="value of the variable", xlim=c(0,80) )
```



Interestingly enough, the simulation confirmed the Birthday Paradox. Of the 1,000 simulations, 513 iterations had a repeated before at or before the 23rd person added to the room. Of course with enough iterations an low result of 2 people can be expected, but the maximum number of people needed to have a repeated birthday was only 77 people. This means that you can almost guarantee that in a room of 77 people or more, there will be a repeated birthday. The math makes sense and the simulation confirms it, but it still hurts my brain. Out of 365 potential birthdays, you only need between 23 and 77 of them. Why do we have so many days in a year then?