CFerrari_Assignment6

Charley Ferrari

Thursday, March 05, 2015

Problem Set 1

(1) When you roll a fair die 3 times, how many possible outcomes are there?

Assuming the die has 6 sides and we care about the order:

```
outcomes <- 6 * 6 * 6

outcomes
```

```
## [1] 216
```

(2) What is the probability of getting a sum total of 3 when you roll a die two times?

```
die1 <- c(rep(1,6),rep(2,6),rep(3,6),rep(4,6),rep(5,6),rep(6,6))
die2 <- rep(1:6,6)

Samplespace <- data.frame(die1 = die1, die2 = die2, sum = die1 + die2)

nrow(Samplespace[Samplespace$sum == 3,]) / nrow(Samplespace)
```

```
## [1] 0.05555556
```

(3) Assume a room of 25 strangers. What is the probability that two of them have the same birthday? Assume that all birthdays are equally likely and equal to 1/365 each. What happens to this probability when there are 50 people in the room?

To simplify this problem, I'll calculate the probability that everyone has a unique birthday, and subtract this from 1 to get the probability that at least 2 people share a birthday (I'm making the assumption that at least 2 people can share a birthday, and this isn't asking if ONLY 2 people share a birthday.)

The first person's birthday can be any day, so it's 365/365. The second person's birthday now only has a choice of 364 potential days, so the probability that these two don't share is (365/365) * (364/365). Similarly, the third person will have 363 potential birthdays, so (365/365) * (364/365) * (363/365).

To repeat this for 25 people, we'll go down to 341/365.

```
pdiff <- prod(365:341)/(365^25)
```

And now we'll need to subtract this from 1 to get the probability that two people have the same birthday:

```
psame <- 1 - pdiff
```

If there are 50 people in the room, we'll go down to 316/365 in our cumulative product, and do the same thing:

```
pdiff <- prod(365:316)/(365^50)

psame <- 1-pdiff
```

The probability goes up to around 97%!

Problem Set 2

Sometimes you cannot compute the probability of an outcome by measuring the sample space and examining the symmetries of the underlying physical phenomenon, as you could do when you rolled die or picked a card from a shuffled deck. You have to estimate probabilities by other means. For instance, when you have to compute the probability of various english words, it is not possible to do it by examination of the sample space as it is too large. You have to resort to empirical techniques to get a good enough estimate. One such approach would be to take a large corpus of documents and from those documents, count the number of occurrences of a particular character or word and then base your estimate on that.

Write a program to take a document in English and print out the estimated probabilities for each of the words that occur in that document. Your program should take in a file containing a large document and write out the probabilities of each of the words that appear in that document. Please remove all punctuation (quotes, commas, hyphens etc) and convert the words to lower case before you perform your calculations.

```
library(stringr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
setwd("E:/Downloads/Courses/CUNY/SPS/Git/IS 605 Fundamentals of Computational Mathematics/Assignment 6")

filename <- "assign6.sample.txt"

singleword <- function(filename){

  textlistorig <- scan(filename, character(0), encoding = "UTF-8", quote=NULL)

  textlist <- tolower(str_replace_all(textlistorig, "[^A-Za-z/']", ""))
  textlist <- str_replace_all(textlist,"/'s$", "")
  textlist <- textlist[textlist != ""]

  textlistDF <- data.frame(Word = textlist)


  textlistDFS <- textlistDF %>%
```

```
    group_by(Word) %>%
    summarize(Count = n()) %>%
    mutate(Probability = Count / sum(textlistDFS$Count))

  return(textlistDFS)

}
```

Extend your program to calculate the probability of two words occurring adjacent to each other. It should take in a document, and two words (say the and for) and compute the probability of each of the words occurring in the document and the joint probability of both of them occurring together. The order of the two words is not important.

```
bigramtest <- function(filename, word1, word2){

  textlistorig <- scan(filename, character(0), encoding = "UTF-8", quote=NULL)

  textlist <- tolower(str_replace_all(textlistorig, "[^A-Za-z/']", ""))
  textlist <- str_replace_all(textlist,"/'s$", "")
  textlist <- textlist[textlist != ""]

  textlistDF <- data.frame(Word = textlist)


  textlistDFS <- textlistDF %>%
    group_by(Word) %>%
    summarize(Count = n())
  textlistDFS <- textlistDFS %>%
    mutate(Probability = Count / sum(textlistDFS$Count))

  textlistDF$NextTo <- c(NA, head(textlist, -1))
  textlistDF <- textlistDF[2:nrow(textlistDF),]

  textlistDF$Flag <- ifelse((textlistDF$Word == word1 & textlistDF$NextTo == word2) |
                              (textlistDF$Word == word2 & textlistDF$NextTo == word1),1,0)

  return(sum(textlistDF$Flag) / nrow(textlistDF))

}
```